IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 20 February 2023, accepted 27 March 2023, date of publication 3 April 2023, date of current version 7 April 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3264016

RESEARCH ARTICLE

Arbiter PUF–A Review of Design, Composition, and Security Aspects

S. HEMAVATHY, (Graduate Student Member, IEEE), AND V. S. KANCHANA BHAASKARAN, (Senior Member, IEEE)
Vellore Institute of Technology, Chennai 600127, India
Corresponding author: V. S. Kanchana Bhaaskaran (vskanchana@gmail.com)

ABSTRACT Hardware security modules play a crucial role in protecting and preserving technologically integrated systems that are used in daily life. They employ cryptographic protocols to secure a system against adversaries. Generally, cryptographic algorithms and security keys are quintessential for maintaining the security of a system. Cryptography uses a secret key to encipher and decipher the data. These secret keys are stored in a non-volatile memory that attackers can easily access. The hardware security primitive, Physical Unclonable Function (PUF) is a promising alternative for enhancing the security of interconnected devices. A PUF produces an output in response to an input based on the physical structure and intrinsic manufacturing variations of an integrated circuit (IC). The generated random response being unpredictable, act as a robust secret key in cryptographic protocols. The first silicon PUF is the Arbiter PUF, which can instantly produce significantly more secret keys based on the input with a lightweight design. Due to its advantage, it is best suited for device authentication in resource-constrained applications employing the Internet of Things (IoT). The PUF is also suitable for applications such as the Internet of Vehicles, the Internet of Medical Things, RFID (radio frequency identification) tags, and smart cards. In this paper, the basic Arbiter PUF design is implemented in ZedBoard to analyze the PUF performance characteristics for 16, 32, and 64-bit responses. A review of Arbiter PUF design, different compositions of Arbiter PUF, their individual characteristics, and vulnerabilities against machine-learning attacks have been presented at their broader best in this paper.

INDEX TERMS Arbiter PUF, cryptography, hardware security, machine learning attacks, physical unclonable function, ZedBoard.

I. INTRODUCTION

Industry 4.0 is a platform of boundless digitized technological innovations. It relates, to the development of industrial automation, privacy, and secret data exchange between consumer and security devices interconnected through the Internet of Things and cloud computing, among other things. With Cyber-Physical Systems (CPS) set to evolve as the imminent target for adversaries to be attacked through the core component of the system, it has become a challenging task for security professionals to safeguard the devices connected to CPS.

The authentication, authorization, and privacy of interconnected devices play a crucial role in IoT-based application platforms [1], [2], [3]. Cryptographic algorithms are employed efficiently for efficient endorsement and enrolment to encipher and decipher the data by saving the secret keys in battery-backed Static Random Access Memory (SRAM), Non-Volatile Memory (NVM) such as flash memory, and Electrically Erasable Programmable Read-only Memory (EEPROM) [4]. If a cryptanalyst succeeds in inferring the key by invasive attacks, the effect is catastrophic. A contemporary technology that requires minimum resources while possessing the propensity to resist attacks is essential to avoid these security problems. Efforts in that direction have lead the research community to develop PUFs. Gassend et al. introduced the concept of Physical Random Function, later known as Physical Unclonable Function for reliable authentication, in 2002 [5].

Integrated circuits are essentially manufactured to have identical digital logic functionality so that their digital response to input remains unaffected. However, in the

The associate editor coordinating the review of this manuscript and approving it for publication was Jiafeng Xie.

semiconductor industry, the ICs housed in the same die incur inadvertent manufacturing variations, inter and intra-die variations, doping level variations, and even a few logic process modules exhibiting different outputs for the same logic functionality [6]. This unique behavior, which is even more dominant for contemporary lower technology nodes, and usually considered detrimental in the semiconductor industry, is being exploited in developing PUFs.This feature of PUF is exploited for generating its fingerprint, which is specific to the chip. IoT applications that necessitate lightweight security measures can integrate PUF for impenetrable security solutions at a lower cost factor. The paper offers a deeper insight into the design, reproducibility, and security of PUFs, with a focus on the Arbiter PUF, its variants, and their resilience to vulnerability.

A. RELATED WORK

PUFs can be classified as strong PUFs and weak PUFs based on the number of Challenge and Response Pairs (CRPs) employed [7]. Strong PUFs have a larger number of CRPs, and the size of each CRP increases significantly with the number of PUF circuits, and they ensure a secure environment without additional cryptographic hardware. A weak PUF, also termed "physically obfuscated key" [8], on the other hand, generally possesses one challenge that is used for secured key storage. Several strong PUFs such as Arbiter PUF (APUF) [6], Loop PUF [9], Configurable Ring Oscillator PUF (CRO PUF) [10], Bistable Ring PUF (BR PUF) [11], Dual Mode PUF [12], and Configurable Tristate PUF (CT PUF) [13] have been proposed in the literature with significantly enhanced performance features. Loop PUF suffers from replicating the delay chains when implemented in a Field Programmable Gate Array (FPGA) and needs more investigation for its robustness against machine learning attacks. CRO PUF includes a specific inverter to the delay chain based on the configuration of the challenge to the multiplexer. This delay information is easily traceable by the adversary to model the PUF. Comparing CT PUF to APUF, CRO PUF, BR PUF, and Dual Mode PUF, the modeling accuracy of CT PUF utilizing an artificial neural network equals 50% [13]. Almost all the promising designs have been successfully modeled with and without a mathematical model. Yet, the hardware complexity of APUF is significantly lower than the other strong PUFs.

Additionally, various other weak PUFs have also been developed and studied in addition to strong PUFs. They employ adiabatic logic [14], [15], delay-based design [16], memory-based logic [17], [18], and the metastability feature [19] to generate a secret key. These include the Ring Oscillator PUFs (RO PUFs) [16] that use the phase delay between the oscillation of two or more connected inverters to generate a unique response. The SRAM PUFs [17] use the inherent variations in the threshold voltages of transistors in an SRAM cell to generate a unique response. The weak PUFs being harder to model, are resilient to machine learning attacks. However, the weak PUF size linearly increases to enable the extraction of more responses in tune with the bit length. Each type of PUF has its own set of advantages and disadvantages, and the specific type that is most suitable for a given application depends on the requirements and constraints of that application.

Arbiter PUF is the most extensively studied work in literature since it depends upon delay and timing information provided by the ICs. In addition, they can be implemented using standard manufacturing processes, which makes them relatively easy to integrate into existing electronic systems. As a result, they have gained significant attention from researchers in fields such as computer security, license management, and secured communication. Rigorous researches have emerged in the mathematical modeling of the APUF and its compositions using Machine Learning, Side-Channel Analysis, and Deep Learning. They motivate more research on APUF to make it increasingly resistant to modeling attacks.

The significant contributions of the paper are as follows:

- Understanding of the concept of PUF and its performance characteristics with illustrative examples.
- Design of a basic Arbiter PUF with emphasis on the placement and routing of the switch block and the arbiter. The design is implemented in the ZedBoard platform to evaluate its PUF metrics for the 16, 32, and 64-bit responses.
- Application of Arbiter PUF in security protocols during the enrollment and authentication phases.
- Elucidation of the characteristics and vulnerabilities of different compositions of Arbiter PUF.
- The mathematical modeling of the variants of APUF and its resilience against adversary attacks.

The paper is structured as follows. Section II provides a brief introduction to PUF and its metrics. Section III elaborates the Arbiter PUF with a focus on design, timing delay modeling, placement and routing, and typical applications. Section IV presents the compositions of Arbiter PUF. Section V describes the machine learning attacks achieved in different types of Arbiter PUFs for various variants published in the literature. Section VI concludes by highlighting the opportunities and future scope of research.

II. PHYSICAL UNCLONABLE FUNCTIONS

PUF can be precisely defined as a simulated physical system using an input or challenge to generate a response or output [20], [21]. Mathematically, a PUF can be defined as,

$$f : Ch \rightarrow R. \tag{1}$$

$$f(ch) : r(ch \in Ch, r \in R). \tag{2}$$

where Ch is an external stimulus or an applied challenge, and R describes the output or response produced by the PUF.

A PUF design is made such that replicating two identical PUFs, even with extensive computational resources, becomes virtually impossible [5], [6]. The responses for the same PUF received from different ICs are unique, which acts as the

TABLE 1. Basic symbols and notations.

Symbol	Definitions
HD	Hamming Distance
L	Length of the response / token R
N	Number of devices / chips
K	Number of different responses / tokens generated per device
T	Number of tests performed per response / token
l	Bit position of the response / token,$1 \leq l \leq L$
n	Index of a device, $1 \leq n \leq N$
k	Index of response / token, $1 \leq k \leq K$
t	Index of a test, $1 \leq t \leq T$
h	The value of Hamming distance between L response bits
$ID_{n,k}$	The correct ID k that is expected to be generated by device n
$b_{n,l}$	The expected right response bit to be generated in device n of length l
$b_{n,k,l}$	The expected right response bit to be generated in device n for token k, length l
$b_{j,k,l}$	The actual response bit l of token k estimated to be generated in device j
$b_{n,k,t,l}$	The empirically generated response bit in device n for token k, length l and t-test
$S_{n,k,l}$	The steadiness of l^{th} bit of $ID_{n,k}$
$C_{n,k,l}$	Correctness of the l^{th} bit of $ID_{n,k}$
$d_{n,l}$	Sum of HD of the possible bit combinations
$b_{n,i,l}$ and $b_{n,j,l}$	The correct response generated for i^{th} and j^{th} challenge
$\oplus$	Exclusive-OR operation

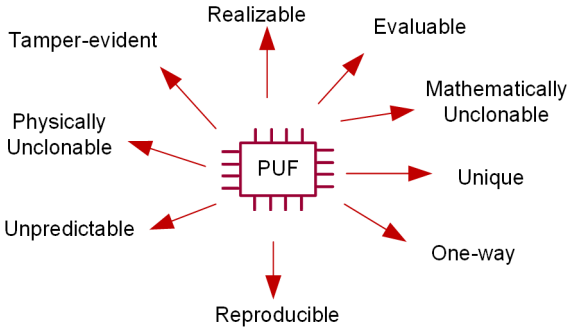

FIGURE 1. Properties of PUF.

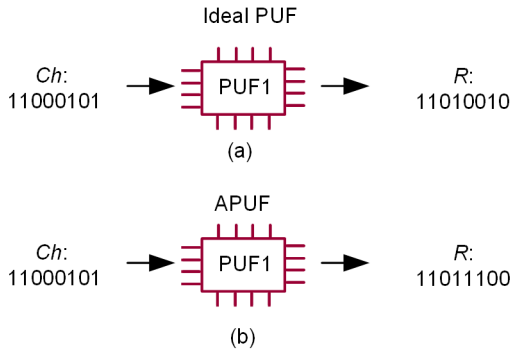

FIGURE 2. Illustrative example for Uniformity metric.

fingerprint for individual ICs. Thus, the PUF circuits are unclonable, robust, easy to evaluate, and unique.

In silicon-based PUFs, the individual responses depend on the hidden delay/timing feature [6], manufacturing variations of ICs incurred even while using the same mask, uneven wiring length, and non-stochastic doping processes [22]. These random deviations have been modeled using Gaussian and probabilistic distributions [23], [24]. It is impossible for an adversary to physically clone a PUF even if one acquires all the necessary data. This is because the CRPs generated by the PUF cannot be described even by the manufacturer if the process variations are high enough. If an attacker tries for an invasive attack to study the nature of a device, it is apparent that it invariably results in a change in the delay of the devices or wires. Fig. 1 presents the properties of PUF, which determine the capabilities and limitations of the PUF and can influence the suitability of the PUF for a given application. Being tamper-evident and one-way mapped enables the use of PUFs for security better than their counterpart security measures. The realizable, evaluable, and reproducible nature of the device-level security of PUF enhances the use of the inherent feature of IC technology to the specific instance of the physical device. Physically and mathematically unclonable, unpredictable ensures a unique response from PUF. Table 1 presents the basic symbols and notations used in this paper.

A. PUF METRICS

A quantitative evaluation to explore the performance of Arbiter PUF was first defined and evaluated by Hori et al. [25], [26]. The evaluation parameters specified had one inter-device performance indicator: Uniqueness, and four intra-device performance indicators: Randomness, Steadiness, Correctness, and Diffuseness. Maiti et al. [27] defined additional PUF parameters to compare different PUF models of strong and weak PUFs. Inter-chip and Intra-chip variations are fundamental concepts that explain variations among devices or within a single device. These variations have been explained in three-axes, namely, device, space, and time [27].

- *Inter-chip variation:* Inter-chip variation (HD_{inter}) [28] between any two different devices for the same digital design PUF with a challenge vector Ch=Ch_1, $Ch_2, \ldots, Ch_m$ embedded in it can be expressed as,

$$HD_{inter} = P(R_i(Ch) \neq R_j(Ch)). \tag{3}$$

  where $R_i(Ch)$ and $R_j(Ch)$ represents responses of $i^{th}$and j^{th} device, for a given challenge vector.
- *Intra-chip variation:* Intra-chip variation (HD_{intra}) [28] is a measure to identify the number of biased bits in R of length L for a given challenge in the same device.
- *Randomness Vs Uniformity:* Randomness (H_n) [26] can be well-defined as the balance of 0 and 1 in the response of PUF and can be expressed as,

$$H_n = -\log_2 max(p_n, 1 - p_n). \tag{4}$$

  The randomness of a bit-sequence can be described using H_n for a single device. H_n can take the maximum value 1 when $p_n = 0.5$ and the most negligible value 0 for

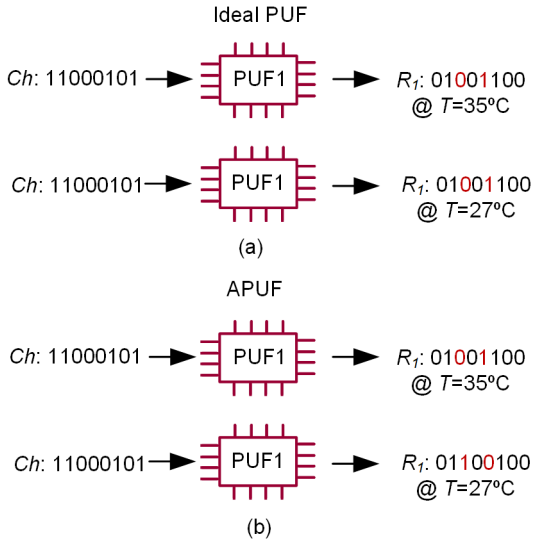

FIGURE 3. Illustrative example for Reliability metric.

$p_n = 0$ or $p_n = 1$. The relative frequency of appearance of p_n as 1 can be expressed as,

$$p_n = \frac{1}{K.T.L} \sum_{k=1}^{K} \sum_{t=1}^{T} \sum_{l=1}^{L} b_{n,k,t,l}. \tag{5}$$

Uniformity ($Unif$) [27] for a chip n can be evaluated by determining the average hamming weight of the response and represented as,

$$(Unif)_n = \frac{1}{L} \sum_{l=1}^{L} b_{n,l} \times 100\%. \tag{6}$$

For K different IDs,

$$(Unif)_K = \frac{1}{K.L} \sum_{k=1}^{K} \sum_{l=1}^{L} b_{n,k,l} \times 100\%. \tag{7}$$

The primary difference in the definition of Randomness defined by Hori et al. [26] and Uniformity by Maiti et al. [27] is the total number of samples measured for each ID, T. Considering an example illustrated in Fig. 2.a, the uniformity for an ideal PUF is expected to be 50%. If it is equal to either 100% or 0%, all the bits in an L-bit response are either at logic-1 or logic-0. As in Fig. 2.b, the hamming weight for an 8-bit response (11011100) counts to 5. The uniformity for the given challenge (11000101) is 62.5% representing the number of 1 is more than the number of 0.

- *Reliability:* The delay difference between the upper and lower paths ought to be significant enough to generate a reliable bit, albeit the delay increases with temperature at different rates [29]. However, the delay between two paths can intersect over the working temperature when the path delay difference is negligibly small and increases the chance of bit flip. Reliability is a measure to quantify the capability of the PUF to reproduce a reliable response R for a given set of challenges irrespective of the changes in environmental conditions such as temperature and supply voltage [27]. A PUF can be reliable only if it can reproduce the same response (100%) despite hostile environmental conditions. The intra-chip hamming distance (HD_{intra}) is used to estimate the reliability metrics within a chip n by comparing the responses obtained at ambient temperature and nominal supply voltage $R_n(l)$ with that of varying voltage and temperature $R'_n(l)$ for m samples.

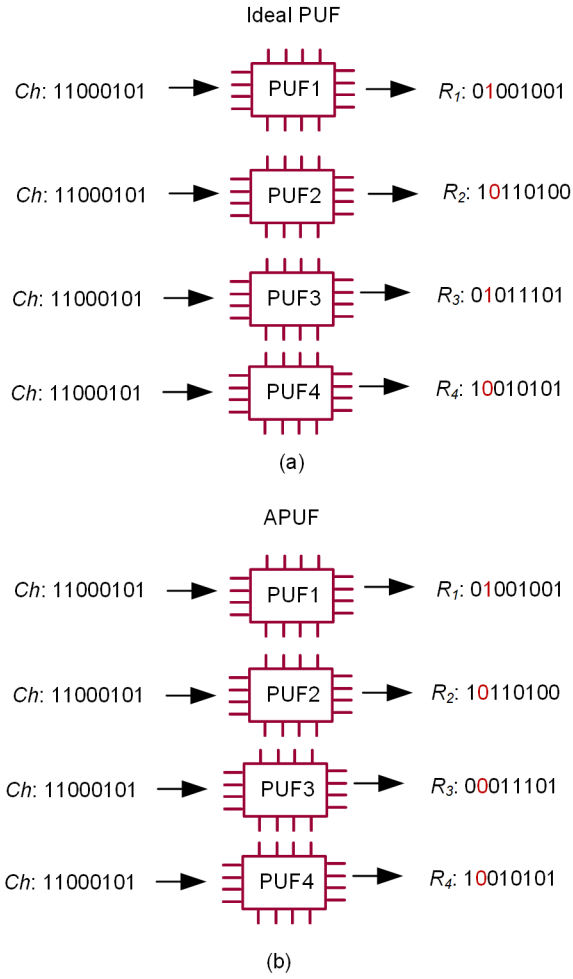

FIGURE 4. Illustrative example for Bit-aliasing metric.

$$HD_{intra} = \frac{1}{m} \sum_{i=1}^{m} \frac{HD(R_n(l), (R'_n(l))}{n} \times 100\%. \tag{8}$$

The reliability of a PUF in terms of HD_{intra} can be expressed as,

$$Reliability = 100\% - HD_{intra}. \tag{9}$$

Consider an example of a comparison of PUF for the ideal condition with an APUF, as depicted in Fig. 3. An ideal PUF shown in Fig.3.a generates a similar output (01001100) for a given input of 11000101 for

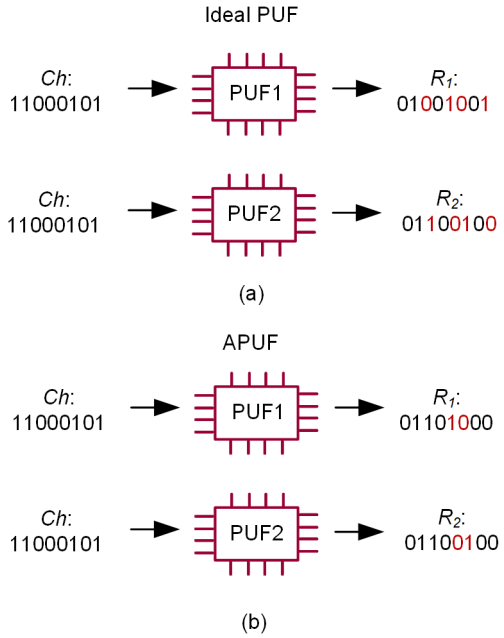

FIGURE 5. Illustrative example for Uniqueness metric.

two different temperatures, thus achieving reliability of 100%. However, in APUF, a bias exists in the output for varying temperature conditions, as illustrated in Fig.3.b. For the same challenge of 11000101, the *HD* for the two responses is two, and the reliability reduces to 75% compared to the ideal PUF, indicating the PUF design is unreliable at varying temperatures.

- *Bit-aliasing:* Bit-aliasing for different chips is said to happen when they produce similar response bits. Bit-aliasing within the chip is said to happen when homologous bits are biased due to static variations [27]. Both these are undesirable and is expressed as,

$$(Bit - aliasing)_{k,l} = \frac{1}{N}\sum_{n=1}^{N} b_{n,k,l}. \tag{10}$$

Fig. 4 elucidates the bit-aliasing for ideal APUF for four different ICs given the challenge as 11000101. For example, in Fig.4.b, consider the second bit of the 8-bit response; the bit-aliasing corresponds to 25%, indicating that the bit is more biased towards 0. One possible reason can be inferred as an unsymmetrical physical layout of APUF resulting in delay variations and orienting the output to generate 0, as shown in Fig.4.a. Thus, it is challenging to use PUFs as a means of securely identifying a device, since it is possible for multiple devices to have the same "fingerprint."

- *Uniqueness:* Uniqueness (U_n) [26] is a measure used to indicate how unique are the generated IDs between different devices. Uniqueness ought to be close to 50% and is expressed as,

$$U_n = \frac{4}{K.L.N}\sum_{k=1}^{K}\sum_{l=1}^{L}\sum_{j=1,j\neq n}^{L}(b_{n,k,l} \oplus b_{j,k,l}). \tag{11}$$

The sample mean $\bar{U}$ is the average of U_n of all the devices used for evaluation [25]. It is to be close to 1 to predict that the CRPs are indistinguishable.

$$\bar{U} = \frac{4}{K.L.N^2}\sum_{k=1}^{K}\sum_{l=1}^{L}\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}(b_{n,k,l} \oplus b_{j,k,l}). \tag{12}$$

Uniqueness can also be evaluated using Inter-chip Hamming Distance (HD_{INTER}) between two chips, i and $j(i \neq j)$, having n-bit responses $R_i(n)$ and $R_j(n)$. A chip is deemed unique when the HD_{INTER} is close to an ideal value of 50%. The average inter-chip *HD* (Uniqueness) can be expressed as in [27], $U(HD_{INTER}) =$

$$\frac{2}{k(k-1)}\sum_{i=1}^{k-1}\sum_{j=i+1}^{k}\frac{HD(R_i(n),(R_j(n))}{n} \times 100\%. \tag{13}$$

Fig. 5 illustrates the uniqueness with an example. Let the responses from PUF1 and PUF2 be 01001001 and 01100100, respectively, for a challenge of 11000101, as shown in Fig.5.a. The *HD* between the two responses for an ideal PUF is four, and the uniqueness equals 50%, illustrating the fact that the responses are unique for PUF1 and PUF2. On the other hand, for the APUF shown in Fig.5.b, for the same challenge, the *HD* is two, and the uniqueness is 25%, elucidating the number of unique responses that are determined from an APUF is comparatively less. The reason is that the responses are biased to 0 or 1 because the average delay variation is lesser than the average delay difference.

With the background of the importance of PUF properties and PUF metrics, Section III presents the Arbiter PUF.

III. ARBITER PHYSICAL UNCLONABLE FUNCTION

Following the above discussions on PUF metrics as a background, this section presents a holistic review of the conventional Arbiter PUF found in the literature. The performance characteristics of the conventional APUF have been studied using the programmable SoC (system-on-chip) ZedBoard Zynq-7000 board from Xilinx®, and various PUF metrics have been evaluated. This process step is followed by comparisons against other variants of APUF found in the literature, as explained in the subsequent sections. Hence, an actual comparison of various PUF metrics pertaining to each contribution by the researchers and interpretation of their designs and their impact on the metrics have been made. This is the first comprehensive study and analysis of the literature from 2002 to the present, as perceived by the authors.

Arbiter PUF [6] utilizes the property of intrinsic delay variations of each device to generate a unique identity or token. It is devised to be the first silicon PUF structure and found to

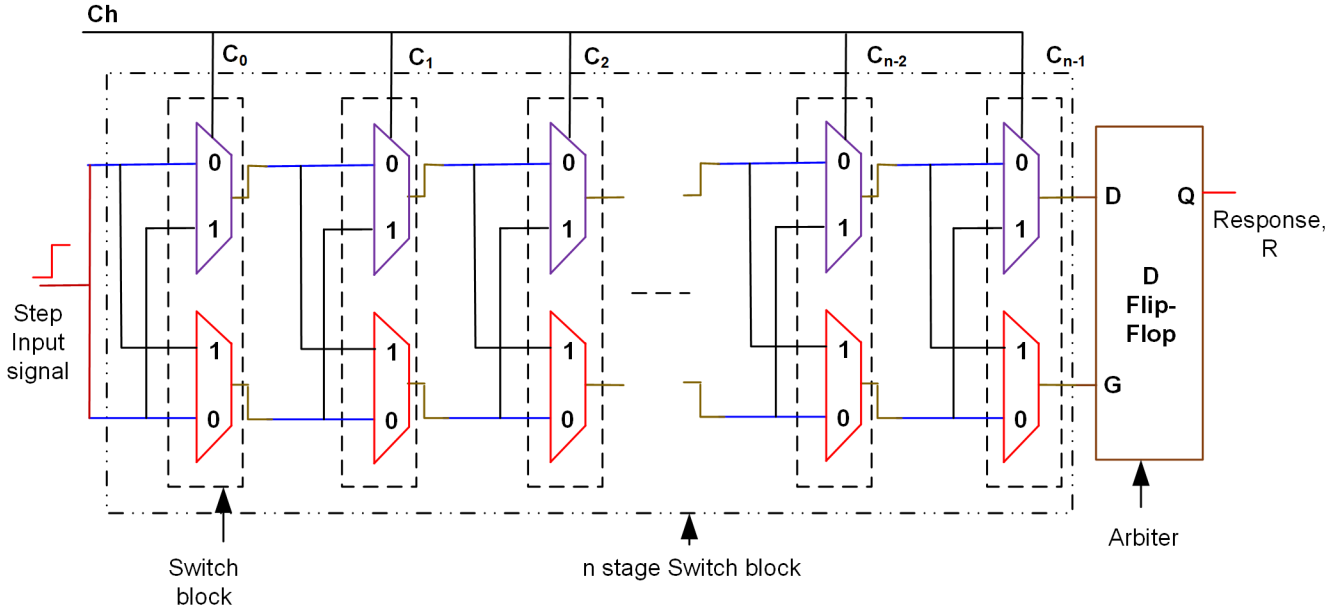

FIGURE 6. Arbiter PUF [6].

be best suited as lightweight hardware security fundamental component. The APUF is built based on a linear additive delay model consisting of a path-swapping switch/switch block (SB) and an arbiter circuit. The SB, made of two parallel 2-to-1 multiplexers (MUX), introduces the delay in the design and has n-stages depending on the challenge length (Fig. 6). The switch block delays are increased by adding buffers and cross-coupling the path between the upper and lower MUXes. The first switch block (Fig. 6) is initially excited by a step input, and the parallel MUXes share the rising edge. As depicted in Fig. 7, when the Ch=0, input signals (i_0 and i_1) flow parallelly. If Ch=1, the signal path is cross-coupled between the upper and lower MUXes, influencing the wanton signal delay while traveling through each switch block. Finally, whichever signal reaches the arbiter first wins the race and decides the random number value to be a 0 or 1. The response will be 0 if the lower path signal (o_1) is faster than the upper path signal (o_0) and vice-versa.

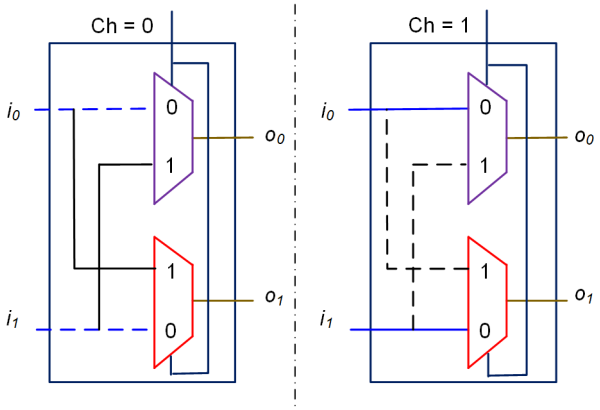

FIGURE 7. Switch block for Ch=0 and Ch=1 [6].

A. ARBITER DESIGN

An AND gate was initially used as an arbiter to capture the signal from the last SB and ascertain the response based on the edges of the two signals, and later, it was replaced by a D Flip-Flop (D-FF) [6], and a Reset/Set latch (RS latch) [8]. As shown in Fig.6, when the input is applied to the APUF, the upper and lower paths pass the input signal with a specific delay based on the challenge. Let the delay created by the upper and lower paths be T_1 and T_2. As shown in Fig.8, the delay difference, $\Delta T = T_1 - T_2$ will be the input to the arbiter. Fig. 8 illustrates that the D-FF samples the response to be 1 when the positive edge of input D arrives before the positive edge of input *clk* by a time value more significant than the setup time of the D-FF, or else, the output will remain at 0. A logical 1 or 0 is achieved when the delay difference is prominent. If the delay difference is considerably less, the arbiter enters a metastable condition, resulting in an unstable response. Since the path from D to Q and Clock to Q are asymmetrical, the output bits in a D-FF are more biased. As shown in Fig. 9, RS-latch features a symmetric propagation delay between the input and the output that is used to reduce the bias [32]. A race condition is experienced when both the inputs S and R change from high to low. To mitigate the impacts of metastability, an RS-latch can be used in place of the D-FF. While implementing in FPGA, the D-FF utilizes only one Look-Up table (LUT), resulting in biased output due to uneven symmetrical routing in either the upper or lower path. On the other hand, the use of a cross-coupled NAND gate for an RS-latch engages one

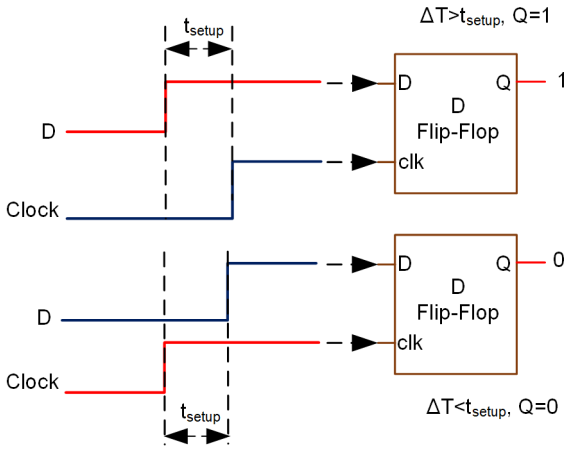

FIGURE 8. Response based on the signal transition in D- FF [6].

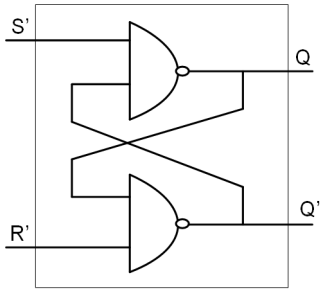

FIGURE 9. RS latch.

LUT for each NAND gate, thus achieving a symmetrical path with reduced bias in response.

B. IMPLEMENTATION OF SWITCH BLOCK

Considerable research contributions are found in achieving symmetrical routing between the switching blocks. The various configurations of SB design include employing MUX primitives [33], [34], [35], Programmable Delay Line (PDL) logic [32], [36], a 6-2 input LUT combining two parallel MUX in one LUT [37], PDL and MUX without cross-coupling [38], tri-state buffers [39], and Path Changing Switch (PCS) [40] focusing on response bias and enhanced uniqueness. A Schmitt trigger has been used as a buffer between the switch blocks instead of a CMOS (Complementary Metal-Oxide Semiconductor) buffer as it is susceptible to process variations in [41]. Recently, a 4 × 4 switch block has been presented instead of a 2 × 2 switch block, which can generate a maximum of 6 responses [42]. However, it is claimed that the improved performance is only for random placement. The path delay of a classical APUF has been transformed based on the acquired response of 0 or 1 using an adjustable module in an Adjustable APUF (A-APUF) [43]. A sub-threshold APUF [44], [45] has been designed based on the premise that these are more prone to process variations in 45 nm deep sub-micron technologies. The switch blocks made of multiplexers have been replaced with NAND gates in the design, along with additional interconnects. The delay between the top and the bottom paths stands extended in the upper path by 2 μm. An optimal sub-threshold voltage of 0.43 V has been applied based on the minimum power-delay product value. The design model has also been verified with a super-threshold voltage of 1.1 V. Even though the mean *HD* and reliability are close to the ideal value in the proposal, the number of CRPs considered is found to be significantly less.

C. TIMING DELAY MODEL, PLACEMENT AND ROUTING IN FPGA

The prerequisite for modeling is the delay difference introduced between the top and bottom path signals while transitioning from one stage to another. Let us consider that for stage i, the delay difference corresponding to challenge bit 0 and 1 be represented as $\delta_{0,i}$ and $\delta_{1,i}$, respectively [46]. The final delay difference can be easily computed if the delay difference for every stage is calculated individually, considering crossed paths in the computation when the challenge bit is 1. The delay difference ΔD_i after i^{th} stage can be evaluated recursively as

$$\Delta D_i = \Delta D_{i-1}.(-1)^{c_i} + \delta_{c_i,i}. \tag{14}$$

where c_i represents the i^{th} challenge bit. The final output R is calculated from the delay difference ΔD_n after computing the last delay stage n.

$$R = \begin{cases} 1 & if\ \Delta D_n > 0 \\ 0 & if\ \Delta D_n < 0 \end{cases} \tag{15}$$

The recursive delay vector $w = (w_1, .., w_{(n+1)})$ needs to be computed to design an efficient model for an APUF.

$$w_1 = \delta_{0,1} - \delta_{1,1}, \tag{16}$$

$$w_i = \delta_{0,i-1} + \delta_{1,i-1} + \delta_{0,i} - \delta_{1,i}, \tag{17}$$

and

$$w_{n+1} = \delta_{0,n} - \delta_{1,n}. \tag{18}$$

The total delay time of an n-stage APUF with a scalar multiplication $\Phi = (\Phi_1, \ldots \Phi_{n+1}) \epsilon \{-1, 1\}^{n+1}$ is given by,

$$\Delta D_n = w^T \phi. \tag{19}$$

APUFs require identical logic and routing for their combinational paths so that the difference in delay across a defined path occurs only due to process variations. In order to validate this fact, initially, the APUF was implemented by the authors in a Spartan 2 FPGA device with the average inter-chip and intra-chip values being 1.05% and 0.10%, respectively. It ensures that, on average, only one response remains unique, and the reliability of obtaining the same response is also significantly less, as also claimed in [6]. The response bits are biased depending on the relative delays of the different paths through the circuit, addressing the necessity of symmetrical placement and routing of the delay paths in the FPGA. In comparison, the symmetrical placement improved the inter-chip to 23% when implemented in an

Application-Specific Integrated Circuit (ASIC) environment, though it remained far from the desired value of 50% [6], [47]. To accurately implement delay paths, FPGA requires hard macros to integrate fixed placement and routing, and the designer has to follow a lower level of abstraction instead of RTL.

The responses are biased either due to the unsymmetrical routing of instances or the architecture of the APUF model and they can be termed implementation and architectural biases [48], [49]. The technology node, implementation platform, and external environmental factors will not affect the architectural bias. As a result, the bias in the CRPs will be only due to the architectural flaws of the PUF design. The implementation bias would be much influenced by the evaluated FPGA platform that impacts the output based on CRPs. Different authors have handled broad categories of placement of the SB for an APUF. While designing the MUX logic of the SB, one MUX primitive [33], [34], [35] from each slice can be chosen for the upper and lower paths placed adjacently, thus setting the APUF instance horizontally or vertically. Generally, in FPGA, each slice consists of four LUTs in it. Tcl commands (BEL) will help fix the LUT in positions A, B, C, and D in each slice. The input pins in the LUTs can be locked by using the lock-pin attribute in the Hardware Description Language (HDL) code [32]. The MUX primitive can be replaced either by a 3-1 LUT for each MUX logic or a 6-2 LUT for one switch block [37]. Each MUX/SB LUTs can be placed in one slice to have a unique token. If more than one LUTs are occupied in a slice while routing, it can result in a random placement of the switch block in an APUF. A random sequence of LUT for one APUF instance will result in reduced uniformity [50]. Thus, it is preferable to place the arbiter in a separate slice.

D. IMPLEMENTATION IN ZedBoard

To validate the APUF, we have implemented it as a customized IP formed in the ZedBoard using a 3-1 LUT for a MUX configuration, as shown in Fig.10. The upper and lower paths are symmetrically placed in the adjacent slices to enhance the manufacturing variations. Fig. 10 illustrates the design of APUF as a customized IP (Intellectual Property) to communicate between Programmable Logic (PL) and Processing System (PS). Using Xilinx SDK (Software Development Kit), the challenges were given to the APUF, and responses were received. Pre-processing of responses to evaluate the PUF parameters was realized using MATLAB. The APUF was designed for 16-bit, 32-bit, and 64-bit challenges, and responses of similar size were achieved. The PUF metrics, namely, uniqueness, uniformity, and reliability, have also been evaluated. The uniformity of APUF is evaluated to be 63.65%, 63.78%, and 53.76% for 16-bit, 32-bit, and 64-bit challenges. Thus, the responses of the basic APUF are more biased towards 1. The reliability for 16-bit, 32-bit, and 64-bit challenges are 98.86%, 99.52%, and 97.96%, validating a reliable random output. Table 2 illustrates that the symmetrical placement of APUF in the SoC board has resulted in a significantly high uniqueness of 42.20% (16-bit) for 50,000 CRPs compared to other FPGA platforms. The uniqueness for 64-bit is 37.75% which is comparatively higher than other technology boards. The challenges were incremented by 1-bit to have a one-bit flip in the challenge, and the corresponding responses were recorded. This customized IP design for a SOC board confirms that it is the most preferable one for lightweight and low-cost IoT applications, and it necessitates no additional overhead resources for error correction codes and fuzzy extractors.

TABLE 2. Uniqueness of APUF in various FPGA boards.

FPGA board	Uniqueness (%)	Ideal Value (%)	PUF (64-bit)
Spartan 3E (90 nm)	7.20	50	APUF [27]
Virtex 5 (65 nm)	10.2	100	APUF [25]
Spartan 6 (45 nm)	29.02	100	APUF [27]
Kintex 7 (28 nm)	20.1	100	APUF [25]
Artix 7 (28 nm)	9.42	100	APUF [25]
ZedBoard SoC (28 nm)	42.20	50	16-bit APUF
ZedBoard SoC (28 nm)	38.52	50	32-bit APUF
ZedBoard SoC (28 nm)	37.75	50	64-bit APUF

E. APPLICATION, SECURITY PROTOCOLS, AND NIST STANDARDS

Physical Unclonable Functions can be used in various applications due to their unique property and unclonability, ranging from secure cryptographic key storage [51], [52], entity authentication [52], [53], authentication framework [54], keyed-hash message authentication code [4], RFID tags [55], [56], [57], smart cards [5], [58], Certified execution and software licensing [59], Digital rights management, set-top boxes, and distributed computation [6], and IP protection [60].

The demand for an increased number of heterogeneous devices in the IoT has also raised concerns for its security. Hence, the IC vendors exploit resource-constrained hardware-based technology solutions to secure IoT devices. The usage of PUF in security protocols has also gained more importance due to its lightweight structure and more CRP space. The PUF models integrate additional techniques to obfuscate the challenge and hide the responses from malicious attacks. Different authentication protocols [6], [59] have been proposed to secure the secret key generated by the PUF employing fuzzy extractors and error correction code [52] in providing reliable CRPs.

The security offered by the protocol increases as the prover and the verifier can opt for more CRPs while using APUF. Furthermore, the used CRPs are discarded after successful authentication and updated with a new set of CRPs, which is a tremendous advantage of using APUF. Table 3 shows the various protocols that use APUF and its variants for secured authentication. It presents a glimpse of a few protocols. The Two-Stage Multiple-Choice Arbiter-based PUF (TSMCA PUF) proposed in [2] uses bi-directional authentication to

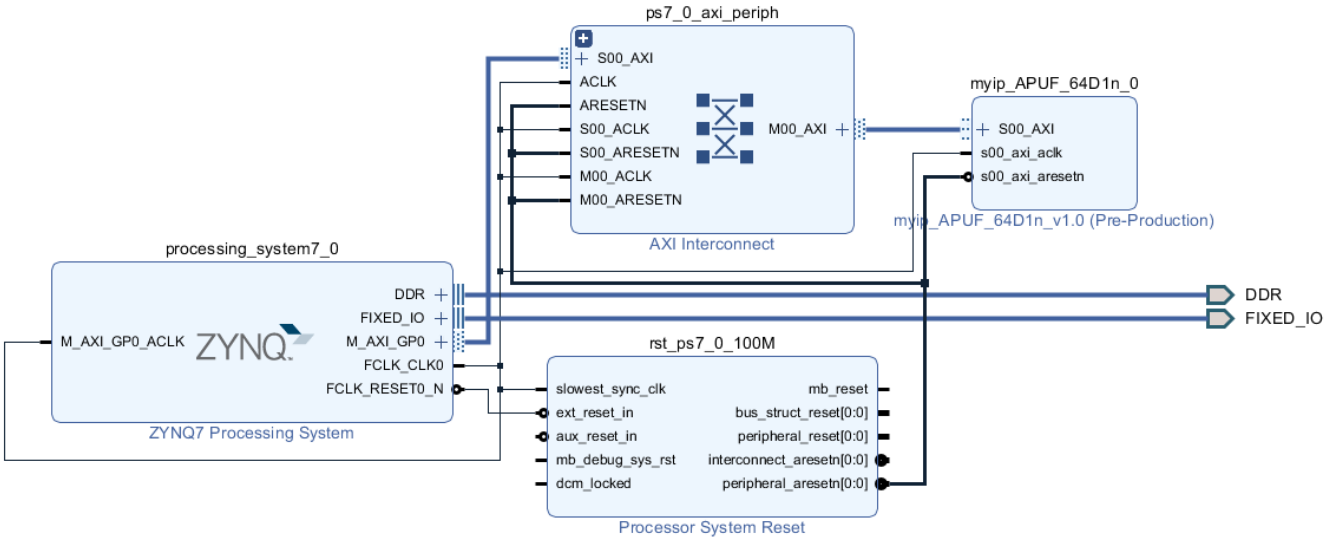

FIGURE 10. Customized IP design of APUF connected with ZYNQ processing system.

TABLE 3. Security protocols used in authentication using APUF.

Security Protocols	PUF model	Obfuscation technique	Authentication
Bidirectional PUF protocol [2]	TSMCA PUF	NA*	String-matching method
Secret Pattern Recognition [61]	APUF	NA	Secret pattern
Trinary quadruple response [62]	APUF	Trinary digit quadruple response	Bitwise comparison
OB-PUF Protocol [63]	APUF	Obfuscated Challenge	Prover response equals the response for obfuscated challenge
Lightweight PUF based Authentication Protocol [64]	APUF	Encryption	Ciphertext comparison
Set-based Obfuscation Protocol [65]	APUF	Random Set based Obfuscation	Bitwise comparison
Slender PUF Protocol [66]	XOR APUF	Substring Matching	Substring response match for predefined threshold time
Noise Bifurcation Protocol [67]	XOR APUF	Noise Bifurcation	Bitwise comparison
Lockdown Technique [68]	XOR APUF	Bypass mechanism	FHD of the prover to be within threshold
Helper Data Algorithm [69]	BST PUF	Reliability Flag	Message authentication code and Pseudo Random Function

*Not specified in the paper

improve the resilience between the server and tag and vice versa while authenticating RFID devices. The PUF has been embedded in the RFID tag and provides the seed as a password to the random number generator to generate challenges. During authentication, the responses get verified by a string-matching method.

In [61], a lightweight PUF-based authentication protocol utilizes APUF for authentication between the prover and verifier as shown in Fig. 11. The authentication protocol considers that the prover and the verifier possess the PUF circuit model to produce a response to the unexpected challenge. The steps are as follows,

- Exchange of PUF IDs and initialization messages
- Verifier sends a pseudo-challenge C1 to the prover
- Prover transforms C1 to dynamically generate pseudo-challenges C2 and C3 based on the transformed response (R_V)
- Creation of unique secret patterns (g)
- Verifier searches for the matches of the secret pattern received with any secret patterns specific to the particular device

The communication protocol in [62] has three phases, viz., enrollment, authentication, and update, to authorize the prover device as shown in Fig. 12. It is illustrated as follows:

- *Enrollment:*Trusted party delivers an initial message to the server and the authenticating device
- FSM in the authentication server model responds to the message
- Quadruple challenge is issued to the APUF and the majority voter to generate a reliable 8-bit response
- APUF model is built using a tripartite classification algorithm
- *Authentication:*Device and the server communicate with each other to confirm authorization
- Server acknowledges the device request by providing a random number (R_N)
- Random number produces a quadruple response in the device using the PUF model (R_F)
- The server generates the response (R_M) for the same R_N using a software PUF model
- Device is authorized only if both responses are equal
- *Update:*After a fixed number of successful authentication attempts, the relevant parameters are periodically updated from memory during the update phase.

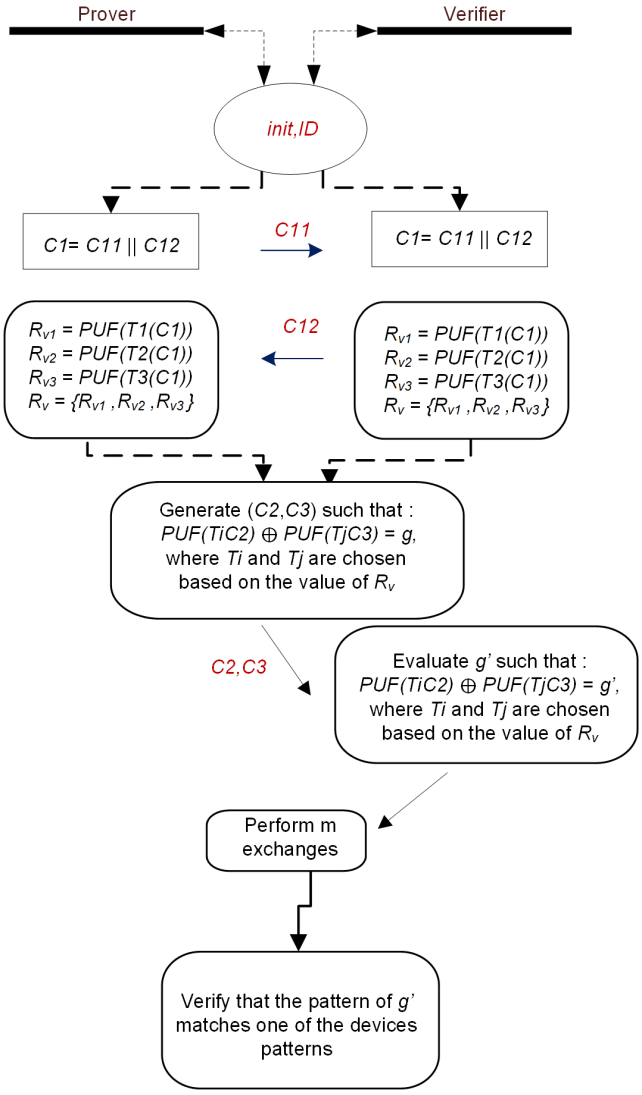

FIGURE 11. Authentication protocol [61].

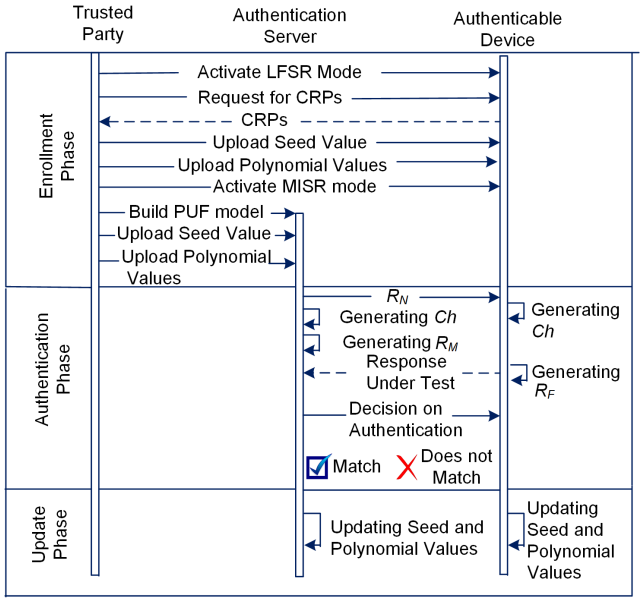

FIGURE 12. Authentication protocol [62].

Lockdown protocol of [68] deploys XOR-APUF to secure authentication for lightweight applications on the server side. BST-PUF (Bit-Self-Test PUF) uses the reliability flag value as helper data during the registration and recovery phase without additional logic [69]. It embeds a delay module to set the threshold value based on the delay difference and raises a reliability flag to determine a reliable response. PUF-FSM (Finite State Machine) [70], [71] proposes a method that realizes CPUF without helper data and error correction code. The responses from the APUF have been combined with FSM and a random number generator. This approach is claimed robust by the author against reliability-based attacks and the most suitable for authentication. The other strategy for authentication is by using the remote configuration of APUF on FPGA [72], which means that the PUF is not physically available before authentication.

The cryptographic applications and the protocols rely on the random numbers generated by the Random Number Generator (RNG). Securing the unpredictable random number is the most challenging part of the cryptographic process. The random numbers used as private keys, public keys, nonce, and block padding are generated using a True Random Number Generator (TRNG) and a Pseudo-Random Number Generator (PRNG). The malicious adversaries can impersonate the random number generated through various invasive and non-invasive attacks. The quality of random numbers is tested using the statistical test, National Institute of Standards and Technology (NIST) test-suite [73]. NIST has 15 tests to evaluate the randomness of the bit string. Any RNG must pass the NIST 800-22 test for commercial acceptance, though the NIST does not provide any suggestions for designing the RNG.

Random number generation using PUF has also gained more importance due to its unpredictability and its robustness against malicious attacks in producing random numbers [74], [75], [76], [77], [78], [79]. As discussed in the previous section, the basic APUF produces random outputs, which depend on timing and delay information. The responses thus achieved are easily predicted using machine learning (ML) algorithms due to their increased noise level content. Hence, the basic APUF successfully passes only a part of the NIST test. Various techniques are adopted to generate a random number to pass the NIST test. The APUF-based TRNG [74] uses a single incoming request to generate a random output. The PUF output is produced after several iterative processes to identify a meta-stable challenge that gives unpredictable results. The PUF output corresponding to the meta-stable challenge goes through a post-processing procedure using a Von Neumann corrector to extract randomness.

In [75], the author presents a TRNG by introducing non-linearity using APUF in the non-linear feedback shift register. The stability of the responses are improved by comparing the signals from the last switch block using two

cross-coupled arbiters. The output from TRNG is considered valid only if the two arbiters acknowledge the output. RNG undergoes extensive tests when designed, after production, and in-field for validation. Statistical tests, online health tests, and known answer tests (KATs) are general tests performed to verify the stability of RNGs. In [76], KAT is used to test the TRNG designed using APUF. KAT is performed in-field during power-ups. For a known input, the output from the TRNG is validated only if the output for the given input is the same, or KAT will not be detected.

IV. VARIANTS OF ARBITER PUF

In this section, different variants of APUF are discussed in terms of their performance, the robustness of the design, and their resilient nature against adversary attacks.

A. NON-LINEARITY AND RECONFIGURABILITY IN AN APUF MODEL

1) FEED-FORWARD APUF (FF-APUF)

The linear delay model of APUF has been used to design a software-cloned APUF. The adversaries were able to build the delay models to determine the CRPs by reverse engineering. In the Feed-Forward APUF (FF-APUF) [80], the challenges to the intermediate stages of the switch blocks originate from the feed-forward arbiter itself instead of the user issuing them. Note that it is essential for the challenges to reach the corresponding switch component before the delay signals reach. Non-compliance can lead to bias in the final output. The addition of the feed-forward arbiter for obfuscating the challenges reduces the reliability and makes it resilient to modeling attacks.

Introducing non-linearity in FF-APUF does not result in achieving a unique identity. With heterogeneous types of FF-APUF and XOR in [81], significant improvements in uniqueness and enhanced resilient property to adversary attack have been proven. In the homogeneous approach, the FF-arbiters are placed identically after a particular stage in each APUF instance, while the arbiters are placed arbitrarily in the heterogeneous approach. The authors claim that placing the FF-arbiter close to the last switch block degrades its performance. For security analysis, FF-XOR PUF has been simulated as a black box using ANN (Artificial Neural Network), and the results demonstrate that these models are proved resilient when the number of FF-APUF instances in the design is greater than 5, with a prediction accuracy of 50%. Different security analyses have been carried out using Logistic Regression (LR), ANN, and Covariance Matrix Adaptation-Evolution Strategy (CMA-ES) for more than 100 x 10^3 to a 1 million training sets, and authors conclude that the heterogeneous FF-XOR PUF is more secure than the standard XOR PUFs.

2) RECONFIGURABLE APUF

CRPs are reconfigured by different approaches, such as adding reconfigurable LFSR (linear feedback shift register) [57], hash functions [59], and obfuscating the responses [82]. Authors in [82] propose reconfigurable PUFs to increase the reliability and security of the PUF performance during authentication without altering its properties by 1) reconfiguring CRPs without configuring PUF and 2) reconfiguring the PUF, thereby making CRPs reconfigurable. Reference [83] secures against the ML attack on APUF by applying partial reconfiguration and the tribes function.

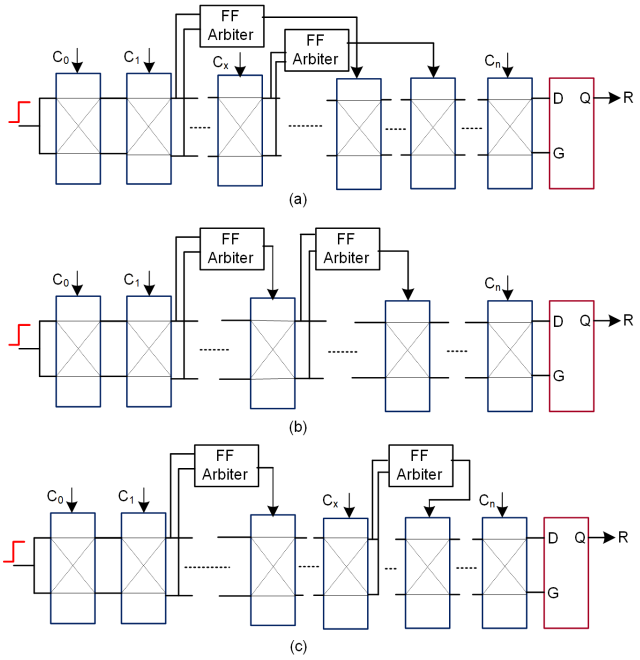

FIGURE 13. Feed-Forward MUX PUF (a) overlap (b) cascade and (c) separate structure [82].

Reconfiguring the FF-APUF overcomes the problem of degraded reliability. The Feed-Forward structure [80] does not specify the feed-forward point in the design. Fig. 13 depicts the three reconfigurable feed-forward MUX PUF structures [82], viz., 1) Feed-Forward Overlap (FFO), where the feed-forward arbiter is added after every five to eight stages so that the challenges from the feed-forward reach before the delay signal traverses to it and with one stage overlap between every two feed-forward paths, 2) Feed-Forward Cascade (FFC) where the final stage of one feed-forward arbiter acts as the starting stage of another feed-forward path, and 3) Feed-Forward Separate (FFS), with no overlap or cascade of feed-forward paths.

The Modified Feed-Forward MUX PUF (MFF MUX PUF) [84] architecture has been proposed in continuation with his previous work [82] by Lao and Parhi In this PUF design, the feed-forward arbiter output was given as a challenge to two successive MUX stages with feed-forward arbiter connected in overlap, cascade, and separate structures (MFFO, MFFC, MFFS). With no improvement in security, the reliability has been enhanced in the modified design compared to the feed-forward MUX structure.

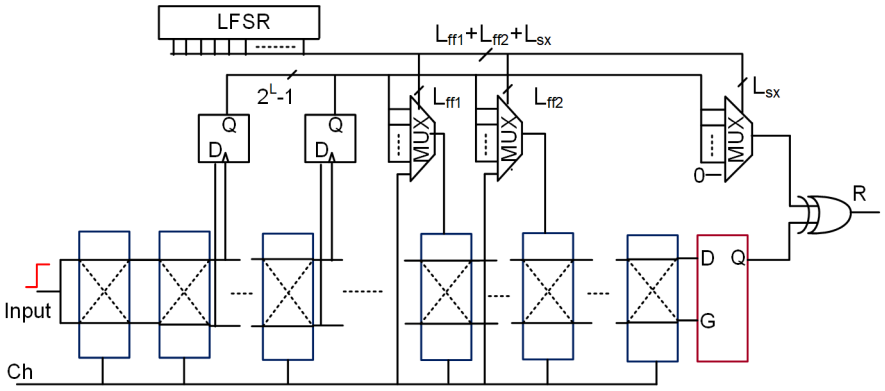

FIGURE 14. Dynamically Configurable Hybrid PUF [85].

In Dynamically Configurable Hybrid (DCH) PUF [85], LFSR has been used to configure the APUF to generate the response. As a result, different response bits are obtained depending on the configuration signals. The configurations can be enhanced based on the LFSR bit length, which raises the computational complexity of a modeling attack. The author of [85] has also proposed configurable self-XOR (DC SX-APUF) and configurable Feed-Forward (DC FF-APUF) structures. In the former, the intermediate signals from various switch blocks are fed to the feed-forward arbiter and then to the MUX. Based on the configuration signal of the MUX, the output of the MUX is XOR-ed to generate the output. This result is XOR-ed with the APUF output to determine the final 1-bit response. The resource utilized by this model is comparatively less compared to the n-XOR PUF. The DC FF-APUF is similar to FF-APUF with an additional MUX added at the output of the Feed-Forward with configuration signal. The DCH PUF shown in Fig. 14, combines these two designs with one branch of self-XOR and two Feed-Forward overlapped structures. For an n-XOR model, only one LFSR has been used to produce the configuration signal. Fig. 15 demonstrates that introducing non-linearity and reconfigurability to the basic APUF has improved the parameter, however, failing to satisfy the primary goal of improved inter-chip and intra-chip variations (ideal value 100%). The reason can be the lack of control over the placement and routing of parallel paths and feed-forward arbiter. In comparison, the DCH PUF model has shown a significant improvement with a uniqueness of 41% (ideal value = 50%), which can be studied further. However, it may be noted that experimental validation is realized only for a 7-bit response.

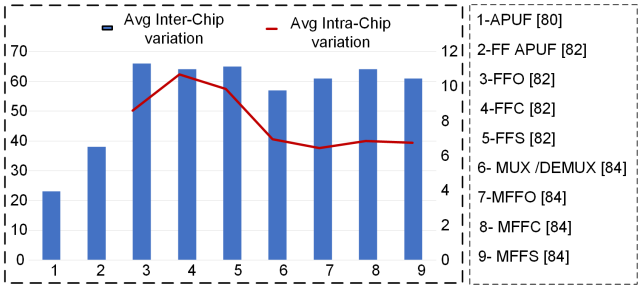

FIGURE 15. Inter-chip and Intra-chip variation of APUF and variants of FFAPUF.

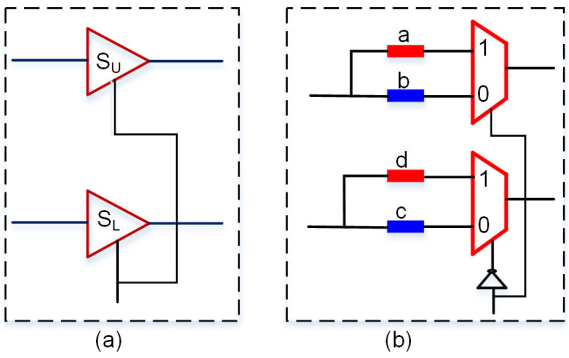

FIGURE 16. PDL based switch [36].

B. UTILIZING THE FPGA ARCHITECTURE TO ENHANCE SYMMETRICAL ROUTING

1) PDL LOGIC

To meet the symmetrical routing requirements, the MUX logic in the FPGA was replaced by PDL with a single LUT [36]. PDL was used to fine-tune the delay skews caused by irregularity in signal routing. To implement PDL, any *n*-input LUT can be configured in which the LUT inputs other than the first one were used as don't-care bits. The don't-care inputs will regulate the propagation of signals inside LUT. The output of LUT is the inverted value of the first LUT input. Instead of path-swapping structures, new non-swapping switch structures are designed using two PDLs (S_U and S_L) as shown in Fig. 16.a. Fig. 16.b shows the equivalent circuit of the non-swapping switch structure with the nominal delay values of 'a' and 'd' and the crosswise paths of 'b' and 'c' are identical.

The structure of Programmable delay lines based APUF (PAPUF) [36] realized using PDL consists of N switch components and K tuning blocks. The PDL was fine-tuned using

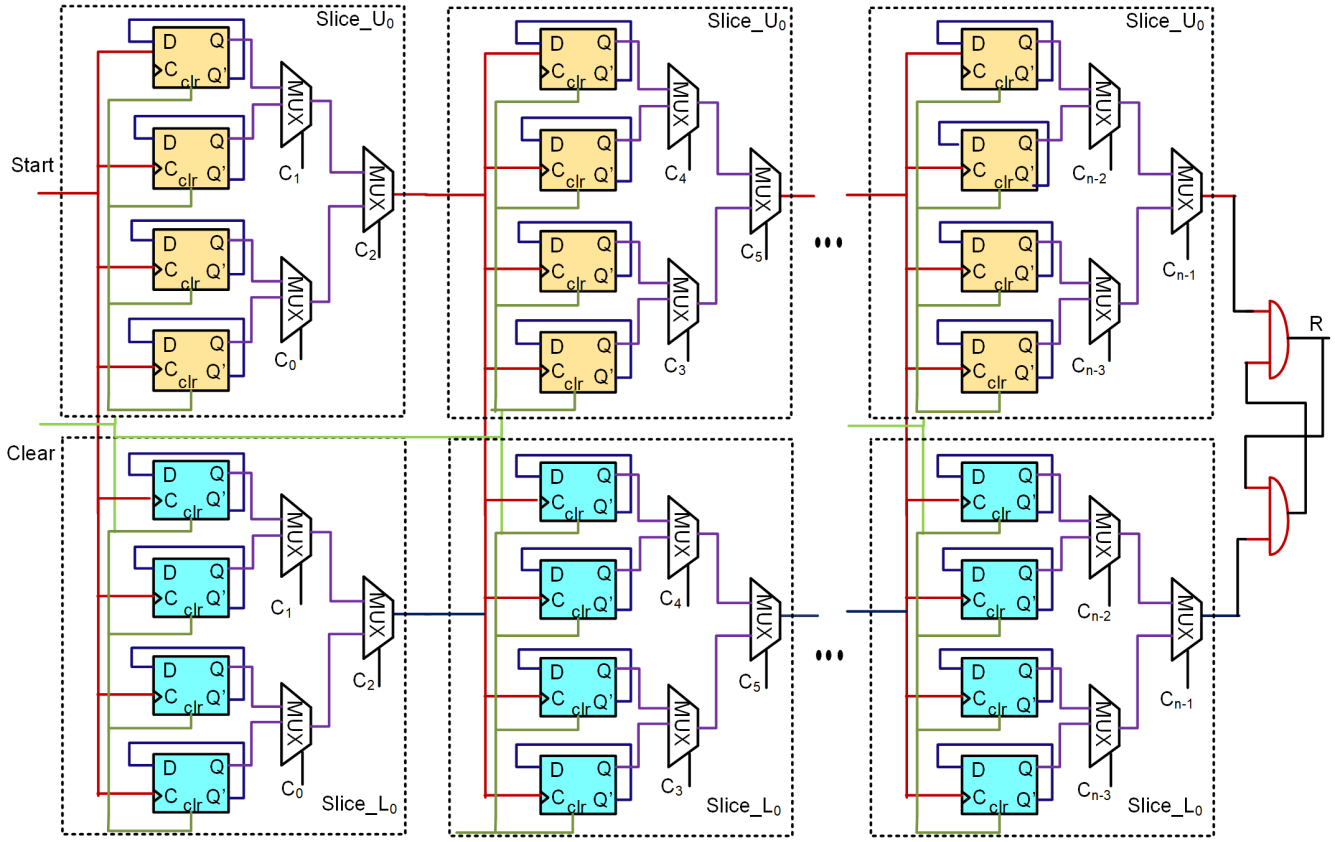

FIGURE 17. Flip-Flop APUF [86].

the delay characterization technique to obtain a maximum delay for different input patterns. The tuning blocks are free to be located at any place in the structure of PUF. Depending on their selector inputs, these tuning blocks can insert additional delays in the top or bottom paths. The significant difference between the switches and the tuning blocks lies in the connection of selector lines. The selector lines are standard for upper and lower paths, while in tuning blocks, the upper and lower paths have different selector lines. The top and bottom paths and the arbiter have been symmetrically routed. The tuning blocks also eliminate the bias in the delay. The design is evaluated for the same challenges, and the responses have been obtained after majority voting to improve reliability. It enhances the response robustness and alleviates the meta-stability problem of the arbiter. The author has also arrived at a new hypothesis that the responses were robust when the delay difference is found to be more significant at the arbiter input.

2) FLIP-FLOP BASED APUF DESIGN

Flip-Flop APUF proposed by Gu *et al.* inserts additional delay and symmetrical routing to increase the uniqueness with four flip-flops and three MUXes in place of a single MUX structure in APUF [86], [87]. The design reduces resource utilization significantly since the number of challenges is increased three times compared to the conventional APUF. The cross-coupled NAND arbiter has been chosen over the D-FF since it contributes a 10% skew to the routing path. Thus, Flip-Flop APUF shown in Fig. 17 uses only 44 slices to generate a 1-bit response compared to 129 slices for a 64-bit challenge for an APUF design by saving 66% of hardware resources.

The flip-flops are initially reset by CLEAR and then enabled by the rising edge of the clock signal using START. At the end of the three MUXes for every slice, the output will select any of the four FFs to determine the delay path corresponding to the given challenge. The output from each upper or lower path (TU or TL) of the PUF cell becomes an input to the clock port of the next PUF cell. It continues till the last stage, and its output is fed to the arbiter with the response bit generated based on the racing signal entering the arbiter. The design [80] has also been evaluated for min-entropy, conditional Shannon entropy, and conditional min-entropy, as found to be 0.54, 0.90, and 0.61, respectively.

Additional delay incorporated into the Flip-Flop APUF in FOXFF-APUF (Feedback Oriented XORed FF-APUF) slightly improves the uniqueness of an 8-bit challenge [88]. This design introduces a delay before and after the third MUX using a D-FF with feedback. Two configurations have been proposed, the first with three challenges as input and the second with only one challenge as input. The author verifies the uniqueness in Spartan-3 and Virtex-6 FPGA boards for

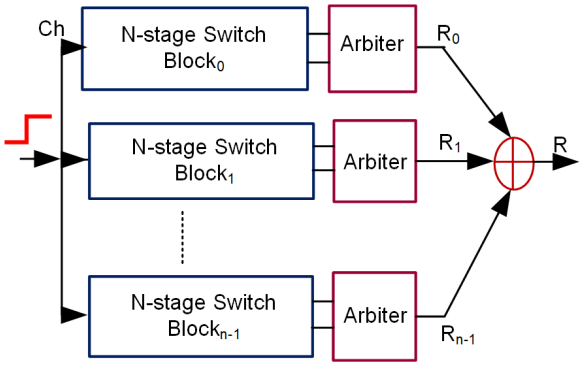

FIGURE 18. XOR APUF [52].

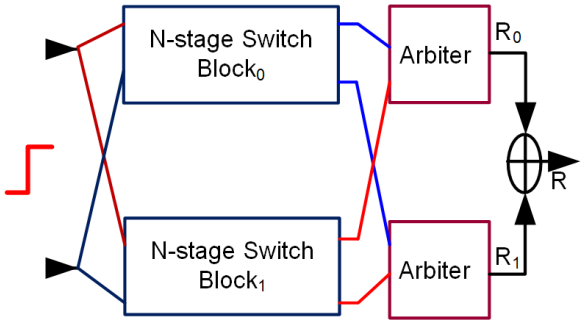

FIGURE 19. DAPUF [33].

an 8-bit challenge, concluding that it was better than the Flip-Flop APUF. The above results imply that the Flip-Flop APUF and FOXFF-APUF have shown better results than the conventional APUF design. This design requires fewer slices than APUF, which can reinforce the lightweight design focus. Despite the trade-off for uniqueness and resource utilization, the inclusion of tuning blocks and symmetrical routing in PAPUF realized enhanced uniqueness for 90 nm compared to 28 nm FPGA, though with increased resource utilization than Flip-Flop APUF.

C. ROBUSTNESS TO MODELING ATTACKS

The primary consideration while designing a new variant of APUF would be the symmetric layout, augmenting the impact of intrinsic variations to improve the uniqueness [89], thus increasing the reliability against temperature sensitivity [90].

1) REPLICATING APUF INSTANCES

Cryptography exploits the XOR function to generate a random number with an equal probability of 0 and 1. In that direction, to increase the randomness of APUF significantly, n-APUF instances are connected in parallel and their outputs are XOR-ed to produce a 1-bit response (XOR APUF) as depicted in Fig. 18 [52]. The embedding of XOR logic also obscures the outcome of each APUF instance. As n is increased, the computational time required to attack the XOR APUF increases. The addition of XOR, however, introduces non-linearity, at the cost of reduced reliability since it involves n APUF instances.

Double APUF (DAPUF) shown in Fig. 19 incurs less bias than the basic APUF, and hence the uniqueness is comparatively better than the conventional APUF [33], [34], [35]. The SB for the n-bit challenge has been duplicated for DAPUF and a hardcore MUX primitive has been used in the design. The significant difference between the XOR APUF [52] and the DAPUF is the cross-connections from the switch blocks to the arbiter and the response generated. In DAPUF, the input signal, and the output of the last switch block entering the arbiter are cross-coupled. The outputs from the arbiters are XOR-ed to produce the final response as depicted in Fig. 19. Similarly, the 3-1 DAPUF is designed using three switch blocks and 6 arbiters, and the 6-bit output generated by the SR latch is XOR-ed to generate the final 1-bit response. The author has compared the DAPUF design with APUF, which has the same set of responses being generated. The uniqueness results are thus improved for 2-1 DAPUF and 3-1 DAPUF compared to 2-1 and 3-1 APUF by 41% and 50%, respectively. However, the performance of 2-2 DAPUF was found to be inferior to 2-2 APUF. At the same time, Component-differentially-challenged XPUF (CDC-XPUF) employs different challenges for each APUF instance [91].

Due to the improved uniqueness of the DAPUF model, various APUF variants have subsequently been designed on its premise. The Dual APUF discards the unreliable response to improve the reliability by incorporating two additional paths to determine the delay difference and filter out the inconsistent responses with lesser delay difference [92]. In Multi-Block APUF (MB APUF), n-stage APUF has been divided into two or four blocks with different voltages for each block to achieve enhanced entropy without increasing area and power consumption, compared to DAPUF [22]. In Two-Stage Multiple-Choice Arbiter PUF (TSMCA PUF) shown in Fig. 20, the reconfigurability has been achieved by connecting the switch block and the arbiter with a 5-1 MUX [2]. The select lines of the MUX receive a part of the challenges. The output from the MUX undergoes XORing and character padding operations to produce the response. The MA-PUF (Mixed Arbiter-PUF) generates a response by integrating an additional two 4-1 MUX with four switch blocks between the switch block and the arbiter to reduce the resilience against modeling attacks [93].

The drawback of Yilmaz et al. [64] proposal is that it cannot be utilized to authenticate a time-critical system, due to the fact that along with the selected APUF configuration file, the challenges to FPGA also are needed to be sent. Furthermore, the CRPs have been generated from a single APUF, so that the adversary can quickly rebuild its tentative model with known CRPs. Sahoo et al. have presented a Multiplexer-based APUF (MPUF) of Fig. 21.a, in which the output of primitive APUFs is given as an input to MUX to achieve the final response [94]. Implementation of MPUF increases area utilization. However, this also enhances the computational complexity, necessitating each APUF instance

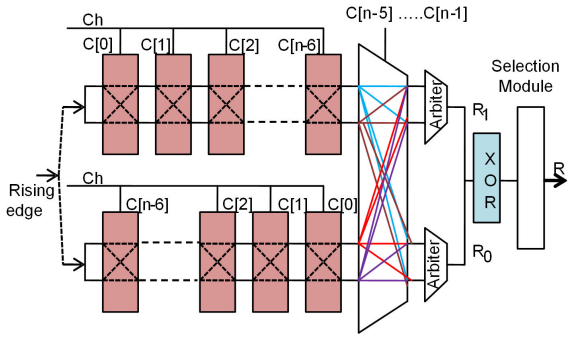

FIGURE 20. TSMCA PUF [2].

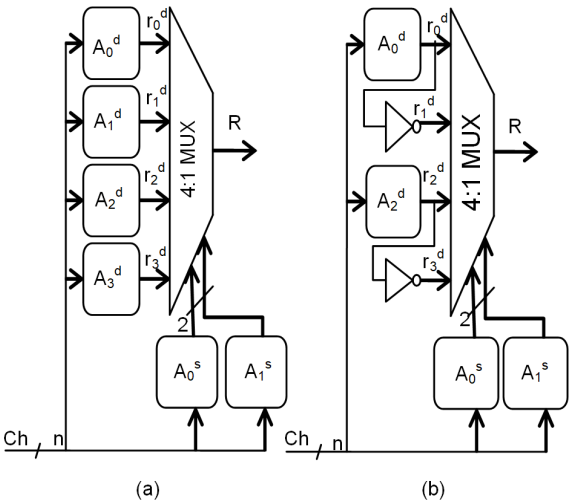

FIGURE 21. (a) (n, 2)-MPUF [94](b) (n, 2)-cMPUF [94].

to be extensively studied by the adversary while modeling the design. To make the MPUF robust against reliability-based modeling attacks, a new variant, rMPUF has also been presented [95]. In rMPUF, the selection inputs generated by independent APUFs are given to 2:1 MUXes. The selection input for APUFs is modeled sequentially along the path from the input to the response. The rMPUF has good reliability and security properties with a trade-off in the cost of hardware overhead due to the large increase in the number of APUFs. Another variant of basic MPUF is the cMPUF depicted in Fig. 21.b designed to be resistant to linear cryptanalysis [96]. To avoid the linear approximation between the response and data inputs connection pattern, half of the data inputs of MUX have been complimented. It aims to introduce 50% noise in training CRPs. Thus, it becomes hard to model the cMPUF by the adversaries. Reference [94], has all the theoretical findings validated using MATLAB, without any implementation in the FPGA board for the design verification.

The divide and conquer approach used in reliability-based modeling has successfully modeled the XOR APUF. The challenges of the XOR APUF are obtained from the responses of another XOR APUF to protect against the above-said attack in Multiple XOR PUF (MXPUF) [97] and iPUF [37].

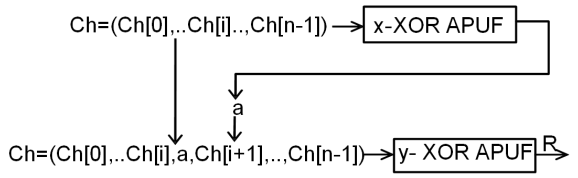

FIGURE 22. MXPUF [97].

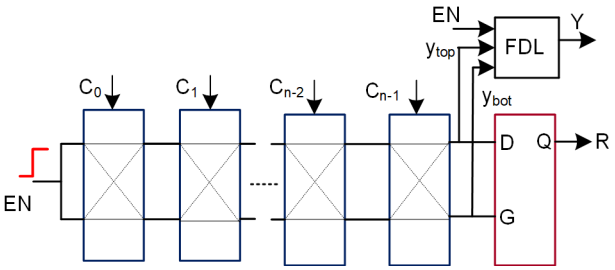

FIGURE 23. APUF with FDL [98].

It has two layers, identified as the upper and lower layers, as shown in Fig. 22, and the PUF is intended to be more resilient to measurement noise due to the feedback. The upper and lower layers comprise an n-bit x-XOR APUF and $(n+1)$-bit y-XOR APUF, respectively. For a given n-bit challenge Ch, the responses generated by the upper layer have been interleaved in the middle of n-bit challenge Ch to construct a new lower layer that generates the final response bit. The reliability of (x, y)-MXPUF was found to be twice that of the $(x + y)$-XOR APUF and less secured against Becker's attack.

A Fault Detection Logic (FDL) proposed in [98] helps detect the run-time alteration caused by fault injection in the PUF instances as depicted in Fig. 23. Fault injection can be stuck-at 0 faults in the arbiter (clock input of D-FF), which an attacker can add to the last switch block. FDL has three inputs, viz., Enable (EN), y_{top}, y_{bot}, and output Y. If the value of Y is 1, then the response obtained from that particular APUF instance indicates the absence of fault, or else, the presence of a fault. Adding FDL to the APUF instances makes it hard to introduce a fault in determining the CRPs by the adversaries. The APUFs are combined parallelly to an XOR APUF to increase the number of challenge bits, thereby increasing the uniqueness and the data complexity with reduced reliability. The reliability of the design is reduced due to the addition of parallel instances of APUF.

2) PRE-PROCESSING THE CHALLENGES AND OBFUSCATING THE RESPONSES

- *Obfuscating both challenge and response:* The Lightweight Secure PUF (LSPUF) illustrated in Fig. 24 transforms the challenges and the responses obtained from the APUFs [58], [99]. The challenges to the APUF embedded in LSPUF are fed using XOR logic to satisfy the Strict Avalanche Criterion (SAC). According to SAC, even if one-bit flips in the challenge, each response

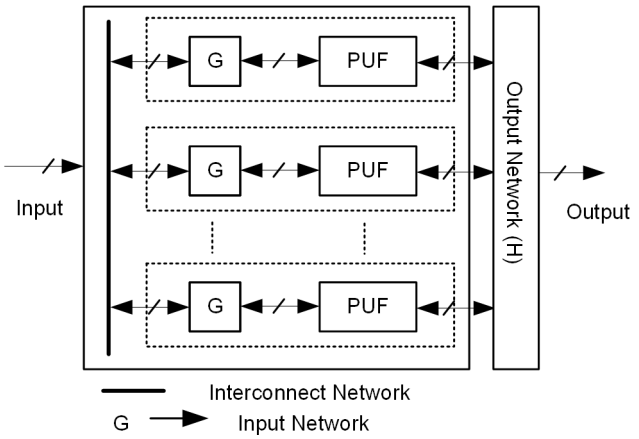

FIGURE 24. LSPUF [99].

bit ought to flip, with a 50% probability. The challenge bits for multiple APUFs have been given in a circular shift using an interconnect to satisfy SAC. The responses from several rows of APUF are combined using XOR logic. These aim to improve the randomness of the responses and fortify them against reverse engineering. However, the XOR logic for obfuscating necessitates more resources to cater to the number of switch blocks and the responses.

PUF presented as Random Set-based Obfuscation (RSO) utilizes the stable PUF responses obtained from the PUF and pre-stores them as a set for obfuscation in [65]. Initially, a few stable CRPs of the PUFs are collected and stored in NVM using DMA (Direct Memory Access) from the testing phase. In the second stage, the stable challenges stored are applied as input to the PUF circuit. The responses from the PUF get temporarily stored in registers that would later be utilized as a subset for obfuscation, followed by a TRNG, which selects two keys to obfuscate challenges and responses with XOR operations. RSO-based APUF has been proven to be resilient against machine learning attacks. Even if the adversary successfully collects 1 million CRPs, the threshold limit set in this method automatically updates the CRPs. Finally, the obfuscated response is used for authentication in the fourth stage. The prediction accuracy for 64 × 64 APUFs with a set size of 32 and 1 million CRPs demonstrates that an adversary can manage with only a random guess.

- *Obfuscating the challenges:* Non-linear masking of original challenges has been proposed with two-party authentication using a C-REG in [100] and Multiple Input Signature Registers (MISR) in [101]. In the former, the challenges have been classified as weak and strong, based on their vulnerability to influence a response bit to flip without adding any additional hardware cost. 128-bit weak challenges are initially generated using a pseudorandom sequence generator in the host machine, which is then padded with a leading and trailing binary bit to convert to 130-bit challenges. These challenges are non-linearly processed using a register (C-REG) with N number of T Flip-Flops (T FF) thus producing a robust challenge set. Randomized Arbiter PUFs (R-APUFs) obfuscate the CRPs by exploiting the randomizing property using ''Shamir's secret key'' algorithm to reconstruct the polynomials, thus resulting in increased robustness [53].

The tripartite classification algorithm [62] has been used to build an accurate APUF model to segregate the CRPs. The CRPs segregate into five classes: unstable, meta-stable zero, meta-stable one, stable zero, and stable one. The built-in logic observer (BILBO) module breaks the linear dependency between the challenge and response mapping. The input challenge and the quadruple response have been obfuscated in this way. After ascertaining the response stability by repeating each quadruple challenge, the data are fed into the Deep Neural Network (DNN) for training. Then, the DNN is trained to model two different path lengths of APUF, N=24 for short APUF and N=128 for standard APUF. The APUF model at the authentication server does not require helper data or raw CRPs to be stored externally. Hence, it significantly increases its resistance against modeling attacks. In Obfuscated Challenge APUF (OC-APUF) [103], the server sends the challenges to a challenge obfuscation module before authentication. The obfuscated challenges will be sent to the OC-APUF, where the original challenges are retrieved with the help of a recovery module, and the corresponding responses get generated. Similarly, in the Challenge Pre-Processing APUF structure (CPP-APUF) [104], a CPP structure with a 4-input modified RS flip-flop is found embedded to mask the original challenge.

The Current-Starved Inverter APUF (CSI-APUF) [102] has been proposed with NMOS-based CSI, biased at zero temperature coefficient points. As a result, temperature-induced response bit flip will significantly decrease, thus improving reliability characteristics. The current required to charge and discharge the CSI is considerably smaller than the basic inverter. Therefore, the manufacturing variations have been more pronounced in the CSI, increasing the delay difference between the upper and lower paths. The 64-bit Fibonacci LFSR provides the necessary input challenges to the APUF. The D latch has been replaced with a pair of RS-latch, with an AND gate connected to an active low enable input of the second stage of RS-latch to act as an arbiter as shown in Fig.25 to reduce the systematic bias. It also helps in reading a stable response bit from APUF.

The Dual LFSR PUF [57] combines a Randomness Adjustment Module to generate a masked challenge as an input. This module enhances the randomness and reliability by counting the number of zeros from the arbiter. The counting is initialized using the RST key on FPGA.

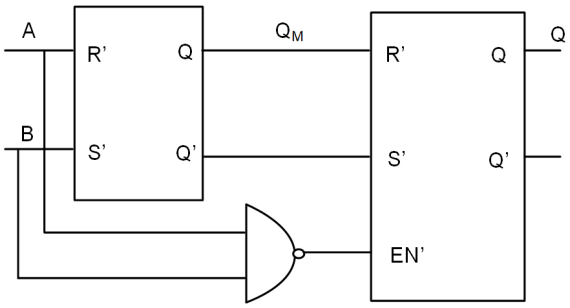

FIGURE 25. Arbiter in CSI-APUF [102].

If the number of zeros is not close to 50%, the delay is included in the path to make it symmetric using the compensation circuit module. The reliability of APUF gets strengthened by directing the output of the arbiter to the voter and the XOR module, followed by extraction of the response bit. Two Galois LFSRs are added to obfuscate the challenge in a time-variant approach to make the PUF more robust during authentication against man-in-the-middle attacks. Both the LFSRs depend on the response from the APUF. The challenge to the APUF is from the first LFSR when the response is 0, or else, from the second LFSR. Moreover, the author does not discuss the security analysis of the design.

Multi-PUF [105] and MMPUF (MUX-based Multi-PUF) [106] designs propose to obscure the challenge bits with a weak PUF design. Both use Pico PUF to obfuscate the challenges of the strong APUF [108]. They differ in the way Pico PUF gets connected to the APUF. In Multi-PUF, the original challenge and the output from weak PUF are XOR-ed, and the masked challenges are provided to APUF. On the other hand, in MMPUF, the MUX receives a response from two Pico PUFs, and the original challenges are fed to the selected signals. The probability of 0 and 1 is comparatively higher in MMPUF than in Multi-PUF. Even though the uniqueness has been significantly improved in both designs compared to classical APUF, it does not fit precisely in the normal distribution curve. While discussing the security, it is shown that as the challenge bit length is increased, the prediction rate for Multi-PUF also increases, whereas, for MMPUF, it is less than 59%. Similarly, MMPUF prediction accuracy ranges from 52% to 74% when Support Vector Machine (SVM) and CMA-ES are used. However, the prediction accuracy increases as the CRP size increases (64 and 128-bit).

Fig. 26 illustrates that masking the challenges and responses in RSO-PUF, MISR-APUF, and CPP-APUF has significantly improved the uniqueness by almost near to 49%, while CSI-APUF, Multi-PUF, MMPUF, and CMPUF have uniqueness greater than 40%. The obfuscation technique increases the complexity to the adversary since the input or output dataset required for training the algorithm is lost. Hence, LR analysis for the above-said design has a reduced

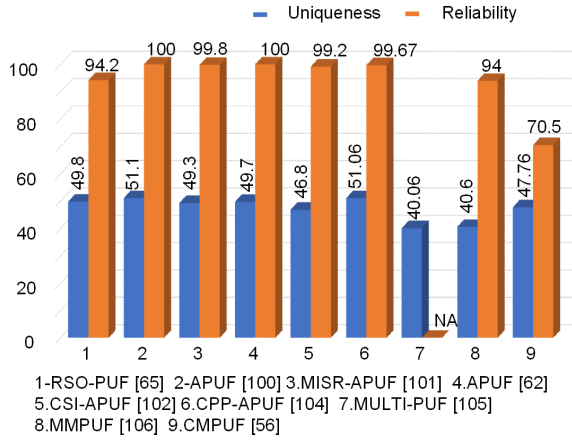

FIGURE 26. Performance parameters of the variants of APUF after pre-processing the challenges and obfuscating the responses. The values that are not defined are shown as NA.

prediction accuracy of <65% due to the use of an obfuscation technique rather than classical APUF, which had a prediction accuracy of 100%.

D. COMBINING BASIC PUF MODULES AS SMALL BUILDING BLOCKS

Yet another prototype in the design of PUFs is the Composite PUF [109], [110], which employs smaller PUFs as design blocks. Composite PUFs have larger CRPs and reduced resource overhead with improved performance than the constituent PUF models generating the same CRPs. The authors in [16] provide a theoretical framework to choose the best possible PUF composition of RO PUF, and APUF to improve PUF parameters significantly. The delay lines of APUF are replaced with RO PUF in S-ArbRO PUF to enhance the number of CRPs [111]. The Mem-APUF [112] is a digitized version of the APUF that resembles the behavior of the APUF realized by integrating a weak PUF circuit with fixed-point arithmetic. Splitting attack on iPUF [114] has motivated the design of an LP-PUF (composite PUF) [113], which combines an FF-APUF and iPUF compatible with CMOS design.

Table 4 summarizes Section IV based on the technology design environment employed, performance parameters and methods adopted to improve PUF characteristics, and security analysis conducted in the respective literature. The conventional APUF designed with a linear additive delay model suffers from poor uniqueness. The explanation is that the lack of symmetrical routing results in reduced latency between two paths. Adding many instances of APUF and XOR functions to the basic APUF design model has reduced the bit-biasing towards 1 or 0. Thereby, the unique ID of each device is enhanced with increased uniqueness and reliability in DAPUF, XOR APUF, TSMCA PUF, and MPUF. Further, obfuscation of the challenge and response of an APUF improves the uniqueness near the ideal value of 50% by making it more difficult for an attacker to predict, or reproduce the

outputs of the APUF based on knowledge of the challenges and responses.

V. MODELING ATTACK

Software modeling builds a numerical model of PUF using a computer algorithm with knowledge of datasets. In non-invasive attacks of APUF, presume that Bob, an adversary has obtained a few sets of CRPs and the adversary can design a numerical model with delay value and CRPs. In that case, developing the PUF model is possible, which is subjective to ML attacks.

Mathematical modeling of APUF, XOR APUF, and LSPUF [115] have been discussed in brief under this section.

A. MATHEMATICAL MODELING OF APUF

The linear delay additive model is used to design the APUF. The final delay difference (Δ) between the top and bottom paths for an n-bit APUF can be expressed as,

$$\Delta = \vec{w}^T \vec{\phi}. \tag{20}$$

The parameter vector $\vec{w}$ and feature vector $\vec{\phi}$ are of $n+1$ dimensions. The vector $\vec{w}$ includes the delays from all the multiplexer components in the APUF stages. The feature vector $\vec{\phi}$ is a function of the input challenges. The response (R) of APUF is dependent on the sign of Δ. The response $R = -1$ when the APUF output is 0 or else, $R = 1$.

$$R = sgn(\Delta) = sgn(\vec{w}^T \vec{\phi}). \tag{21}$$

B. MATHEMATICAL MODELING OF XOR APUF

The XOR APUF is modeled with m APUFs in parallel, each consisting of n stages. A set of similar challenges are applied to all the APUF instances, and their responses are XOR-ed to generate a final response. Mathematically, the response from each instance can be given as $R_i \ \epsilon \{-1, 1\}$. Hence, for an m-XOR APUF, the final response becomes $R_{XOR} = \prod_{i=1}^{m} R_i$. The mathematical model with the parameters and feature vector of m-XOR APUF is written as,

$$R_{XOR} = \prod_{i=1}^{m} sgn(\vec{w}_l^T \vec{\phi}_l). \tag{22}$$

C. MATHEMATICAL MODELING OF LSPUF

The LSPUF is also similar to m-XOR APUF, with the difference in introducing the challenges. As mentioned in Section IV, the challenges to the LSPUF are delivered in a circular shift, and the output bits are XOR-ed to generate a multi-bit output response denoted as $out_1, out_2, .., out_j$. This approach can challenge the adversaries in extracting the exact single-bit output.

The design parameters for LSPUF are (i) the number of output bits (r), (ii) the values that affect the single output bit response (l), and (iii) the circular shift in choosing l values. For $n = 1, 2, \ldots, j$,

$$out_n = \oplus_{i=1,2,\ldots l} resp_{(n+cs+i) mod\ m}. \tag{23}$$

In the last few decades, adaptations of the XOR function in the design of APUF compositions have become inevitable for enhanced security due to the increased computational complexity. Reference [115] has experimentally verified that an exponential increase of XOR function enhances the rigidity of XOR PUFs against attack, however, trading-off reliability for the resource. Interestingly, Tobisch and Becker [46], [95] invalidate this claim by performing a cloning attack on a commercially available RFID tag with a 4-way XOR PUF model using divide and conquer approach in the Reliability-based CMA-ES and LR with R-prop algorithm. Reference [116] finds ML attacks still successful while using obfuscated response bits by extracting the necessary information from padded strings and helper data. CMA-ES has been used to attack Slender PUF and reverse fuzzy extractor-based protocols that use PUF to create secret keys in [116]. It emphasizes the need for protocols to be robust enough. Recent research focuses on using Deep Learning (DL) [117], [118], genetic programming [119], and ANN [120], [121] based techniques to model the design with reduced error and improved prediction accuracy of 64-bit and 128-bit CRPs.

D. MODELING ATTACK ON APUF

APUF remains the focus of most modeling attacks since it is the primary building block for analyzing attacks on APUF variants. Non-invasive, semi-invasive, and mathematical modeling attacks are used to extract the CRPs from APUF.

1) SIDE-CHANNEL ATTACK

Side-channel attack (SCA) is one of the most prominent non-invasive attacks used to determine the secret key from a PUF model [122], [123]. For every input and output in the PUF design, FPGA implementation employs a register. These registers serve as the data source for the power trace analysis. SCA depends on the architecture of PUF as well as the pseudo-random functions. In [123], using correlation power analysis, SCA is performed on LSPUF to collect the CRPs. Two approaches were focused on the register that stores the result of the XOR operation. The first method recovers the hamming weight of the targeted register, and the second obtains the difference of mean-based Differential Power Analysis (DPA).

2) PHOTONIC EMISSION AND LASER-BASED TECHNIQUES

These techniques collect the CRPs from the FPGA using photonic emission [124], and fault injection [125] from individual APUF instances. An intrusive attack on the APUF [126] successfully detects the CRPs with a high-resolution time of 6 ps and assists in determining the internal timing delay between each stage. This analysis also demonstrates that APUF can be modeled even without a minimal number of CRPs. This technique measures the time delay between the enabled signal and photon emission at the output of the final stage for each APUF occurrence. The authors of [126] experimentally evaluated their findings by applying the

TABLE 4. Compositions of Arbiter PUF.

Composition	Variants	Year	Technology Design Environment	Uniqueness (%)	Uniformity (%)	Reliability (%)	Method adopted to improve PUF Characteristics	Security Analysis Conducted
Variation in switch blocks	Tristate PUF [39]	2008	MATLAB	-[+]	-	-	Tristate	-
	Schmitt-trigger based APUF [41]	2015	SPICE	-	-	-	Schmitt-trigger	-
	APUF [42]	2021	Artix 7	15.15	∼0.45 -0.5	98.9714	MUX	-
	APUF [38]	2022	ZYNQ 7000	45.2	-	-	PDL and MUX	-
	Subthreshold voltage based APUF [44][45]	2010, 2012	SPICE	12.52 (25-bit)	-	-	NAND gate	SVM, SCA
	Adjustable APUF [43]	2012	Xilinx board	-	-	-	Adjustable module	-
Non-Linearity and Reconfigurability	Feed-Forward APUF [73]	2010	SPICE 180 nm	38	-	70	Feed-Forward arbiter	-
	Feed-Forward XOR PUF [81]	2020	Artix -7	-	-	-	Feed-Forward arbiter and XOR Function	LR, CMA-ES
	Feed-Forward MUX PUF [82]	2011	SPICE	-	-	-	Reconfigurable Feed-Forward Arbiter	-
	Modified Feed-Forward MUX PUF [84]	2014	SPICE	-	-	-	Reconfigurable Feed-Forward MUX PUF	-
	DCH PUF [85]	2021	Artix -7	0.4	0.45-0.5	1.36 (BER)	Dynamically configurable	CMA-ES
	MXPUF,IPUF [97][37]	2017, 2019	MATLAB	-	-	-	XOR logic and masking the challenge	CMA-ES
Utilizing the FPGA architecture to enhance symmetrical routing	PAPUF [36]	2012	Virtex 5, Spartan 3	-	-	-	Programmable Line	-
	Flip-Flop APUF [87]	2016	Artix-7	41.53	-	97.10% t*, 93.90% v*	Flip-Flop MUX primitive in CLB	LR, CMA-ES
	FOXFF APUF [88]	2018	Spartan 3, Virtex 6	42, 44	-	-	Flip-Flop and MUX primitive in CLB	-
Replicating APUF instances	XORAPUF[52]	2007	Virtex-4	46.15	-	48%	XOR function	-
	2-1 DAPUF [35]	2014	Virtex-5	46.4	-	-	XOR and additional Arbiters	SVM^{light}
	3-1 DAPUF [35]	2014		50.2	-	-		
	MB APUF [22]	2018	ASIC	127.89 (256-bit)	-	83.18	Multiple blocks	-
	CDC-XPUF [91]	2021	Artix-7	17	-	97.5	XOR function	-
	Dual APUF [92]	2017	Artix -7	19.39	-	0.24 (BER)	Additional delay path	-
	MA-PUF [93]	2021	SPICE 65 nm	49.77	48.34	96	MUX	ANN
	TSMCA PUF [2]	2019	Virtex-II	49.6	-	97.4	MUX, reversal of challenges	-
	MPUF(64,4) [94]	2018	MATLAB	49.6	-	97.4	MUX logic	CMA-ES, cryptanalysis
	APUF [98]	2016	Artix -7	-	-	-	Fault detection logic	-
Obscuring the challenges	LSPUF [99]	2008	Virtex-5	-	-	-	XOR logic and circular shift	-
	RSO-PUF [65]	2021	Artix-7	49.8	-	94.2	Subset of responses	LR
	APUF [100]	2017	Artix-7	51.1	-	100	Pseudo Random Sequence Generator and T-FF	SVM
	MISR-APUF [101]	2017	Xilinx ZC706	49.3	-	99.8	Multiple Input Signature Register	SVM
	R-APUF [53]	2017	Virtex-5	-	-		Shamir's secret key algorithm	ES
	APUF [62]	2019	Artix-7	49.7	-	100	Tripartite algorithm	CMA-ES
	OC-APUF [103]	2019	-	-	-	-	Challenge Obfuscation module	LR
	CSI-APUF [102]	2019	Altera	46.8	-	0.8(BER)	Fibonacci LFSR	-

TABLE 4. *(Continued.)* **Compositions of Arbiter PUF.**

	CPP-APUF [104]	2020	Altera	51.06	50.18	99.67	Challenge Pre-processing module	Linear regression, LR, SVM and BPNN
	Dual LFSR PUF [57]	2020	-	-	-	-	Galois LFSR	-
	Multi-PUF [105]	2018	Artix-7	40.06	37.03	-	Weak PUF	LR, CMA-ES
	MMPUF [106]	2020	Artxi-7, Kintex-7	40.6	-	96%t*, 94%v*	Weak PUF	LR, SVM, CMA-ES (64-bit)
	APUF [107]	2019	-	-	-	-	Bent Function	-
	CMPUF [56]	2018	SPICE 65 nm	47.76	39.06	70.5	Current Mirror	ANN
Combining basic PUF as small building blocks	Composite PUF [109]	2014	Spartan-3	25-49	-	-	Combining APUF and RO PUF	-
	S-ArbRO PUF [111]	2013	Spartan-3	-	-	15.5 (45°C), 22.75 (65°C)	RO PUF as delay path	-
	Mem-APUF [112]	2020	SPICE 45 nm	0.4-0.44	-	100	SRAM and APUF	-
	LP-PUF [113]	2022	-	99 (Ideal 100%)	-	∼99	FF-APUF and iPUF	LR, CMA-ES

*t-reliability for change in environmental temperature,v-reliability for change in voltage,+Not specified in the paper

photonic emission on the rear side of the 180 nm Altera FPGA board.

3) CRYPTANALYSIS ATTACK

It is known that cryptanalysis and modeling attacks are the two standard mathematical methodologies employed for examining the security of PUFs. In the domain of PUF, cryptanalysis refers to computational attacks that use CRPs and PUF design attributes to determine the response and they don't require a mathematical model explicitly declared. Although a strong key is determined utilizing fuzzy extractors and error-correcting codes, [96] and [127] successfully model the composite PUF using a cryptanalysis approach.

4) PREDICTABILITY TEST

The unpredictable nature of CRPs is the quintessential feature expected from a PUF. Hamming Distance Test ($HDT(t)$) and Propagation Criteria ($PC(t)$) are two of the few unpredictability tests used to validate the APUF variants, viz., XOR APUF, LSPUF, and Composite PUF [128]. These tests estimate the likelihood of output transition probability to evaluate the t flipping bits due to the challenge pairs(Ch_i, Ch_j) with $HD(Ch_i, Ch_j) = t$ and $HD(Ch_i, Ch_j) \leq t$. Reference [128] proves that the said testing schemes are insufficient to test the unpredictability and incorporate a new parameter (flipping bit pattern vector e) to HDT. The adversary finds fewer chances to rebuild the model if the value of $HDT(e, t)$=0.5.

E. MODELING ATTACK ON XOR APUF

In [129] and [130], x-XOR PUF ($x \leq 6$) is demonstrated to be robust against attacks since it takes more computational time to achieve the final response. However, there exists a significant correlation between the challenges assigned randomly to any one of the APUF components. CMA-ES algorithm [95], thus utilizes these correlated values along with a divide and conquer attack to build each component of APUF. The application of the neural network method has efficiently modeled the x-XOR PUF ($x \leq 9$) [131], [132], [133], [134], [135], and the LR approach is successful in obtaining the entire CRPs [114], [115]. YU and Wen [136] propose a novel hybrid ML attack model on XOR APUF using CNN and side-channel analysis. As CNN did not effectively extract the critical correlation between the N input challenge bits of the XOR APUF, SCA has been integrated to pre-process the challenge. Later, the correlated challenge enters the CNN, which significantly raises the training accuracy to 98% [136]. Generally, XOR APUF employs the same challenges for all the APUF instances. The literature study reveals that CDC-XPUF variants with different challenges for different APUF instances have lower resilience against modeling attacks [91]. CDC-XPUF [91], [137] employs different challenges for each APUF instance in contrast to XOR APUF. The author has modeled the CDC-x-XPUF with $x(3, 4, 5)$ and demonstrated for $x > 5$. However, it fails to model the PUF with a training size of 100 million using four-layer, No-Neuron, and LR with R-prop.

F. MODELING ATTACK ON DAPUF

The DAPUF (3-1 and 4-1) [34] has been proven significantly more resilient against modeling attacks by using SVM^{light}. With 1000 CRPs given to the SVM for testing, the prediction rate was reduced to 57% compared with APUF. The CRPs are then increased to 17 million to model the DAPUF using LR [118]. This technique has modeled the 2-1 DAPUF albeit with increased complexity while applying it to 3-1 and 4-1 DA-APUF. Though 3-1 DAPUF has been modeled with an accuracy of 86%, modeling 4-1 DAPUF has not been successful. On the other hand, the DL technique has effectively managed to model the 3-1 and 4-1 DAPUF with 1 million CRPs [118], posing a threat to PUF design. In [138], the authors have used Multilayer Perceptron (MLP) neural network with an Adaptive Moment Estimation optimizer [139] and ReLU (activation function for hidden layers) to build the XOR APUF and DAPUF. A comparatively reduced

TABLE 5. Modeling attack of various compositions of APUF.

Techniques applied for modeling attack	PUF under modeling attack	No. of XORs/ FF-loops	No. of CRPs	Prediction Accuracy (%)	Training time	Challenge size
LR	APUF [124]	-#	18,050 39,200	99.9	0.60 s 2.10 s	64 128
Fault injection	APUF [142]	-	25000	-	-	32
DL	APUF [117]	-	6800 8000	99.5	11.25 s 18.21 s	64 128
LR	DAPUF [118]	2-1 3-1	100 K+ 17 M++	∼80 - 93 76.2	-	64
DL	DAPUF [118]	3-1 4-1	17 M	85.8 71.3-81.5	-	64
Neural network with Multi-Layer Perceptron (MLP)	DAPUF [138]	2-1 3-1 4-1	1 M	95.62 91.63 81.46	-	64
SVM^{light}	DAPUF [34]	2-1 3-1 4-1	1 K	56.64 -80.72 ∼56 54.73 -56.11	-	64
LR	LSPUF [129]	5 5	300 K 1M	99	13:06 Hrs 267 days	64 128
LR	XOR APUF [129]	6 5	200 K 500 K	99	31:01 Hrs 16:36 Hrs	64 128
LR	XOR APUF [46]	6 5	1.40 M 2.2 M	-	1.01 min 54 s	64 128
SVM^{light}	XOR APUF [34]	2 3	1 K	93.50 - 95.41 85.25 - 85.95	-	64
Neural network with MLP	XOR APUF [138]	2 3 4	1 M	98.23 96.72 96.19	-	-
Reliability based CMA-ES	XOR APUF [95], [116]	8 4	100 K 1.2 M	84.6 92.3	4:6 Hrs 155 Hrs	128 64
DL	XOR APUF [117]	6 7	680 K 1.2 M	97.68 NP*	20.52 min NP	64
		5 6	655 K 1.2 M	97.87 NP	29.21 min NP	128
Hybrid (SCA and CNN)	XOR APUF [136]	4	150 K	98.4	-	128
Neural network	XOR APUF [134]	9	4.2 M	99	10 min	6
ES	FFAPUF [129]	10 10	50 K 50 K	97.86 97.31	47.07 min 3:15 Hrs	64 128
Neural network with MLP	FFAPUF [120]	5 6	200 K	87.1367 86.866	117.0242 min 374.7134 min	64
		5 6	200 K	86.978 84.0047	81.6782 min 324.0404 min	128
DL	iPUF [117]	4,4 5,5	319 K 1.2 M	97.44 NP	5 min 23 s NP	64
		4,4 5,5	647 K 1.2 M	97.68 NP	32 min 17 s NP	128

#Not specified in the paper,+K=10^3,++M=10^6,*modeling attack failed

CRPs around 500 × 10^3 is used to model N XOR APUF (N = 2,3,4) and DAPUF (2-1, 3-1, and 4-1) and has achieved an accuracy of around 80% for 4-1 DAPUF. For 1 million CRPs, the accuracy was reported to be 81% for 4-1 DAPUF. Hence, it is more claimed that among XOR APUF (N = 2,3,4) and DAPUF (2-1, 3-1, and 4-1), 4-1 DAPUF has proved robust to modeling attacks with lower prediction accuracy.

G. MODELING ATTACK ON FEED-FORWARD APUF

Feed-Forward APUF is considered robust against modeling attacks as the number of loops increases in N stage APUF with l-loops of feed-forward arbiter. FF-APUF [129] modeled with the same number of loops achieved a prediction accuracy of about 95% for $l \geq 6$ loops. The MLP neural network [120] with three layers of the neuron at the input and one at the output has been used to model the FF-APUF (64 and 128-bit) with varying lengths, l ($1 \leq l \leq 6$). The maximum training time to model was 374.7134 minutes and 324.0204 min with 200 × 10^3 CRPs for 64 and 128-bit challenges (l-6), respectively. The training time was reduced to 73 minutes for 1 million CRPs with ten loops by combining additional bits (ghost bits) in [121] along with challenge bits to FF-APUF and XOR APUF using the above neural network method.

H. MODELING ATTACK ON LSPUF

The responses from the LSPUF have a significant degree of entropy due to the obfuscated CRPs. For $x > 6$ [115], the LSPUF is resilient to LR-based modeling. However, while adopting parallel implementation, it failed for $x = 9$ [46], [95]. In [128], the modeling of y-XOR PUF helped to model

the x-XOR PUF ($y < x$), thus reducing the time and data complexity even without side-channel analysis. Furthermore, obtaining CRPs is found indeed successful using SCA [124] and cryptanalysis [96].

I. MODELING ATTACK ON iPUF

Reference [117] uses the gradient-based numerical optimization method to model the iPUF (4,4) with a training time of 5.23 minutes. This method uses the knowledge of parity vectors obtained from the challenge. The challenges have been collected using MATLAB simulation models and Rectified Linear Unit (ReLU) as an activation function for learning mapping and patterns of the dataset. Probably Approximately Correct (PAC) [140], the sub-space pre-learning method [141] uses randomness and known CRPs to model the iPUF. The uniformity metric is found helpful in splitting the iPUF (4,4) [50], [117] applying LR. The latest technique employs the splitting attack [114] using the divide and conquer ML algorithm for 64-bit iPUF (8,8), and (1,9) with a training size ranging between 300 million to 750 million CRPs, and the prediction accuracy was near 95%.

Table 5 summarizes the modeling attack conducted by respective authors on various APUF variants to understand their resiliency against attacks for different machine-learning techniques. One can easily model an attack on a conventional APUF because of its linear additive model using LR, DL, and fault injection with reduced time and prediction accuracy of 99.9%. Adding APUF instances parallelly in the DAPUF increases the computation time by reducing the prediction accuracy between 71% and 95% with increased CRPs. Similarly, the XOR APUF and FFAPUF increase the training time to predict its random output as the parallel instances of APUF, the number of XOR and feed-forward loops are increased in the design model. Thus, XOR APUF, FF-APUF, and iPUF are challenging to get modeled in machine learning because they are designed to be unpredictable and exhibit high variability in their outputs. This makes it difficult for a machine learning model to accurately predict the response of an XOR APUF, FFAPUF, and iPUF, as the model needs to account for many possible input combinations and corresponding output values.

VI. CONCLUSION AND FUTURE SCOPE

The Arbiter PUF, being a strong PUF in nature, operates on the delay difference incurred due to the intrinsic manufacturing variations, thus possessing an exponential number of CRPs, which makes it more suitable for authentication in several applications. The APUF is well suited for low-weight applications since the resource utilization is much lower compared to other strong PUFs. Some of the key benefits of APUFs include the following:

- *Ease of implementation:* APUFs can be implemented using standard digital logic gates.
- *High degree of variability:* APUFs can generate a large number of possible CRPs, which can improve the uniqueness of the PUF.
- *Robustness:*APUFs can be designed to be resistant to tampering and against attempts to reverse engineer the response. This feature improves the security of the PUF.

The factors that need to be considered while designing APUF include the delay difference, placement, and routing of the switch blocks, size of the PUF model, cost, power requirements, the execution time, and the compatibility of the APUF with the manufacturing process of the device in which it is being used. Factors that can influence the composition of an APUF include the desired level of security, the size and complexity of the device, and the resources available for implementing the PUF. Considering the above discussions, the APUF can be used securely for IP protection, IoT device authentication, Internet of Vehicles, and resource-constrained applications such as RFIDs and smart cards.

As PUF technology evolves, there are several areas in which future research on APUFs could focus on improving their performance and capabilities. Typical directions can be as follows:

- Developing methods that will further strengthen the confidentiality of APUFs against tampering and reverse engineering.
- Investigating measures to increase the number of possible CRPs and further improve the uniqueness of APUFs.
- Developing new fabrication techniques to lower the cost of APUFs, to enable their integration into a broader range of devices.
- Reduce the power consumption of APUFs, to make them more suitable for low-power and battery-powered devices.
- Integrate APUFs with other security features, such as secure communication protocols, to provide an additional layer of protection.
- Investigate the robustness of PUFs against physical attacks such as differential power analysis, electromagnetic analysis, and side-channel analysis.

REFERENCES

[1] A. Babaei and G. Schiele, "Physical unclonable functions in the Internet of Things: State of the art and open challenges," *Sensors*, vol. 19, no. 14, p. 3208, Jul. 2019, doi: 10.3390/s19143208.
[2] W. Liang, S. Xie, J. Long, K.-C. Li, D. Zhang, and K. Li, "A double PUF-based RFID identity authentication protocol in service-centric Internet of Things environments," *Inf. Sci.*, vol. 503, pp. 129–147, Dec. 2019, doi: 10.1016/j.ins.2019.06.047.
[3] A. Shamsoshoara, A. Korenda, F. Afghah, and S. Zeadally, "A survey on physical unclonable function (PUF)-based security solutions for Internet of Things," *Comput. Netw.*, vol. 183, Dec. 2020, Art. no. 107593, doi: 10.1016/j.comnet.2020.107593.
[4] C. Herder, M.-D. Yu, F. Koushanfar, and S. Devadas, "Physical unclonable functions and applications: A tutorial," *Proc. IEEE*, vol. 102, no. 8, pp. 1126–1141, Aug. 2014, doi: 10.1109/JPROC.2014.2320516.
[5] B. Gassend, D. Clarke, M. Van Dijk, and S. Devadas, "Silicon physical random functions," in *Proc. ACM Conf. Comput. Commun. Secur.*, 2002, pp. 148–160, doi: 10.1145/586131.586132.
[6] B. Gassend, D. Lim, D. Clarke, M. van Dijk, and S. Devadas, "Identification and authentication of integrated circuits," *Concurrency Comput., Pract. Exper.*, vol. 16, no. 11, pp. 1077–1098, Sep. 2004, doi: 10.1002/cpe.805.

[7] H. Ning, F. Farha, A. Ullah, and L. Mao, "Physical unclonable function: Architectures, applications and challenges for dependable security," *IET Circuits, Devices Syst.*, vol. 14, no. 4, pp. 407–424, Jul. 2020, doi: 10.1049/iet-cds.2019.0175.

[8] B. Gassend, "Physical random functions," M.S. thesis, Dept. Elect. Eng. Comput. Sci., Massachusetts Inst. Technol., Cambridge, MA, USA, Jan. 2003.

[9] Z. Cherif, J.-L. Danger, S. Guilley, and L. Bossuet, "An easy-to-design PUF based on a single oscillator: The loop PUF," in *Proc. 15th Euromicro Conf. Digit. Syst. Design*, Sep. 2012, pp. 156–162, doi: 10.1109/DSD.2012.22.

[10] M. Gao, K. Lai, and G. Qu, "A highly flexible ring oscillator PUF," in *Proc. 51st Annu. Design Autom. Conf.*, Jun. 2014, pp. 1–6, doi: 10.1145/2593069.2593072.

[11] Q. Chen, G. Csaba, P. Lugli, U. Schlichtmann, and U. Ruhrmair, "The bistable ring PUF: A new architecture for strong physical unclonable functions," in *Proc. IEEE Int. Symp. Hardw.-Oriented Secur. Trust*, San Diego, CA, USA, Jun. 2011, pp. 134–141, doi: 10.1109/HST.2011.5955011.

[12] Q. Wang, M. Gao, and G. Qu, "A machine learning attack resistant dual-mode PUF," in *Proc. Great Lakes Symp. VLSI*, May 2018, pp. 177–182, doi: 10.1145/3194554.3194590.

[13] J. Zhang, C. Shen, Z. Guo, Q. Wu, and W. Chang, "CT PUF: Configurable tristate PUF against machine learning attacks for IoT security," *IEEE Internet Things J.*, vol. 9, no. 16, pp. 14452–14462, Aug. 2022, doi: 10.1109/JIOT.2021.3090475.

[14] S. Hemavathy and V. S. K. Bhaaskaran, "Design and analysis of secure quasi-adiabatic tristate physical unclonable function," *IEEE Consum. Electron. Mag.*, vol. 11, no. 4, pp. 109–114, Jul. 2021, doi: 10.1109/MCE.2021.3117541.

[15] S. Hemavathy and V. S. K. Bhaaskaran, "Design and analysis of secure quasi-adiabatic tristate physical unclonable function," in *Proc. IEEE Int. Symp. Smart Electron. Syst. (iSES)*, Dec. 2020, pp. 109–114, doi: 10.1109/ISES50453.2020.00034.

[16] Z. Zulfikar, N. Soin, S. F. W. M. Hatta, M. S. A. Talip, and A. Jaafar, "Routing density analysis of area-efficient ring oscillator physically unclonable functions," *Appl. Sci.*, vol. 11, no. 20, p. 9730, Oct. 2021, doi: 10.3390/app11209730.

[17] D. E. Holcomb, W. P. Burleson, and K. Fu, "Power-up SRAM state as an identifying fingerprint and source of true random numbers," *IEEE Trans. Comput.*, vol. 58, no. 9, pp. 1198–1210, Sep. 2009, doi: 10.1109/TC.2008.212.

[18] F. Tehranipoor, N. Karimian, W. Yan, and J. A. Chandy, "DRAM-based intrinsic physically unclonable functions for system-level security and authentication," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 3, pp. 1085–1097, Mar. 2017, doi: 10.1109/TVLSI.2016.2606658.

[19] S. Hemavathy and V. S. K. Bhaaskaran, "Double edge-triggered tristate flip-flop physical unclonable function for secure IoT ecosystem," in *Proc. IEEE Int. Symp. Smart Electron. Syst. (iSES)*, Dec. 2021, pp. 44–47, doi: 10.1109/ISES52644.2021.00022.

[20] U. RÃührmair and D. E. Holcomb, "PUFs at a glance," in *Proc. Design, Automat. Test Eur. Conf. Exhib. (DATE)*, Mar. 2014, pp. 1–6, doi: 10.7873/DATE2014.360.

[21] U. Rührmair, J. Sölter, and F. Sehnke, "On the foundations of physical unclonable functions," *IACR Cryptol. ePrint Arch.*, vol. 2009, p. 277, 2009.

[22] Y. Guo, T. Dee, and A. Tyagi, "Multi-block APUF with 2-level voltage supply," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI (ISVLSI)*, Jul. 2018, pp. 327–332, doi: 10.1109/ISVLSI.2018.00067.

[23] D. Lim, J. W. Lee, B. Gassend, G. E. Suh, M. van Dijk, and S. Devadas, "Extracting secret keys from integrated circuits," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 13, no. 10, pp. 1200–1205, Oct. 2005, doi: 10.1109/TVLSI.2005.859470.

[24] Z. C. Jouini, J.-L. Danger, and L. Bossuet, "Performance evaluation of physically unclonable function by delay statistics," in *Proc. IEEE 9th Int. New Circuits Syst. Conf.*, Bordeaux, France, Jun. 2011, pp. 482–485, doi: 10.1109/NEWCAS.2011.5981324.

[25] 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, doi: 10.2197/ipsjjip.22.344.

[26] 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, doi: 10.1109/ReConFig.2010.24.

[27] A. Maiti, V. Gunreddy, and P. Schaumont, "A systematic method to evaluate and compare the performance of physical unclonable functions," *IACR Cryptol. ePrint Arch.*, vol. 2011, p. 657, 2011.

[28] A. Sadr and M. Zolfaghari-Nejad, "Weighted Hamming distance for PUF performance evaluation," *Electron. Lett.*, vol. 49, no. 22, pp. 1376–1378, Oct. 2013, doi: 10.1049/el.2013.2326.

[29] M. Kaur, R. Rashidzadeh, and R. Muscedere, "Reliability of physical unclonable function under temperature and supply voltage variations," *Midwest Symp. Circuits Syst.*, vol. 2018, pp. 1008–1011, Aug. 2019, doi: 10.1109/MWSCAS.2018.8623849.

[30] 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, doi: 10.1109/JSSC.2007.910961.

[31] C. Gu, W. Liu, N. Hanley, R. Hesselbarth, and M. O'Neill, "A theoretical model to link uniqueness and min-entropy for PUF evaluations," *IEEE Trans. Comput.*, vol. 68, no. 2, pp. 287–293, Feb. 2019, doi: 10.1109/TC.2018.2866241.

[32] 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. Digit. Syst. Design*, Aug. 2015, pp. 559–562, doi: 10.1109/DSD.2015.51.

[33] 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 Design Autom. Conf.*, Jan. 2015, pp. 6–7, doi: 10.1109/ASPDAC.2015.7058919.

[34] T. Machida, D. Yamamoto, M. Iwamoto, and K. Sakiyama, "A new arbiter PUF for enhancing unpredictability on FPGA," *Sci. World J.*, vol. 2015, pp. 1–13, Jan. 2015, doi: 10.1155/2015/864812.

[35] T. Machida, D. Yamamoto, M. Iwamoto, and K. Sakiyama, "A new mode of operation for arbiter PUF to improve uniqueness on FPGA," in *Proc. Ann. Comput. Sci. Inf. Syst.*, Sep. 2014, pp. 871–878.

[36] M. Majzoobi, A. Kharaya, F. Koushanfar, and S. Devadas, "Automated design, implementation, and evaluation of arbiter-based PUF on FPGA using programmable delay lines," *IACR Cryptol. ePrint Arch.*, vol. 2014, p. 639, 2014. [Online]. Available: http://eprint.iacr.org/2014/639

[37] P. H. Nguyen, D. P. Sahoo, C. Jin, K. Mahmood, U. Rührmair, and M. Van Dijk, "The interpose PUF: Secure PUF design against state-of-the-art machine learning attacks," *IACR Trans. Cryptograph. Hardw. Embedded Syst.*, vol. 2019, no. 4, pp. 243–290, Aug. 2019, doi: 10.13154/tches.v2019.i4.243-290.

[38] J. Yang, X. Yu, and R. Wei, "A low resource consumption arbiter PUF improved switch component design for FPGA," *J. Phys., Conf. Ser.*, vol. 2221, no. 1, May 2022, Art. no. 012011, doi: 10.1088/1742-6596/2221/1/012011.

[39] E. Ozturk, G. Hammouri, and B. Sunar, "Physical unclonable function with tristate buffers," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2008, pp. 3194–3197, doi: 10.1109/ISCAS.2008.4542137.

[40] M. H. Mahalat, S. Mandal, A. Mondal, and B. Sen, "An efficient implementation of arbiter PUF on FPGA for IoT application," in *Proc. 32nd IEEE Int. Syst.-Chip Conf. (SOCC)*, Sep. 2019, pp. 324–329, doi: 10.1109/SOCC46988.2019.1570548268.

[41] C. W. Lin and S. Ghosh, "A family of Schmitt-Trigger-based arbiter-PUFs and selective challenge-pruning for robustness and quality," in *Proc. IEEE Int. Symp. Hardw. Oriented Secur. Trust (HOST)*, May 2015, pp. 32–37, doi: 10.1109/HST.2015.7140232.

[42] C. Aknesil and E. Dubrova, "An FPGA implementation of 4×4 arbiter PUF," in *Proc. Int. Symp. Mult. Log.*, May 2021, pp. 160–165, doi: 10.1109/ISMVL51352.2021.00035.

[43] J. Zhou, "FPGA oriented design method of adjustable arbiter physical unclonable circuit," *J. Phys., Conf. Ser.*, vol. 1449, no. 1, Jan. 2020, Art. no. 012060, doi: 10.1088/1742-6596/1449/1/012060.

[44] L. Lin, D. Holcomb, D. K. Krishnappa, P. Shabadi, and W. Burleson, "Low-power sub-threshold design of secure physical unclonable functions," in *Proc. 16th ACM/IEEE Int. Symp. Low Power Electron. Design*, Aug. 2010, pp. 43–48, doi: 10.1145/1840845.1840855.

[45] L. Lin, S. Srivathsa, D. K. Krishnappa, P. Shabadi, and W. Burleson, "Design and validation of arbiter-based PUFs for sub-45-nm low-power security applications," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 4, pp. 1394–1403, Aug. 2012, doi: 10.1109/TIFS.2012.2195174.

[46] J. Tobisch and G. T. Becker, "On the scaling of machine learning attacks on PUFs with application to noise bifurcation," in *Radio Frequency Identification* (Lecture Notes in Computer Science), vol. 9440. New York, NY, USA: Springer, Jun. 2015, pp. 17–31, doi: 10.1007/978-3-319-24837-0_2.

[47] Z. Cherif, J.-L. Danger, F. Lozac'h, Y. Mathieu, and L. Bossuet, "Evaluation of delay PUFs on CMOS 65 nm technology: ASIC vs FPGA," in *Proc. 2nd Int. Workshop Hardw. Architectural Support Secur. Privacy*, Jun. 2013, pp. 1–8.

[48] D. P. Sahoo, P. H. Nguyen, R. S. Chakraborty, and D. Mukhopadhyay, "Architectural bias: A novel statistical metric to evaluate arbiter PUF variants," *IACR Cryptol. ePrint Arch.*, vol. 2016, p. 57, Jan. 2016. [Online]. Available: http://dblp.uni-trier.de/db/journals/iacr/iacr2016.html#SahooNCM16

[49] D. P. Sahoo, P. H. A. Nguyen, and R. S. Chakraborty. (2018). *On the Architectural Analysis of Arbiter Delay PUF Variants*. [Online]. Available: http://dblp.uni-trier.de/db/journals/iacr/iacr2016.html#SahooNCM16

[50] A. Aghaie and A. Moradi, "Inconsistency of simulation and practice in delay-based strong PUFs," *IACR Trans. Cryptograph. Hardw. Embedded Syst.*, vol. 2021, no. 3, pp. 520–551, Jul. 2021, doi: 10.46586/tches.v2021.i3.520-551.

[51] 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 *Symp. VLSI Circuits, Dig. Tech. Papers*, Jun. 2004, pp. 176–179, doi: 10.1109/VLSIC.2004.1346548.

[52] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *Proc. 44th ACM/IEEE Design Automat. Conf.*, San Diego, CA, USA, 2007, pp. 9–14.

[53] S. Chen, B. Li, F. Dan, and J. Chen, "A machine learning resistant arbiter PUFs scheme based on polynomial reconstruction," in *Proc. IEEE 2nd Int. Conf. Signal Image Process. (ICSIP)*, Singapore, Aug. 2017, pp. 465–469, doi: 10.1109/SIPROCESS.2017.8124585.

[54] W. Liang, Z. Ning, S. Xie, Y. Hu, S. Lu, and D. Zhang, "Secure fusion approach for the Internet of Things in smart autonomous multi-robot systems," *Inf. Sci.*, vol. 579, pp. 468–482, Jun. 2021, doi: 10.1016/J.INS.2021.08.035.

[55] W. Liang, S. Xie, D. Zhang, X. Li, and K.-C. Li, "A mutual security authentication method for RFID-PUF circuit based on deep learning," *ACM Trans. Internet Technol.*, vol. 22, no. 2, pp. 1–20, May 2022, doi: 10.1145/3426968.

[56] H. Su, M. Zwolinski, and B. Halak, "A machine learning attacks resistant two stage physical unclonable functions design," in *Proc. IEEE 3rd Int. Verification Secur. Workshop (IVSW)*, Costa Brava, Spain, Jul. 2018, pp. 52–55, doi: 10.1109/IVSW.2018.8494839.

[57] D. A. Puf, Y. Wang, and Z. Chang, "Authentication against man-in-the-middle attack with a time-variant reconfigurable dual-LFSR-based arbiter PUF," 2020, *arXiv:2007.10755*.

[58] M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Techniques for design and implementation of secure reconfigurable PUFs," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 2, no. 1, pp. 1–10, 2009, doi: 10.1145/1502781.1502786.

[59] B. Gassend, M. Van Dijk, D. Clarke, and S. Devadas, "Controlled physical random functions," in *Security With Noisy Data*. London, U.K.: Springer, 2007, pp. 235–253, doi: 10.1007/978-1-84628-984-2_14.

[60] J. Guajardo, S. S. Kumar, G.-J. Schrijen, and P. Tuyls, "Physical unclonable functions and public-key crypto for FPGA IP protection," in *Proc. Int. Conf. Field Program. Log. Appl.*, Aug. 2007, pp. 189–195, doi: 10.1109/FPL.2007.4380646.

[61] T. A. Idriss, H. A. Idriss, and M. A. Bayoumi, "A lightweight PUF-based authentication protocol using secret pattern recognition for constrained IoT devices," *IEEE Access*, vol. 9, pp. 80546–80558, 2021, doi: 10.1109/ACCESS.2021.3084903.

[62] S. S. Zalivaka, A. A. Ivaniuk, and C.-H. Chang, "Reliable and modeling attack resistant authentication of arbiter PUF in FPGA implementation with trinary quadruple response," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 4, pp. 1109–1123, Apr. 2018, doi: 10.1109/TIFS.2018.2870835.

[63] Y. Gao, G. Li, H. Ma, S. F. Al-Sarawi, O. Kavehei, D. Abbott, and D. C. Ranasinghe, "Obfuscated challenge-response: A secure lightweight authentication mechanism for PUF-based pervasive devices," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops*, Sydney, NSW, Australia, Mar. 2016, pp. 1–6, doi: 10.1109/PERCOMW.2016.7457162.

[64] Y. Yilmaz, S. R. Gunn, and B. Halak, "Lightweight PUF-based authentication protocol for IoT devices," *Proc. IEEE 3rd Int. Verification Secur. Workshop (IVSW)*, Jul. 2018, pp. 38–43, doi: 10.1109/IVSW.2018.8494884.

[65] J. Zhang and C. Shen, "Set-based obfuscation for strong PUFs against machine learning attacks," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 68, no. 1, pp. 288–300, Jan. 2021, doi: 10.1109/TCSI.2020.3028508.

[66] M. Majzoobi, M. Rostami, F. Koushanfar, D. S. Wallach, and S. Devadas, "Slender PUF protocol: A lightweight, robust, and secure authentication by substring matching," in *Proc. IEEE Symp. Secur. Privacy Workshops*, May 2012, pp. 33–44, doi: 10.1109/SPW.2012.30.

[67] M.-D. Yu, D. M'Raihi, I. Verbauwhede, and S. Devadas, "A noise bifurcation architecture for linear additive physical functions," in *Proc. IEEE Int. Symp. Hardw.-Oriented Secur. Trust (HOST)*, May 2014, pp. 124–129, doi: 10.1109/HST.2014.6855582.

[68] 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, doi: 10.1109/TMSCS.2016.2553027.

[69] Z. He, W. Chen, L. Zhang, G. Chi, Q. Gao, and L. Harn, "A highly reliable arbiter PUF with improved uniqueness in FPGA implementation using bit-self-test," *IEEE Access*, vol. 8, pp. 181751–181762, 2020, doi: 10.1109/ACCESS.2020.3028514.

[70] Y. Gao, H. Ma, S. F. Al-Sarawi, D. Abbott, and D. C. Ranasinghe, "PUF-FSM: A controlled strong PUF," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 37, no. 5, pp. 1104–1108, May 2017, doi: 10.1109/TCAD.2017.2740297.

[71] J. Kokila and N. Ramasubramanian, "Enhanced authentication using hybrid PUF with FSM for protecting IPs of SoC FPGAs," *J. Electron. Test.*, vol. 35, no. 4, pp. 543–558, Aug. 2019, doi: 10.1007/s10836-019-05808-w.

[72] A. Spenke, R. Breithaupt, and R. Plaga, "An arbiter PUF secured by remote random reconfigurations of an FPGA," in *Proc. Int. Conf. Trust Trustworthy Comput.* in Lecture Notes in Computer Science, vol. 9824, 2016, pp. 140–158, doi: 10.1007/978-3-319-45572-3_8.

[73] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, and M. Levenson, "NIST special publication 800–22: A statistical test suite for the validation of random number generators and pseudo random number generators for cryptographic applications," Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Tech. Rep. 800-22 Rev 1a, 2010.

[74] C. W. O'Donnell, G. E. Suh, and S. Devadas, "PUF-based random number generation," MIT, Cambridge, MA, USA, Tech. Rep., 481, 2004.

[75] A. Sadr and M. Zolfaghari-Nejad, "Physical unclonable function (PUF) based random number generator," *Adv. Comput., Int. J.*, vol. 3, no. 2, pp. 139–145, 2012, doi: 10.5121/acij.2012.3214.

[76] Y. Yu, E. Dubrova, M. Naslund, and S. Tao, "On designing PUF-based TRNGs with known answer tests," in *Proc. IEEE Nordic Circuits Syst. Conf. (NORCAS), NORCHIP Int. Symp. Syst.-Chip (SoC)*, Oct. 2018, pp. 18–23, doi: 10.1109/NORCHIP.2018.8573489.

[77] K. Pratihar, U. Chatterjee, M. Alam, D. Mukhopadhyay, and R. S. Chakraborty, "A tale of twin primitives: Single-chip solution for PUFs and TRNGs," *IACR Cryptol. ePrint Arch.*, vol. 2021, p. 1067, Dec. 2021.

[78] M. Ayat, R. E. Atani, and S. Mirzakuchaki, "On design of PUF-based random number generators," *Int. J. Netw. Secur. Appl.*, vol. 3, no. 3, pp. 30–40, May 2011, doi: 10.5121/ijnsa.2011.3303.

[79] A. Maiti, R. Nagesh, A. Reddy, and P. Schaumont, "Physical unclonable function and true random number generator: A compact and scalable implementation," in *Proc. 19th ACM Great Lakes Symp. VLSI*, May 2009, pp. 425–428, doi: 10.1145/1531542.1531639.

[80] M. Brookes, "Extracting secret keys from integrated circuits," *IEEE Trans. Circuits Syst. Analog Digit. Signal Process.*, vol. 47, no. 9, pp. 930–935, Oct. 2000.

[81] S. V. S. Avvaru, Z. Zeng, and K. K. Parhi, "Homogeneous and heterogeneous feed-forward XOR physical unclonable functions," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 2485–2498, 2020, doi: 10.1109/TIFS.2020.2968113.

[82] Y. Lao and K. K. Parhi, "Novel reconfigurable silicon physical unclonable functions," in *Proc. Workshop Found. Dependable Secure Cyber-Phys. Syst.*, 2011, pp. 30–36.

[83] F. Ganji, S. Tajik, P. Stauss, J.-P. Seifert, M. Tehranipoor, and D. Forte, "Rock'n'roll PUFs: Crafting provably secure PUFs from less secure ones (extended version)," *J. Cryptograph. Eng.*, vol. 11, no. 2, pp. 105–118, Jun. 2021, doi: 10.1007/s13389-020-00226-7.

[84] Y. Lao and K. K. Parhi, "Statistical analysis of MUX-based physical unclonable functions," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 33, no. 5, pp. 649–662, May 2014, doi: 10.1109/TCAD.2013.2296525.

[85] Y. Wang, C. Wang, C. Gu, Y. Cui, M. O'Neill, and W. Liu, "A dynamically configurable PUF and dynamic matching authentication protocol," *IEEE Trans. Emerg. Topics Comput.*, vol. 10, no. 2, pp. 1091–1104, Apr. 2021, doi: 10.1109/TETC.2021.3072421.

[86] C. Gu, W. Liu, Y. Cui, N. Hanley, M. OrNeill, and F. Lombardi, "A flip-flop based arbiter physical unclonable function (APUF) design with high entropy and uniqueness for FPGA implementation," *IEEE Trans. Emerg. Topics Comput.*, vol. 9, no. 4, pp. 1853–1866, Oct. 2021, doi: 10.1109/TETC.2019.2935465.

[87] C. Gu, Y. Cui, N. Hanley, and M. O'Neill, "Novel lightweight FF-APUF design for FPGA," in *Proc. 29th IEEE Int. Syst.-Chip Conf. (SOCC)*, Sep. 2016, pp. 75–80, doi: 10.1109/SOCC.2016.7905439.

[88] R. Sushma and N. S. Murty, "Feedback oriented XORed flip-flop based arbiter PUF," in *Proc. Int. Conf. Electr., Electron., Commun., Comput., Optim. Techn. (ICEECCOT)*, Dec. 2018, pp. 1444–1448, doi: 10.1109/ICEECCOT43722.2018.9001605.

[89] R. Kumar, S. N. Dhanuskodi, and S. Kundu, "On manufacturing aware physical design to improve the uniqueness of silicon-based physically unclonable functions," in *Proc. 27th Int. Conf. VLSI Design 13th Int. Conf. Embedded Syst.*, Jan. 2014, pp. 381–386, doi: 10.1109/VLSID.2014.72.

[90] R. Kumar, H. K. Chandrikakutty, and S. Kundu, "On improving reliability of delay based physically unclonable functions under temperature variations," in *Proc. IEEE Int. Symp. Hardw.-Oriented Secur. Trust*, Jun. 2011, pp. 142–147, doi: 10.1109/HST.2011.5955012.

[91] K. T. Mursi and Y. Zhuang, "Experimental examination of component-differentially-challenged XOR PUF circuits," *J. Phys., Conf. Ser.*, vol. 1729, no. 1, Jan. 2021, Art. no. 012006, doi: 10.1088/1742-6596/1729/1/012006.

[92] H. Idriss, T. Idriss, and M. Bayoumi, "A highly reliable dual-arbiter PUF for lightweight authentication protocols," in *Proc. IEEE Int. Conf. RFID Technol. Appl. (RFID-TA)*, Sep. 2017, pp. 248–253, doi: 10.1109/RFID-TA.2017.8098903.

[93] M. S. Mispan, H. Sarkawi, A. Z. Jidin, R. H. Ramlee, and H. M. Nasir, "Design and implementation of multiplexed and obfuscated physical unclonable function," *Indonesian J. Electr. Eng. Informat. (IJEEI)*, vol. 9, no. 1, pp. 91–100, Mar. 2021, doi: 10.11591/ijeei.v9i1.2664.

[94] D. P. Sahoo, D. Mukhopadhyay, R. S. Chakraborty, and P. H. Nguyen, "A multiplexer-based arbiter PUF composition with enhanced reliability and security," *IEEE Trans. Comput.*, vol. 67, no. 3, pp. 403–417, Mar. 2018, doi: 10.1109/TC.2017.2749226.

[95] G. T. Becker, "The gap between promise and reality: On the insecurity of XOR arbiter PUFs," in *Proc. Int. Workshop Cryptograph. Hardw. Embedded Syst.*, in Lecture Notes in Computer Science, vol. 9293, Sep. 2015, pp. 535–555, doi: 10.1007/978-3-662-48324-4_27.

[96] D. P. Sahoo, P. H. Nguyen, D. Mukhopadhyay, and R. S. Chakraborty, "A case of lightweight PUF constructions: Cryptanalysis and machine learning attacks," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 8, pp. 1334–1343, Aug. 2015, doi: 10.1109/TCAD.2015.2448677.

[97] P. H. Nguyen, D. P. Sahoo, C. Jin, K. Mahmood, and M. V. Dijk, "MXPUF: Secure PUF design against state-of-the-art modeling attacks," *IACR Cryptol. ePrint Arch.*, vol. 2017, p. 572, Jan. 2017.

[98] D. P. Sahoo, S. Patranabis, D. Mukhopadhyay, and R. S. Chakraborty, "Fault tolerant implementations of delay-based physically unclonable functions on FPGA," in *Proc. Workshop Fault Diagnosis Tolerance Cryptogr. (FDTC)*, 2016, pp. 87–101, doi: 10.1109/FDTC.2016.10.

[99] M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Lightweight secure PUFs," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, Nov. 2008, pp. 670–673, doi: 10.1109/ICCAD.2008.4681648.

[100] S. S. Zalivaka, A. A. Ivaniuk, and C.-H. Chang, "FPGA implementation of modeling attack resistant arbiter PUF with enhanced reliability," in *Proc. 18th Int. Symp. Quality Electron. Design (ISQED)*, Mar. 2017, pp. 313–318, doi: 10.1109/ISQED.2017.7918334.

[101] S. S. Zalivaka, A. A. Ivaniuk, and C.-H. Chang, "Low-cost fortification of arbiter PUF against modeling attack," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2017, pp. 1–4, doi: 10.1109/ISCAS.2017.8050671.

[102] Y. Cao, W. Zheng, X. Zhao, and C.-H. Chang, "An energy-efficient current-starved inverter based strong physical unclonable function with enhanced temperature stability," *IEEE Access*, vol. 7, pp. 105287–105297, 2019, doi: 10.1109/ACCESS.2019.2932022.

[103] B. Chen, P. Wang, and G. Li, "An obfuscated challenge design for APUF to resist machine learning attacks," in *Proc. IEEE 13th Int. Conf. ASIC (ASICON)*, Oct. 2019, pp. 3–6, doi: 10.1109/ASICON47005.2019.8983648.

[104] W. Ge, S. Hu, J. Huang, B. Liu, and M. Zhu, "FPGA implementation of a challenge pre-processing structure arbiter PUF designed for machine learning attack resistance," *IEICE Electron. Exp.*, vol. 17, no. 2, pp. 1–6, 2020, doi: 10.1587/elex.16.20190670.

[105] 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. 23rd Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Jan. 2018, pp. 97–104, doi: 10.1109/ASPDAC.2018.8297289.

[106] Y. Cui, C. Gu, Q. Ma, Y. Fang, C. Wang, M. O'Neill, and W. Liu, "Lightweight modeling attack-resistant multiplexer-based multi-PUF (MMPUF) design on FPGA," *Electronics*, vol. 9, no. 5, pp. 1–21, 2020, doi: 10.3390/electronics9050815.

[107] M. El-hajj, A. Fadlallah, M. Chamoun, and A. Serhrouchni, "Secure PUF: Physically unclonable function based on arbiter with enhanced resistance against machine learning (ML) attacks," in *Proc. 5th Int. Conf. Sensors Electron. Instrum.*, Sep. 2019, pp. 216–220.

[108] 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. 1–10, 2017, doi: 10.1145/3053681.

[109] D. P. Sahoo, S. Saha, D. Mukhopadhyay, R. S. Chakraborty, and H. Kapoor, "Composite PUF: A new design paradigm for physically unclonable functions on FPGA," in *Proc. IEEE Int. Symp. Hardw.-Oriented Secur. Trust (HOST)*, May 2014, pp. 50–55.

[110] N. Pundir, F. Amsaad, M. Choudhury, and M. Niamat, "Novel technique to improve strength of weak arbiter PUF," in *Proc. IEEE 60th Int. Midwest Symp. Circuits Syst. (MWSCAS)*, Boston, MA, USA, Aug. 2017, pp. 1532–1535, doi: 10.1109/MWSCAS.2017.8053227.

[111] D. Ganta and L. Nazhandali, "Easy-to-build arbiter physical unclonable function with enhanced challenge/response set," in *Proc. Int. Symp. Quality Electron. Design (ISQED)*, Santa Clara, CA, USA, Mar. 2013, pp. 733–738, doi: 10.1109/ISQED.2013.6523692.

[112] H. Idriss, T. Idriss, P. Williams, and M. Bayoumi, "Memory-based arbiter PUF: A novel highly reliable and scalable strong PUF design," in *Proc. IEEE 6th World Forum Internet Things (WF-IoT)*, New Orleans, LA, USA, Jun. 2020, pp. 1–6, doi: 10.1109/WF-IoT48130.2020.9221015.

[113] N. Wisiol, "Towards attack resilient delay-based strong PUFs," in *Proc. IEEE Int. Symp. Hardw. Oriented Secur. Trust (HOST)*, McLean, VA, USA, Jun. 2022, pp. 5–8, doi: 10.1109/HOST54066.2022.9839791.

[114] N. Wisiol, C. Mühl, N. Pirnay, P. H. Nguyen, M. Margraf, J.-P. Seifert, M. van Dijk, and U. Rührmair, "Splitting the interpose PUF: A novel modeling attack strategy," *Trans. Cryptograph. Hardw. Embedded Systems*, vol. 2020, no. 3, pp. 97–120, 2020, doi: 10.13154/tches.v2020.i3.97-120.

[115] U. RÃührmair, F. Sehnke, J. Sölter, G. Dror, S. Devadas, and J. Schmidhuber, "Modeling attacks on physical unclonable functions," in *Proc. 17th ACM Conf. Comput. Commun. Secur.*, New York, NY, USA, 2010, pp. 237–249, doi: 10.1145/1866307.1866335.

[116] G. T. Becker, "On the pitfalls of using arbiter-PUFs as building blocks," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 8, pp. 1295–1307, Aug. 2015.

[117] P. Santikellur, A. Bhattacharyay, and R. S. Chakraborty, "Deep learning based model building attacks on arbiter PUF compositions," *IACR Cryptol. ePrint Arch.*, vol. 2019, p. 566, 2019.

[118] M. Khalafalla and C. Gebotys, "PUFs deep attacks: Enhanced modeling attacks using deep learning techniques to break the security of double arbiter PUFs," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Florence, Italy, Mar. 2019, pp. 204–209, doi: 10.23919/DATE.2019.8714862.

[119] R. S. Chakraborty, R. R. Jeldi, I. Saha, and J. Mathew, "Binary decision diagram assisted modeling of FPGA-based physically unclonable function by genetic programming," *IEEE Trans. Comput.*, vol. 66, no. 6, pp. 971–981, Jun. 2017, doi: 10.1109/TC.2016.2603498.

[120] M. S. Alkatheiri and Y. Zhuang, "Towards fast and accurate machine learning attacks of feed-forward arbiter PUFs," in *Proc. IEEE Conf. Dependable Secure Comput.*, Aug. 2017, pp. 181–187, doi: 10.1109/DESEC.2017.8073845.
[121] Y. Zhuang, K. T. Mursi, and L. Gaoxiang, "A challenge obfuscating interface for arbiter PUF variants against machine learning attacks," 2021, *arXiv:2103.12935*.
[122] U. Rührmair, X. Xu, J. Sölter, A. Mahmoud, M. Majzoobi, F. Koushanfar, and W. Burleson, "Efficient power and timing side channels for physical unclonable functions," in *Cryptographic Hardware and Embedded Systems*, vol. 8731. Berlin, Germany: Springer, 2014, pp. 476–492, doi: 10.1007/978-3-662-44709-3_26.
[123] D. P. Sahoo, P. H. Nguyen, D. B. Roy, D. Mukhopadhyay, and R. S. Chakraborty, "Side channel evaluation of PUF-based pseudorandom permutation," in *Proc. Euromicro Conf. Digit. Syst. Design (DSD)*, Aug. 2017, pp. 237–243, doi: 10.1109/DSD.2017.79.
[124] F. Ganji, J. Krämer, J.-P. Seifert, and S. Tajik, "Lattice basis reduction attack against physically unclonable functions," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2015, pp. 1070–1080, doi: 10.1145/2810103.2813723.
[125] S. Tajik, F. Ganji, J. P. Seifert, H. Lohrke, and C. Boit, "Laser fault attack on physically unclonable functions," in *Proc. Workshop Fault Diagnosis Tolerance Cryptogr.*, 2016, pp. 85–96, doi: 10.1109/FDTC.2015.19.
[126] S. Tajik, E. Dietz, S. Frohmann, J.-P. Seifert, D. Nedospasov, C. Helfmeier, C. Boit, and H. Dittrich, "Physical characterization of arbiter PUFs," in *Proc. Int. Workshop Cryptograph. Hardw. Embedded Syst.*, in Lecture Notes in Computer Science, vol. 8731, 2014, pp. 493–509, doi: 10.1007/978-3-662-44709-3_27.
[127] Z. Wu, H. Patel, M. Sachdev, and M. V. Tripunitara, "Strengthening PUFs using composition," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Nov. 2019, pp. 1–6, doi: 10.1109/ICCAD45719.2019.8942176.
[128] P. H. Nguyen and D. P. Sahoo, "An efficient and scalable modeling attack on lightweight secure physically unclonable function," *IACR Cryptol. ePrint Arch.*, vol. 2016, p. 428, Jan. 2016.
[129] 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 Security*, vol. 8, no. 11, pp. 1876–1891, Aug. 2013, doi: 10.1109/TIFS.2013.2279798.
[130] N. Wisiol and M. Margraf, "Why attackers lose: Design and security analysis of arbitrarily large XOR arbiter PUFs," *J. Cryptograph. Eng.*, vol. 9, no. 3, pp. 221–230, Sep. 2019, doi: 10.1007/s13389-019-00204-8.
[131] A. O. Aseeri, Y. Zhuang, and M. S. Alkatheiri, "A machine learning-based security vulnerability study on XOR PUFs for resource-constraint Internet of Things," in *Proc. IEEE Int. Congr. Internet Things (ICIOT)*, Jul. 2018, pp. 49–56, doi: 10.1109/ICIOT.2018.00014.
[132] A. O. Aseeri, Y. Zhuang, and M. S. Alkatheiri, "A subspace pre-learning approach to fast high-accuracy machine learning of large XOR PUFs with component-differential challenges," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2018, pp. 1563–1568, doi: 10.1109/BIGDATA.2018.8621890.
[133] A. Aseeri, "Noise-resilient neural network-based adversarial attack modeling for XOR physical unclonable functions," *J. Cyber Secur. Mobility*, vol. 9, pp. 331–354, Mar. 2020, doi: 10.13052/JCSM2245-1439.926.
[134] K. T. Mursi, B. Thapaliya, Y. Zhuang, A. O. Aseeri, and M. S. Alkatheiri, "A fast deep learning method for security vulnerability study of XOR PUFs," *Electronics*, vol. 9, no. 10, pp. 1–13, 2020, doi: 10.3390/electronics9101715.
[135] N. Wisiol, K. T. Mursi, J. P. Seifert, and Y. Zhuang, "Neural-network-based modeling attacks on XOR arbiter PUFs revisited," *IACR Cryptol. ePrint Arch.*, vol. 2021, p. 555, Sep. 2021.
[136] W. Yu and Y. Wen, "Efficient hybrid side-channel/machine learning attack on XOR PUFs," *Electron. Lett.*, vol. 55, no. 20, pp. 1080–1082, Oct. 2019, doi: 10.1049/el.2019.1363.
[137] G. Li, K. T. Mursi, A. O. Aseeri, M. S. Alkatheiri, and Y. Zhuang, "A new security boundary of component differentially challenged XOR PUFs against machine learning modeling attacks," *Int. J. Comput. Netw. Commun.*, vol. 14, no. 3, pp. 1–15, May 2022, doi: 10.5121/ijcnc.2022.14301.
[138] M. A. Alamro, K. T. Mursi, Y. Zhuang, A. O. Aseeri, and M. S. Alkatheiri, "Robustness and unpredictability for double arbiter PUFs on silicon data: Performance evaluation and modeling accuracy," *Electronics*, vol. 9, no. 5, p. 870, May 2020, doi: 10.3390/electronics9050870.
[139] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Learn. Represent.*, 2015, pp. 1–15.
[140] D. Chatterjee, D. Mukhopadhyay, and A. Hazra, "Interpose PUF can be PAC learned," *IACR Cryptol. ePrint Arch.*, vol. 2020, p. 471, 2020.
[141] G. Li and K. T. Mursi, "A subspace pre-learning strategy to break the interpose PUF," *Electronics*, vol. 11, no. 7, pp. 1–10, 2022, doi: 10.3390/electronics11071049.
[142] J. Delvaux and I. Verbauwhede, "Fault injection modeling attacks on 65 nm arbiter and RO sum PUFs via environmental changes," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 6, pp. 1701–1713, Jun. 2014, doi: 10.1109/TCSI.2013.2290845.

S. HEMAVATHY (Graduate Student Member, IEEE) received the master's degree in VLSI design. She is currently pursuing the Ph.D. degree with VIT University, Chennai, with a focus on hardware security using physical unclonable functions. She has been in the teaching field for ten years. She is a Research Associate with the School of Electronics Engineering, VIT University. She has published in *IEEE Consumer Electronics Magazine*, two international conferences, and a book chapter concentrating on low-power PUF design. Her research interests include low-power VLSI design, adiabatic logic, hardware security in SoC and FPGA, and physically unclonable functions.

V. S. KANCHANA BHAASKARAN (Senior Member, IEEE) received the bachelor's degree in engineering from IE(I), the master's degree from BITS, and the Ph.D. degree from the Vellore Institute of Technology (VIT). With nearly 40 years of experience, she has industry, teaching, and research experiences. After 11 years of teaching at the SSN College of Engineering, she joined VIT Chennai and had been donning the role of the Pro Vice-Chancellor. She has published around 140 papers in international journals and reputed conferences and has three patents published in her field of research. She writes on higher education and the impact of ICT in learning, and delivers invited lectures. She is the author of four books on linear integrated circuits and related subjects published by McGraw Hill Education and has contributed book chapters on low-power VLSI circuit design. Her research interests include VLSI design for low-power, microprocessor architectures, and linear integrated circuits. She is a fellow of the Institution of Electronics and Telecommunication Engineers (IETE) and the Institution of Engineers (India) (IEI). She is a member of the IET and a Life Member of the ISTE. She is a reviewer of international journals and conferences.