Cost-Efficient Design for Modeling Attacks Resistant PUFs

Mohd Syafiq Mispan*†, Haibo Su†, Mark Zwolinski†, Basel Halak†
†Electronics and Computer Science, University of Southampton, United Kingdom
e-mail: {msm1g14,H.Su,mz,bh9@ecs.soton.ac.uk}
*Faculty of Engineering Technology, Technical University of Malaysia Malacca, Malaysia

***Abstract*—Physical Unclonable Functions (PUFs) exploit the intrinsic manufacturing process variations to generate a unique signature for each silicon chip; this technology allows building lightweight cryptographic primitive suitable for resource-constrained devices. However, the vast majority of existing PUF design is susceptible to modeling attacks using machine learning technique, this means it is possible for an adversary to build a mathematical clone of the PUF that have the same challenge/response behavior of the device. Existing approaches to solve this problem include the use of hash functions, which can be prohibitively expensive and render PUF technology as the suitable candidate for lightweight security. This work presents a challenge permutation and substitution techniques which are both area and energy efficient. We implemented two examples of the proposed solution in 65-nm CMOS technology, the first using a delay-based structure design (an Arbiter-PUF), and the second using sub-threshold current design (two-choose-one PUF or TCO-PUF). The resiliency of both architectures against modeling attacks is tested using an artificial neural network machine learning algorithm. The experiment results show that it is possible to reduce the predictability of PUFs to less than 70% and a fractional area and power costs compared to existing hash function approaches.**

Keywords*—Physical Unclonable Function (PUF); Arbiter-PUF; TCO-PUF; Machine Learning; Security

I. Introduction

Identification and authentication are two essentials process in any security field. Designing an electronic system which provides robust security of these two essentials process with low manufacturing cost and low power consumption is really demanding. Examples of applications include Radio-Frequency Identification (RFID) tags for secure access, health and social-security cards, supply chain control and etc. To protect these devices against counterfeiting, they must be identified and authenticated in a secure way. Nevertheless, an application such as RFID tags requires a small footprint and low power consumption. Hence, providing fundamental security services such as authentication and identification introduce a significant challenge.

Physical Unclonable Function (PUF) is an emerging hardware-based security technology which offers a promising solution for an IC identification and authentication with a relatively low cost. PUFs exploit the random intrinsic manufacturing process variations that map a set of challenges to a set of responses. The mapping of challenge-response pairs (CRPs) is device-specific and random to each PUF instance, which make PUFs very promising technology as hardware fingerprinting that can be used as a secure alternative to memory tags and are proposed as an anti-counterfeit security for resource-constraint devices.

Several works, [1], [2], [3] have demonstrated the suitability of Strong-PUFs to provide a lightweight authentication protocol for area and energy constrained platforms such as RFIDs and Internet-of-Things (IoTs). The Strong-PUFs are PUFs with CRPs that grow exponentially as the number of bit challenges increases as in Arbiter-PUF [4], XOR Arbiter-PUF [5] and Lightweight-PUF [6]. Nonetheless, as discussed in [7], ML techniques were able to model aforementioned PUFs with high accuracy. An ML-attack is considered as the most plausible way to attack Strong-PUFs since it offers great advantages in terms of cost efficiency and high prediction accuracy. As a countermeasure, the concept of Controlled-PUF can be used to reduce the vulnerability of Strong-PUFs against ML-attack [8]. However, it introduces significantly large area and power overhead to implement the hash functions and error-correction code (ECC) blocks. Hence, invalidate the suitability of Controlled-PUF for low-cost identification and authentication applications such as RFID tags, as mentioned earlier.

In this study, we focus on such commercial resource-constraint PUF-based RFID tags in which the Strong-PUFs combined with a technique such as permutation and substitution is used to provide a certain level of security against ML-attack. Hence, it is unworthy for the low-financial attackers to spend their resources to counterfeit these products. Our goal is to improve the security of Strong-PUFs against an ML-attack and significantly achieves low-cost implementation. As a case study, we evaluate and perform analysis on Arbiter-PUF and TCO-PUF.

The main contributions of this work are:

1) We show that using a challenge permutation technique can alter the output transition probability of Arbiter-PUF, resulted in an increase of the resiliency against an ML-attack.
2) We also show an iterative process to obtain a random challenge permutation that produces a certain level of unpredictability by controlling the behavior of the output transition probability of Arbiter-PUF.
3) We further show that a challenge substitution technique can increase the resiliency of both Arbiter-PUF and TCO-PUF against ML-attack. Both PUFs achieved a prediction accuracy of less than 65%.

The rest of the paper is organized as follows. Section II describes the background which related to this work. Section

III describes the CRPs generation, the ML algorithm and the threat model used in our work. The analysis of an ML-attack is presented in Section IV. Finally, conclusions are drawn in Section V.

II. Background

A. Related Work

ML-attack resistance describes the complexity of the challenge to response mapping for a particular PUF. Arbiter-PUF and its derivative such as XOR Arbiter-PUF, Lightweight-PUF have been shown to be susceptible to ML-attack [7]. According to [7], one might disable the ML-attack by implementing the output network of XOR Arbiter-PUF and Lightweight-PUFs with 8 XORs or more. Nevertheless, the reliability of the PUF worsens as the number of XOR increases. The low reliability might result in device authentication errors such as false positive and false negative [9]. Besides, the XORing technique increases the area overhead due to the parallel architecture. The concept of Controlled-PUF is introduced by Gassend *et al.*, [8] which uses a secure one-way function to increase the complexity of the challenge to response mapping in a Strong-PUF. For any attacker that only has access to the interface of the device, the Controlled-PUF successfully disables an ML-attack. Nevertheless, a one-way hash function is too costly for resource-constraint devices as shown in Table I.

TABLE I. Area and power of hash function [10].

Type	Area [GE]	Power [mW]
SHA-1	9567	1.256
SHA-256	12980	1.688

Several works focus on how to increase the resilience of Arbiter-PUF against an ML-attack. In recent studies, Ye *et al.* proposed an obfuscation logic based PUF (OPUF) [11] and randomized challenge PUF (RPUF) [12] to increase the resiliency of Arbiter-PUF against ML-attack. However, an OPUF suffers reliability issue as described above since it adapted XOR network technique at the output stage and RPUF has a potential issue of bias in random number generator (RNG), use to randomize the challenge. Elsewhere, Gao *et al.*, [2] proposed an Obfuscated-PUF (OB-PUF) in which a partial challenge is sent by the verifier to the OB-PUF (i.e., the prover). Subsequently, within an OB-PUF, a partial challenge is padded with a random pattern generated by RNG to make up a full-length challenge. Earlier, Rostami *et al.*, [1], proposed a sub-string matching technique in which only a subset of PUF response strings is sent to the verifier during authentication. Generally, both works, [2], [1], use the same technique by only exposing a subset of either challenges or responses. However, this might increase the authentication time to run the matching algorithm, as well as the area, on the verifier side. One might argue, however, that the area is not a concern since the verifier has always been assumed to be resource rich. In our work, we explore a lightweight technique which is a challenge permutation on Arbiter-PUF.

B. Arbiter-PUF and TCO-PUF Architectures

The architecture of Arbiter-PUF and TCO-PUF have been described in details in [13], [10], and are omitted for the sake of brevity. We refer k as the bit challenge length for both architectures throughout our discussion unless otherwise stated.

III. Methodology

In this section, we briefly discuss the simulation setup for CRP generation, the ML algorithm, and the threat model used in this study.

A. CRP Generation

An Arbiter-PUF and TCO-PUF have been implemented using a 65-nm technology node and they are simulated using the BSIM4 (V4.5) transistor model with a nominal supply voltage of 1.2V and a room temperature of 25^oC. Intrinsic variations such as effective length, effective width, oxide thickness and threshold voltage are modeled in Monte Carlo simulations using the built-in fabrication standard statistical variation (3σ variations) in the technology design kit. For a CRP generation, an arbitrary challenge can be applied on both implemented PUFs to generate its corresponding response. A total of 32000 CRPs have been generated for ML-attack analysis.

B. Machine Learning

An Artificial Neural Network (ANN) has been considered in our study since it is known to have the capability of modeling highly non-linear systems. A feed-forward network with multilayer perceptron is required for solving non-linear problems and therefore has been chosen in our study. A resilient back-propagation has been chosen as the best training algorithm considering the prediction accuracy and fast convergence time. The test data for prediction accuracy computation is not part of the training data. ANN has been implemented using a built-in function in MATLAB.

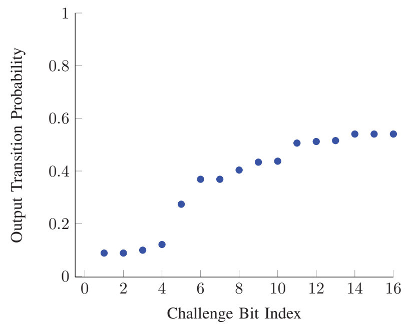
C. Threat Model

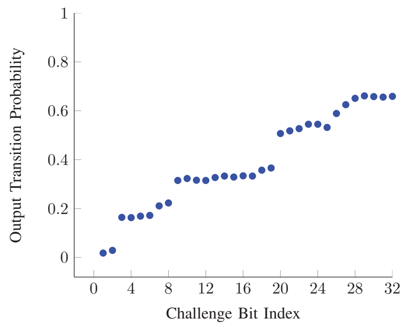
An adversary may have several motives for attacking deployed low-end devices, such as gaining secure access through RFID, or to counterfeit products. In order to achieve one of the adversary goals, we assume that the attacker is a low-financial attacker and restricted to a non-invasive CRPs measurement. The attacker only has an access to the primary interfaces of the PUF-based RFID tag, and can apply a polynomial number of challenges to the device and then collect the corresponding responses. Subsequently, the attacker tries to derive a numerical model from the CRPs data by using ML techniques, as described in Section III-B.

IV. Analysis

A. Challenge Permutation Technique

1) Output Transition Probability: According to [14], an Arbiter-PUF with k-bit challenge and a 1-bit response is said to satisfy strict avalanche criteria (SAC) if its output transition occurs with a probability of 0.5 whenever a single challenge bit


(a) $k = 16$


(b) $k = 32$

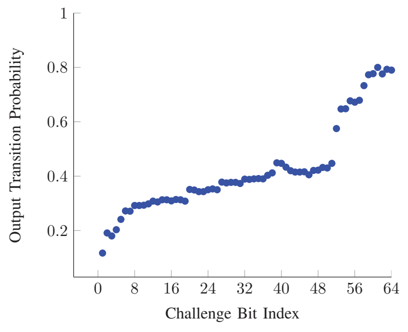

(c) $k = 64$

Fig. 1. Output transition probability for k-bit Arbiter-PUF

is complemented. The output transition probability for Arbiter-PUF can be estimated as described in Algorithm 1. The CRPs generation in step c) follows the setup as explained in Section III-A.

Algorithm 1 Computation of Output Transition Probability, $N = 1000$.

```
Require:
    a) Generate $N$ random challenges, $\mathbf{c}$.
    b) Generate $N$ challenges, $\hat{\mathbf{c}}$ from $\mathbf{c}$ with HD=1 for every challenge bit index.
    c) Simulate $k$-bit Arbiter-PUF using a TSMC 65-nm technology node.
    d) Collect CRPs for $\mathbf{r} \leftarrow PUF(\mathbf{c})$ and $\hat{\mathbf{r}} \leftarrow PUF(\hat{\mathbf{c}})$
Ensure:
    Value of output transition probability
 1: $count \leftarrow 0$
 2: for index= 1 to $k$ do
 3:     for $i = 1$ to $N$ do
 4:         if $\mathbf{r} \neq \hat{\mathbf{r}}$ then
 5:             $count \leftarrow count + 1$
 6:         end if
 7:     end for
 8:     probability[index] $\leftarrow count/N$
 9:     $count \leftarrow 0$
10: end for
```

Figure 1 shows the computed output transition probability for $k = \{16, 32, 64\}$. A challenge bit index in the x-axis is referring to a bit position of the challenge which was flipped. As can be seen from Figure 1, for all Arbiter-PUFs, they produce a generic pattern where the output transition probability increases as the challenge bit index arising from 1 to k. It can be observed that the probability values for the first index is significantly poor, very close to zero. Based on this observation, given a set of CRPs, one can simply generate another set of CRPs by merely flipping the first-bit position of all the challenges where the corresponding responses remain the same. It can be concluded that an Arbiter-PUF does not fulfills SAC, resulted in CRPs mapping can be predicted easily using ML technique. Next, the impact of challenge permutation on output transition probability and the correlation with ML-attack prediction accuracy will be discussed.

2) Challenge Permutation Technique: For a given size of Arbiter-PUF, the challenge permutation space is extremely huge. As a starting point, we analyze the permutation as shown in Figure 2 and investigate the impact on the output transition probability. k is the bit-length of the challenge and n is the length of the bit challenge that gets combined. The value of n is a power of two. Afterwards, this permutation scheme is known as an n-block permutation.

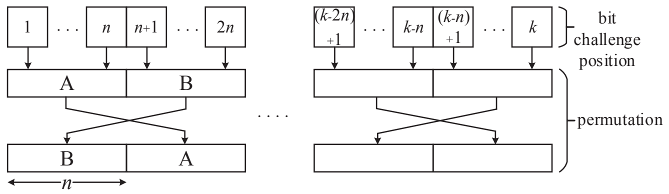


Fig. 2. n-block permutation scheme.

We applied the n-block permutation on generated CRPs used to compute the output transition probability in Figure 1. Next, the output transition probability based on n-block permutation is computed using Algorithm 1 and the values are illustrated in Figure 3[1] for 32-bit Arbiter-PUF. As can be seen from Figure 3, the probability for $n = \frac{k}{2}$ is essentially the probability in Figure 1 in which the corresponding probability value of index 1 to $\frac{k}{2}$ and index $(\frac{k}{2}+1)$ to k has been swapped over. It similarly applies to the rest of n values. A similar pattern has been observed for $k = \{16, 64\}$, hence, they are not shown for brevity.

Next, an n-block permutation is applied on 32000 randomly generated challenges. Subsequently, the permutated challenges are applied on Arbiter-PUF to generate corresponding responses as described in Section III-A. The ML-attack is performed on the CRPs and the prediction accuracy is listed in Table II. For each Arbiter-PUF configuration, as n reduces, the predictability of the response reduces. For small n, e.g., 2-block, as the Arbiter-PUF size grows the prediction accuracy is further reduces. Re-examine Figure 3, the output transition probability tends to fluctuate as n reduces and as a consequence, the original probability pattern (Figure 1) is not being preserved. Indeed, this is correlated with the findings in Table II in which the fluctuation in the output transition probability indicates that the CRPs mapping has been disordered and low prediction accuracy could be achieved.

[1]Challenge bit index is a discrete value. Figure 3 has been plotted with continuous lines, however for better visibility.

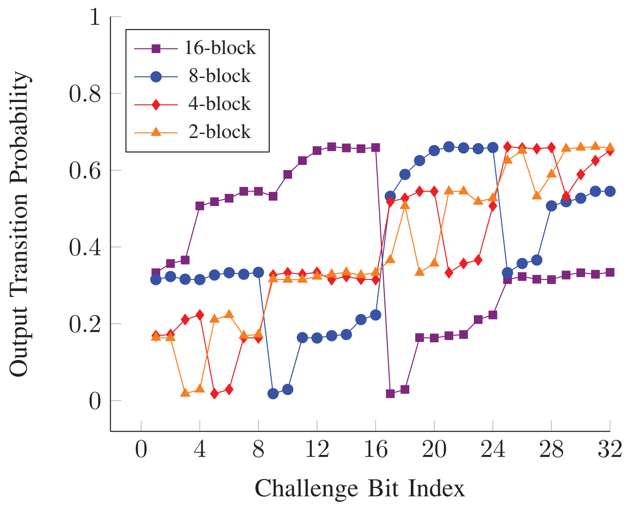


Fig. 3. Output transition probability for 32-bit Arbiter-PUF with n-block permutation.

TABLE II. Prediction Accuracy of k-bit Arbiter-PUF with n-block Permutation Scheme.

Permutation	k=16	k=32	k=64
	Prediction Accuracy [%]		
32-block	NA	NA	99.27
16-block	NA	99.35	91.58
8-block	99.46	95.23	87.79
4-block	96.38	91.77	85.31
2-block	94.77	91.58	83.25

3) Random Challenge Permutation Generation: In order to maximize the fluctuation in the output transition probability, the generic pattern of the output transition probability as in Figure 1 must not be preserved. To achieve this, we define two set of conditions as below:

Condition 1: $| Pr(\text{index-1}) - Pr(\text{index}) | \leq \Delta\text{max}_k$

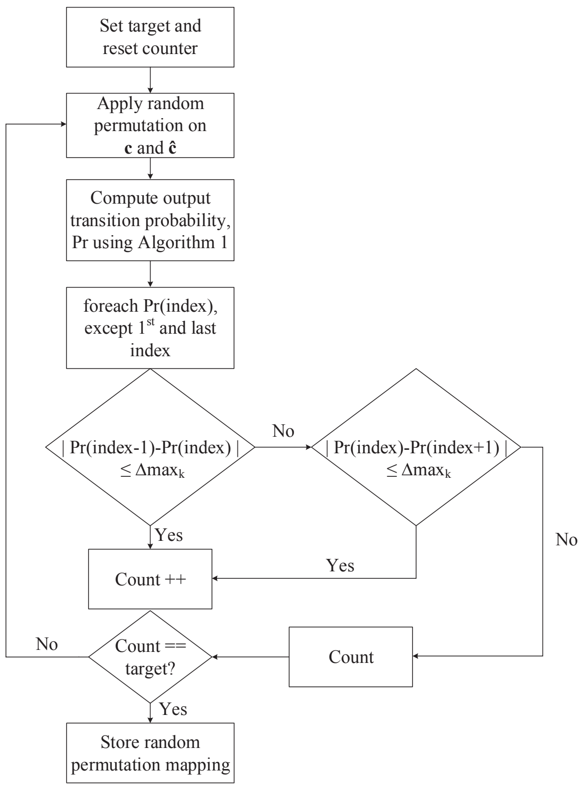
Condition 2: $| Pr(\text{index}) - Pr(\text{index+1}) | \leq \Delta\text{max}_k$

where Pr is the output transition probability. Above conditions are not applicable for the very first and last bit challenge positions. A parameter called Δmax_k is introduced to constraint both conditions which can be calculated as below:

$$\mu_k = \sum_{\text{index}=1}^{k-1} \frac{|Pr(\text{index}+1) - Pr(\text{index})|}{k-1} \tag{1}$$

$$\Delta\text{max}_k = \mu_k + \sigma_k \tag{2}$$

where μ_k is the average of the difference between two consecutive output transition probability and σ_k is its standard deviation. Both conditions must be avoided to ensure the fluctuation in the output transition probability is maximized.



Set target and reset counter
Apply random permutation on c and ĉ
Compute output transition probability, Pr using Algorithm 1
foreach Pr(index), except 1st and last index
| Pr(index-1)-Pr(index) | ≤ Δmax_k
No
| Pr(index)-Pr(index+1) | ≤ Δmax_k
Yes
Count ++
Yes
No
No
Count == target?
Count
Yes
Store random permutation mapping

Fig. 4. Iteratively finding a random challenge permutation mapping.

Figure 4 shows the iterative process of finding a good random challenge permutation mapping which maximizes the fluctuations in the output transition probability based on the conditions explained above. The target in Figure 4 indicates the number of occurrence of both conditions in the output transition probability. Given an Arbiter-PUF with k-bit challenge length, the maximum occurrence of both conditions is $k-2$. To demonstrate the correlation between the occurrence of both conditions and ML prediction, the target is varied within the range of $0 \leq \text{target} \leq k-2$. Once the required challenge permutation mapping is found, it is applied to 32000 randomly generated challenges and their corresponding responses are generated following a methodology as described in Section III-A. Subsequently, the ML-attack is performed on 32000 CRPs for 16-bit, 32-bit and 64-bit Arbiter-PUF. Figure 5 illustrated the correlation between the ML prediction and the occurrence of both conditions in the output transition probability for aforementioned Arbiter-PUFs. Clearly, it shows that as the occurrence of condition 1 and 2 reduces, the prediction accuracy reduces. The lowest prediction accuracy that we could achieve for 16-bit, 32-bit and 64-bit are 86%, 65.13% and 69.04%, respectively.

Generally, for an explicit frequency of occurrence (e.g., 10), it can be observed that as k increases, the prediction accuracy reduces. Following this observation, let consider a worst case condition where given a k-bit challenge length with a total number of 1's is given as $i = 1$ (at first index)

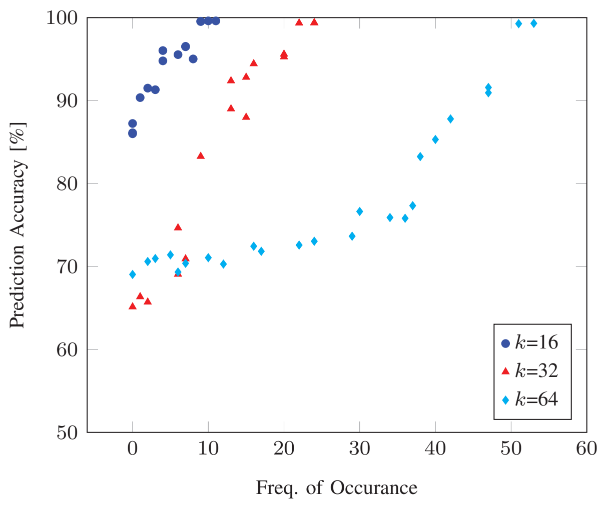


Fig. 5. Correlation between the ML prediction and the occurrence of condition 1 and 2 for k-bit Arbiter-PUF.

and $(k-i)$ 0's, a total unique permutation can be computed as $\binom{k}{i} = {}^kC_i$. As mentioned in Section IV-A1, without the challenge permutation technique, by flipping the first index or bit challenge position, another set of CRPs can be derived easily because the output transition probability is almost zero. However, with a challenge permutation technique, the output transition probability has now $\binom{k}{i} = {}^kC_i$ possible values. As k increases, the possible values of output transition probability increases accordingly and increase the resilient of Arbiter-PUF against ML-attack.

All of the above shows the principle of a challenge permutation technique which has been applied on Arbiter-PUF. Based on our observation, however, this technique is not effective for TCO-PUF. Hence, a challenge substitution technique is explored and will be discussed in the next section.

B. Challenge Substitution Technique

In this section, we explore a challenge substitution technique to reduce the mapping correlation between the challenge and response of the Arbiter-PUF and TCO-PUF. A standard Rijndael substitution box (S-box) which based on advanced encryption standard (AES) cryptographic algorithm is utilized to implement a challenge substitution technique. The S-box is generated using a built-in function in MATLAB. A 32000 randomly generated challenges is applied on the S-box to generate the substituted challenges, which further applied on Arbiter-PUF and TCO-PUF to generate the corresponding responses, as described in Section III-A. The ML-attack is performed on the 32000 CRPs and Figure 6 shows the resiliency of both PUFs against the ML-attack. Without a challenge substitution technique, as the amount of CRPs used for training increases, the prediction accuracy increases and achieved more than 95% predictability at 30000 CRPs for both PUFs. However, with a challenge substitution technique, the predictability of Arbiter-PUF and TCO-PUF reduce to 66.4% and 62.5%, respectively at 30000 CRPs.

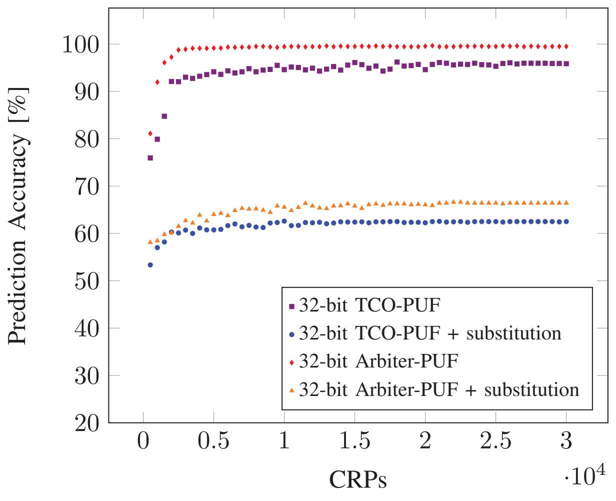
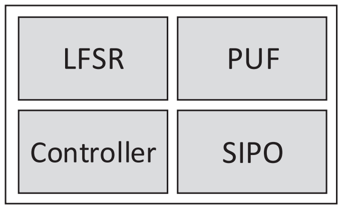


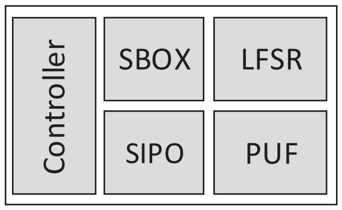
Fig. 6. ML-attack for 32-bit Arbiter-PUF and TCO-PUF with and without a challenge substitution technique.

C. Hardware Implementation

In this section, we present a hardware implementation of the PUF configuration for both permutation and substitution techniques. Figure 7 shows the top level architecture of a PUF to generate m-bit of response which consists of a controller, k-bit LFSR, and m-bit serial-in and parallel-out (SIPO) register, and S-box.



(a) Challenge permutation.



(b) Challenge substitution.

Fig. 7. Top level functional unit blocks.

As can be seen in Figure 7a, there is no functional unit block to implement the challenge permutation technique. We argue that this technique can be implemented by the routing obfuscation between challenge interface and internal ports of LFSR unit block. Hence, a challenge permutation technique introduces no additional logic gates. As for the challenge substitution technique, an S-box has been implemented using a compact arithmetic, following [15]. The area and power have been estimated using Synopsys Design Compiler (DC) and listed in Table III. However, TCO-PUF is not included in Table III as the area and power for analog circuit could not be estimated using Synopsys DC.

TABLE III. Area and Power Estimation for m=128.

Technique	k-bit Arbiter-PUF	Area [GE]	Power [mW]
Permutation	k=32	1130	0.2492
	k=64	1512	0.3949
Substitution	k=32	1445	0.3801
	k=64	1787	0.4467

D. Predictability Comparison

Our findings in Section IV-A and IV-B are summarized in Table IV and compared with the previously reported data. With a challenge permutation and substitution techniques, the resistant of Arbiter-PUF against ML-attack increases, better than XOR Arbiter-PUF and Voltage Transfer Characteristic PUF (VTC-PUF), and approximately the same as Current Mirror-PUF and Lightweight OB-PUF. Although a challenge permutation technique is not effective for TCO-PUF, the ML-attack resistant of TCO-PUF increases with a substitution challenge technique and achieved approximately similar performance as Arbiter-PUF with the proposed techniques. As mentioned in Section II-A, Controlled-PUF which uses a one-way hash function makes the input-output relations of PUF unpredictable and therefore, disables the ML-attack [7]. For an area comparison, two hash functions (SHA-1) in Controlled-PUF configuration as in [8] already consumed 19134 GE. Clearly, our proposed techniques have much less area and power - at least 10.71× and 5.62× smaller, respectively

TABLE IV. Comparison of Prediction Accuracy.

Type	CRPs	Prediction Accuracy [%]
32-bit Arbiter-PUF + proposed permutation	3×10^4	65.13
64-bit Arbiter-PUF + proposed permutation	3×10^4	69.04
32-bit Arbiter-PUF + proposed substitution	3×10^4	66.40
32-bit TCO-PUF + proposed substitution	3×10^4	62.50
4-XOR 64-bit Arbiter-PUF [7]	3.9×10^4	99
4-XOR 64-bit Lightweight-PUF [7]	1.2×10^4	99
80-bit Current Mirror-PUF [16]	3×10^4	≈68
Lightweight OB-PUF [2]	2×10^4	63.27
64-bit RPUF [12]	2×10^2	69.1
64-bit VTC-PUF [17]	3×10^4	≈75
Controlled-PUF [8]	NA	unpredictable [7]

V. Conclusion

Providing security such as authentication for resource constraint devices is a significant challenge. Strong-PUFs are a promising technology to provide low-cost authentication for pervasive devices. However, the susceptibility to ML-attack is still a major concern. In this paper, we have proposed a challenge permutation and substitution techniques to increase the ML-attack resistant of Strong-PUFs. Using these techniques, we are able to reduce the predictability of Arbiter-PUF and TCO-PUF responses to less than 70%. Our proposed techniques consume less than 2000 GEs for a 128-bit identifier. Hence, this technique is suitable for lightweight security devices.

VI. Acknowledgments

The authors would like to thank Ministry of Education Malaysia (MOE) and Technical University of Malaysia Malacca (UTeM) for financial support.

References

[1] M. Rostami, M. Majzoobi, F. Koushanfar, D. Wallach, and S. Devadas, "Robust and reverse-engineering resilient PUF authentication and key-exchange by substring matching," *IEEE Transactions on Emerging Topics in Computing*, vol. 2, pp. 37–49, 2014.

[2] 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 *IEEE International Conference on Pervasive Computing and Communication Workshops*, 2016, pp. 1–6.

[3] B. Halak, Y. Hu, and M. S. Mispan, "Area efficient configurable physical unclonable functions for FPGAs identification," in *IEEE International Symposium on Circuits and Systems*, 2015, pp. 946–949.

[4] D. Lim, J. W. Lee, B. Gassend, G. E. Suh, M. V. Dijk, and S. Devadas, "Extracting secret keys from integrated circuits," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 13, no. 10, pp. 1200–1205, 2005.

[5] G. E. Suh and S. Devadas, "Physical Unclonable Functions for device authentication and secret key generation," in *ACM/IEEE Design Automation Conference*, 2007, pp. 9–14.

[6] M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Lightweight secure PUFs," in *IEEE/ACM International Conference on Computer-Aided Design*, 2008, pp. 670–673.

[7] 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 Transactions on Information Forensic and Security*, vol. 8, pp. 1876–1891, 2013.

[8] B. Gassend, M. van Dijk, D. Clarke, E. Torlak, and S. Devadas, "Controlled physical random functions and applications," *ACM Transactions on Information and System Security*, vol. 10, no. 4, pp. 15:1 –15:22, 2008.

[9] A. Maiti and P. Schaumont, "The impact of aging on a physical unclonable function," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 9, pp. 1854–1864, 2014.

[10] M. S. Mispan, B. Halak, and M. Zwolinski, "Lightweight Obfuscation Techniques for Modeling Attacks Resistant PUFs," in *IEEE International Verification and Security Workshop*, 2017, pp. 19–24.

[11] J. Ye, Y. Hu, and X. Li, "OPUF: Obfuscation logic based physical unclonable function," in *IEEE International On-Line Testing Symposium*, 2015, pp. 156–161.

[12] ——, "RPUF: Physical unclonable function with randomized challenge to resist modeling attack," in *IEEE Asian Hardware Oriented Security and Trust Symposium*, 2016, pp. 1–6.

[13] M. S. Mispan, B. Halak, Z. Chen, and M. Zwolinski, "TCO-PUF: A subthreshold physical unclonable function," in *IEEE PRIME*, 2015, pp. 105–108.

[14] P. H. Nguyen, D. P. Sahoo, R. S. Chakraborty, and D. Mukhopadhyay, "Security analysis of Arbiter PUF and its lightweight compositions under predictibility test," *ACM Transactions on Design Automation of Electronic Systems*, vol. 22, no. 2, pp. 1–28, 2016.

[15] L. Kuang, "Lightweight design for PUF resilient to model building attacks," MSc. Thesis, University of Southampton, 2017.

[16] R. Kumar and W. Burleson, "On design of a highly secure PUF based on non-linear current mirrors," in *IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, 2014, pp. 38–43.

[17] A. Vijayakumar and S. Kundu, "A novel modeling attack resistant PUF design based on non-linear voltage transfer characteristics," in *Design, Automation & Test in Europe Conference & Exhibition*, 2015, pp. 653–658.