Side-Channel Resistant Implementation Using Arbiter PUF

RajaAdhithan Radhakrishnan, Suganya Annadurai
r.rajaadhithan@gmail.com, and suganya.annadurai@gmail.com
Society For Electronic Transactions and Security(SETS)

Abstract. The goals of cryptography are achieved using mathematically strong crypto-algorithms, which are adopted for securing data and communication. Even though the algorithms are mathematically secure, the implementation of these algorithms may be vulnerable to side-channel attacks such as timing and power analysis attacks. One of the effective countermeasures against such attacks is Threshold Implementation(TI). However, TI realization in crypto-device introduces hardware complexity, so it shall not be suitable for resource-constrained devices. Therefore, there is a need for efficient and effective countermeasure techniques for resource-constrained devices. In this work, we propose a lightweight countermeasure using an Arbiter Physical Unclonable Function (A-PUF) to obfuscate intermediate values in the register for rolled and unrolled implementation of Advanced Encryption Standard (AES). The countermeasure is realized in rolled (iterative) implementation of AES in a 65nm Field Programmable Gate Array (FPGA). We have analyzed the security strength and area of the obfuscated AES using A-PUF and compared it with conventional (rolled AES) and masked TI of AES. Further, we have illustrated the effectiveness of pre-charge and neutralizing countermeasures to strengthen the side channel resistance. We have discussed the complexity of mounting a side-channel and modeling attacks on obfuscated AES using A-PUF.

Keywords: Side-channel attacks · Countermeasures· obfuscating· PUF· AES· Masking· TI· TVLA· Pre-charge· Neutralizing.

1 Introduction

To safeguard highly valuable data, encryption is the technique used to convert the data into an unreadable format using a key. Advanced Encryption Standard (AES) [1] is the most popular encryption algorithm used in many applications. In [2], differential power analysis side-channel attack is demonstrated on AES implementation to retrieve the secret key. Various countermeasure techniques such as masking [3, 4], hiding [5], and leakage resilient design had been proposed to increase the complexity of the attack. However, higher-order attacks such as probing attacks had been demonstrated against few masking scheme [6], which reveals the secret key from masked implementation. Though the countermeasures

increase the complexity of the side-channel attack, many of these countermeasures rely on the security of a random number generator, which provides random mask values.

In [7] Threshold Implementation (TI) had been proposed as a provable solution to increase the security strength of AES. But the area overhead of TI is thrice that of the naive implementation. Since the emerging of new technologies such as the Internet of Things(IoT) [8] requires lightweight cryptography solutions for protecting its data, TI may not be suitable for ubiquitous devices. On the other hand, Quantum technologies [9] and Artificial intelligence(AI) are focusing on improving computational resources to reduce the attack complexity. This alarms the need for a secure encryption algorithm with a lightweight countermeasure. To fulfill the need, we propose an obfuscation technique using Physical Unclonable Function (PUF) [10] to prevent side-channel attacks. PUF generates the Unique nature of chips like biometrics for humans due to manufacturing process variation. Depending upon the Challenge Response Pairs (CRPs) space, it can be classified into strong PUF and weak PUF. Strong PUF has large CRPs Eg: Arbiter based PUFs, which is used for device authentication. Wherein, weak PUF has fewer CRPs Eg: Ring Oscillator(RO) PUF, and Static Random Access Memory(SRAM) PUF. Weak PUF is used for key generation, seed for Pseudo Random generator(PRNG), secure key storage, obfuscation, and device identification [13]. Further, PUF side-channel attacks, modeling attacks, and their countermeasures are detailed in [11] [12][30][31][32]. In this paper, our contributions are as follows:

- We proposed the lightweight countermeasure using Arbiter PUF to prevent the side channel attack
- The countermeasure is realized in rolled (iterative) implementation of Advanced Encryption Standard (AES) in 65nm Field Programmable Gate Array (FPGA) platform.
- We have analyzed the security strength and area of the obfuscated AES using A-PUF and compared it with conventional (rolled AES) and masked Threshold Implementation (TI) of AES.
- Further, we have illustrated the effectiveness of pre-charge and neutralizing countermeasures to strengthen the side-channel resistance of the proposed countermeasure in a noise-free environment.

The rest of the paper has organized as follows. Section 2 describes the Side-channel overview. Section 3 discusses obfuscated countermeasure using Arbiter PUF. Section 4 describes the implementation of obfuscating countermeasures in rolled and unrolled AES and experimental results. Section 5 discusses the pre-charge and neutralizing countermeasures to boast up the Side channel resistance of obfuscated rolled AES using A-PUF. We conclude the paper in Section 6.

2 Side-channel attack overview

Implementation attacks are classified into three types [14] 1. Non-invasive attack. 2. Semi-invasive attack. 3. Invasive. Side-channel attack comes under non-

invasive attack, where the attacker observes the normal behavior of the device and exploits the physical characteristics of the device for key extraction without destructing the device. Eg: Through Power or Electro-Magnetic (EM) emission from the device. Power and EM attack mechanisms are similar, but there is an extra advantage on EM, which targets a particular portion of the device. Power analysis is again classified into Simple power analysis[15] and Differential power analysis. Simple power analysis uses one or a few samples to attack the device.

2.1 Differential power analysis:

Differential power analysis is a technique used to retrieve secret information by finding the correlation between the hypothetical power model vs original power or EM traces. A hypothetical power model is nothing but calculating the switching activities of the register from the input or output data for all possible keys.

Power measurement from the device is depicted in Fig 1. The input for the cryptography algorithm in the device is given from the PC and the corresponding power consumption of the device is captured by the PC via oscilloscope. The captured power traces are then compared with the modeled power consumption to retrieve the secret key.

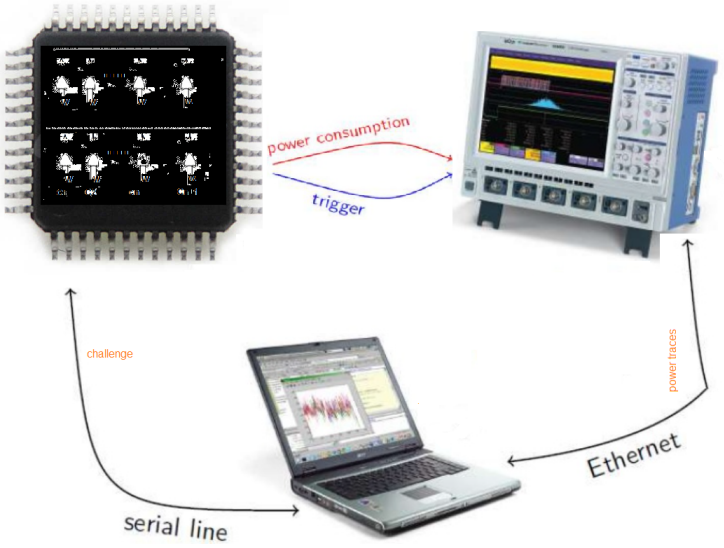

Fig. 1. Power analysis based SCA

The process of key retrieval depends on the power model, which in turn depends on the architecture of the device. The power model is classified into two types 1. Hamming weight power model 2. Hamming distance power model

Hamming weight power model: Traditionally, the Hamming weight of a value is the number of non-zeroes. For example, in the binary number 1100 0010 the Hamming weight would be 3. For this power model, the input and output of the function are stored in an accumulator/register after refreshing the previous value in the accumulator/register as shown in Fig 2. The attacker mostly targets this power model in a micro-controller device, since the accumulator resets its value to zero for every operation. To compute this power model, we need to calculate the number of one's in the accumulator/register.

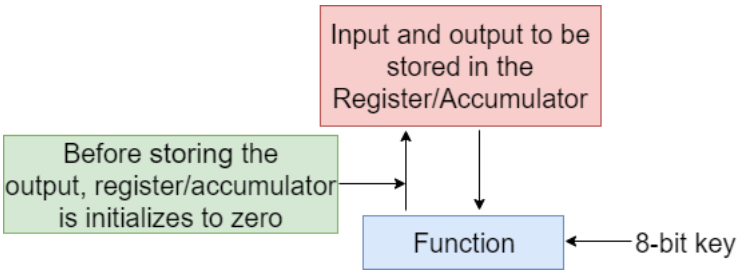

Fig. 2. Hamming weight power model

Hamming distance power model: Hamming distance is a metric for comparing two binary data strings. While comparing two binary strings of equal length, Hamming distance is the number of bit positions in which the two bits are different. For this power model, the input and output of the function are stored in a register without refreshing a previous value in the register as shown in Fig 3. The attacker mainly targets this power model for devices like FPGA and ASIC. To compute this power model, we need to calculate the hamming weight of the hypothetical intermediate value from the previous state of output with the current state of the output of a function.

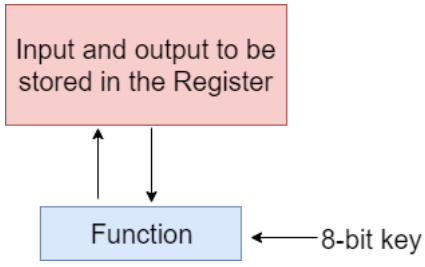

Fig. 3. Hamming distance power model

2.2 General countermeasures

There are three popular types of existing countermeasures that can be implemented to prevent the side-channel attack hiding, masking, and leakage resilience.

Hiding This type of countermeasure tries to make it difficult to measure the leakage by adding electronic noise, shielding to prevent electromagnetic emission, and dummy logic insertions [16]. However, a disadvantage of hiding is, that often if physical access to the device is granted, shielding can be removed, adding more traces to the attack to overcome noise as well as dummy operations.

Leakage Resiliency Leakage resiliency is nothing but key cycling. The device changes the key frequently, which is derived from a deterministic mechanism. This reduces the number of traces required to attack the particular key in the device. The disadvantage of this technique includes changing the cryptography protocol for a system and device performance will be severely impacted. Due to these facts, the technique only adds up the attack complexity [17].

Masking In masking the goal is to break the statistical significance of the algorithm from the values processed in the algorithm such as the key or plain text. One can split the sensitive information into two shares for example. Operations are then performed on the shares individually. These implementations however are subject to higher order attacks[18].

In this paper, we proposed a lightweight countermeasure using Arbiter PUF, which is efficient in terms of area overhead and verified to be secure using TVLA. In the next section, we discuss the testing methodology for protected implementation.

2.3 Testing methodology for side channel evaluation

Evaluation style testing and conformance style testing are the two testing methods to evaluate the side-channel leakage in cryptography circuits. Evaluation style testing is performed by evaluating the circuit for every attack model. Whereas, conformance style testing gives the amount of leakage of the circuit for successful key extraction. This test is called Test Vector Leakage Assessment (TVLA) [19]. In this work, we performed both testing methods to evaluate our proposed countermeasure. We discuss the secure lightweight countermeasure using PUF in the next section.

3 Proposed countermeasure using PUF

3.1 PUF

The circuits have the same functionality that uses the manufacturing process variation to generate the unique nature of ICs. The circuit is termed PUF (the

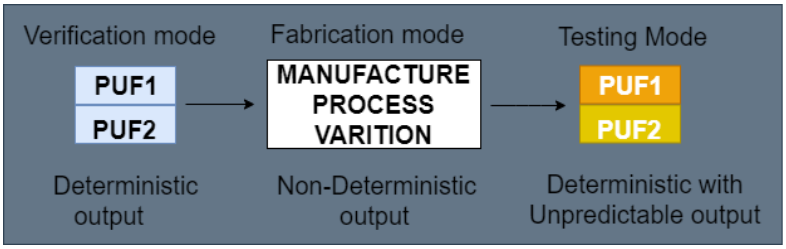

Fig. 4. PUF

unique nature of ICs is used as a digital fingerprint for ICs, like biometrics for humans)

For example, consider two PUFs, PUF1 and PUF2 with the same functional design, In verification mode, PUF1 and PUF2 output are the same, But after fabrication mode PUF1 and PUF2 output are different. The difference in output occurs due to manufacturing process variation during fabrication mode as shown in Fig 4. Depending upon the implementation, PUF is classified into weak PUF and strong PUF. Strong PUF has a large number of challenge-response pairs. It is used for device authentication and identification. Eg: Arbiter-based PUF [10]. Week PUF has a small number of challenge-response pairs (sometimes no or one fixed challenge). This is used for key generation, and seed for PRNG. Eg: Memory-based PUFs [10] [12].

3.2 Proposed countermeasure using PUF

Differential power analysis is successful when the attacker is able to guess the switching activities of the registers in a circuit depending on the key variation. The main objective is to obfuscate the switching activities of the register. This could be possible by obfuscating the data with random numbers. To retrieve the original data, the same random number value is used with obfuscated data stored in the register as shown in Fig 5.

For every data value, there is a need for a new random number to obfuscate the switching activity. Generation of random numbers for every data is a challenging process. Usually, a random number generator is constructed using a Linear feedback shift register(LFSR), but it requires a random seed to operate securely. Again, the LFSR depends on the true random number generator (TRNG), which in turn adds a new circuit to the device. In addition, TRNG has to be evaluated properly. To overcome this issue and strengthen the process, PUF is an emerging technology to resolve this problem. PUF can be used in two ways for generating a random number. The first way is the PUF response can be used in a deterministic random number generator as a seed. The second way is to use strong PUF for generating the random number as illustrated in [20]. This section discusses different ways of generating a random number using PUF.

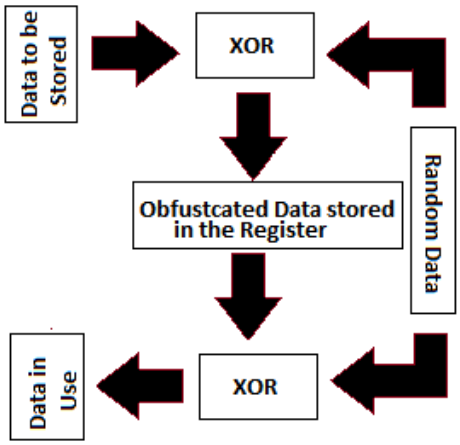

Fig. 5. Storing the obfuscated data in register to prevent side-channel attack

TRNG using Ring Oscillator Ring Oscillators (RO) are used as the source of randomness for TRNGs [21, 22]. In 2007, Sunar et al. used many distinct ring oscillators arranged on a chip [23] for TRNG. In Fig 6, it is shown that each

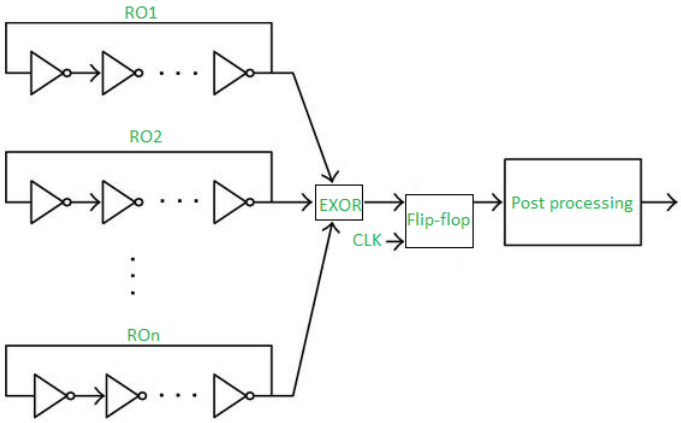

Fig. 6. RO based TRNG

ring oscillator is composed of some inverter. The ring oscillators' outputs are fed to an XOR tree and then sampled at a regular clock frequency to generate a true random number generator using manufacturing process variation.

TRNG using Arbiter PUF An A-PUF is composed of two identically configured delay paths that are stimulated by a clock signal. It is constructed by using a sequence of switch components (i.e., pair of 2-to-1 multiplexers). Each o interconnects have two input ports to two output ports with straight or crossed configurations depending on the applied challenge bit (0 or 1). The output ports of the last stage are connected to an Arbiter, which determines which signal arrived first. Based on this result, the Arbiter generates an unpredictable single-bit response. To generate the random number from Arbiter PUF, the input can be fed through the shift register as discussed in [20]. Since LFSR and hash-based

Table 1. Different ways of constructing random number generation and its comparison

Different random number Generation	Area	Area with fault tolerant	Seeding	Speed	Security
LFSR based [20]	Low	Medium	Required	High	Low
Hash Based	High	High	Required	VeryHigh	Medium
RO's Based [24][25] [20]	low	Medium	Not Required	Medium	High
Arbiter Based [24][25] [20]	low	Low	Not Required	High	High but varies with application

random number generators are deterministic and their security depends on the random seed and underlying Boolean functions. Ring oscillator-based TRNG is not deterministic, but may be vulnerable after some period due to aging effects. Arbiter PUF is quite flexible to control and generate an unpredictable random number with less area to reach high security as shown in Table 1, we have chosen Arbiter PUF to generate an unpredictable random number to obfuscate the switching activity of register.

Two ways of generating a random PUF data for proposed countermeasure using Arbiter PUF

1. In the first method, a single Arbiter PUF is implemented to generate an unpredictable one-bit response. The one-bit output is fed to the shift register to create n-bit random PUF data to obfuscate the data to be stored. To retrieve the data, the reverse procedure is followed with the same random PUF data as shown in Fig 7. Depending upon the application, Data to be stored or random PUF data can be used as challenges for Arbiter PUF. To strengthen the attack complexity, we can fix the random number in the register using weak PUF or feed the input through a counter to A-PUF and generate random bits initially.
2. In the second method, we can create (instantiate) n-number of A-PUF to generate n-bit random data in parallel as shown in Fig 8. The rest of the procedure remains the same as the previous method. When compared to the first method the area overhead will be high in this method, but there is no initialization required using weak PUF for this method.

In the next section, our approach is to discuss the implementation of obfuscated countermeasures in AES using Arbiter PUF.

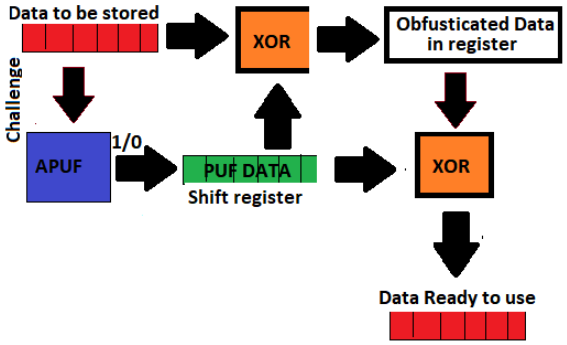

Fig. 7. Arbiter PUF Random data using shift register

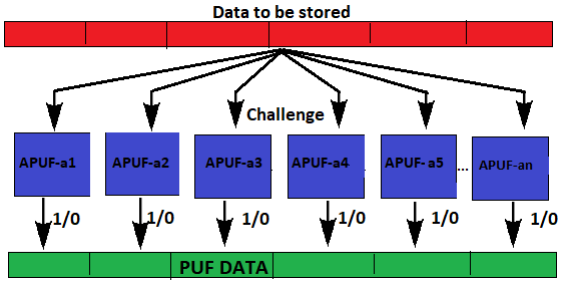

Fig. 8. Arbiter PUF Random data without shift register

4 Implementation of proposed countermeasure for unrolled and rolled AES

In this section, we discuss the proposed countermeasure for hamming weight power model and hamming distance power model with an example. The unrolled implementation of AES shown in [26] can be vulnerable to DPA using hamming weight power model. The naïve implementation of AES in FPGA is vulnerable to DPA using hamming distance power model as shown in [2]. To protect the implementation, we present the countermeasure using Arbiter PUF to obfuscate the hamming weight and hamming distance power model in unrolled and rolled implementations of AES respectively.

4.1 Obfuscating a power model in rolled AES

In the rolled implementation of AES-128 (block size 128-bit, Key size 128-bit), there will be 10 rounds, which are executed in 10 clock cycles. The first nine rounds consist of subytes (S-Box), mix Columns, shift rows, and add round Keys and in the tenth round, the mix Column operation is skipped as shown in

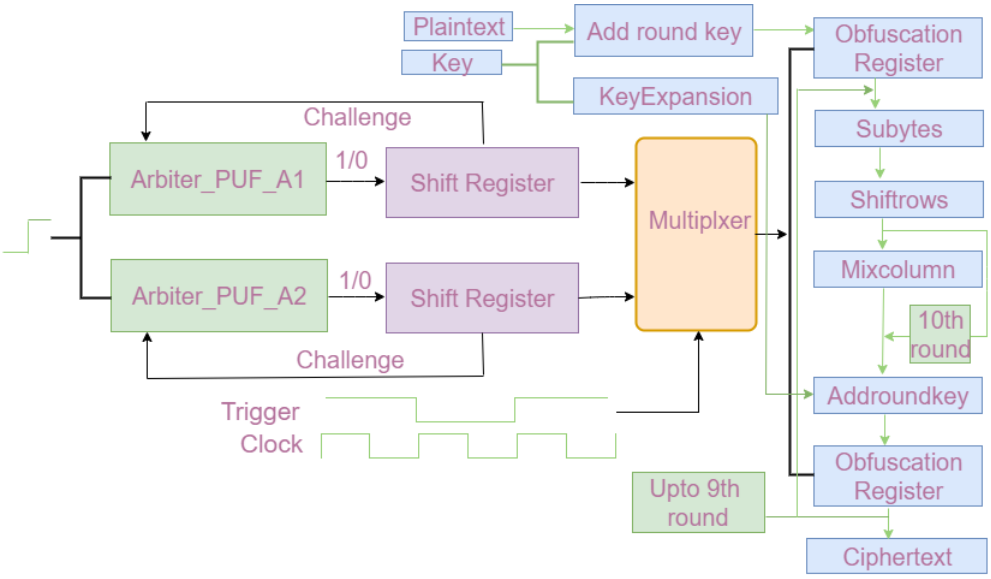

Fig. 9. Rolled AES countermeasure using Arbiter PUF

Fig 9. Every round output is stored and updated in the single register for every clock cycle. By taking advantage of this in [2] the author has demonstrated the CPA on AES to retrieve the tenth round key. In order to avoid this attack, we have proposed the obfuscation countermeasure for rolled AES. In this countermeasure, we obfuscated the switching activity of the register, by generating two random values (128-bit each) using Arbiter PUF. To obfuscate the switching activities of the 128-bit register, one of the random number is xor-ed with the intermediate value before it is stored in the register. The random numbers are chosen alternatively for consecutive clock cycles. Since the design follows hamming distance model, if we use the same random values to obfuscate the switching activity of a consecutive clock cycle, the switching activity will remain the same or unchanged. Hence, our approach is to obfuscate the intermediate value with two different random values for the consecutive clock cycle of each encryption as shown in Fig 9.

4.2 Obfuscating a power model in unrolled AES

The unrolled implementation of AES-128 is proposed for less time consumption and to improve the security of the design discussed in [26]. In [29], Hamming weight power model is used to retrieve the secret key using a power attack. To protect the implementation, we have used the obfuscation technique to obfuscate the switching activities of the register as shown in Fig 10. In this technique, we need random values, which are generated using Arbiter PUF for every plain

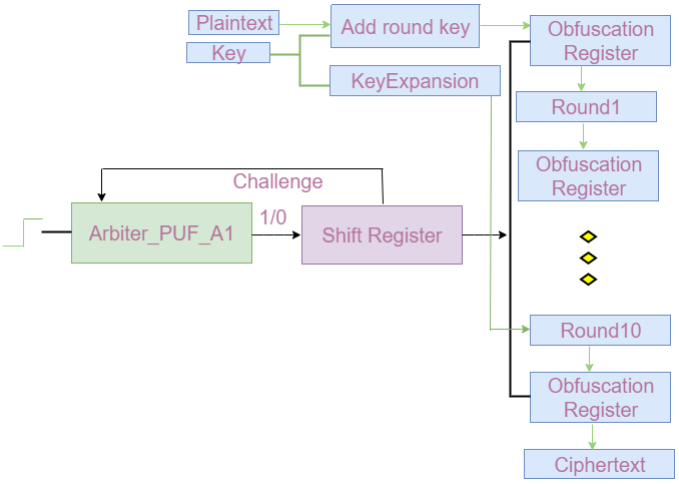

Fig. 10. Unrolled AES countermeasure using Arbiter PUF

text and xor-ed to obfuscate the switching activities of the register. Thereby hamming weight power models can be prevented in unrolled AES.

In Fig 9 and Fig 10, the obfuscation register block contains xor logic which is used to obfuscate the register-switching activities as shown in Fig 5. In the next Section, we compare the performance and security of the proposed countermeasure with naïve implementation of rolled AES and masked AES using the TI scheme.

4.3 Experimental result and comparison of rolled AES

Table 2. Experimental setup

Oscilloscope	Keysight DSOS204A
Board	SASEBO-GII
Target FPGA	Virtex5(65nm)
Sampling rate	10 Giga Sample per sec
Operating frequency of target	3MHZ
Interface	MATLAB

To compare the security strength of the proposed countermeasure with the conventional countermeasure, we have collected 1,00,000 power traces of naïve AES, masked TI [27], and Obfuscated rolled AES using A-PUF is implemented in SASEBO GII board with Spartan 3A as control FPGA and Virtex-5 as target FPGA. We have developed and used the MATLAB interface to communicate with the SASEBO GII board and an oscilloscope. Operating frequency and sampling rate as mentioned in Table 2. We have performed two types of testing to

validate the above design: 1. Evaluation style testing. 2. Conformance style testing.

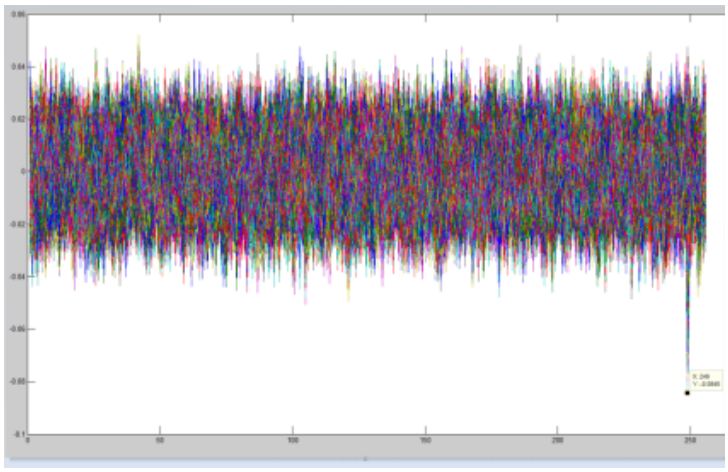

Fig. 11. CPA for rolled AES

Evaluation style testing In this testing methodology, we performed correlation power analysis(CPA) for rolled naïve AES using 10,000 traces and proposed obfuscated AES using 1,00,000 traces. From the Fig 11, Fig 12 clearly shows that the highest peak to differentiate key for rolled naïve AES and there is no highest peak to uniquely differentiate the key byte for obfuscated rolled AES by first order CPA. Further, we have collected an additional 1,00,000 traces for an obfuscated rolled AES. To analyze the second-order attack by combining two intermediate results of power trace and correlation between combinational (data path) switching activities versus power traces. It is observed that there is no highest peak to differentiate the key for 2,00,000 traces.

Conformance style testing In order to ensure security by conformance style testing Methodology, we have evaluated the non-specific method of Test Vector Leakage Assessment (TVLA) for Rolled naïve AES, masked TI AES, and proposed countermeasures of AES. Since the non-specific TVLA method of testing is not suitable for all types of countermeasures, we have evaluated the implementations using the specific method of TVLA. The last round intermediate value is targeted for all three implementations (rolled AES with 10,000 traces, masked TI AES with 100,000 traces, and obfuscated rolled AES with 100,000 traces).

We have plotted the t-score for the specific TVLA of every individual bit of 128-bit intermediate value for every 1,000 traces incrementally.Fig 13, Fig 14, and Fig 15 show the t-score plot on 128-bit intermediate value for a corresponding

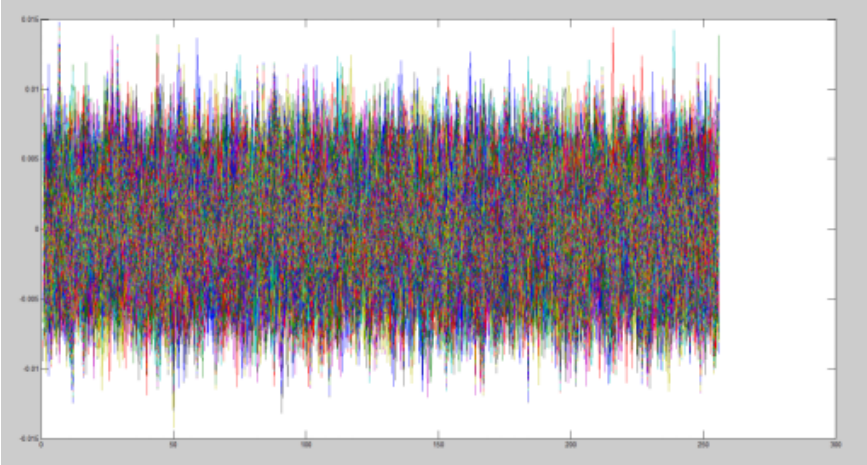

Fig. 12. CPA for proposed obfuscated rolled AES using A-PUF

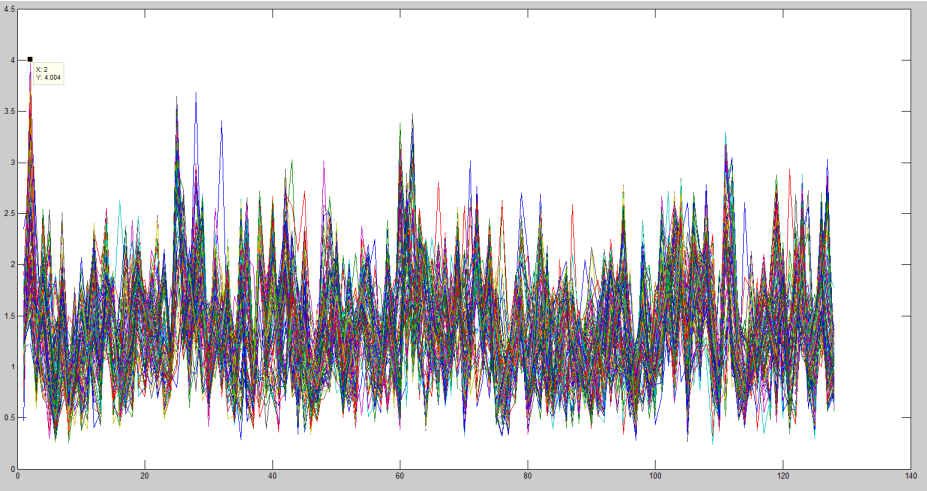

Fig. 13. Specific TVLA for masked TI

number of traces. The sigma value is higher than the 4.5 threshold for 10,000 traces of rolled AES Implementation. Whereas, the sigma value for masked TI AES and proposed obfuscated rolled AES is less than the 4.5 thresholds for 100,000 traces. Further, in Table 3 and Fig 16, we have compared the area and security strength of rolled, masked TI, and obfuscated Rolled AES. From these results, it is observed that the obfuscated AES is not revealing the secret key, at the same time not introducing the area overhead as TI schemes.

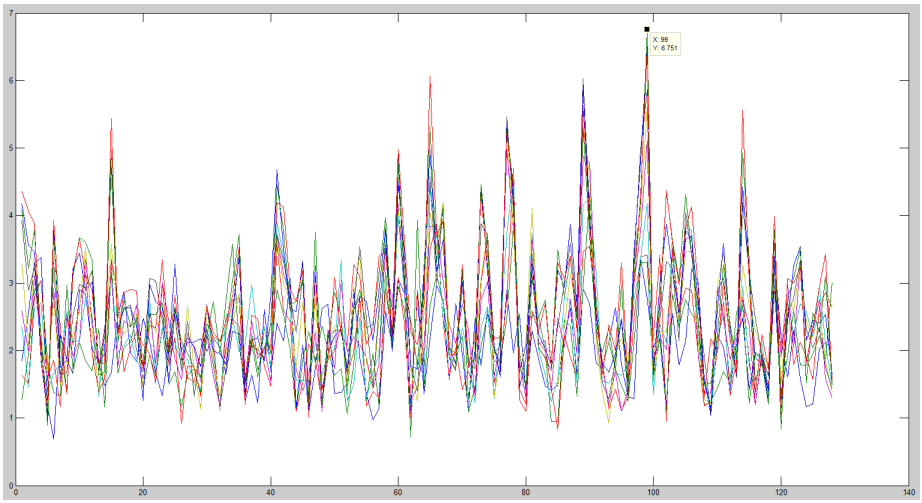

Fig. 14. Specific TVLA for Rolled AES

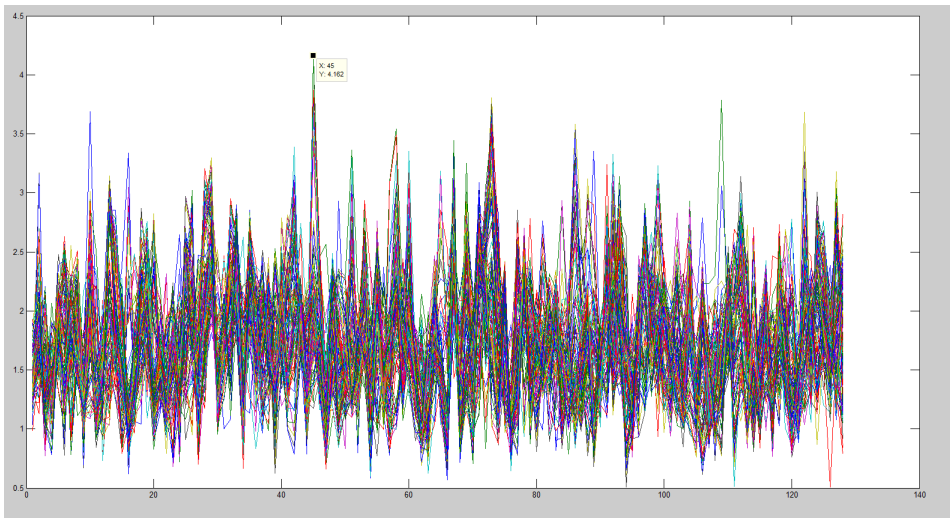

Fig. 15. Specific TVLA for obfuscated rolled AES using A-PUF

4.4 Attack complexity of the proposed obfuscated rolled AES countermeasure using Arbiter PUF

To attack the obfuscating countermeasure, the attacker needs to compute the intermediate value of the register. To compute the intermediate value, the random number for every encryption needs to be found. That would be challenging since Arbiter PUF uses manufacturing process variation to generate the random number. Though this PUF has a modeling attack issue, we are keeping a PUF re-

Table 3. t-score and area comparison of rolled AES, masked TI AES, and Proposed obfuscated Rolled AES

Design	Area(Slice)	Specific TVLA	No of traces
Rolled AES	827	6.75	10000
Masked TI AES (4input and 3 output shares)	4560	4.00	100000
Proposed Obfuscated Rolled AES	2069	4.1	100000

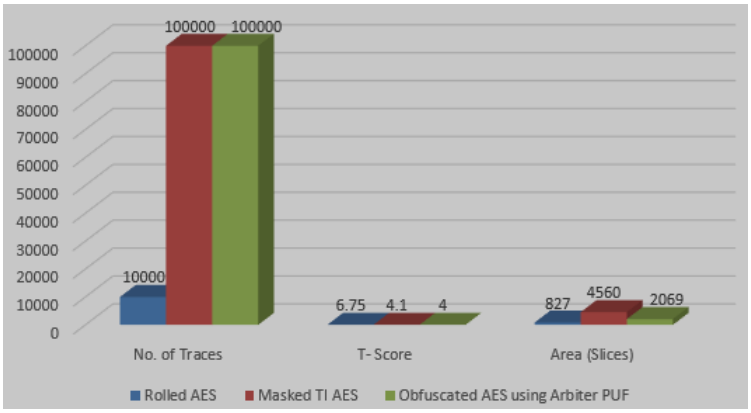

Fig. 16. Area and security strength comparison

sponse private. Therefore predicting the intermediate value is difficult to perform first-order CPA. On the other hand, we try to validate the design by combining a power consumption of two different intermediate values(power consumption random values and 9th-10th round intermediate value) as demonstrated in [28]. Since we are generating two random numbers in parallel, predicting two random values of a large size, quite increases the attack complexity of the design. In order to ensure the security strength further, implementing an Arbiter PUF with side-channel countermeasure is shown in [30]. In the next Section, we have discussed the other countermeasure technique to boost up the side-channel resistance of obfuscated AES.

5 Pre-charge and neutralizing countermeasure to boost up the side-channel resistance of obfuscated rolled AES using A-PUF

In this section, we have discussed two countermeasures to boost up the side-channel resistance of obfuscated rolled AES in a noise-free environment. 1. Pre-charge countermeasure 2. Neutralizing countermeasures

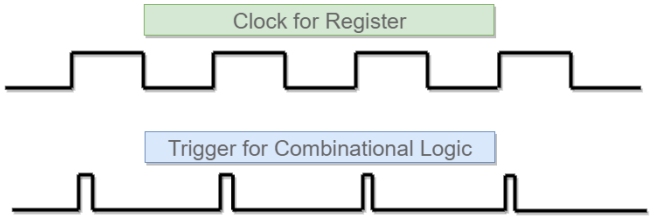

Fig. 17. Pre-charge countermeasure for combinational design

5.1 Pre-charge Countermeasure

- When Trigger is '1', random data is processed in the design.
- When Trigger is '0', functional data is processed in the design.

The countermeasure will boost the side-channel resistance of combinational logic as shown in Fig .17.

5.2 Neutralizing Countermeasure

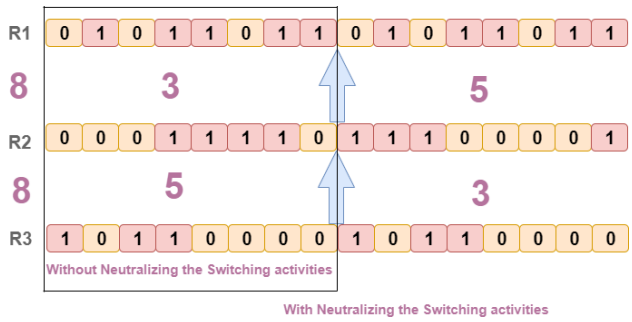

Fig. 18. Neutralizing countermeasure for Register

- Without neutralizing countermeasure switching activities between register R3 to R2 and R2 to R1 is different as shown in Fig.18.
- With neutralizing countermeasure switching activities between register R3 to R2 and R2 to R1 are the same, which helps to boost the side-channel resistance of register switching activities of design combined with an obfuscated countermeasure.

6 Conclusion

In this work, we have proposed the lightweight implementation of countermeasures using Arbiter PUF and demonstrate the effectiveness of obfuscated rolled AES implementation using A-PUF in the FPGA platform. Further, we have included pre-charge countermeasure, neutralizing countermeasure for enhancement of side-channel resistance in a noise-free environment, and robustness of IC design.

7 Futher Work

Our next work is to test the side channel vulnerability of design with obfuscated, precharge, and neutralizing countermeasures using AI-based techniques.

References

1. F. J. D'souza and D. Panchal, "Advanced encryption standard (aes) security enhancement using hybrid approach," in 2017 In- ternational Conference on Computing, Communication and Automation (ICCCA), 2017, pp. 647–652.
2. A. Amaar, I. Ashour, and M. Shiple, "Efficient implementation of aes algorithm immune to dpa attack," in 2012 UKSim 14th International Conference on Computer Modelling and Simulation, 2012, pp. 396–401.
3. E. Prouff and M. Rivain, "Masking against side-channel attacks: A formal security proof," in Advances in Cryptology – EUROCRYPT 2013, T. Johansson and P. Q. Nguyen, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 142–159.
4. H. Maghrebi, J.-L. Danger, F. Flament, S. Guilley, and L. Sauvage, "Evaluation of countermeasure implementations based on boolean masking to thwart side-channel attacks," in 2009 3rd International Conference on Signals, Circuits and Systems (SCS), 2009, pp. 1–6.
5. Y.-i. Hayashi and N. Homma, "Introduction to electromagnetic information security," IEICE Transactions on Communications, vol. E102.B, 08 2018.
6. Y. Ishai, A. Sahai, and D. A. Wagner, "Private circuits: Securing hardware against probing attacks," in CRYPTO, 2003.
7. B. Bilgin, B. Gierlichs, S. Nikova, V. Nikov, and V. Rijmen, "A more efficient aes threshold implementation," 05 2014.
8. S. Surendran, A. Nassef, and B. D. Beheshti, "A survey of cryp- tographic algorithms for iot devices," in 2018 IEEE Long Island Systems, Applications and Technology Conference (LISAT), 2018, pp. 1–8.
9. T. M. Fern andez-Caram'es and P. Fraga-Lamas, "Towards post- quantum blockchain: A review on blockchain cryptography re- sistant to quantum computing attacks," IEEE Access, vol. 8, pp. 21 091–21 116, 2020.
10. S. Joshi, S. Mohanty, and E. Kougianos, "Everything you wanted to know about pufs," IEEE Potentials, vol. 36, pp. 38–46, 11 2017.
11. R. R. Adhithan and N. N. Anandakumar, "Modeling attacks and efficient countermeasures on interpose PUF," in Foundations and Practice of Security - 13th International Symposium, FPS 2020, Montreal, QC, Canada, December 1-3, 2020, Revised Selected Papers, ser. Lecture Notes in Computer Science, G. Nicolescu, A.

Tria, J. M. Fernandez, J. Marion, and J. Garc ıa-Alfaro, Eds., vol. 12637. Springer, 2020, pp. 149–162. [Online]. Available: https://doi.org/10.1007/978-3-030-70881-8 10

12. C. Yehoshuva, R. R. Adhithan, and N. N. Anandakumar, "A survey of security attacks on silicon based weak PUF architectures," in Security in Computing and Communications - 8th International Symposium, SSCC 2020, Chennai, India, October 14-17, 2020, Revised Selected Papers, ser. Communications in Computer and Information Science, S. M. Thampi, G. Wang, D. B. Rawat, R. K. L. Ko, and C. Fan, Eds., vol. 1364. Springer, 2020, pp. 107–122. [Online]. Available: https://doi.org/10.1007/978-981-16-0422-5 8
13. C. Herder, M.-D. Yu, F. Koushanfar, and S. Devadas, "Physical unclonable functions and applications: A tutorial," Proceedings of the IEEE, vol. 102, no. 8, pp. 1126–1141, 2014.
14. J. Bajorat, R. Hofmockel, D. Vagts, M. Janda, B. Pohl, C. Beck, and G. Noeldge-Schomburg, "Comparison of non-invasive, semi- invasive and invasive techniques of cardiac output measurement under different haemodynamic conditions in a pig model," Euro- pean journal of anaesthesiology, vol. 23, pp. 23–30, 02 2006.
15. T. Fujino, T. Kubota, and M. Shiozaki, "Tamper-resistant cryp- tographic hardware," IEICE Electronics Express, vol. 14, pp. 20 162 004–20 162 004, 01 2017.
16. H. Bar-El, "Introduction to side-channel attacks," 2010, http://gauss.ececs.uc.edu/Courses/c653/lectures/SideC/intro.pdf.
17. P. Schaumont, "Key updating for leakage resiliency with applica- tion to aes modes of operation," IEEE Transactions on Information Forensics and Security, vol. 10, pp. 519–528, 03 2015.
18. S. Mangard, E. Oswald, and T. Popp, Power Analysis Attacks: Revealing the Secrets of Smart Cards, 1st ed. Springer Publishing Company, Incorporated, 2010.
19. G. T. Becker, J. Cooper, E. K. DeMulder, G. Goodwill, J. Jaffe, G. Kenworthy, T. Kouzminov, A. J. Leiserson, M. E. Marson, P. Rohatgi, and S. Saab, "Test vector leakage assessment (tvla) methodology in practice," 2013.
20. A. Sadr and M. Zolfaghari-Nejad, "Physical unclon- able function (PUF) based random number generator," CoRR, vol. abs/1204.2516, 2012. [Online]. Available: http://arxiv.org/abs/1204.2516
21. P. Kohlbrenner and K. Gaj, "An embedded true random number generator for fpgas," in Proceedings of the 2004 ACM/SIGDA 12th International Symposium on Field Programmable Gate Arrays, ser. FPGA '04. New York, NY, USA: Association for Computing Machinery, 2004, p. 71–78. [Online]. Available: https://doi.org/10.1145/968280.968292
22. C. Klein, O. Cret, and A. Suciu, "Design and implementation of a high quality and high throughput TRNG in FPGA," CoRR, vol. abs/0906.4762, 2009. [Online]. Available: http: //arxiv.org/abs/0906.4762
23. Y. Ma, J. Lin, T. Chen, C. Xu, Z. Liu, and J. Jing, "Entropy evaluation for oscillator-based true random number generators," in Cryptographic Hardware and Embedded Systems – CHES 2014, L. Batina and M. Robshaw, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 544–561.
24. D. P. Sahoo, S. Patranabis, D. Mukhopadhyay, and R. S. Chakraborty, "Fault tolerant implementations of delay-based physically unclonable functions on fpga," Cryptology ePrint Archive, Paper 2016/441, 2016, https://eprint.iacr.org/2016/441. [Online]. Available: https://eprint.iacr.org/2016/441
25. C. W. O'Donnell, G. E. Suh, and S. Devadas, "Puf-based random number generation," 2004.

26. L. Ali, I. Aris, F. S. Hossain, and N. Roy, "Design of an ultra high speed aes processor for next generation it security," Computers Electrical Engineering, vol. 37, no. 6, pp. 1160– 1170, 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045790611000875
27. B. Bilgin, "Threshold implementations: as countermeasure against higher-order differential power analysis," Ph.D. dissertation, Uni- versity of Twente, Netherlands, May 2015, cum laude. 9
28. T. S. Messerges, "Using second-order power analysis to attack dpa resistant software," in Proceedings of the Second International Workshop on Cryptographic Hardware and Embedded Systems, ser. CHES '00. Berlin, Heidelberg: Springer-Verlag, 2000, p. 238–251.
29. Ville Yli-Mäyry and Rei Ueno Noriyuki Miura and Makoto Nagata and Shivam Bhasin " Diffusional Side-Channel Leakage From Unrolled",IEEE Trans. Inf. Forensics Secure,16,1351–1364.2021.https://doi.org/10.1109/TIFS.2020.3033441
30. T. Kroeger, W. Cheng, S. Guilley, J. -L. Danger and N. Karimi, "Making Obfuscated PUFs Secure Against Power Side-Channel Based Modeling Attacks," 2021 Design, Automation and Test in Europe Conference and Exhibition (DATE), 2021, pp. 1000-1005, doi: 10.23919/DATE51398.2021.9474137.
31. Merli, Dominik. "Attacking and protecting ring oscillator physical unclonable functions and code-offset fuzzy extractors." (2013).
32. Ulrich Rührmair, Xiaolin Xu, Jan Sölter, Ahmed Mahmoud, Mehrdad Majzoobi, Farinaz Koushanfar, and Wayne Burleson. 2014. Efficient Power and Timing Side Channels for Physical Unclonable Functions. In Proceedings of the 16th International Workshop on Cryptographic Hardware and Embedded Systems — CHES 2014 - Volume 8731. Springer-Verlag, Berlin, Heidelberg, 476–492. https://doi.org/10.1007/978-3-662-44709-3$_2$6