PPAP and iPPAP: PLL-based Protection Against Physical Attacks

Prasanna Ravi, Shivam Bhasin, Jakub Breier, Anupam Chattopadhyay
Nanyang Technological University, Singapore
{prasanna.ravi, sbhasin, jbreier, anupam}@ntu.edu.sg

***Abstract*—Digital security practitioners are facing enormous challenge in face of the growing repertoire of physical attacks, e.g., Side Channel Attack (SCA) and Fault Injection Attack (FIA). Countermeasures to such threats are usually very different in nature and come with a significant performance penalty. While the FIA countermeasures rely on fault-detecting sensors or concurrent error detection schemes, SCA countermeasures are based on data masking or dual-rail logic circuits. Recently, a low-overhead FIA countermeasure has been proposed that utilises a ring oscillator circuit with Phase-Locked Loop (PLL). In this paper, we extend that countermeasure to further provide protection against SCA, thereby proposing PLL based Protection Against Physical attacks (PPAP). We demonstrate the `PPAP` on an FPGA prototype under rigorous SCA and FIA testing. We evaluate SCA resistance using the TVLA metric and observe a 2000× increase in SCA protection (in terms of number of traces) with `PPAP`. We further improve the security of `PPAP` using statistical analysis through an improved PPAP design (`iPPAP`) with an increase in SCA resistance of at least 5000× compared to the unprotected implementation with a minimal area overhead.**

I. Introduction

The notion of security evaluation of cryptographic systems using only mathematical cryptanalysis of the underlying schemes was challenged by the seminal work of Kocher et al. [1] who introduced Side Channel Attacks (SCA) that target weaknesses in implementations of the cryptographic scheme. One of the most well-known side channel attacks is the Differential Power Analysis (DPA) which exploits the relation between the processed data and the power consumption of the device during the operation. On the other hand, there are Fault Injection Attacks (FIA) [2] which assume an active attacker, who induces faults in the internal state or maliciously alters the way of operation of the cryptographic algorithm. Attacker exploits the propagation of induced error to learn information of internal secrets like the key. Since their inception, numerous attacks and corresponding countermeasures have been reported in literature.

Countermeasures against SCA and FIA are quite different in nature. While SCA countermeasures depend on noise generation to hide the leakage, FIA countermeasures are generally based on the ability to detect/repair faults via information/space/time redundancy. For both SCA and FIA, there have been several works that introduce countermeasures at different levels of design abstraction. When protecting against both, usually SCA and FIA countermeasures are developed independently and later combined, leading to significant overheads [3], [4].

It must be noted that timing precision is in general crucial to both SCA and FIA. Externally introduced timing misalignment has shown to be effective against both the attacks [5], [6]. In [5], a PLL based clock randomiser is used for timing misalignment to boost SCA security.

In this paper, we propose **PLL based Protection Against Physical attacks (PPAP)**, a low cost combined hardware countermeasure against SCA and FIA with a very low performance overhead. The main building blocks of `PPAP` are a *fault attack sensor* and a *timing misalignment module*. Fault attack sensors were originally proposed by Miura et al. [7] to detect electromagnetic fault injection. The sensor is composed of a ring oscillator as fault injection detector and PLL for triggering an alarm. We first integrate both countermeasures to provide resistance against SCA and FIA. Next, we study the relation of timing misalignment and its statistical impact on SCA security, to design `iPPAP` (i.e., improved `PPAP`), which further boosts the SCA security.

The contributions in this works are as follows:

- We propose `PPAP`, an integrated plug and play hardware-based countermeasure against SCA and FIA.
- We develop a metric for measuring SCA security introduced by timing misalignment, to understand the key parameters contributing to SCA resistance.
- We next propose `iPPAP`, improved for SCA security.
- Both countermeasures are practically evaluated on an FPGA and demonstrated to be low-cost, easy to implement and independent of the circuit under protection.
- We report near perfect FIA security and SCA security boost of 2000× and 5000× for `PPAP` and `iPPAP` respectively.

II. Preliminaries

In this section, we provide general background information on side-channel and fault attack concepts used. We also recall related works on clock randomisation, fault sensor and combined countermeasures.

A. Test Vector Leakage Assessment

Side-Channel Attacks (SCA) target physical observation of the secret-key related computation. A novel approach – Test Vector Leakage Assessment (TVLA [8]) has recently emerged as a testing strategy for SCA leakage. It detects any data dependent leakage through Welsh T-test to give a *PASS/FAIL*

decision. The commonly used fixed vs random (FVR) or non-specific TVLA partitions traces based on varying and fixed plaintext, and computes T-test to check for data-dependent leakage. It is computed as $TVLA = (\mu_r - \mu_f)/\sqrt{\frac{\sigma_r^2}{m_r} + \frac{\sigma_f^2}{m_f}}$, where μ_r, σ_r and m_r (resp. μ_f, σ_f and m_f) are mean, standard deviation and cardinality of the set with varying plaintexts (resp. fixed plaintexts), respectively. The device is considered to leak side-channel information and *FAIL* the test if TVLA value crosses the threshold of $[-4.5, 4.5]$. Unlike attack techniques like correlation power analysis etc., TVLA stays independent of the leakage model and thus makes countermeasure comparison easy.

B. Clock Randomisation

There have been a number of works proposed to perform temporal misalignment [5], [9], [10]. In [5], a technique for generating an irregular clock from multiple phase shifted clocks of a PLL output using clock multiplexers was proposed. An EDA friendly automated design flow for a clock randomiser design was proposed in [9] to drive sequential elements using multiple phase shifted clocks. Further proposals for hardware timing disarrangement include insertion of random delays in the input data path or attaching random wait registers [10], [11]. The idea of timing disarrangement has also been explored in software, e.g. through shuffling the instruction sequence, insertion of dummy loops and Random Delay Interrupts (RDI) using special state machines or non-deterministic processors [12], [13].

C. Fault Injection Attacks & Laser/EM Sensor

Countermeasures against fault attacks are either based on concurrent error detection or physical sensors. In this work, we focus on sensor based countermeasures. A Phase-Locked Loop (PLL) based sensor for electromagnetic (EM) fault injection was first proposed by Miura et al. in [7]. The sensor is built up of two main components: a watchdog ring oscillator (WRO) and an alarm circuitry. The WRO runs at a stable frequency during normal operation, but is very sensitive to environmental variations. Hence, a high energy EM or laser injection can alter the frequency and phase of the WRO. Detecting this abnormal event on the WRO through an alarm generation circuit forms the core idea of the FIA sensor.

The PLL is a widely used clock timing control tool available in modern FPGAs. It comprises of a phase detector which converts frequency and phase differences between the feedback and the input clock to up or down pulses. These pulses further drive a loop filter and a voltage control oscillator to nullify the difference. In [7], PLL was used to monitor the clock generated from WRO and thus, detecting the malicious behavior.

D. Combined SCA and FIA countermeasures

As the modus operandi of SCA and FIA are different, protection strategies also differ vastly as discussed in the previous sections.

Mentens et al [11] used the dynamic reconfiguration feature in modern FPGAs to achieve resistance against SCA and FIA. Bhasin et al. [14] exploited the in-built fault tolerance of dual-rail logic to propose it as a combined countermeasure. Combination of threshold implementations, i.e. a provable side-channel countermeasure with parity-based error detection was proposed in [3]. Another approach to use enhanced private circuits against side-channel and fault attack was proposed in [4]. Linear and non-linear codes have also been investigated for combined SCA and FIA countermeasure in software by Breier et al. [15]. Most of these countermeasures were applied at the information level, while one used device specific features. They require design modification and incur significant performance overhead, as high as $20\times$ compared to unprotected equivalent, in some of the above mentioned cases.

III. PLL based Physical Attack Protection

The general idea of **PLL based Protection Against Physical attacks (PPAP)** is illustrated in Fig. 1. Modern FPGAs contain PLLs as DCMs (Digital Clock Managers) in Xilinx Virtex 4, 5, 6 series FPGAs, and as MMCMs (Multi Mode Clock Managers) in the latest Xilinx 7 series FPGA and Zynq UltraScale processors. The latter ones offer more capabilities, such as multiple programmable clock outputs, static and dynamic fine phase shifting and Dynamic Port Reconfiguration Ports (DRP) which can alter the characteristics of clock during runtime. The PLL configured to generate multiple phase shifted clock, which leads to timing misalignment when multiplexed, was the initial idea proposed in [5]. By deriving the source clock from an internal free running ring oscillator (RO), fault attack detection capabilities are added to `PPAP`. The RO is placed over the sensitive core in order to detect any active fault injections like laser or EM. Fault detecting properties of RO were already demonstrated in [7]. `PPAP` has three fold impact on security. The timing misalignment contributes directly to SCA security. Moreover, the misalignment also makes fault injection harder as injection timing is often crucial for localizing injection targets. Finally, for more powerful attacks like laser/EM, the clock generating RO detects the attack before a fault is injected in the sensitive circuits.

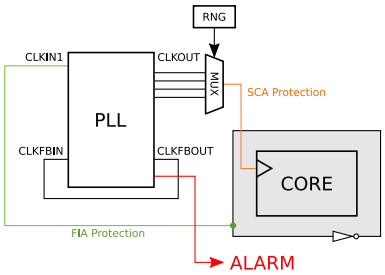


Fig. 1: Design of the combined FIA and SCA countermeasure based on PLL.

The main advantages of using such approach are listed as follows:

- This approach does not call for any changes to the implementation of the circuit under protection.

- Utilization of components that are readily available on numerous FPGAs.
- Since the implementation stays the same, it has zero impact on the maximum operating frequency of the design. Albeit, the clock randomiser itself has an impact on the throughput of the design, which we attempt to minimize.

Through statistical analysis, we also improve the design of the clock randomiser in PPAP to propose iPPAP to further boost SCA security. Though we only demonstrate results for SCA and FIA resistance of PPAP and iPPAP on FPGA, we would like to note that our design is easily compatible with the design flow of [7] that automatically implements the combined FIA sensor along with the clock randomiser. The modifications needed are limited to only *RTL* level and thus straightforward to implement. Therefore, PPAP and iPPAP is readily compatible with existing standard-cell design flows used in current EDA tools.

IV. Improving Clock Randomiser

In this section, we first explain the impact of known RDI techniques on SCA protection followed by design of improved clock randomiser.

A. Random Delay Insertion Techniques

The time instance of occurrence of a particular operation depends on the sum of the individual delays introduced until the execution of that operation. Hence, from an attacker's perspective, only the cumulative effect of the individually inserted delays is observable. Thus the security and efficiency of timing misalignment is heavily dependent on the behaviour of the cumulative delay (*CD*) distribution. This *CD* can be modeled as a random variable X with a distribution of finite mean μ and variance σ^2. Thus, one would ideally look to maximize the variance of the *CD* distribution (spreading the time of occurrence of the operation) for a fixed mean (overall average performance). The most straightforward RDI method is generating uniformly distributed independent delays, whose *CD* exhibits a Gaussian distribution.

Since the Gaussian distribution has a large peak and a small variance, it provides much lower randomness compared to an equivalent uniform distribution. A near uniform *CD* distribution is achievable through adoption of the "Floating Mean Method" as proposed in [13]. The intuition is to make a random selection of a small section within a given interval and generate uniform delays within that selected interval for a single run of the algorithm. The selection of the small section is updated for every run of the algorithm and the same is repeated. Statistically speaking, it generates a *Gaussian mixture* which comprises of a series of smaller Gaussian distributions with shifted means which together have much wider span, thus achieving a near uniform distribution. This work was further improved by the same authors in [12] as "Improved Floating Mean" method where they introduced finer granularity in generating the delays to flatten out the cogs formed in the *CD* distribution. Refer to Fig. 4(a) for the *CD* distribution generated by all the three different methods.

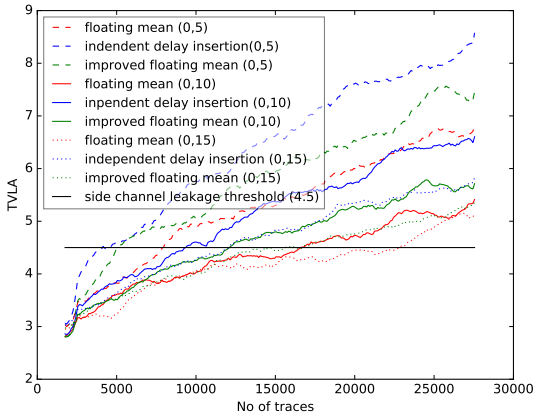


Fig. 2: Comparison of TVLA metric values for the simulated RDI methods for varying delay intervals.

They proposed a performance-security metric M to evaluate the efficiency of the RDI countermeasure, which is stated as follows,

$$M = 1/(2.\hat{p}.\mu) \tag{1}$$

where $\hat{p}$ and μ are the maximum peak and mean of the probability distribution of the *CD* distribution, respectively. Considering this metric, a "uniform" *CD* distribution will provide the best efficiency and security.

B. Simulation-Based Analysis

We further analyse the security performance of these different RDI techniques using the TVLA testing methodology. We analyse their behaviour in a simulated setting when incorporated into a secure algorithm. We use the parallel AES-128 implementation, with a perfect Hamming weight (HW) leakage model with added white Gaussian noise. Random delays modeled based on the above three RDI techniques are inserted after every round of the AES algorithm. A weaker implementation will cross the threshold faster. As shown in Fig. 2 for the fixed delay intervals (5,10 and 15), we can see that the floating mean method performs best. The floating mean method (FM) also performs equally or better than its improved counterpart (IFM) for varying delay intervals. Another observation is that the increase in variance leads to increase in security.

Summing up the above discussion, the factors that primarily affect side channel leakage of a timing disarrangement countermeasure are:

- Behaviour of *CD* distribution.
- Unique number of CD instances.
- Span of *CD* distribution.

Thus, increasing the variance of the *CD* distribution while maintaining a fixed mean, along with improving the granularity (in terms of number of different instances of the operation), will improve the SCA resistance of the RDI countermeasure.

C. Clock Randomiser Design by [5]

The clock randomiser design in [5] uses multiple phase shifted clock outputs of the MMCM which are fed to a tree

of clock multiplexers (with select lines connected to PRNG) to output a jittery clock. In order to maintain a glitch free operation, the clock multiplexers do not change the output immediately after a select input change. Upon the change of the select input, the output follows the current input until a falling transition on the current input. The output stays in the low state until the new input has also transitioned from high to low, and then follows the new input. The time taken for the output to switch from one input to the other can be referred to as the *Trance* state. The duration of this *Trance* state can be varied by continuously switching the select input of the multiplexer. We implemented the design on FPGA and plotted the distribution of duration of AES encryption, when driven by clock randomiser from [5]. From practical measurements on FPGA, we can see that the time taken for AES execution follows a Gaussian distribution (Fig. 4(c)). Gaussian CD was shown to provide less security in previous simulations, compared to uniform CD. Improvements to the current clock randomiser are further proposed to bring the delay CD close to uniform.

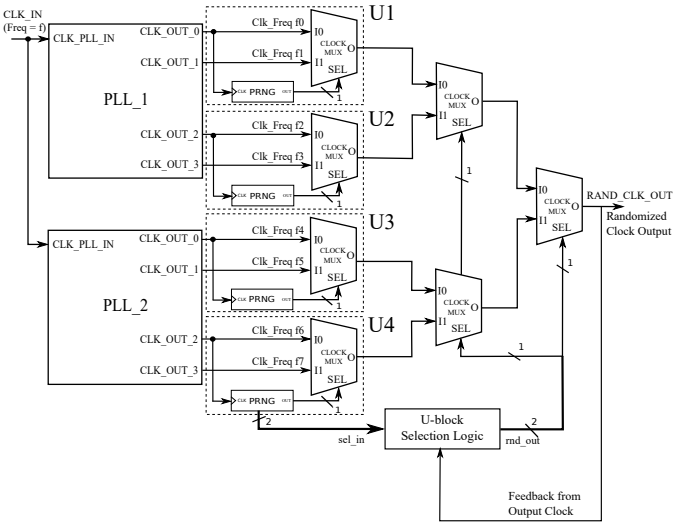


Fig. 3: Design of improved clock randomiser.

D. Improved Clock Randomiser Design

To improve the design of the clock randomiser, we analyse the output of a single clock multiplexer with two phase shifted inputs. The clock output of a single clock multiplexer with input clocks of time period t phase shifted by $p < t$, can have the following four values of time period: t, $t+p$, $2t-p$ and $2t$. Thus, increasing the number of phase shifts increases the number of possible values of time period of the output clock. Hence, we propose to use *frequency shifted* clocks to increase the number of instantaneous phase shifts between the clocks, thus yielding a higher number of *CD* instances.

We call U-block a macro which is a combination of a clock multiplexer with two frequency shifted inputs and a PRNG which generates a random select line input for the multiplexer. Fig. 4(b) shows the *CD* distribution of different combination of frequency shifted clocks. Thus, we can see that the set of frequencies of the clock inputs to a U-block determine

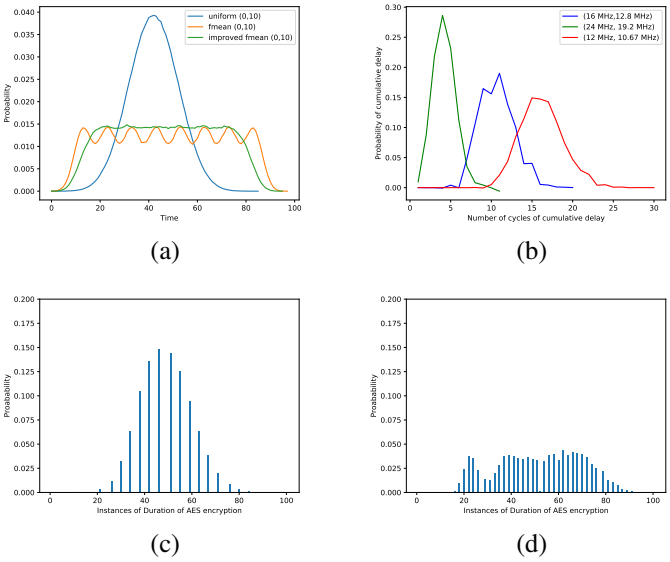


Fig. 4: (a) Cumulative Delay (*CD*) distribution generated by the three RDI (Random Delay Interrupts) methods. (b) Shifted Gaussians generated by different choices of frequencies and frequency shifts. Histogram of measurement from FPGA of cumulative delays by (c) clock randomiser in [5], (d) improved clock randomiser.

the mean and variance of the corresponding Gaussian *CD* distribution. With $2P$ clock outputs from the MMCM, we can thus have P such U-blocks and $log(2P)$ levels of multiplexers which form the improved clock randomiser as shown in Fig. 3 ($P = 4$ in our design). Thus, one U-block is selected randomly for each encryption, similar to the random selection of the small section in "Floating Mean method" (Refer Sec.IV(A)). With suitable choice of frequencies for the clock inputs to the U-blocks, one can generate a Gaussian mixture (combination of mean shifted Gaussians), thus arriving close to a uniform distribution. In order to equalize the peaks of all the smaller Gaussians, we bias and fine tune the selection of U-blocks to generate a distribution as close as possible to uniform. Refer Fig. 4(d) for the near uniform CD distribution from practical FPGA measurements.

The implementation results of the clock randomiser from [5] and our improved proposal are summarised in Tab. I. The area overhead comes from the individual PRNG required for each U-block as compared to a single PRNG in [5]. Instead of a PRNG, any randomness source like TRNG or a stream cipher could be used as select inputs for the U-blocks. Thus, barring the area used up for the PRNG, both the designs are equally efficient. It also reports the mean encryption delay (μ) and distribution peak $\hat{p}$ to compute performance security trade off as per Eq. (1). Evaluation of the metric defined in Eq. (1) reveals a theoretical 2.4× increase in efficiency of our countermeasure (`iPPAP`) compared to [5].

Not all choices of frequencies lead to a synthesizable design in FPGA as it fails to achieve timing closure. The possible reasons for this behaviour could be:

- The apparent inability of the PLL to generate precise output frequencies from a given input frequency.

TABLE I: Area overhead and performance-security trade-off of Clock Randomiser (CR) and Improved Clock Randomiser (ICR)

	LUT	DFF	BUFG	PLL	$\mu(\mu s)$	$\hat{p}$	M(μs^{-1})
CR [5]	23	64	8	2	0.9	0.15	3.7
ICR	102	266	11	2	1.25	0.045	8.88

- Violation due to instances when the instantaneous phase shift between the clocks is very small compared to the time period of the clocks, resulting in glitches in the clock output (although no such considerations are specified in the design documents).

One could possibly provide explicit timing constraints [9] to achieve timing closure or choose a constrained set of frequencies that avoid timing violations. In this work, we carefully arrived at certain combination of frequencies of the clock inputs to the multiplexers that lead to a synthesizable design. For a base period of the clock as t, the other time periods are computed as perturbations by a fraction of the base period. Thus, the 8 clock outputs are generated with time periods $(t, t + t/4), (3t/2, 3t/2 + (3t/2)/4), (2t, 2t + 2t/8), (5t/2, 5t/2 + (5t/2)/5)$. For our design which is also depicted in Fig.3, we utilized the t value of 41.666 nsec, which corresponds to 24 MHz. The process of automatic generation of the timing constraints file to achieve timing closure is left for future work.

V. Security Evaluation

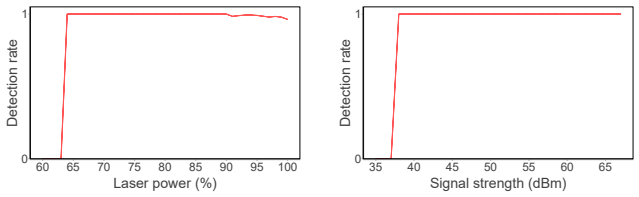
In this section, we perform practical security evaluation of the proposed PPAP and iPPAP module against SCA and FIA.

A. Experimental Platform

For evaluating fault injection detection capabilities, we have tested the PPAP and iPPAP experimentally, against both laser fault injection (LFI) and electromagnetic fault injection (EMFI). As a device under test (DUT), we used a Xilinx Virtex-5 FPGA, mounted on a Genesys board from Digilent. We have used PRESENT-80 cipher for testing the countermeasure, where the RO was routed around this cipher circuit to protect it. The reason behind using a lightweight cipher like PRESENT-80 is to limit the size of ring oscillator. Trigger signal for LFI/EMFI activation was generated by the cipher implementation and sent during the last round. The board was placed on a X-Y positioning table with a step precision of 0.05 μm.

In the case of LFI, we used a diode pulse laser station as a fault injection device, with a near-infrared (1064 nm) laser source with a maximum output power of 20 W. We have mounted a 5$\times$ magnification lens on the laser source, decreasing the power to a maximum of 10 W and spot size to 60$\times$14 μm^2. Fault injection was controlled by a triggering device connected to a computer. For checking the precision of injection parameters, such as offset and length, we have used a digital oscilloscope.

When it comes to EMFI, three main components were used. A pulse generator that was capable of capturing the trigger signal and setting a precise offset for the pulse generation. An amplifier, with a constant 54 dB gain. And a probe that was taking the signal from the amplifier and injecting it into the DUT.



(a) (b)

Fig. 5: (a) LFI detection rate (b) EMFI detection rate

The side-channel testing was performed on SASEBO GII board which also contains the same Xilinx Virtex-5 FPGA. The usage of SASEBO board for side-channel measurements was motivated by a special support for power/EM measurements. Side-channel traces were measured using a near-field EM probe, placed over a decoupling capacitor on the power line. Measurements were sampled on the Teledyne LeCroy Waverunner 610zi oscilloscope at a sampling frequency of 500 MS/s. The PPAP module was then used to generate clock for a AES-128 module, which encrypts one round per clock.

B. Evaluation against FIA

Since LFI is a localized attack method, we first had to profile the DUT in order to estimate the area where the implementation resides. For this purpose, we have used the same methodology as described in [16]. It is important to note that the EMFI and LFI detection capabilities are the same for both PPAP and iPPAP designs. After the target area was localized, we had run several tests in order to estimate the effectiveness of the proposed countermeasure. We varied the laser power up to 100%, while scanning the whole region of the implemented cipher. The detection rate of the sensor against LFI is depicted in Fig. 5(a). We can see that the sensor detects all the injections up to $\approx$ 90% power. Afterwards, the rate slightly drops but still stays very high ($>$ 0.971). This is caused by the cipher faults that go undetected when the laser power reaches its maximum. In this case, the laser spot with the peak power becomes tightly focused at one point and is able to reach the implementation without disturbing the surrounding ring oscillator. This can be encountered with smaller loops in oscillator.

In case of EMFI, the disturbance is more global, therefore we have not observed a high variance w.r.t. chip area. Again, we varied the signal strength up to the limit of our setup (68.5 dB). The detection rate is depicted in Fig. 5(b). Results for EMFI are consistent throughout the varied power and all the faults were detected by the implemented sensor. The reason for this is the aforementioned global effect, ensuring that any EM pulse affecting the cryptographic circuit will also affect the surrounding sensor.

C. Evaluation against SCA

We used the TVLA methodology to evaluate the SCA resistance of PPAP and iPPAP implemented for an AES-128 design on the FPGA. The randomised clock from PPAP (resp. iPPAP) serves as the system clock for AES. We utilized two PLLs with 8 clock outputs, as in [5] for fair comparison. To

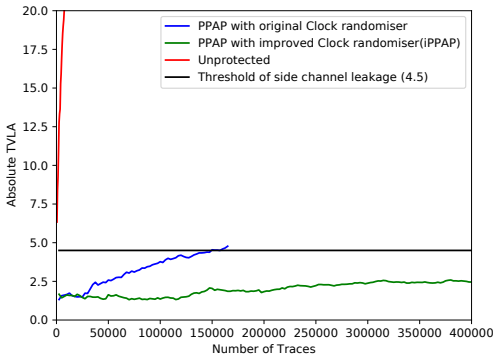


Fig. 6: TVLA evaluation for unprotected and PPAP-protected AES with original and improved clock randomiser.

TABLE II: Area overhead of PPAP and iPPAP with AES-128 on Xilinx Virtex 5 FPGA

Implementation	LUT	Registers	BUFG	PLL_ADV
Unprotected	1850	742	1	0
PPAP	1874	806	9	2
iPPAP	1954	1008	12	2

compare the resistance, we plot the evolution of the t-value against number of traces. The results are shown in Fig. 6. Analysis revealed that unprotected AES already failed the t-test at 80 traces, the PPAP required around $145,000$ traces to cross the TVLA threshold, while iPPAP stays below the threshold until 400,000 traces. Thus, iPPAP provides enhanced security PPAP at minor area and performance overhead.

The area results are reported in Tab. II. The main overhead of iPPAP (compared to PPAP) comes from the use of multiple PRNGs. Note that area of PPAP and iPPAP are constant and independent of size of sensitive circuits. Thus, for larger and resource-heavy designs (like RSA or ECC), the resource overhead will be near negligible.

D. Discussions

Realignment techniques can be potentially applied to defeat PPAP and iPPAP. Briefly looking into popular methods, algorithms like Threshold Phase Only Correlation (T-POC [17]) and Rapid Alignment (RAM [18]) can resync traces shifted in time (phase misalignment), and algorithms like Elastic alignment (Dynamic Time Warping [19]) and Re-synchronization by Moments [20] can also work on aligning time stretched waveforms, thus improving attacker efficiency. The complexity of most of these algorithms depends on the number of sample points on the trace s. While the computational complexity is quasilinear with s ($\mathcal{O}$(slog(s))) for T-POC and linear with s ($\mathcal{O}$(s)) for the elastic alignment algorithm [19]. However, the elastic alignment algorithm can only work on aligning two traces at a time, which requires a reference waveform. Hence the computational complexity of these pre-processing algorithms can be increased by simply increasing the misalignment variance, albeit with some performance overhead. Additionally, there are also certain assumptions like the constant frequency of the input clock for some of the resync algorithms (POC, T-POC), which thus do not work with iPPAP which outputs a variable frequency output clock.

VI. Conclusions

This paper presents a combined SCA and FIA hardware-based countermeasure termed PPAP. It is a low-cost, plug and play countermeasure, independent of the protected circuit. Further, an improvement in form of iPPAP is proposed to boost SCA security, by ensuring a near-uniform distribution of the introduced cumulative delay. Both countermeasures are validated against practical SCA & FIA, reporting high security gain. Further research can focus on all-digital alternatives of PPAP and iPPAP. Moreover, resistance against resynchronisation technique should be evaluated.

References

[1] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Advances in cryptology – CRYPTO99*. Springer, 1999, pp. 789–789.
[2] E. Biham and A. Shamir, "Differential fault analysis of secret key cryptosystems," in *CRYPTO'97*. Springer, 1997, pp. 513–525.
[3] T. Schneider, A. Moradi, and T. Güneysu, "Parti–towards combined hardware countermeasures against side-channel and fault-injection attacks," in *Annual Cryptology Conference*. Springer, 2016, pp. 302–332.
[4] T. De Cnudde and S. Nikova, "More efficient private circuits ii through threshold implementations," in *Fault Diagnosis and Tolerance in Cryptography (FDTC), 2016 Workshop on*. IEEE, 2016, pp. 114–124.
[5] T. Güneysu and A. Moradi, "Generic side-channel countermeasures for reconfigurable devices," in *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer, 2011, pp. 33–48.
[6] V. Lomné, T. Roche, and A. Thillard, "On the need of randomness in fault attack countermeasures-application to aes," in *Fault Diagnosis and Tolerance in Cryptography 2012*. IEEE, 2012, pp. 85–94.
[7] N. Miura, Z. Najm, W. He, S. Bhasin, X. T. Ngo, M. Nagata, and J.-L. Danger, "Pll to the rescue: a novel em fault countermeasure," in *Design Automation Conference (DAC) 2016*. IEEE, 2016, pp. 1–6.
[8] B. J. Gilbert Goodwill, J. Jaffe, P. Rohatgi *et al.*, "A testing methodology for side-channel resistance validation," in *NIST non-invasive attack testing workshop*, 2011.
[9] A. G. Bayrak, N. Velickovic, F. Regazzoni, D. Novo, P. Brisk, and P. Ienne, "An eda-friendly protection scheme against side-channel attacks," in *Proceedings of the Conference on Design, Automation and Test in Europe*, 2013, pp. 410–415.
[10] S. Mangard, "Hardware countermeasures against dpa–a statistical analysis of their effectiveness," in *Cryptographers Track at the RSA Conference*. Springer, 2004, pp. 222–235.
[11] N. Mentens, B. Gierlichs, and I. Verbauwhede, "Power and fault analysis resistance in hardware through dynamic reconfiguration," in *CHES*. Springer, 2008, pp. 346–362.
[12] J.-S. Coron and I. Kizhvatov, "Analysis and improvement of the random delay countermeasure of ches 2009," in *CHES 2010*, 2010, pp. 95–109.
[13] ——, "An efficient method for random delay generation in embedded software," in *CHES 2009*, 2009, pp. 156–170.
[14] S. Bhasin, J.-L. Danger, F. Flament, T. Graba, S. Guilley, Y. Mathieu, M. Nassar, L. Sauvage, and N. Selmane, "Combined sca and dfa countermeasures integrable in a fpga design flow," in *ReConFig'09*. IEEE, 2009, pp. 213–218.
[15] J. Breier and X. Hou, "Feeding two cats with one bowl: On designing a fault and side-channel resistant software encoding scheme," in *Cryptographers Track at the RSA Conference*. Springer, 2017, pp. 77–94.
[16] W. He, J. Breier, S. Bhasin, D. Jap, H. G. Ong, and C. L. Gan, "Comprehensive laser sensitivity profiling and data register bit-flips for cryptographic fault attacks in 65 nm fpga," in *SPACE*, 2016, pp. 47–65.
[17] S. Guilley, K. Khalfallah, V. Lomne, and J.-L. Danger, "Formal framework for the evaluation of waveform resynchronization algorithms." *WISTP*, vol. 6633, pp. 100–115, 2011.
[18] R. A. Muijrers, J. G. van Woudenberg, and L. Batina, "Ram: Rapid alignment method," in *International Conference on Smart Card Research and Advanced Applications*. Springer, 2011, pp. 266–282.
[19] J. G. van Woudenberg, M. F. Witteman, and B. Bakker, "Improving differential power analysis by elastic alignment," in *Cryptographers Track at the RSA Conference*. Springer, 2011, pp. 104–119.
[20] N. Debande, Y. Souissi, M. Nassar, S. Guilley, T.-H. Le, and J.-L. Danger, "re-synchronization by moments: an efficient solution to align side-channel traces," in *WIFS 2011*, pp. 1–6.