IACR Transactions on Cryptographic Hardware and Embedded Systems
ISSN 2569-2925, Vol. 2022, No. 4, pp. 589–613. DOI:10.46586/tches.v2022.i4.589-613

A Power to Pulse Width Modulation Sensor for Remote Power Analysis Attacks

Brian Udugama, Darshana Jayasinghe, Hassaan Saadat, Aleksandar Ignjatovic and Sri Parameswaran

School of Computer Science and Engineering, University of New South Wales, Sydney, Australia.
b.udugama@unsw.edu.au, darshanaj@cse.unsw.edu.au, h.saadat@unsw.edu.au,
a.ignjatovic@unsw.edu.au, sri.parameswaran@unsw.edu.au

Abstract. Field-programmable gate arrays (FPGAs) deployed on commercial cloud services are increasingly gaining popularity due to the cost and compute benefits offered by them. Recent studies have discovered security threats than can be launched remotely on FPGAs that share the logic fabric between trusted and untrusted parties, posing a danger to designs deployed on cloud FPGAs. With remote power analysis (RPA) attacks, an attacker aims to deduce secret information present on a remote FPGA by deploying an on-chip sensor on the FPGA logic fabric. Information captured with the on-chip sensor is transferred off the chip for analysis and existing on-chip sensors demand a significant amount of bandwidth for this task as a result of their wider output bit width. However, attackers are often left with the only option of using a covert communication channel and the bandwidth of such channels is generally limited. This paper proposes a novel area-efficient on-chip power sensor named PPWM that integrates a logic design outputting a pulse whose width is modulated by the power consumption of the FPGA. This pulse is used to clear a flip-flop selectively and asynchronously, and the single-bit output of the flip-flop is used to perform an RPA attack. This paper demonstrates the possibility of successfully recovering a 128-bit Advanced Encryption Standard (AES) key within 16,000 power traces while consuming just 25% of the bandwidth when compared to the state of the art. Moreover, this paper assesses the threat posed by the proposed PPWM to remote FPGAs including those that are deployed on cloud services.

Keywords: Cloud FPGA · Multi-Tenant FPGA · Side-Channel Attack · Remote Power Analysis Attack · On-chip Sensors

1 Introduction

Recent studies have revealed that field-programmable gate arrays (FPGAs) available on cloud services, such as Amazon Web Services Elastic Compute Cloud (AWS EC2), are vulnerable to remotely carried out power analysis attacks [GCRS20]. These attacks can deduce secret information, such as cryptographic keys, used in remote FPGAs by monitoring the fluctuations in dynamic power consumption of a targeted victim device [GCRS20]. In remote power analysis (RPA) attacks, an on-chip sensor that can measure the power consumption of a remote FPGA is deployed onto the targeted remote FPGA to monitor dynamic power fluctuations occurring on the FPGA. The power consumption fluctuations measured by the on-chip sensor are then transferred to an off-chip analysis module for deducing the secret information. Since on-chip sensors are remotely deployable onto FPGAs, RPA attacks can be performed without physical access to the target FPGA, as opposed to conventional power analysis attacks which require physical access to the target FPGA [SGMT18].

Received: 2022-04-15 Accepted: 2022-06-15 Published: 2022-08-31

Understanding possible on-chip sensor designs and attack techniques enable the building of reliable defenses against threats posed by RPA attacks. Hence, it is necessary to conduct extensive research on hitherto unknown sensor designs and remote attack techniques. Existing on-chip sensors, such as time-to-digital converter (TDC) based sensors [SGMT18] and ring oscillator (RO) based sensors [GDTL19], are bulky in terms of area consumption and communication bandwidth requirement for transferring the recorded power consumption fluctuations. The recently proposed VITI [UJS+21] shows a lightweight and calibratable on-chip sensor design that can be covertly inserted into an FPGA. However, VITI too requires significant bandwidth to transfer the recorded power consumption information to the off-chip analysis module. In a typical RPA attack, the attacker has to rely on covert communication channels for communicating with the on-chip sensor. Covert communication channels in general can only support lower bandwidths [GRS20] and therefore, it is crucial for the attacker to devise an on-chip sensor that demands minimum communication bandwidth.

This paper proposes PPWM (**P**ower to **P**ulse **W**idth **M**odulation Sensor), a novel on-chip sensor that exploits the difference in the impact of the momentary power consumption on the signal propagation delay of two signal paths to induce a pulse that is sensitive to power fluctuations. The design of PPWM uses a single bit to record the presence or the absence of a noticeable pulse which is correlated with the power consumption. Further, using PPWM, this paper demonstrates the possibility of exploiting such single bit information to successfully attack a cryptographic circuit. In addition to being area efficient, PPWM requires significantly less storage and communication bandwidth to transfer power fluctuation information to the attacker when compared to existing on-chip sensors [UJS+21]. Less storage and communication bandwidth requirement of PPWM makes it the ideal on-chip sensor for contexts where the attacker has to rely on covert communication channels to communicate with the sensor.

Paper Contributions:

- A new type of core logic design to construct on-chip sensors for RPA attacks is presented. The proposed core logic design exploits the differences in the power consumption induced fluctuations on the signal propagation delay of two signal paths to create narrow pulses and thereby measure the power fluctuations.
- Introducing PPWM, an area-efficient sensor that uses the newly proposed core logic design along with an asynchronously cleared storage mechanism to store one of the binary states: the presence of a considerable pulse or the absence of a considerable pulse.
- With an extensive set of experiments, demonstrate the use of single-bit information to attack cryptographic circuits and support the experiment results using design insight analysis. Experiment results also demonstrate the possibility of using PPWM to perform successful attacks while keeping the communication bandwidth to a minimum, making PPWM the preferable on-chip sensor for the attackers in low communication bandwidth setups.
- For the first time, propose that lower sensor resolution does not significantly affect the RPA attack efficacy and support the proposition using experiment results and design insights.
- Evaluate the performance of PPWM under various design parameters and operating conditions.

Paper Organization: Subsequent sections of this paper are organized as follows. Section 2 briefly introduces some essential background knowledge required to apprehend this paper. Prior works related to the work proposed in this paper are presented in Section 3.

The design and construction of PPWM along with its working principle are presented in Section 4. Section 5 details the experiment setups used to evaluate the performance of PPWM. Results are presented in Section 6 which is followed by the conclusion in Section 7. As supplementary information, a theoretical analysis of the insights that contributed to designing PPWM is presented in Appendix A.

2 Background

2.1 Correlation Power Analysis

Correlation power analysis (CPA) [BCO04] is a powerful statistical power analysis technique used by cryptanalysts to infer secret keys used in cryptographic modules. In a typical CPA attempt, the attacker partitions the secret key into bytes to obtain a tractable search space to exhaust all possible key guesses. Then, for each guess for a key byte, the attacker models the power consumption for a targeted logical operation in the cryptographic module. The promising model used in a typical CPA attempt is the Hamming distance model [BCO04] in which the power consumption is modeled by the number of bits changed by the targeted operation. After modeling the power consumption for each key byte guess, the attacker estimates the correlation factor (also known as the Pearson correlation coefficient) of the observed power consumption and the modeled power consumption for all the key guesses [BCO04]. With sufficiently many power consumption observations, the correlation factor of the correct key byte guess eventually emerges from the rest of the guesses. Finally, the attacker assembles the correct key byte guesses to discern the secret key of the targeted cryptographic module.

2.2 Key Rank

Key rank (KR) [VCGS13] is a metric used by cryptographic researchers to measure the success of a key inference attempt. Following related work in the domain [UJS+21], this paper adopts KR as the performance metric for evaluating experiment results.

KR calculates the rank of the correct key, given all possible key candidates are sorted based on their likelihoods derived from the output of the key inference attempt. Thus, KR is an estimate of the number of brute force tries the attacker has to perform in order to identify the correct key. Typically, secret keys used in cryptographic modules are at least 128 bits making the process of ranking all key candidates infeasible. Consequently, researchers have proposed key rank estimation algorithms [Gro19] which can be used to obtain a key rank estimate with a bounded error without enumerating all key candidates. A lower KR implies that the attacker must perform fewer brute force tries which in turn implies an easier secret key recovery. Readers are advised to refer [Gro19] for an in-depth understanding of key rank estimation algorithms.

2.3 Threat Model

Similar to prior works, such as [SGMT18] and [UJS+21], this paper assumes the context of a shared FPGA as the threat model. Complex logic designs often feature design components sourced from 3rd party vendors. Deployment of such complex logic designs on FPGAs results in the resources on the FPGA being shared among components from various origins, effectively making the FPGA a shared FPGA. Further, with the increased interest in supporting multi-tenancy in cloud FPGAs to maximize resource utilization [IAS18], shared FPGAs are likely to become more prevalent in the near future.

As shown in Figure 1, the shared FPGA assumed in this paper houses two shells: (1), the attacker shell and (2), the victim shell. The attacker shell contains a PPWM to monitor the power fluctuations occurring on the FPGA. The victim shell contains a

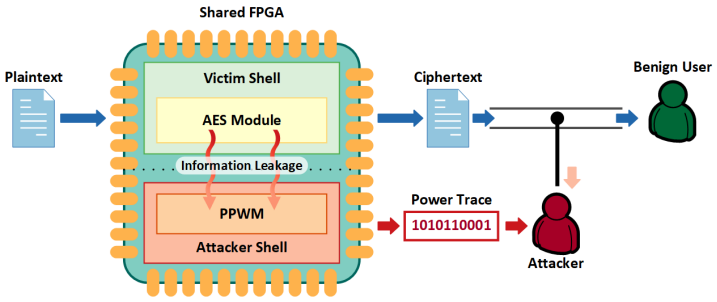

Figure 1: Threat Model.

cryptographic module that implements Advanced Encryption Standard (AES) [NIS01] in hardware. The cryptographic module hereinafter referred to as the AES module encrypts repeatedly the plaintext supplied to it and sends the ciphertext via a medium transparent to the attacker. The attacker intends to perform an RPA attack and therefore, logs the ciphertext along with the power fluctuations sensed by PPWM during the corresponding encryption. Then, the logged information is subjected to CPA to discern the secret AES key.

3 Related Work

Current research on the security of FPGA deployments in cloud services examines potential threats posed by cloud service vendors, co-users, 3rd party intellectual properties (IPs) and toolchains. Such threats include but are not limited to IP theft [BSH12], long wire crosstalk exploits [RPD+18], fault-injection attacks [KGT18], power-based covert channels [GRS19], thermal-based covert channels [TS19] and RPA attacks [SGMT18]. RPA attacks utilize on-chip sensors to exploit the power consumption induced by the momentary instability in the power delivery network (PDN) to discern the secret information of a victim [UJS+21].

On-chip sensors proposed by researchers thus far exploit the dependency of signal propagation delay on the momentary power consumption to measure the power fluctuations over time and they can be grouped into two types based on their core logic design: (1), delay lines [ZS+13] and (2), ring oscillators [ZH12]. Delay line based sensors, such as TDC [SGMT18] and VITI [UJS+21], use the impact of the propagation delay changes on a signal along the delay line to measure the dynamic power fluctuations. RO based sensors [GDTL19] measure the dynamic power fluctuation by utilizing a counter to measure the impact of the power fluctuation on the oscillation frequency of the RO.

RPA attacks were proposed by Schellenberg et al. in [SGMT18] using a TDC sensor featuring a 23 FPGA slices long delay line. The delay line of the TDC sensor featured a seven slices long initial delay segment and a 16 FPGA slices long observable segment. The initial delay segment was implemented using lookup tables (LUTs) and latches to lower the area overhead of the TDC sensor. The observable segment was implemented using carry-chains to obtain the finest resolution for the sensor reading. The Xilinx FPGA used in [SGMT18] makes use of CARRY4 primitives for carry-chains and each CARRY4 primitive is a 4-bit long portion of the carry-chain producing a 4-bit output. As the observable segment was 16 FPGA slices long with each slice having a CARRY4 primitive, the delay line of the TDC sensor used in [SGMT18] generated a 64-bit reading. The notion of using a phase-shifted clock to build smaller TDC sensors was presented by Zick et al. in [ZS+13]. Krautter et al. in [KGT20] proposed a TDC sensor comprising a chain of coarse calibration and fine calibration slices which allows the TDC sensor to self-calibrate at runtime. However, the authors of both [ZS+13] and [KGT20] used carry-chains to

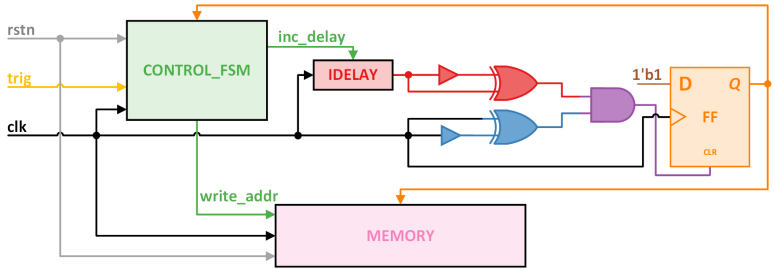

Figure 2: The Abstract Design of the Proposed PPWM Sensor.

preserve the resolution per bit in the output of the respective TDC sensor.

The first RPA attack using RO based sensors was demonstrated by Gravellier et al. in [GDTL19]. Each RO based sensor used in [GDTL19] was constructed with LUTs and flip-flops with each sensor having an RO consuming a LUT and a counter consuming eight flip-flops and a LUT. The counter produced an 8-bit output capturing the RO frequency shift incurred due to the momentary instability in the PDN. As demonstrated in the results of [GDTL19], successful RPA attacks required several RO based sensors each generating an 8-bit output.

VITI, proposed by Udugama et al. in [UJS+21], is a compact and self-calibrating delay line based sensor build on LUTs, flip-flops and adjustable input delay elements. In contrast to previous delay line based on-chip sensor designs, the authors of [UJS+21] used LUTs for the delay line to save area at the cost of the resolution of the sensor reading. By using adjustable input delay elements available on FPGAs, the authors of [UJS+21] designed an automatic calibration mechanism to detect the effects of momentary instability in the PDN without requiring trial-and-error placement and routing attempts. The length of the delay line (therefore the bit width of the sensor reading) of VITI was parameterized and the experiment results presented in [UJS+21] are based on a 4-bit VITI sensor producing a 4-bit output.

Inspired by the trade-off between area cost and resolution presented in [UJS+21] and the preliminary experiments detailed in Appendix A, this paper proposes PPWM, an area-efficient on-chip sensor that creates power consumption sensitive pulses and records the created pulses as single bit information that can be used to perform successful RPA attacks. Since each reading from PPWM is a single bit, the communication bandwidth requirement of PPWM is minimal and therefore, PPWM is perfectly suited for deployment in RPA attack contexts where the communication bandwidth is limited.

4 The PPWM Sensor

This section details the design, construction and working principle of PPWM. The abstract design of PPWM is presented in Subsection 4.1 followed by Subsection 4.2 which explains the working principle of PPWM. Subsection 4.3 explains the aim of calibrating PPWM. Subsection 4.4 presents information on constructing PPWM on Xilinx Spartan-6 and UltraScale FPGAs.

4.1 Abstract Design

Illustrated in Figure 2 is the abstract design of the proposed PPWM which consists of a finite state machine (labeled CONTROL_FSM) for controlling, a flip-flop (labeled FF) for capturing the power fluctuations, a memory (labeled MEMORY) for storing captured power fluctuations, an adjustable input delay element (labeled IDELAY) for

runtime calibration and an AND gate connected to a pair of XOR gates each of which has one buffered input for creating pulses. Adjustable input delay elements are primitive components available on FPGAs to allow aligning incoming data to the FPGA. The use of adjustable input delay elements for runtime calibration of on-chip sensors was presented in [UJS+21] and by following the adjustable input delay element based calibration algorithm proposed in [UJS+21], the runtime calibration mechanism for PPWM is implemented within CONTROL_FSM. The creation of the power fluctuation sensitive pulses using the gate network and recording the pulses using the flip-flop is detailed in Subsection 4.2.

As shown in Figure 2, the inputs to PPWM are a reset signal, a trigger signal and a clock signal which are labeled *rstn*, *trig* and *clk* respectively. The reset signal is used to reset the control logic in CONTROL_FSM and the stored values in MEMORY. The start of the encryption execution on the targeted cryptographic module is signaled by the trigger signal. The clock signal is used to clock CONTROL_FSM, MEMORY and FF. Further, the clock signal is routed via the IDELAY element and the gate network to obtain a signal to drive the asynchronous clear input of the flip-flop FF. The output of FF is supplied to MEMORY to be written to the address pointed by *write_addr*. Following prior work, such as [UJS+21], a single value written to MEMORY would hereinafter be referred to as a *sample*. A series of samples sequentially stored in MEMORY and corresponding to a single encryption execution on the targeted cryptographic module would hereinafter be referred to as a *power trace* from PPWM.

4.2 Power Fluctuation Measurement with Pulses

PPWM is an on-chip power sensor made of digital primitives available on FPGAs. Thus, PPWM makes use of the byproducts emanating from dynamic power fluctuations, such as propagation delay variations, to estimate dynamic power fluctuations occurring on the FPGA. PPWM captures the propagation delay variations on the FPGA by asynchronously clearing a flip-flop using a propagation delay sensitive signal derived from two pulse signals propagating along two signal paths.

Subfigures (b) and (c) of Figure 3 show the expected waveforms of the signals labeled in Figure 3 (a). As seen from the portion of PPWM depicted in Figure 3 (a), signal a is the output of the buffer labeled as BUF_2 which is driven by the *clk* signal. When the PDN is not stressed (sometimes referred to as a stable PDN), the buffer induces a δ_{BUFFER} delay on its output signal. Therefore, as shown in Figure 3 (b), a has a δ_{BUFFER} offset from the *clk* signal in the case of an unstressed PDN. However, when the PDN is stressed (sometimes referred to as an unstable PDN), the delay induced by BUF_2 increases to Δ_{BUFFER} causing the offset of a to grow to Δ_{BUFFER} as depicted in Figure 3 (c). As shown in Figure 3 (a), the XOR_2 gate which produces signal b, takes a and *clk* as the input signals. The dotted line waveform immediately below the waveform of a in both Figure 3 (b) and Figure 3 (c) shows the result of XORing *clk* signal with respective signal a, for both unstressed and stressed PDN scenarios, respectively. Since the XOR_2 gate itself is a logic element that induces power consumption sensitive propagation delay, signal b is identical to *clk XOR a* waveform except for the offset denoted by Δ_{XOR} (Figure 3 (c)) and δ_{XOR} (Figure 3 (b)) for instances corresponding to a stressed PDN and an unstressed PDN, respectively.

As depicted in Figure 3 (a), signal c is the output of XOR_1 which is followed by BUF_1 in a configuration similar to that generates signal b. However, as opposed to signal b, which is derived directly from the *clk* signal, signal c is derived from a delayed *clk* signal arriving through the IDELAY element. Comparing the waveforms of b and c corresponding to a stressed PDN with that of an unstressed PDN, it could be concluded that, when the PDN is unstressed, the buffer element induces a smaller delay (δ_{BUFFER}) and thereby results in a narrower pulse width of size δ_{BUFFER}. On the other hand, when the PDN is stressed, the delay induced by the buffer element increases (Δ_{BUFFER}) resulting in a broader pulse

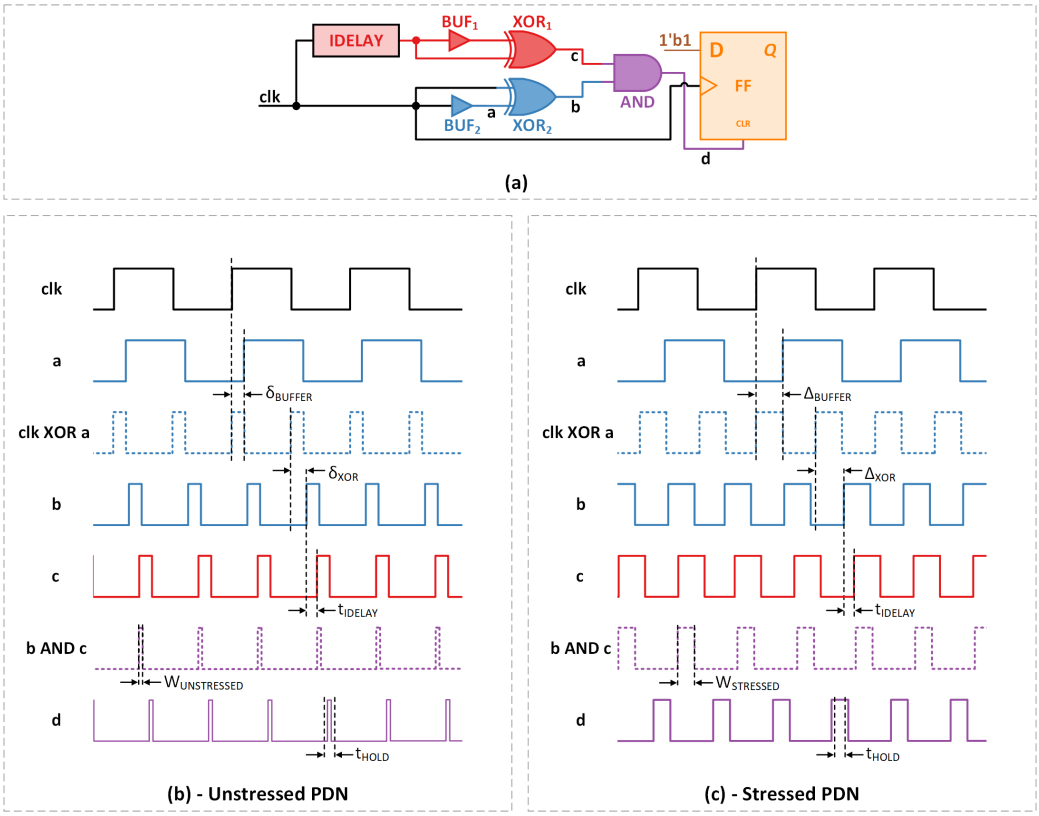

Figure 3: Working Principle of PPWM Sensor Showing (a), Labelled Block Diagram; (b), Waveforms for Unstressed PDN; and (c), Waveforms for Stressed PDN.

width of size Δ_{BUFFER}. Please note that the width of the pulse observed in signal b (also, applicable to signal c) in Figure 3 (a) is beyond the control of the deployer of PPWM. Therefore, the subsequent explanation presents the method of deriving a signal with a deployer controllable pulse width using signals b and c.

Let t_{IDELAY} denote the delay induced by the IDELAY element and the t_{IDELAY} value demonstrated in Figure 3 (b) and Figure 3 (c) be the value chosen by the calibration state machine to satisfy the conditions subsequently stated in this subsection. For simplicity, assume that the power fluctuations do not cause a noticeable change to the delay induced by the IDELAY element and therefore the offset between signal b and signal c remains the same regardless of the condition of the PDN. Therefore, performing AND operation on signals b and c would result in waveforms that resemble the plots against the label b *AND* c in Figure 3 (b) and Figure 3 (c). Signal d is essentially the result of ANDing signals b and c (therefore, similar to b *AND* c waveforms on each of Figure 3 (b) and Figure 3 (c)) with the delay of AND gate added to the output. Note that when the PDN is stressed, the delay induced by AND gate too gets increased when compared to the delay induced when the unstressed PDN.

The flip-flop on PPWM, FF, is clocked with *clk* and the D input is supplied with a constant logic one signal resulting in FF refreshing its storage to a logic one at every rising edge of *clk*. Thus, if FF is not cleared with the asynchronous clear input, the output of FF would be a logic one. Since signal d drives the asynchronous clear input of FF, FF could be cleared with a pulse on d if the width of the pulse on d satisfies the hold time requirement of the asynchronous clear input (denoted t_{HOLD}). As shown in Figure 3 (b), the width of the pulse on signal d ($W_{UNSTRESSED}$) is narrower when PDN is unstressed

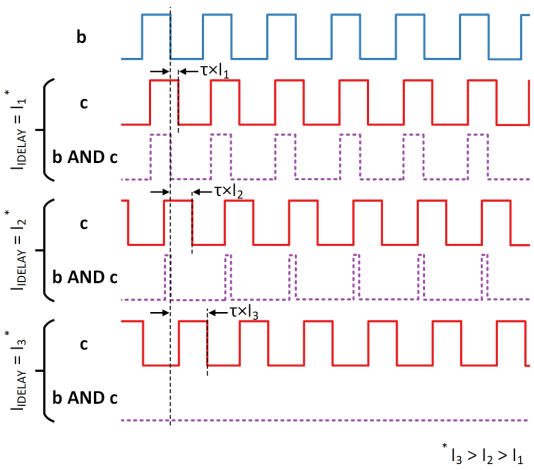

Figure 4: Calibrating t_{IDELAY} with CONTROL_FSM.

and it could be formally expressed as shown in Equation 1. Conversely, when the PDN is stressed, the pulse width on signal d ($W_{STRESSED}$) becomes relatively broader as shown in Figure 3 (c). The mathematical expression in Equation 2 formally describes the magnitude $W_{STRESSED}$.

$$W_{UNSTRESSED} = \max\left(0, \delta_{BUFFER} - t_{IDELAY}\right) \tag{1}$$

$$W_{STRESSED} = \max\left(0, \Delta_{BUFFER} - t_{IDELAY}\right) \tag{2}$$

Hence, if t_{IDELAY} could be adjusted such that the pulse width on signal d is less than t_{HOLD} when the PDN is unstressed while it is at least t_{HOLD} when the PDN is stressed, then, signal d could be used to selectively clear FF to reflect the power consumption of the FPGA. The same condition can be expressed mathematically by the inequality shown in Equation 3. As a result, the output of FF could be used to estimate dynamic power fluctuations occurring on the FPGA.

$$\begin{gathered} W_{UNSTRESSED} < t_{HOLD} < W_{STRESSED} \\ \max\left(0, \delta_{BUFFER} - t_{IDELAY}\right) < t_{HOLD} < \max\left(0, \Delta_{BUFFER} - t_{IDELAY}\right) \end{gathered} \tag{3}$$

Please note that the assumption that power fluctuations do not cause a noticeable change in t_{IDELAY} is for the sake of simplicity and the working principle of PPWM does not rely on the aforesaid assumption. As shown in Equation 3, the sufficient condition for the functioning of PPWM is based on the difference between the delay induced by the buffer element and the delay induced by the IDELAY element. Hence, any shift in the value of the delay induced by the IDELAY element is immaterial to the functioning of PPWM as long as the inequality in Equation 3 is satisfied. Therefore, the method of power fluctuation measurement presented in this section can also be applied to FPGAs where the t_{IDELAY} is susceptible to voltage and temperature changes. However, please take note that newer FPGAs, manufactured by vendors such as Xilinx, feature IDELAY implementations that maintain t_{IDELAY} by compensating for voltage and temperature changes [Xil21].

4.3 Calibrating t_{IDELAY} with CONTROL_FSM

Adjustable input delay elements, such as IDELAY, can induce runtime adjustable delay on a signal passing through it and the amount of delay induced can be controlled externally.

Table 1: Mapping Abstract Elements of PPWM to Xilinx FPGAs.

Abstract Element	Primitive used in the FPGA	
	Xilinx Spartan-6 [Xil13]	**Xilinx UltraScale [Xil18]**
IDELAY	1 × IODELAY2	1 × IDELAYCTRL and 1 × IDELAYE3
MEMORY	1 × RAMB8BWER (Described using BRAM_SDP_MACRO macro)	1 × RAMB18E2 (Described using XPM_MEMORY_SDPRAM macro)
CONTROL_FSM	LUTs and flip-flops (Described at register transfer level)	
FF	1 × FDCE	
Logic gate network for creating pulses (2 × buffer + 2 × XOR gate + 1 × AND gate)	2 × LUT6_2 and 1 × LUT6 (Shown in Figure 5)	

PPWM uses a state machine implemented within CONTROL_FSM to control the IDELAY element and as shown in the abstract design in Figure 2, *inc_delay* from CONTROL_FSM is used to signal the IDELAY element whether to hold or increase the induced delay.

Referring to the signals in Figure 3 (a), Figure 4 illustrates the behavior of signal b, signal c and the hypothetical signal b *AND* c for different delay indices of the IDELAY element. Waveforms illustrated in Figure 4 assume that the stress level of the PDN is similar across the three instances under consideration. Let the delay index of the IDELAY element be denoted by I_{IDELAY} and the added delay per increment in the delay index be denoted by τ. When I_{IDELAY} is I_1, t_{IDELAY} is approximately $\tau \times I_1$ and therefore signal c is approximately $\tau \times I_1$ delayed when compared to signal b. $\tau \times I_1$ delay of signal c is an approximate value since the delay induced by buffer and XOR gate configurations generating signal b and signal c are generally not identical and the value of τ varies slightly for different delay indices. When I_{IDELAY} is I_2 and I_3 where $I_1 < I_2 < I_3$, signal c is approximately $\tau \times I_2$ and $\tau \times I_3$ delayed, respectively. As illustrated in Figure 4, when the offset between signal b and signal c grows, the pulse width in the hypothetical signal b *AND* c narrows. Further, as illustrated for the case of $I_{IDELAY} = I_3$, the pulse on b *AND* c disappears completely if the offset between signal b and signal c exceeds a certain threshold. Hence, using the calibration state machine in the CONTROL_FSM, the value of I_{IDELAY} is adjusted to obtain a t_{IDELAY} that satisfies the condition stated in Equation 3.

To this end, the calibration state machine captures a power trace with a fixed I_{IDELAY} setting and examines whether the power trace is made up of samples with a value of one and samples with a value of zero. If the power trace contains samples with different values, the calibration state machine could deduce that t_{IDELAY} satisfies the condition in Equation 3 and therefore FF has been selectively cleared based on the PDN stress level at the time of capturing the sample. Upon reaching the aforementioned observation, the calibration state machine could stop adjusting I_{IDELAY}. If the power trace is made up of identical values, PPWM is yet to be calibrated and therefore I_{IDELAY} should be adjusted until it reaches the calibrated condition.

In summary, the runtime adjustability of the pulse width (shown in Figure 4) along with the phenomenon of power consumption modulating pulse width (shown in subfigures (b) and (c) of Figure 3) are used in PPWM to obtain the asynchronous clear signal for FF which is labeled d in Figure 3 (a).

4.4 Mapping to Xilinx FPGAs

Table 1 succinctly presents the primitives used in Xilinx Spartan-6 and Xilinx UltraScale architectures to implement individual abstract elements of PPWM. The first column of Table 1 lists the elements shown in the abstract design of PPWM. The second and third columns of Table 1 show the FPGA primitives used to implement the corresponding

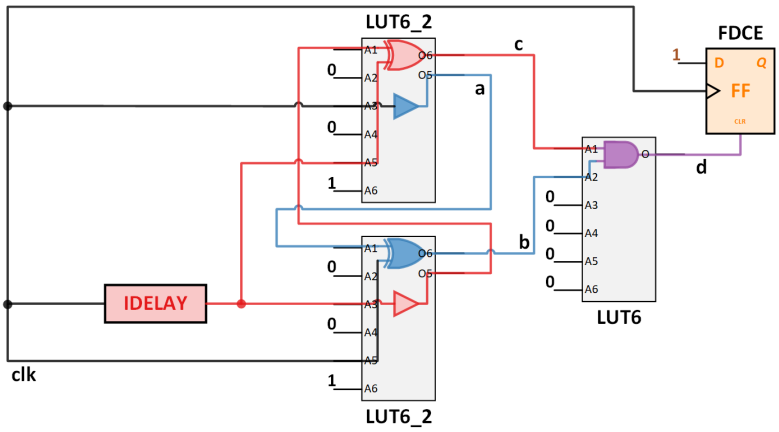

Figure 5: Implementation of Logic Gate Network on LUTs.

abstract element in Spartan-6 and UltraScale architectures, respectively. Xilinx Spartan-6 is a low-cost and low-power FPGA architecture [Xil11]. On the other hand, Xilinx UltraScale is a high-performance architecture [Meh15] that powers FPGAs used in cloud services, such as AWS EC2. Thus, the ability to construct PPWM on both Spartan-6 and UltraScale architectures demonstrates the deployability of PPWM on a wide range of Xilinx FPGAs. Please note that similar primitive components are available on FPGAs from other manufacturers. For example, the IDELAY element can be implemented with Programmable Delay Chains [Alt16] on Intel FPGAs and delay chain within Input Logic Block [Lat10] on Lattice FPGAs.

The five-element logic gate network for creating pulses, namely the two XOR gates, the two buffers and the AND gate, are implemented on two LUT6_2 (Six-input, Two-output, Look-Up Table) primitives and a LUT6 (Six-Input Lookup Table with General Output) primitive to maximally harness the propagation delay difference and to minimize the resource consumption. Figure 5 shows the mapping of the pulse-creating logic gate network to the three primitive components. Each LUT6_2 primitive implements an XOR gate along with the buffer from the opposite signal path. The AND gate is implemented on the LUT6 primitive. Signals labeled with *a*, *b*, *c*, *d* and *clk* in Figure 5 refer to the signals with the same label in Figure 3 (a).

5 Experiment Setup

This section details the two experiment setups that were designed to evaluate the efficacy of PPWM as an RPA attack targeted on-chip sensor. The experiment setups mimic the threat model detailed in Figure 1 where PPWM resides on the same FPGA as the victim AES module. The AES module features a 128-bit datapath and encrypts a 128-bit plaintext, that it receives, using a 128-bit key in 10 transformation rounds. The AES module takes one clock cycle for each transformation round and after the 10^{th} round (i.e., after 10 clock cycles) the ciphertext is generated. The deployed PPWM generates power traces by sensing the power fluctuations. The obtained power traces are used to perform CPA targeting the `SubBytes()` transformation [NIS01] of the last (10^{th}) round of the victim AES module and the success of the CPA is measured using the KR performance metric. The success of CPA is used as a surrogate for the efficacy of PPWM as an RPA attack targeted on-chip sensor.

The first of the two experiment setups, hereinafter referred to as Experiment Setup I, deploys a Xilinx Spartan-6 mapped PPWM to attack the victim AES module. On the

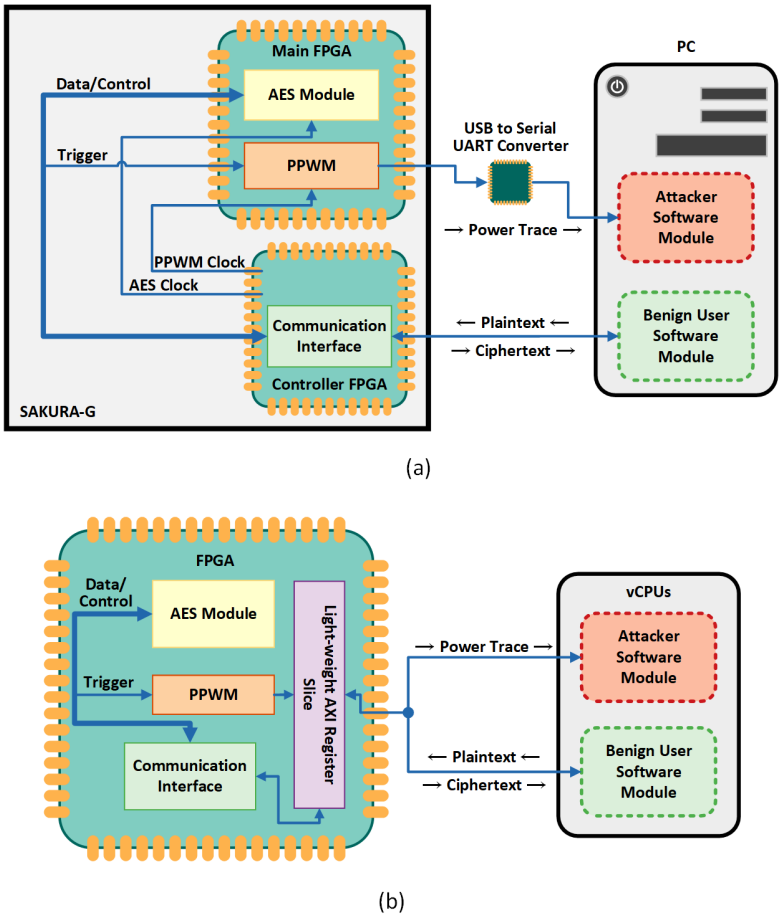

Figure 6: Diagram of (a), Experiment Setup I; and (b), Experiment Setup II.

other hand, the second experiment setup which hereinafter is referred to as the Experiment Setup II, deploys a Xilinx UltraScale mapped PPWM for attacking the victim AES module. Detail descriptions of Experiment Setup I and Experiment Setup II are presented in the following subsections.

5.1 Experiment Setup I

Shown in Figure 6 (a) is the diagram of Experiment Setup I which consists of a personal computer (PC) and a SAKURA-G [Sat14] FPGA board featuring two Xilinx Spartan-6 FPGAs: an XC6SLX75-2CSG484C main FPGA and an XC6SLX9-2CSG225C controller FPGA. The main FPGA of the SAKURA-G board is used to deploy the victim shell containing the AES module together with the attacker shell containing Spartan-6 mapped PPWM. The controller FPGA is used to deploy a logic circuit that establishes communication between the AES module and the PC for sharing plaintexts and ciphertexts. Also, the logic circuit on the controller FPGA generates the clock signals required for all the logic circuits on the SAKURA-G board. The benign user who gets served by the AES module and the attacker who intends to attack the AES module by collecting power traces are implemented as two independent software modules on the PC. Power traces from PPWM are sent to the attacker software module on the PC via an FTDI FT232RL USB to serial UART converter [FTD]. All logic circuits deployed on the SAKURA-G board

Table 2: Design Parameters Applied on SAKURA-G Main FPGA Used in Experiment Setup I.

Evaluated Parameter	PPWM		AES Module	
	Placement	Frequency (MHz)	Placement	Frequency (MHz)
Placement of PPWM *(Results: Subsection 6.1)*	Placements labeled with 1 - 20 in Figure 7 (a)	96	Placement labeled with AES in Figure 7 (a)	6
Placement of AES module *(Results: Subsection 6.2)*	Placement labeled with PPWM in Figure 7 (b)	96	Placements labeled with 1- 5 in Figure 7 (b)	6
Frequency of AES module *(Results: Subsection 6.3)*	Placement labeled with PPWM in Figure 7 (c)	96	Placement labeled with AES in Figure 7 (c)	[6, 12, 24, 48, 96]
Ambient Temperature *(Results: Subsection 6.4)*	Placement labeled with PPWM in Figure 7 (c)	96	Placement labeled with AES in Figure 7 (c)	6

were synthesized, implemented and configured using Xilinx Design Tools ISE 14.7. The AES module consumed ≈ 3084 LUTs and ≈ 650 flip-flops while the number of LUTs and flip-flops consumed by PPWM were ≈ 53 and ≈ 44, respectively. Therefore, note that the number of FPGA slices consumed by PPWM is ≈ 13 and the number of FPGA slices consumed by VITI, TDC and RO based sensors (required 64 for similar efficacy) as reported in [UJS+21] are 8, 34 and 128, correspondingly.

Experiment Setup I was used to evaluate the relationship between the efficacy of PPWM and the design parameters, such as placement of PPWM (results in Subsection 6.1), placement of AES module (results in Subsection 6.2) and frequency of AES module (results in Subsection 6.3). Further, Experiment Setup I was used to assess the impact of environmental conditions, such as temperature (results in Subsection 6.4). Design parameters applied for each Experiment Setup I configuration is shown in Table 2. From left to right, columns of Table 2 show the assessed design parameter followed by the placement of PPWM, the frequency of PPWM, the placement of AES module and the frequency of AES module. For each Experiment Setup I configuration, the attacker software module collected 100,000 power traces along with the corresponding ciphertext. To this end, both PPWM and the AES module were triggered upon receiving a plaintext from the benign user software module on the PC. The plaintexts were randomly generated and sent via the communication interface on the controller FPGA of the SAKURA-G board. After being triggered, PPWM measured the power fluctuations while the AES module encrypted the received plaintext. Subsequently, the ciphertext computed by the AES module was sent back to the benign user software module and the same was captured and logged by the attacker software module. Then, the attacker shell on the FPGA transferred the power trace to the attacker software module which saves it along with the ciphertext. Once 100,000 power traces along with corresponding ciphertexts were collected, they were used to infer the key of the AES module by performing CPA with different numbers of power traces starting at 1,000 power traces and incremented up to 100,000 power traces in steps of 1,000 power traces.

5.2 Experiment Setup II

Figure 6 (b) shows the diagram of Experiment Setup II which was deployed on AWS EC2. Each AWS EC2 F1 instance in Experiment Setup II featured eight virtual processors

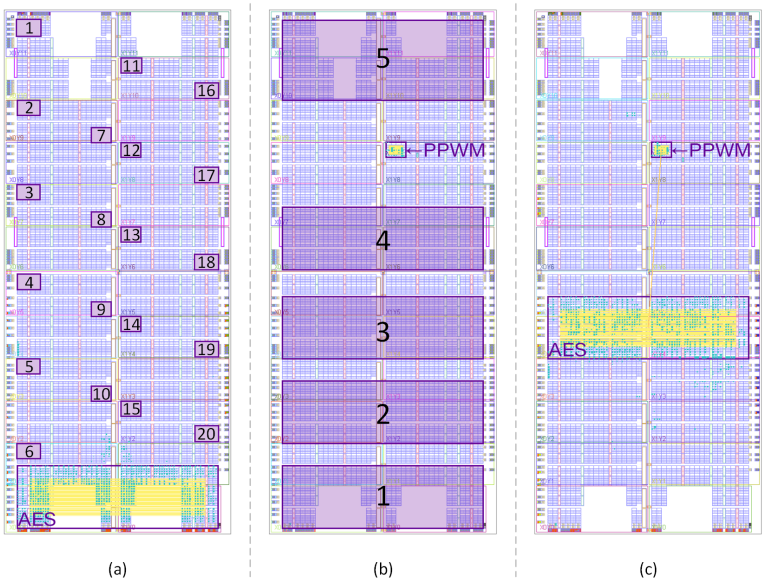

Figure 7: (a) - (c) PPWM and AES Module Placements on the Floorplan of SAKURA-G Main FPGA Used in Experiment Setup I.

(vCPUs) and an XCVU9P-FLGB2104-2-I Xilinx UltraScale+ FPGA. [1] The vCPUs executed attacker and benign user software modules which were similar to that of Experiment Setup I. A logic circuit design that emulated the functionality of the SAKURA-G board in Experiment Setup I was deployed on the FPGA and in the absence of dedicated communication interfaces for the attacker and the benign user, all communication between FPGA and vCPUs was performed via a Light-weight AXI Register Slice [Xil20] which was also implemented on the FPGA. The logic design on the FPGA was synthesized and implemented using Xilinx Vivado v2021.2 which was provided with the FPGA Developer AMI [AWS] available at the time of the experiment.

In addition to demonstrating the viability of deploying PPWM on cloud FPGAs and carrying out RPA attacks using power traces obtained from a cloud-deployed PPWM, Experiment Setup II was used to experimentally verify the portability of PPWM across FPGA instances (results in Subsection 6.5) and thereby evaluate its tolerance to process variation. Experiment Setup II was operated in a single configuration where the AES module and the PPWM were clocked at 10 MHz and 200 MHz, respectively. The procedure followed to collect pairs of power traces and ciphertexts on Experiment Setup II was similar to that of Experiment Setup I, with the exception that the attacker software module collected 500,000 pairs of power traces and ciphertexts instead of 100,000. After collecting 500,000 pairs of power traces and ciphertexts, the collection was transferred to a local PC to be used to perform CPA and infer the key of the AES module with different numbers of power traces from 10,000 power traces to 500,000 power traces in increments of 10,000.

6 Results

This section presents and discusses the results of the experiments that evaluated the impact of design parameters, such as placement and frequency of the logic components, and natural parameters, such as ambient temperature and process variation of the FGPA,

[1] Please note that both Xilinx UltraScale+ and Xilinx UltraScale FPGAs implement the same architecture often referred to as Xilinx UltraScale.

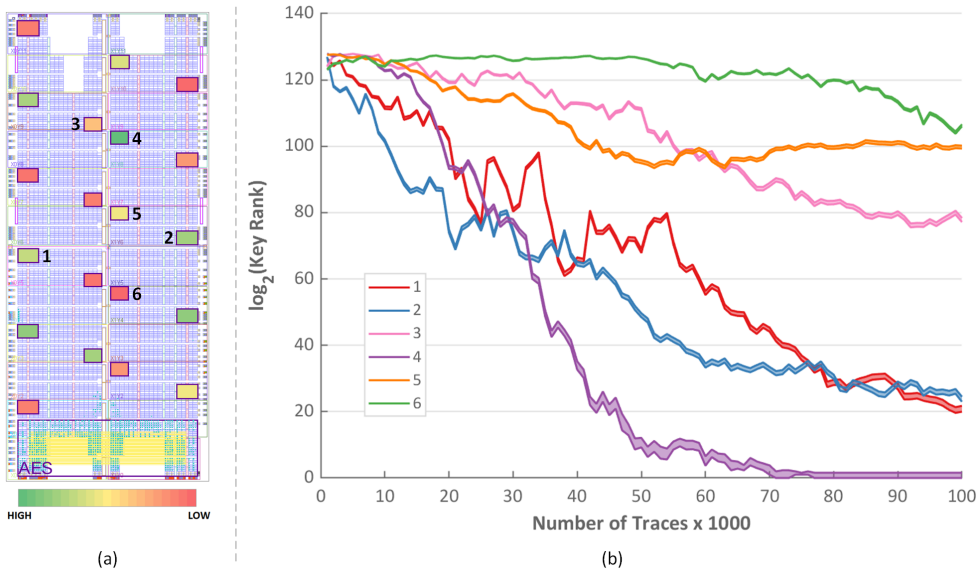

Figure 8: Results - Placement of PPWM. Shown in (a), Heat Map Based on Attack Efficacy; and (b), Variation in KR over Number of Power Traces.

on the efficacy of CPA based key inferences carried out with PPWM. Graphs plotted in this section illustrate the variation of KR, which measures the efficacy of the key inference, against the number of power traces. An efficacious parameter configuration facilitates KR to drop to unity with a fewer number of traces when compared to other parameter configurations and despite relative differences in efficacy, results in this section show that PPWM can be used to generate power traces that realize successful RPA attacks while requiring minimal communication bandwidth.

6.1 Placement of PPWM

Results of the experiments carried out with Experiment Setup I to evaluate the impact of the placement of PPWM on the efficacy of key inferences are shown in the two subfigures of Figure 8. Subfigure (a) of Figure 8 is a heat map overlayed on top of the floorplan of the main FPGA of the SAKURA-G board that indicates the relative efficacy of 20 different PPWM placements, assessed based on KR calculated with 50,000 power traces. Subfigure (b) of Figure 8 plots the variation in KR over the number of power traces for six representative PPWM placements which are identified by the label next to the corresponding placement in subfigure (a) of Figure 8.

As seen from the heat map in Figure 8 (a), PPWM placements with different efficacies are scattered through the floorplan and a relationship between placement and efficacy is unobservable. Hence, the variation in the efficacy can be hypothesized to be a result of the changes in routing delay between logic elements in the gate network of PPWM which in turn is a result of rerouting the logic elements to fit the placement. Given two PPWM placement configurations, it is highly likely to have two different sets of values for the routing delay between logic elements in the gate network of PPWM. Since adjustable input delay elements, such as IODELAY2, take discrete steps (denoted by t in Subsection 4.2) when incrementing or decrementing the induced delay, a change in the routing delay between the logic elements in the gate network can alter either the delay index or the threshold index of the calibrated PPWM. As discussed in Subsection A.2, the threshold index significantly impacts the efficacy of CPA based key inferences and it could be deduced

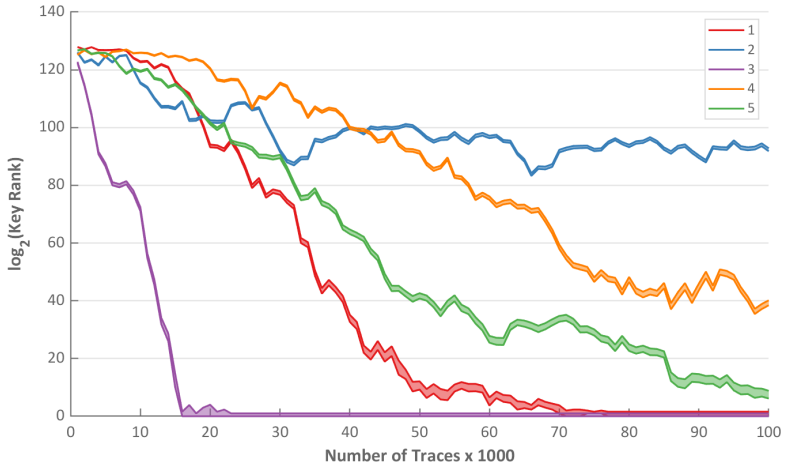

Figure 9: Results - Placement of AES Module.

that the efficacy variation observed in Figure 8 (a) is mainly due to the altered threshold index.

As shown in Figure 8 (b), KR generally tends to drop with the increasing number of power traces, though the rate of the drop varies from configuration to configuration. Further, as elaborated in Subsection 4.2, PPWM uses a single bit to record a single sample and it requires a single bit of information to be transmitted to the attacker to perform the key inference. Thus, from Figure 8 (b), it could be realized that successful CPA based key inferences can be performed by correlating to power consumption information stored in a single bit and therefore communication channels with low bandwidths, such as covert communication channels, can be constructed and utilized by an attacker to perform successful RPA attacks.

6.2 Placement of AES Module

Figure 9 shows the results of the experiments carried out using Experiment Setup I to evaluate the impact of the placement of the AES module on the efficacy of CPA based key inferencing. The results are plotted to show the variation in KR over the number of power traces for the five different placements of the AES module which are labeled by the corresponding placement in Figure 7 (b).

Since examining the plots in Figure 9 along with the floorplan in Figure 7 (b) does not provide any observable relationship between placement and the efficacy of the key inference attempt, it could be hypothesized that the prominent factor that causes the variation in the efficacy is the change in routing delay between logic elements in the gate network of PPWM. This hypothesis falls in line with the observations and the explanation in Subsection 6.1 that describes the impact of the placement of PPWM. Moreover, Figure 9 also shows that KR is generally tending to drop with the increasing number of power traces and successful RPA attacks can be performed utilizing single-bit power consumption information that can be transmitted to the attacker over a low bandwidth medium.

6.3 Frequency of AES Module

Plotted in Figure 10 are the outcomes of the experiments conducted to assess the impact of the frequency of the AES module on the efficacy of key inferencing. The experiments were conducted using Experiment Setup I and the AES module and PPWM were placed as shown in Figure 7 (c). Since the slightest change in design files causes the Xilinx toolchain

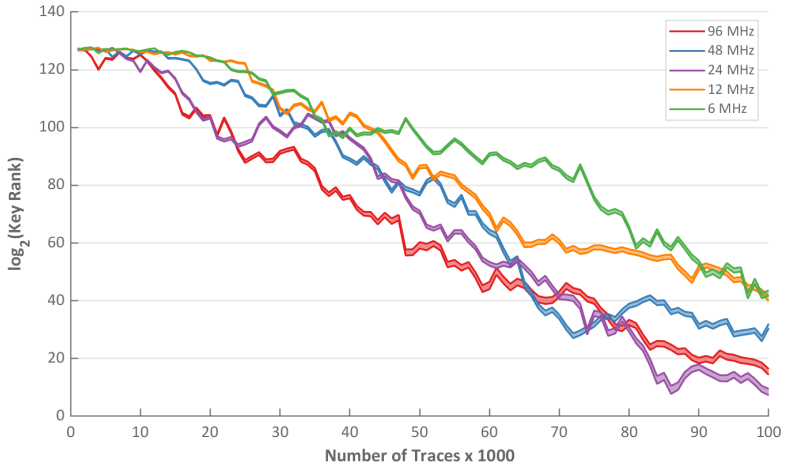

Figure 10: Results - Frequency of AES Module.

to change the placing and routing of components, experiments detailed in this subsection were performed using a single main FPGA logic design featuring a switchable frequency setting to enable communication despite the internal frequency. In this manner, it was possible to eliminate any irregularities resulting from rerouting produced changes in the routing delay between logic elements of the gate network of PPWM. Please note that the clock signals were generated using the Control FPGA and therefore the frequencies could be changed by resynthesizing the Control FPGA logic design after making appropriate changes. Further, please note that the frequency of PPWM was set to 96 MHz for all the experiments detailed in this subsection.

As shown in Figure 10, plots corresponding to each frequency setting of the AES module follow a similar trend line and form a relatively narrow efficacy variation envelop. The slight variations in the efficacy could be attributed to the changes in ambient conditions and random noise. Since the frequency of PPWM was fixed at 96 MHz, the experiment results also demonstrate the relevance of the ratio of frequencies (PPWM sampling frequency to AES frequency) to RPA attack efficacy. As seen in Figure 10, the ratio between PPWM sampling frequency to AES frequency seems to have no relevance to the RPA attack efficacy. It is worth noting that the ratio between PPWM sampling frequency to AES frequency was one to one when the AES module was clocked at 96 MHz and as a result, the entire last round (including the targeted `SubBytes()` transformation) of the AES encryption was captured with a single sample that is stored in a single bit. The observation that the ratio of frequencies seems to not have a relevance to the RPA attack efficacy, falls in line with the observations made in [UJS+21] and it could be assumed that this observation is a result of the AES module having concurrently updating flip-flops that cause a power drop which could be picked up by a single PPWM sample.

Furthermore, the hypothesis presented in Subsection 6.1 could be further supported by the results presented in this subsection. Firstly, the placement constraints defined for placement configuration 3 in Figure 7 (b) were identical to the placement constraints defined for the configuration in Figure 7 (c) even though the efficacies are noticeably different for the two experiments. This observation suggests that the placement has a relatively lesser contribution to the success of a CPA based key inference. Secondly, the aforesaid experiment configurations contained different logic designs despite they were constrained to the same placement. Therefore, they were placed and routed differently and as a result, the routing delay between logic elements of the gate network of PPWM was different. Hence, it could be suggested that the routing delay plays a significant role in determining the efficacy of PPWM as an on-chip power sensor.

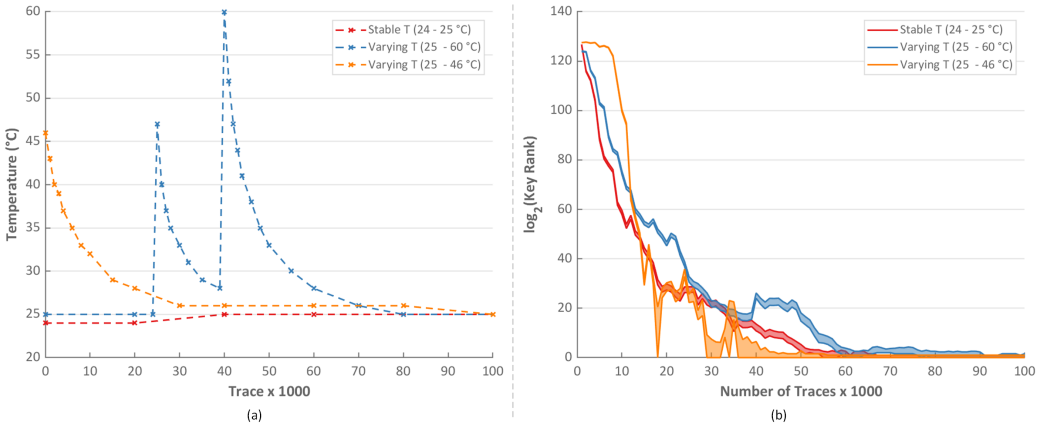

Figure 11: Results - Ambient Temperature. Shown in (a), Temperature Variation Pattern; and (b), Variation in KR over Number of Power Traces.

6.4 Ambient Temperature

This subsection presents and discusses the results of the experiments that were carried out using a modified PPWM to evaluate the impact of the variation in ambient temperature on the key inference efficacy. The modified PPWM used in these experiments features a CONTROL_FSM that retriggers the runtime calibration mechanism if no variation is observed for 10 consecutive power traces. The experiments were carried out by deploying the modified PPWM on Experiment Setup I and the design parameters shown in the fourth row of Table 2 were used to configure the experiment setup. A stream of hot air was used to heat up the main FPGA on Experiment Setup I and to achieve different temperature variation patterns, the hot air stream was blown at different time instances while allowing the main FPGA to naturally cool down to room temperature. The surface temperature of the main FPGA was recorded using a contactless infrared thermometer and the temperature variation pattern for each experiment is plotted in Figure 11 (a). Cross ($\times$) markers in Figure 11 (a) show the measured temperature value. Please note that all the experiment results presented in this section are based on a single main FPGA logic design which was deployed without resynthesis.

Figure 11 (b) shows the variation in KR against the number of power traces for each of the temperature variation patterns shown in Figure 11 (a). When the temperature was stable in the range of 24 – 25 ℃, the variation in KR has a relatively smoother curve (plotted in red) that eventually reaches unity. For the case of the experiment which started at 25 ℃ but intermittently heated up to higher temperatures, the KR plot (plotted in blue) got two bumps around the sample indices that correspond to the power traces that were collected while the FPGA was under the temperature spike. However, the KR eventually reaches unity regardless of the abrupt disturbance. Plotted in orange is the variation in KR for the experiment where the preheated FPGA was allowed to cool down to 25 ℃ from 46 ℃ while the power traces were collected. As seen from the plot, KR initially shows a marginal drop but then rapidly drops to unity even surpassing the plot corresponding to the case of stable temperature. To assess the capability of unmodified PPWM to withstand ambient temperature changes, an experiment setup containing an unmodified PPWM was subjected to ambient temperature variations in the range of 25 – 45 ℃. The experiment conducted using the unmodified PPWM was also successful in recovering the secret key, verifying the capability of PPWM to withstand ambient temperature changes.

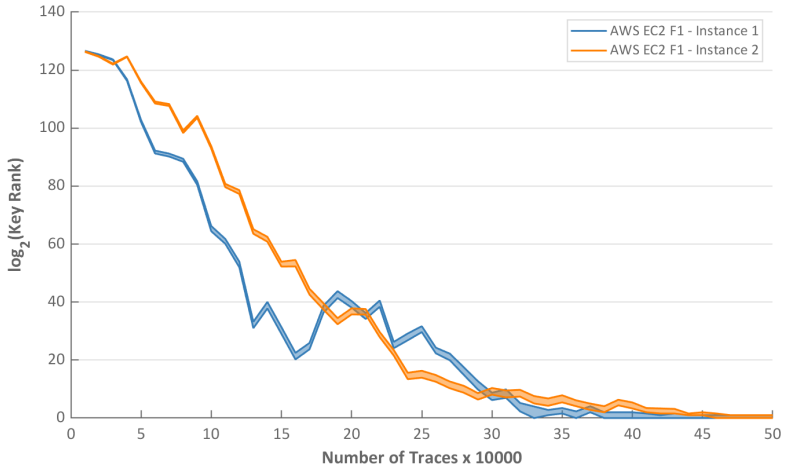

Figure 12: Results - Portability of PPWM.

6.5 Portability of PPWM

Shown in Figure 12 are the results of the experiments that utilized two instances of Experiment Setup II to assess the portability of PPWM. The two Experiment Setup II instances were concurrently deployed on AWS EC2 to ensure that they use two different FPGA instances to execute the RPA attack. As detailed in Subsection 5.2, the power traces from each experiment setup instance were transferred to a local PC at the end of power trace collection and CPA was performed on each trace set individually.

Plots in Figure 12 show that the key inferencing efficacy on both experiment setup instances are quite similar implying that an RPA attack setup based on PPWM could be ported across device instances while retaining the key inferencing efficacy. This enables attackers to locally design and test the attack setup prior to deploying it along with the victim in the remote environment. Further, these experiments also demonstrate the possibility of remapping the abstract PPWM design to different FPGA architectures and the possibility of deploying PPWM on cloud FPGAs.

6.6 RPA Attack Efficacy of PPWM

This subsection presents a comparison of RPA attack efficacy of PPWM and state of the art on-chip sensors, namely RO based sensors, TDC and VITI. The works proposed in [GDTL19] and [SGMT18] were reproduced to construct an RO based sensor and a TDC, respectively. Similar to Experiment Setup I, an attack setup consisting of 64 RO based sensors and an attack setup consisting of a TDC sensor were deployed on the SAKURA-G board and from each attack setup, 100,000 power traces were collected. Using the artifacts of [UJS+21], a collection of 100,000 power traces obtained from a 4-bit VITI sensor was extracted. Afterward, the three collections of power traces were subjected separately to CPA to discern the 128-bit AES key. The placement configuration 3 of Figure 9 was chosen to represent the efficacy of an RPA attack carried out using the power traces obtained from PPWM. Please note that in all four experiments, the frequency of the victim AES module was 6 MHz and the sampling frequency of the sensor was 96 MHz.

Figure 13 shows the variation in KR against the number of power traces for each on-chip sensor. It is evident that PPWM outperforms state of the art sensors for the configurations used in the experiments. In the demonstrated experiment configuration for PPWM, the complete AES key was recovered by performing CPA with 16,000 power traces. However, as discussed in Subsection 6.1, any alteration in routing delay could affect

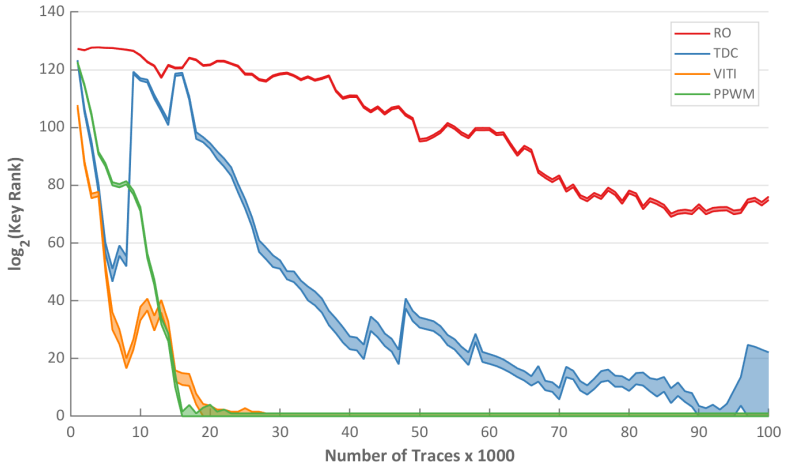

Figure 13: Results - RPA Attack Efficacy of PPWM.

the efficacy of on-chip sensors and it is justifiable to consider that they are of comparable efficacy.

6.7 Bandwidth Requirement of PPWM

As demonstrated by the experiments detailed in Subsection 6.6, PPWM is an effective on-chip sensor that is capable of successfully recovering a complete 128-bit AES key. For ease of use, each power trace obtained from PPWM was made up of 256 samples, each of which was made up of a single bit. Thus, each power trace consumed 32 bytes with PPWM when compared to 128 (4-bit sample $\times$ 256 samples), 256 (8-bit sample $\times$ 256 samples) and 16,384 bytes (8-bit sample $\times$ 256 samples $\times$ 64 sensors) required by VITI, TDC and RO based sensors, respectively to yield a comparable RPA attack efficacy. Thus, the bandwidth requirement of PPWM is 25% when compared to VITI which is the least bandwidth demanding among the existing on-chip sensors. When compared to TDC and RO based sensors producing comparable RPA attack efficacy, the bandwidth requirement of PPWM is 12.5% and less than 0.2%, correspondingly.

Though not proposed in the literature, the bandwidth requirement of VITI and TDC can be lowered by appending a thresholding circuit to the output of the sensor. Based on the observations presented in Subsection A.1, it could be hypothesized that the thresholding circuit would not hamper the efficacy of the on-chip sensor. However, the addition of a thresholding circuit would increase the area cost of the sensor and could hinder the stealthiness of the on-chip sensor.

7 Conclusion

In this paper, a novel on-chip sensor named PPWM featuring a novel core logic design that pulse width modulates the power consumption using signal propagation delay differences on two signal paths is presented. PPWM is an area-efficient on-chip power sensor that couples the newly proposed core logic design with an asynchronously cleared storage mechanism to store power fluctuations as single bit information. Using PPWM, this paper demonstrated the use of such single-bit information to successfully attack cryptographic circuits while keeping the communication bandwidth required to a minimum. With an extensive set of experiments, this paper evaluated the performance of PPWM under various design and natural parameters, such as placement, frequency, ambient temperature and process

variation. In addition, the deployability of PPWM on commercial cloud services was demonstrated by deploying PPWM on AWS EC2 F1 instances. Thus, by disclosing the properties and capabilities of PPWM, this paper revealed the threat posed by covertly deployable on-chip sensors that can be used to successfully carry out RPA attacks despite limited bandwidths often supported by covert communication channels.

References

[Alt16] Altera Corporation. Cyclone IV device handbook - I/O features in Cyclone IV devices. https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/hb/cyclone-iv/cyiv-51006.pdf, 2016.

[AWS] Amazon Web Services. FPGA developer AMI. https://aws.amazon.com/marketplace/pp/prodview-gimv3gqbpe57k.

[BCO04] Eric Brier, Christophe Clavier, and Francis Olivier. Correlation power analysis with a leakage model. In *Cryptographic Hardware and Embedded Systems - CHES 2004*, pages 16–29, 2004.

[BSH12] Florian Benz, André Seffrin, and Sorin A. Huss. Bil: A tool-chain for bitstream reverse-engineering. In *22nd International Conference on Field Programmable Logic and Applications (FPL)*, pages 735–738, 2012.

[FTD] Future Technology Devices International Limited. FT232R – USB UART IC. https://ftdichip.com/products/ft232rq/.

[GCRS20] Ognjen Glamočanin, Louis Coulon, Francesco Regazzoni, and Mirjana Stojilović. Are cloud FPGAs really vulnerable to power analysis attacks? In *2020 Design, Automation Test in Europe Conference Exhibition (DATE)*, pages 1007–1010, 2020.

[GDTL19] J. Gravellier, J. Dutertre, Y. Teglia, and P. Loubet-Moundi. High-speed ring oscillator based sensors for remote side-channel attacks on FPGAs. In *2019 International Conference on ReConFigurable Computing and FPGAs (ReConFig)*, pages 1–8, 2019.

[Gro19] Vincent Grosso. Scalable key rank estimation (and key enumeration) algorithm for large keys. In Begül Bilgin and Jean-Bernard Fischer, editors, *Smart Card Research and Advanced Applications*, pages 80–94, Cham, 2019. Springer International Publishing.

[GRS19] Ilias Giechaskiel, Kasper Rasmussen, and Jakub Szefer. Reading between the dies: Cross-SLR covert channels on multi-tenant cloud FPGAs. In *2019 IEEE 37th International Conference on Computer Design (ICCD)*, pages 1–10, 2019.

[GRS20] Ilias Giechaskiel, Kasper Bonne Rasmussen, and Jakub Szefer. C^{3}APSULe: Cross-FPGA covert-channel attacks through power supply unit leakage. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 1728–1741, 2020.

[IAS18] Zsolt István, Gustavo Alonso, and Ankit Singla. Providing multi-tenant services with FPGAs: Case study on a key-value store. In *2018 28th International Conference on Field Programmable Logic and Applications (FPL)*, pages 119–1195, 2018.

[KGT18] Jonas Krautter, Dennis R. E. Gnad, and Mehdi B. Tahoori. FPGAhammer: Remote voltage fault attacks on shared FPGAs, suitable for DFA on AES. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2018(3):44–68, Aug. 2018.

[KGT20] Jonas Krautter, Dennis Gnad, and Mehdi Tahoori. CPAmap: On the complexity of secure FPGA virtualization, multi-tenancy, and physical design. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2020(3):121–146, Jun. 2020.

[Lat10] Lattice Semiconductor Corporation. LatticeSC PURESPEED I/O adaptive input logic user's guide. https://www.latticesemi.com/-/media/LatticeSemi/Documents/ApplicationNotes/L/NZ/LatticeSCPURESPEEDIOAdaptiveInputLogicUsersGuide.ashx?document_id=25026, 2010.

[Meh15] Nick Mehta. Xilinx UltraScale architecture for high-performance, smarter systems. Technical report, Xilinx, Inc, 2015. https://www.xilinx.com/content/dam/xilinx/support/documentation/white_papers/wp434-ultrascale-smarter-systems.pdf.

[NIS01] The National Institute of Standards and Technology. Announcing the advanced encryption standard (AES). https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf, 2001.

[RPD+18] Chethan Ramesh, Shivukumar B. Patil, Siva Nishok Dhanuskodi, George Provelengios, Sebastien Pillement, Daniel Holcomb, and Russell Tessier. FPGA side channel attacks without physical access. In *2018 IEEE 26th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, pages 45–52, 2018.

[Sat14] Satoh Lab. SAKURA hardware security project. http://satoh.cs.uec.ac.jp/SAKURA/hardware/SAKURA-G.html, 2014.

[SGMT18] F. Schellenberg, D. R. E. Gnad, A. Moradi, and M. B. Tahoori. An inside job: Remote power analysis attacks on FPGAs. In *2018 Design, Automation Test in Europe Conference Exhibition (DATE)*, pages 1111–1116, March 2018.

[TS19] Shanquan Tian and Jakub Szefer. Temporal thermal covert channels in cloud FPGAs. In *Proceedings of the 2019 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, FPGA '19, page 298–303, New York, NY, USA, 2019. Association for Computing Machinery.

[UJS+21] Brian Udugama, Darshana Jayasinghe, Hassaan Saadat, Aleksandar Ignjatovic, and Sri Parameswaran. VITI: A tiny self-calibrating sensor for power-variation measurement in FPGAs. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2022(1):657–678, Nov. 2021.

[VCGS13] Nicolas Veyrat-Charvillon, Benoît Gérard, and François-Xavier Standaert. Security evaluations beyond computing power. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, pages 126–141, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[Xil11] Xilinx. Spartan-6 family overview. https://www.xilinx.com/support/documentation/data_sheets/ds160.pdf, 2011.

[Xil13] Xilinx. Spartan-6 libraries guide for HDL designs. https://www.xilinx.com/support/documentation/sw_manuals/xilinx14_7/spartan6_hdl.pdf, 2013.

[Xil18] Xilinx. UltraScale architecture libraries guide. https://www.xilinx.com/support/documentation/sw_manuals/xilinx2018_1/ug974-vivado-ultrascale-libraries.pdf, 2018.

[Xil20] Xilinx. AXI register slice v2.1. https://www.xilinx.com/support/documentation/ip_documentation/axi_register_slice/v2_1/pg373-axi-register-slice.pdf, 2020.

[Xil21] Xilinx. UltraScale architecture SelectIO resources. https://docs.xilinx.com/v/u/en-US/ug571-ultrascale-selectio, 2021. Revision: 1.13 English.

[ZH12] Kenneth M. Zick and John P. Hayes. Low-cost sensing with ring oscillator arrays for healthier reconfigurable systems. *ACM Trans. Reconfigurable Technol. Syst.*, 5(1), March 2012.

[ZS+13] Kenneth M. Zick, Meeta Srivastav, et al. Sensing nanosecond-scale voltage attacks and natural transients in FPGAs. In *Proceedings of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, FPGA '13, page 101–104, 2013.

A Design Insights

To supplement the understanding of the work presented in this paper, this section presents the insights and motivations that contributed to designing PPWM. A preliminary experiment performed using artificially synthesized power traces to evaluate the impact of the sensor resolution on key inference efficacy is presented in Subsection A.1. Subsection A.2 details a preliminary experiment, based on artificially synthesized power traces, that evaluates key inference efficacy against the power consumption threshold of a sensor that produces single bit information indicating whether the power consumption exceeded the threshold or not.

A.1 Sensor Resolution

To evaluate the impact of the resolution of an on-chip sensor on correctly inferring a key byte, a collection of synthetic power traces was used in an experiment where a series of inference attempts of a key byte was performed by analyzing the correlation between the Hamming distance and each of the synthetic power trace. To obtain the synthetic power trace for each inference attempt in the experiment, firstly a random byte sequence (representing a sequence of ciphertext) of an arbitrary length was generated. Each byte in the byte sequence represented a byte at a fixed index within a ciphertext that was generated using the same key. Hence, the length of the byte sequence corresponds to the number of ciphertexts used for the key byte inference attempt. Then, each ciphertext byte in the generated byte sequence was subjected to a set of operations to reverse a single round of `SubBytes()` and `AddRoundKey()` transformations [NIS01] to obtain a byte-sized intermediate value. Afterward, the Hamming distance between each of the intermediate values and the corresponding ciphertext bytes was calculated (this Hamming distance was used as the baseline synthetic power consumption). Since `SubBytes()` and `AddRoundKey()` are byte-wise transformations, the Hamming distance corresponding to computing each ciphertext byte from the immediate intermediate value is independent of that of each other. Further, the amount of power consumed by a computation is expected to be closely related

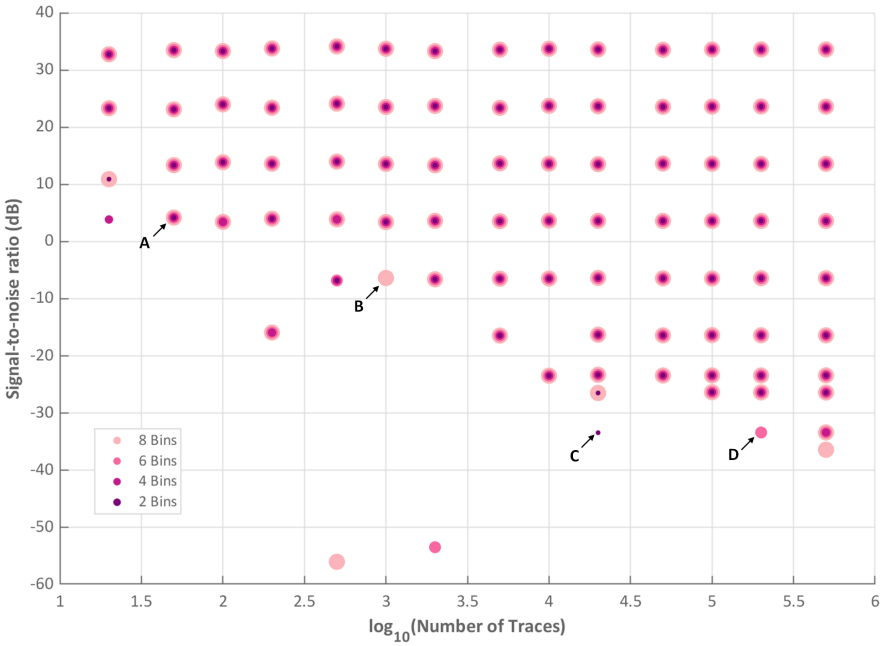

Figure 14: Impact of the Sensor Resolution.

to the Hamming distance incurred by the computation [BCO04]. To model a realistic power consumption perceived by an on-chip sensor, white Gaussian noise was added to the baseline power consumption.

For this preliminary experiment, bytes sequences of lengths (ciphertexts) in the range of 20 - 500,000 were generated. Moreover, white Gaussian noise of various power values in the range of 0.008 - 8,000,000 (corresponds to SNR_{dB} 30 to -60) were added to the baseline power consumption (Hamming distance) values to mimic a range of noise levels unrelated to the computation of interest. Note that the power of the baseline power consumption (Hamming distance) is in the range of zero to seven. To synthesize power traces obtained from a delay line based on-chip sensor with 1-bit, 3-bit, 5-bit and 7-bit long delay line, the perceived power consumption was discretized to two, four, six and eight bins respectively under the assumption that the full scale of the on-chip sensor corresponds to the range of the perceived power consumption. For example, assume a situation where no white Gaussian noise is added (not shown in the experiments - for explanation purposes only). In the case of two bins, representing a 1-bit delay line, the values in the closed interval $[0, 3]$ were mapped to zero while the values in $[4, 7]$ were mapped to one. Similarly, in the case of four bins (representing a 3-bit delay line), the values in the closed intervals $[0, 1]$, $[2, 3]$, $[4, 5]$ and $[6, 7]$ were mapped to zero, one, two and three, respectively. Since the range of the perceived power consumption is mapped to the full scale of the on-chip sensor, shorter delay lines result in larger discretization steps effectively lowering the resolution of the sensor.

Figure 14 shows the results of this preliminary experiment which attempts to infer a key byte by analyzing the correlation between the Hamming distance and the synthetic power trace. The x-axis, on a logarithmic scale, represents the number of traces used for the inference attempt. The number of traces used for the inference attempt is as same as the number of ciphertexts which in turn equates to the length of the random byte sequence. The y-axis, in decibel, represents the signal-to-noise ratio (SNR_{dB}) calculated as per Equation 4, where A_{signal} is the Hamming distance and A_{noise} is the amount of additive noise. The graph in Figure 14 sequentially overlays the successful inferences with

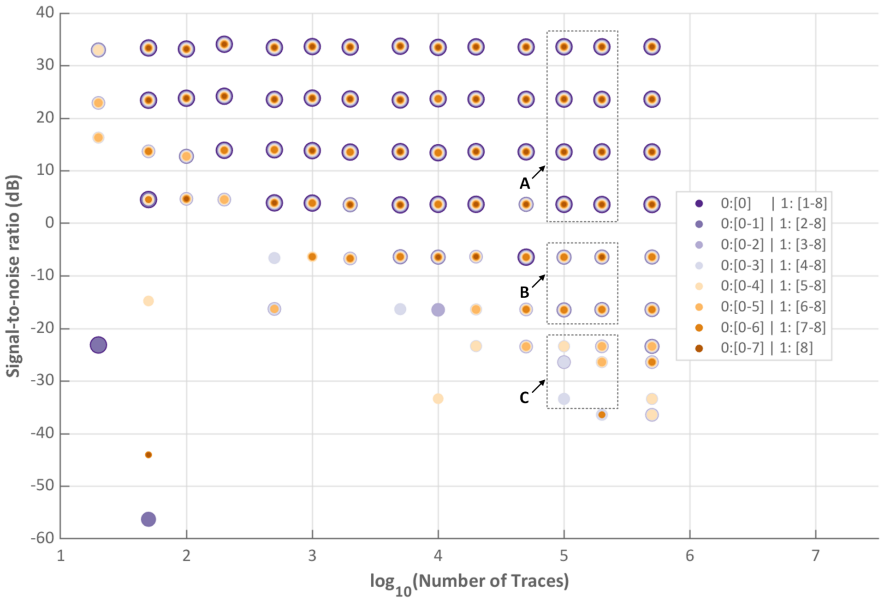

Figure 15: Impact of the Power Consumption Threshold.

eight bins, six bins, four bins and two bins on the same coordinate plane. As seen from Figure 14, the number of bins used for discretization does not significantly impede the key byte inference attempts (almost all light shaded circles have a dark shaded circle overlaid) that exploit the correlation between the power trace and the Hamming distance. For example, the set of overlaid circles labeled with A shows that inferring the key byte was successful with all four binning schemes. However, the circles labeled with B, C and D respectively show the instances that were only recoverable with eight, two and six bins. As seen from Figure 14, such instances are rare implying that the impact of the number of bins used for discretization is insignificant for the success of inferring a key byte.

$$SNR_{dB} = 20 \log_{10} \left(\frac{\sqrt{\sum (A_{signal})^2}}{\sqrt{\sum (A_{noise})^2}} \right) \tag{4}$$

A.2 Power Consumption Threshold

The preliminary experiment detailed in this section assumes power traces obtained from an on-chip sensor that yields a single bit of information by discretizing the power consumption into two bins. The experiment evaluates the relationship between the efficacy of the key byte inference and the threshold value used to discretize the power consumption. Similar to the preliminary experiment detailed in Subsection A.1, the collection of power traces used for the experiment was synthesized using random byte sequences of arbitrary lengths. The power consumption perceived by the on-chip sensor was calculated by adding white Gaussian noise to the Hamming distance between each random byte and the corresponding intermediate value obtained by reversing a single round of `SubBytes()` and `AddRoundKey()` transformations. Since applying a transformation on a single byte of information can result in Hamming distances between zero and seven inclusive, the perceived power consumption was then discretized into eight bins. Please note that if no noise was added to the Hamming distances, the discretized signal would be identical to the Hamming distances. Thereafter, the discretized signal was mapped to two bins by thresholding the discretized signal at an

arbitrarily chosen index to generate the power traces. Each power trace was correlated with Hamming distances calculated for each key byte guess to discern the correct key byte.

Evaluations carried out in this preliminary experiment involved byte sequences having lengths (ciphertexts) in the range of 20 - 500,000 and additive white Gaussian noise having a power in the range of 0.008 - 8,000,000. For each pair of byte sequence length and additive noise power, the evaluation was repeated eight times, each time generating the power trace by thresholding the discretized signal at an index between zero and seven. Results of this experiment are shown in the graph in Figure 15 where the x-axis, in logarithmic scale, represents the number of power traces used for the inference attempt and the y-axis represents the SNR_{dB} calculated as per Equation 4. In Figure 15, plots for different threshold indices are overlaid in the decreasing order of the index for convenient comparison and as seen from the figure, the efficacy of inferring a key byte is significantly reliant on the threshold index. In Figure 15, the dark shaded circles (also appear as rings due to overlaying) are confined to the top right region of the figure whereas the region with light shaded circles (or rings) is much more extended towards lower SNR_{dB} and lower number of traces. For instance, the spots in region A are noticeably darker when compared to the spots in regions B and C because they overlay dark shaded circles (rings) in addition to the light shaded circles (rings). Hence, they illustrate trace count and noise level combinations that are successful in retrieving the key byte regardless of the thresholding index. Spots in region B are lighter when compared to that in region A because of the absence of the darkest circles (rings). They illustrate the scenario where the key recovery failed due to extremely skewed thresholding. Spots in region C are the lightest among the three labeled regions and they illustrate the scenario where the key recovery is successful only with balanced (tend towards the middle of the range) thresholding. Therefore, it can be deduced that the inference attempts can become more efficacious when the threshold index tends toward the middle of the range of power fluctuations.