The Good, the Bad, and the Noisy

A Systematic Analysis of Ring-Oscillator-Based Noise Generators

Florian Stolz*[1], Dina Hesse*[1], Julius Hermelink†[3], Tim Güneysu[1,2] and Elisabeth Oswald[4,5]

[1] Horst Görtz Institute for IT Security, Ruhr University Bochum, Bochum, Germany, firstname.lastname@rub.de
[2] DFKI GmbH, Cyber-Physical Systems, Bremen, Germany
[3] Aarhus University, Aarhus, Denmark, jhermelink@cs.au.dk
[4] University of Birmingham, Birmingham, UK, m.e.oswald@bham.ac.uk
[5] University of Klagenfurt, Klagenfurt, Austria

Abstract. Hiding countermeasures, alongside masking, play an important role in modern hardware security for protecting against side-channel attacks. Additive noise generators are one such hiding countermeasure. They increase the noise, thereby lowering the signal-to-noise ratio and, simultaneously, amplifying the effectiveness of masking countermeasures. Among the different design options for additive noise generators, ring-oscillator-based designs feature regularly in scientific publications. These existing works not only differ in their designs, but also in their evaluation methods; therefore, they do not permit fair comparisons or enable systematic evaluations. Thus, it is currently not possible to design ring-oscillators for noise generation in an informed and tailored way.

In this paper, we address this lack of systematic comparisons of existing designs, and discuss the possibility of filtering out the generated noise and further limitations of ring-oscillator-based noise generators. We find that many existing designs do not consider the need to amplify the output signal or the risk of filtering. As a consequence, the existing designs do not sufficiently lower the signal-to-noise ratio in our experiments. However, using these results, we provide a guideline for tailoring and configuring ring-oscillators so that they efficiently add noise in the desired frequency spectrum. This includes deciding on the inverter chain length, the amplification of the output, and the randomization of the output frequency. Finally, we apply this methodology to build a tailored ring-oscillator-based noise generator for an exemplary AES implementation and find that it significantly outperforms existing ring-oscillator-based designs.

Keywords: Side-Channel Analysis · Hiding · Ring-Oscillator

1 Introduction

Power side-channel attacks exploit information leaked through power consumption: processing of intermediate secret-dependent values influences the power consumption and leaks information to an adversary with physical access. Based on measured power consumption traces, an adversary can then perform a variety of analyses that range from Simple Power Analysis (SPA), over more complex Correlation Power Analysis (CPA) to Deep Learning (DL)-based attacks. These attacks pose a threat to cryptographically secure

*The authors contributed equally to this work.
†The majority of this work was conducted while the author was affiliated with MPI-SP.

algorithms. Consequently, many countermeasures have been proposed in theory, and many are widely deployed in real-world applications.

Power side-channel countermeasures can be broadly grouped into two categories: *Masking* breaks the dependence between the secret and intermediate computation values by splitting the secrets into multiple shares and performing computations on the shares separately. Thus, both the computed values and the side-channels are decoupled from the processing of intermediates. Masking provides provable security guarantees under theoretical models (e.g., [DDF14]), which makes it an attractive research target. However, this comes at a cost: Increasing the number of shares increases the data to be processed, leading to significant time, or area overhead, as well as non-linear operations require special handling and costly mask refreshes. And even if all these rules are considered, masked implementations might still be vulnerable to SCA if the system noise level does not suffice [HFG+25].

Hiding countermeasures reduce the dependency between the observable side channel and the sensitive data. Typically, this does not lead to formal security proofs, but it does predictably increase the trace complexity of attacks. Mangard, Oswald and Popp [MOP07] summarize hiding countermeasures specifically for power side channels: among various options, they refer to the idea of adding random noise. Kocher et al.'s seminal work [KJJ99] already mentioned increasing or adding noise, and [GM11] studied it more deeply for reconfigurable devices. In the patent [KJJ01], Kocher et al. describe a first concrete noise generator. More recent patents from renowned companies [NEC04, Ren09, Sam09, Gun13, Del16, Chu16, CVP24] show that noise generators are applied regularly to sensitive computations. In current real-world devices, noise generation is also achieved by activating additional cores to perform independent computations, such as running another on-board blocks with random data [Kni06, Ped14, KMSS18]. To our knowledge, these noise generation solutions are frequently implemented in an ad hoc manner, often without a thorough evaluation of their efficiency.

[DMN+17] construct a noise generation circuit by using a Linear Feedback Shift Register (LFSR) that controls an additional current source and [GM11, Gün12] analyze how the inherent building blocks of Field-Programmable Gate Arrays (FPGAs) can be utilized to generate noise in the power consumption independent of the protected implementation. In contrast, [KBG09, KBG11, TPZA25] propose protecting an AES-implementation by adding parallel computations on the data with a random key.

Ring Oscillator (RO)-based designs are the most discussed design option for additive noise generators in the academic community[1], as an increasing number of publications show [LCL10, NYH13, MGZS17, KGS+19, PGC+22, NPBH23, Sch23, YKS+23]. In their simplest form, ROs consist of an odd number of inverters, and their oscillation frequency is determined primarily by the physical properties of these inverters, so they promise to require little area, while inducing substantial switching activity, i.e, cause noise. As we discuss later, previous solutions make incomplete comparisons to related work and do not consider adversaries that use even basic filtering techniques. Consequently, it remains unclear if any of the designs offers some tangible real-world protection, or how to choose and configure an RO-based design for practical deployment.

1.1 Our Contributions

We start with reviewing and discussing evaluation methods from prior work to conclude on a more suitable method for our work. Importantly, we consider adversaries that apply trace filtering. Our first contribution (Section 3) is a generic structure for RO-based noise generators that we apply to describe and compare previous works. Second (Subsection 2.3), we comprehensively re-evaluate these published RO designs, avoiding pitfalls from prior

[1] We found no concrete evidence that they are used in commercial products.

Table 1: Overview of Various Noise Generators and their Respective Evaluation Strategies.

Countermeasure		**Evaluation Strategy**			
		Metric	Method	Real-World Exp.?	# traces
Digital Controlled ROs	[LCL10, NYH13]	# traces	CPA	-	1,000,000
		max r_{can}	CPA	☑	
FiGaRO	[MGZS17]	max r_{can}	CPA	☑	2000
Active Fences	[KGS+19]	# traces	CPA	☑	80,000
Xored High Consuming Module	[PGC+22]	$\Delta W/LUTs$	power consumption	☑	-
Configurable RO Chains	[NPBH23]	t-value	TVLA	☑	120,000
Random Number of ROs	[Sch23]	t-value	TVLA	☑	50,000
Programmable RO	[YKS+23]	t-value	TVLA	☑	50,000

work and considering an adversary that filters traces to improve attack outcomes. We show that existing designs which increase the trace complexity of a filtering adversary typically come with undesirably large overheads. However, our results suggest a design strategy that combines ideas from existing work to tailor an efficient RO-based noise generator for a specific target. Thus, we formulate a guideline to achieve a tailored RO noise generator design for a specific, known target (Section 5). We develop this guideline by analyzing key design parameters of ROs, namely chain length, number of ROs, dynamic randomization and output amplification. Finally (Section 6), we discuss environmental, as well as platform and technology effects that impact the effectiveness of RO-based noise generators.

2 Noise Generator Evaluation Strategy

Most previous works use two types of evaluation methods: they either assess the impact of an RO design via the number of needed measurements to succeed in a differential power analysis attack, or, via the number of needed traces to perform a fixed-vs-random leakage detection experiment, see Table 1 for a summary of papers, methods, and outcomes. One single paper reported the overall increase in the power consumption, without reporting either an attack or leakage detection outcome.

In the rest of this section we review and discuss the existing methods, leading to our conclusion that reporting the noise variance directly would be a more efficient strategy.

2.1 Evaluation Methods

The effect of additive noise is well documented in the Side-Channel Analysis (SCA) literature. Comprehensively explained in [MOP07], the power consumption that we measure has several components, e.g., components that directly relate to data and key manipulation, and others that stem from independent device activity. We use the term signal to refer to the data-dependent component, and the term noise to the sum of contributions from all independent activities. The ratio between the variation of the signal and the variation of the noise (also called the signal-to-noise ratio, or short SNR, see 1) crucially impacts the success of a side-channel adversary.

$$\mathrm{SNR} = \frac{P_{signal}}{P_{noise}} = \frac{Var(Signal)}{Var(Noise)}. \tag{1}$$

Several previous works on RO designs [LCL10, NYH13, MGZS17, KGS+19] measure the success of SCA adversaries that can supply multiple different inputs to a device using

a static secret key. Concretely, they use correlation as a distinguishing rule (in a CPA attack), i.e. they estimate r_{cand} with

$$r_{cand} = \frac{\sum(x_i - \mu_x) \cdot (y_i - \mu_y)}{\sum(x_i - \mu_x)^2 \cdot \sum(y_i - \mu_y)^2} \tag{2}$$

whereby X corresponds to the key-dependent predictions and Y corresponds to the measured traces. The key candidate with the maximal r_{cand} is the most likely value.

The intuitive relationship between the SNR and the correlation value of the correct key was formalized in [MOP07]. Under some simplifying assumptions[2] one can link the correlation coefficient of the correct key hypothesis to the number of traces that are required for an attack to succeed (i.e. return the correct key as the most likely value, with near certainty in repeated experiments). Particularly, for small SNR, the "rule of thumb" applies for correlation-based attacks, which is that $n \approx \frac{1}{SNR}$. Consequently, if we increase the noise by a factor of k (i.e. we have $k \cdot Var(Noise)$), then also the number of traces n that are needed for a successful CPA also increases by a factor of k.

The problem with comparing RO designs based on the number of traces for a successful CPA attack is that the number of traces also depends on the signal strength, which in turn depends on the quality of the power model. So, a relative comparison between designs in papers is impossible, because no information about the quality of the respective power model is given.

Other previous work on RO designs use TVLA [NPBH23, Sch23, YKS+23] to evaluate the effectiveness of their designs. They use the "fixed-vs-random" option of TVLA, which attempts to disprove that there is no difference between the sample means of a data set where inputs are fixed and a data set where inputs are randomly varying, by estimating the t-statistic

$$t = \frac{\mu_r - \mu_f}{\sqrt{\frac{\sigma_r^2}{n_r} + \frac{\sigma_f^2}{n_f}}}. \tag{3}$$

The typical decision criterion for rejecting the statement that there is no difference (i.e., rejecting the assumption that there is no leakage) is to compare the resulting t statistic to a threshold (usually 4.5). The number of needed traces for the t-statistic to reliably surpass a threshold is, according to [MOP07], proportional to

$$n = \frac{(\sigma_r)^2 + (\sigma_f)^2}{(\mu_r - \mu_f)^2}. \tag{4}$$

Evidently, increasing the noise variance of an implementation by a factor of k (i.e., $k \cdot \sigma_r$ and $k \cdot \sigma_f$ in Equation 4) increases n by the same factor of k, very much like CPA.

The problem with using this comparison criterion across designs is similar to the problem with CPA: TVLA success also depends on signal strength. In contrast to CPA, where the signal depends largely on the quality of the power model, the signal quality in TVLA is influenced by the "partitioning" implied by the choice of the fixed plaintext value. Durvaux and Standaert discussed this fact in [DS16].

Both approaches in the previous work are based on the (implicit) definition of a signal, and their success depends on the signal strength. Because previous works do not supply any information about the signal, the reported *raw trace numbers* cannot be compared with each other. Furthermore, to our best understanding of the previous work, none of them performed repeat experiments. Thus, their CPA and TVLA outcomes are single experiment estimates: they are statistically unsound; repeat experiments are necessary. Repeat experiments require large trace sets to support independent repetitions. Because

[2] The assumptions do not affect the asymptotic behavior, and therefore they support relative statements when comparing countermeasures.

we compare eight different RO designs, we settled for 10 repetitions for each experiment, and report average outcomes. Additionally, notice that both CPA and TVLA outcomes scale linearly with the noise variance. Thus, a natural approach is to use the increase in noise variance as a direct indicator of an additive noise generator's effectiveness, and to relate the noise variation to other performance metrics to assess its efficiency. We will use this approach in the rest of this paper[3]. Another important aspect, especially when considering RO-based designs, is the frequency-dependency of the noise. We will discuss this effect and its implications in the following.

2.2 Filtering

Power analysis attacks work because of the exploitable relationship between the data dependent part of the overall power consumption (signal) and some predictable secret value. Many attacks and evaluations are performed in the time domain, i.e. power values are sampled over time, and the analysis uses these traces directly. However, they can also be performed partially or in the whole in the frequency domain. When transforming power traces into the frequency domain, it is common that the signal is located in a relatively narrow part of the frequency spectrum (rather than it being spread evenly across the frequency domain). If a noise generator's contribution to the overall power consumption is in a part of the frequency spectrum that is different from the part where the signal is located, then, already simple signal processing techniques such as band pass filters, can effectively remove the added noise. Thus, additive noise from ROs, which typically operate in a narrow frequency band is potentially easily filtered out.

We know from previous work that filtering can improve the success rate of side-channel attacks [PHF08, MOP07]. This can be explained, for instance, with the Signal-to-Noise Ratio (SNR), as the ratio of the signal power and the noise power (see 1). By removing frequencies from the measurements that only contribute to the power of the noise, P_{noise} decreases, while P_{signal} stays constant. Hence, the SNR increases. This is also the case if the removed features contain some parts of the signal, but more noise than signal. For some attack-strategies it might be beneficial to remove as much noise as possible, while other, more noise-tolerant methods, such as Soft Analytical Side-Channel Analysis (SASCA) and DL-based SCA, benefit from more careful filtering. It is important to note that noise present in the same frequencies as the signal cannot be filtered without loss of the SNR. Hence, an optimal noise generator should generate noise in the most dominant frequencies of the signal.

[SWL21] discussed this effect in detail and showed it in practice. They propose to identify the frequencies of the measured signal that have maximum SNR and only analyze them in the final attack. Therefore, the SNR is assessed for all measurable frequency bands individually. By choosing only the frequency band that contains the maximum signal strength and filtering out the other frequency spectrum, the SNR can be significantly increased.

2.3 Our Noise Generator Evaluation Method

Our evaluation method draws on the discussion and insights from before. It compares designs by measuring the increase in noise variance (across multiple repeat experiments) relative to an unprotected target design.

First, we characterise traces from an unprotected reference implementation of a fully parallel AES design (we provide a full description of the setup in Section 4). This means we identify the most leaking frequencies and the point of interest with the highest SNR (and thus the lowest average number of traces needed for a successful CPA attack). Because

[3]For completeness and backwards compatibility to existing work, we also supply the number of needed traces for corresponding CPA attacks, and TVLA tests throughout.

the considered ROs do not affect duration and timing, we can base our comparison on this point of interest.

Then, for each of the RO designs, we derive and report the average noise variance at the point of interest (selected as described above). For completeness, and to enable comparison across designs, we also report CPA, SNR, and TVLA results for all designs (using the same point of interest).

Finally, we report variance-to-area and power-consumption ratios as ways to compare efficiency.

3 Ring-Oscillator-Based Noise Generators

Also in electrical circuit design, Ring Oscillators (ROs) are an important building block. In their simplest form, ROs consist of an odd number of inverters chained together, with the output of the last inverter feeding the first inverter. Because of the odd number of inverters, the output of the final inverter is always different from the previous input of the first inverter. This causes the circuit to start oscillating. The speed of the oscillation, or frequency f_{ro} depends on the number of inverters n and the average delay time of a single inverter t_p [Grö07]:

$$f_{ro} = \frac{1}{2nt_p} \tag{5}$$

Longer chains lead to slower frequencies because the signal requires more time to propagate through the entire chain. The idea of using ROs for noise generation is to utilize this toggling of inverters, also referred to as switching, for secret-independent power consumption. Note that any gate capable of inverting an input can also be used to build an RO and they do not rely on specific building blocks, such as BRAMs. For example, designers can not only use inverters but also `NAND` gates with one of its two inputs tied high. In this work, we use Look-Up Tables (LUTs) to implement the inverting functionality on an FPGA.

In this section, we present the RO-based noise generators from [LCL10, NYH13, MGZS17, KGS+19, PGC+22, NPBH23, Sch23, YKS+23]. To this end, we summarize the different designs and, based on this, propose our generic RO-based noise generator structure and the various configuration options. Our generic structure allows us to then present the individual designs precisely and in a uniform format. This overview of the designs provides an understanding of the current state of research and helps to explain the evaluation results of the designs in Section 4.

3.1 General Ring-Oscillator Architecture

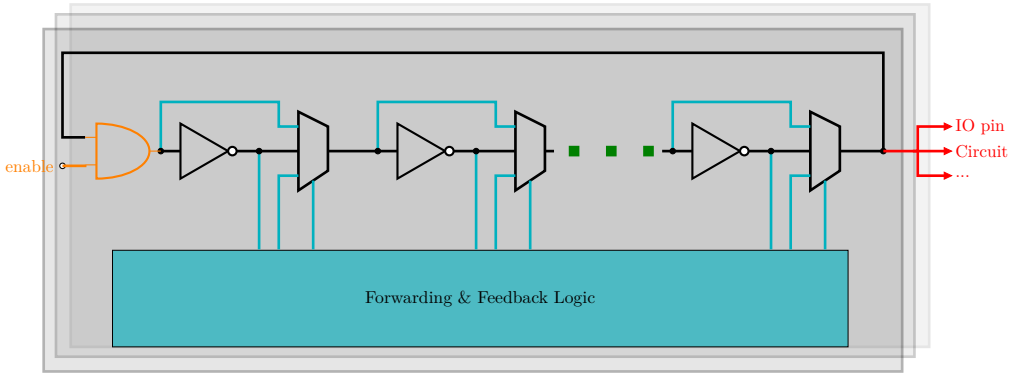


Figure 1: Our Generic Structure of Ring-Oscillator-Based Noise Generators.

Figure 1 illustrates our generic RO-based noise generator structure. The basic inverter chain is shown in black, whereas the configurable options are highlighted in different colors. The green dots indicate the adjustable chain length and the repetitive pattern in the background illustrates that multiple RO instance may be instantiated. The red arrows at the output represent different ways to connect the RO's output to an external load. By fixing these parameters, a *static* RO can be defined. If the configuration can be changed dynamically during operation, we refer to it as a *dynamic* RO. To this end, the orange input logic can be used to en- or disable the ROs. The blue block indicates forwarding and feedback logic that controls the dynamic reconfiguration of the individual RO instance at runtime. Table 2 shows the different RO-based noise generator designs and their configuration decisions. In the following, we elaborate on these different options in more detail, before we present the individual designs from prior work.

Number of Active RO Instances. In principle, one can implement several RO instances in parallel. For dynamic ROs, an `AND` gate enables the activation and deactivation of the individual ROs. This can be used as a power-saving measure when the maximum amount of noise is not needed. That is because setting one input of an `AND` gate forces the output zero, thus stopping the oscillation. It can also be utilized to increase the generated noise if activation and deactivation happen so fast that most of the switching activity stems from the ROs settling into a steady state, as described in [Sch23].

Chain Length. The frequency on which an RO oscillates is determined by the length of its inverter chain. Dynamic ROs allow changing the chain length dynamically by configuring the multiplexers, e.g. one or several inverters can be skipped. This is useful to adjust and control the RO frequency at runtime. System designers can create (arbitrarily) complex ROs by building (chained together) configurable RO blocks.

Randomization Previous work has used two strategies to generate an activation input signal for an RO: an input generated from (or related to) the activity of the cryptographic module we wish to protect, and an input generated independently of the module's activity. The first strategy typically taps some signal from the cryptographic module, e.g. some SubBytes output. This can make the RO activity depend on a portion of the secret key, which, in the worst case, amplifies the information available to an attacker. To implement ROs with an independent input signal, we implement a suitable Pseudorandom Number Generator (PRNG). Another possibility is to use the output of some other component (e.g. sensor) as an input; enabling the *dual use* of the RO as e.g. a fault injection monitor as described in [YKS+23].

Amplification of Output. In order to amplify the high switching activity of the RO output signal, it can be further connected to other circuits. Circuits that can be driven by an RO output include IO pins (as seen in [KGS+19, YKS+23]), LFSRs, and other building blocks. The oscillating output signal can also be used as the clock input of such building blocks or existing on-board modules.

3.2 Previous Designs

We can now apply the general structure introduced in the previous section to describe the noise generator designs from research literature. This enables the evaluation in Section 4 and a more detailed discussion. Table 2 briefly summarizes the configurations, while Table 1 lists the evaluation methods used.

Digital Controlled ROs. The earliest approach, described by Liu, Chang and Lee [LCL10] and Nianhao, Zhou and Liu [NYH13], use twelve, respectively eight, one-stage ROs that are switched on and off depending on the input signal. In this context, stage refers to a chain length of one. The input signal is an XOR combination of the AES S-box input and a random 8-bit number. This strategy was evaluated by simulating power consumption traces and performing a CPA on them. As discussed above, using the secret S-box input to control the ROs will likely alter the leakage function of the secret signal, leading to an incorrect correlation attack value. This assumption is supported by the results in [LCL10] showing that the correlation coefficients do not change for an increasing number of traces. Therefore, for the evaluations in this work, we directly feed the random numbers of a PRNG to the enable logic of the ROs.

FiGaRO. Ma et al. [MGZS17] propose to feedback the RO output signal not only to the beginning of the chain but also to intermediate values in the RO, analogously to an LFSR. The design of the feedback net is based on Fibonacci and Galois LFSRs. The output of 32 Fibonacci and Galois RO instances is fed into an `XOR` gate. The authors evaluate this countermeasure using a CPA with 2,000 traces.

Active Fences. The noise generator of Krautter et al. [KGS+19] is designed to hide side-channel leakage on multi-tenant FPGAs. In this scenario, multiple users share a single FPGA, but utilize different areas. In order to prevent other areas from capturing the power consumption of a victim module, it is intentionally surrounded with single-stage ROs. The overall amount of ROs is roughly equivalent to the utilization of the design that should be protected, which is 672 single-stage ROs for their target system. Additionally, the enable signal of each RO is tied to an output of a random number generator,and all RO outputs are connected to an `XOR` gate, which in term is connected to an IO-pin. For their evaluation, they perform a CPA and report the Measurements to Disclosure (MTD) with and without their countermeasure in place.

Xored High Consuming Module. In [PGC+22], Parrilla et al. advance their previously developed High Consuming Module (HCM) [PCT+15] to the Xored HCM (XHCM). The HCM consists of four parallel single-stage ROs that drive a network of `OR` and `AND` gates. The XHCM replaces the `AND/OR` network with `XOR` gates, as they have a higher switching probability. The authors propose to instantiate the XHCM module with eight randomly activated ROs, controlled by a True Random Number Generator (TRNG). Unlike other works, the authors do not perform a t-test or other side-channel related evaluations and

Table 2: Configurations of the RO-Based Noise Generators from Previous Work

Countermeasure		**Parameters**			
		Chain Length	# of ROs	Output Amplification	Randomization
Digital Controlled ROs	[LCL10, NYH13]	1	12 8	-	RO-activation (dep. on random & input signal)
FiGaRO	[MGZS17]	>3	32	Outputs xored	Linear Feedback RO
Active Fences	[KGS+19]	1	372	Outputs xored to IO-Pin	RO-activation by PRNG
Xored HCM	[PGC+22]	1	8	Outputs xored	RO-activation by PRNG
Configurable RO Chains	[NPBH23]	13 - 65	32	-	Chain length randomization & RO-activation by PRNG
Random Number of ROs	[Sch23]	approx. 11	32	-	RO-activation by PRNG
Programmable RO	[YKS+23]	1 - 57	1	IO-Pin	Chain length randomization by PRNG

base their security claim on measuring the dynamic power consumption of their solution, with a higher power consumption being better.

Configurable RO Chains. Nassar et al. [NPBH23] introduce configurable inverter elements to enable the dynamic reconfiguration of their design. It consists of multiple configurable RO-elements with different routing options. Many different configurations can be dynamically realized by combining single-stage elements with four-stage RO blocks, allowing for fine-grained frequency control. For evaluation purposes, they perform a t-test with 60,000 measurements.

Random Number of ROs. The RO-based design of Schaumont in [Sch23] is designed both as fault injection detection and hiding mechanism. The target device (a RISC-V core running at 4 MHz), is enhanced with 32 ROs that run at 120 MHz and are placed throughout the chip. The software dynamically activates a random number of ROs to generate broadband noise. Applying this countermeasure, a CPA using 50,000 traces does not succeed and a t-test with the same amount of traces showed no significant leakage.

Programmable RO. The Programmable RO (PRO) of Yao et al. in [YKS+23] is also additionally to hiding designed for fault injection monitoring. In order to prevent attackers from filtering out the noise, they also use dynamic ROs where the frequency is changed randomly during runtime. Thus, their example design can achieve frequencies between 22 MHz and 123.44 MHz. This so-called Programmable RO (PRO) instance is connected to an unused IO-pin to amplify its output. The effectiveness of the countermeasure is determined by analyzing the frequency spectrum and performing a t-test with 50,000 traces.

4 Evaluation

In the following section we first detail our measurement setup, and then present the results for the above designs using the previously described evaluation methods.

4.1 Measurement Setup

We evaluate all designs on the ChipWhisperer CW305 board, which is designed for side-channel measurements. It is placed inside a CTS C-40/100 climate chamber, which keeps the ambient temperature within $\pm 0.3\,\mathrm{K}$. For all experiments, the temperature is set to $20\,^{\circ}\mathrm{C}$. We power the FPGA directly with a Korad KD3005P lab power supply, bypassing the board's step-down converter, which would otherwise inject additional noise into the power rail. The power consumption is measured via the on-board shunt resistor. The on-board 20 dB amplifier boosts the shunt signal, which is then fed to a PicoScope 6425E oscilloscope through a $50\,\Omega$ DC blocker to remove voltage offsets and prevent ringing. Each statistical evaluation uses 200,000 traces captured at $2.5\,\mathrm{GS/s}$. To determine the mean power consumption of a design, we remove the DC blocker and connect two oscilloscope channels to either side of the shunt resistor to measure the voltage drop across it. Due to measurement constraints, the sample rate is lowered to $1.25\,\mathrm{GS/s}$. The power consumption is averaged over 500 traces.

All RO designs protect an otherwise unprotected Google Vault Advanced Encryption Standard (AES) [PG15], which is a fully parallelized round-based implementation with 16 S-boxes. One encryption requires 12 cycles, where the first two cycles handle resetting the internal state and the initial key addition respectively. An unprotected implementation allows us to easily analyze the impact of different hiding countermeasures as it provides

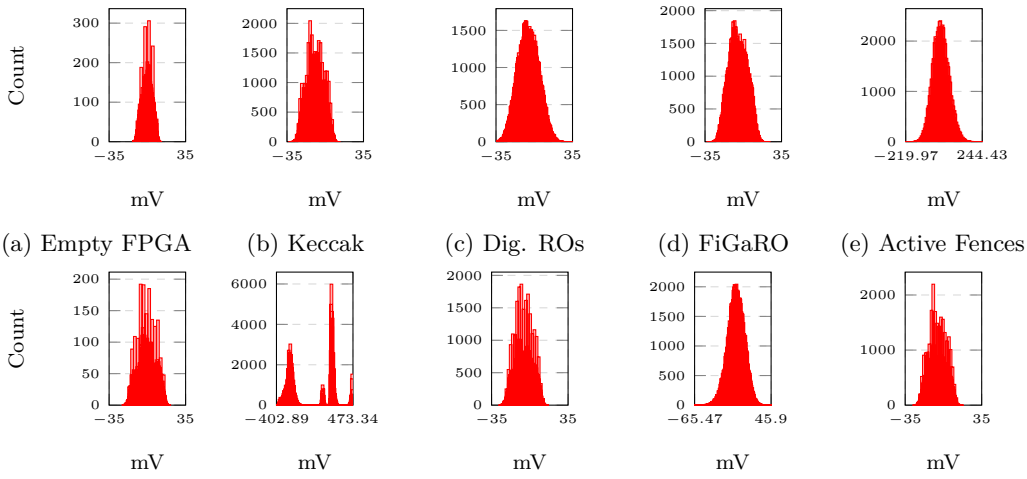
a strong secret-dependent signal. A masked AES would require more traces and computational power to properly evaluate the impact of the countermeasure. However, if a countermeasure works particularly well to hide the unprotected implementation, it will hide the masked implementation just as well, as masking amplifies the noise of a system.

For the designs that require random bits to toggle certain functionality, we include a small `xorshift32` PRNG [Mar03], which produces 32 pseudorandom bits per cycle as described in Algorithm 1. Multiple instances provide more than 32 bits if necessary.

Algorithm 1 Description of the Xorshift32 PRNG.

Require: 32-bit state $x \neq 0$
$x \leftarrow x \oplus (x \ll 13)$
$x \leftarrow x \oplus (x \gg 17)$
$x \leftarrow x \oplus (x \ll 5)$
return x

To ensure reproducibility, we use the standard AMD Vivado partial reconfiguration flow to define two partitions, one for the AES and one for the countermeasure, on our FPGA. These partitions are located apart from one another. This ensures that an RO design cannot interfere with or change the AES design, so the signal remains the same across all comparisons.



(a) Empty FPGA (b) Keccak (c) Dig. ROs (d) FiGaRO (e) Active Fences

(f) Prog. FPGA (g) C. RO Chains (h) Random ROs (i) Prog. RO (j) XHCM

Figure 2: Noise distribution for all evaluated countermeasures.

Signal Selection. We use a consistent definition for the signal component across the estimation of the SNR and CPA. The signal is defined to be the Hamming distance of the SubBytes output of the AES state byte one in the final encryption round. We select the last encryption round because it is the "best attack target" for this AES design: this AES implementation forwards the shifted S-Box results directly to the result registers during the last round. We select the state byte with index one because it ensures that all Hamming distances are possible (this would also be true for the state bytes two to 15, but not state byte zero, which for reasons inherent to the specific implementation cannot produce Hamming distance zero). Our notion of attack success is for the correct key guess to be the most likely (top ranked) key.

We select ciphertexts so that all but the byte in state byte one are fixed. We adjust the testing method for the fixed-vs-random test to match the procedure with the SNR and CPA setup; i.e. in the "random set" we fix all ciphertext bytes with the exception of state byte one. In all our analyses, we only consider the points from the last encryption round.

Checking the noise distribution. As explained in Subsection 2.3, we propose to utilize the noise variance directly to judge the efficiency of RO based designs. Therefore we briefly explore the noise distribution produced by each of the RO designs. To do so, we record the power consumption (200,000 traces) of each design in isolation [4]: we show a representative result from one arbitrarily chosen time point in Figure 2. With the exception of the RO Chains (g), all distributions are approximately Gaussian, fitting with the assumptions and reasoning of how the noise variance impacts on SNR, CPA, and TVLA provided in Subsection 2.3.

In order to relate the variance to the concrete implementation, we measure the SNR of the unprotected AES and compare it to the SNR with the active countermeasures. We define the signal as the variance between mean power traces of different hamming distance groups during the last round. This approach is selected because the chosen AES implementation forwards the shifted S-box output directly to the result registers during the last round. To clearly identify the effect of different RO designs, we compute the SNR on semi-fixed plaintexts, chosen such that only one ciphertext byte is random while all other bytes are fixed to 0. We select byte 1 as the random byte, since for our key all Hamming distance classes are populated. In contrast, a Hamming distance of 0 cannot occur in byte 0, for example. This eliminates noise that could stem from processing completely random plaintexts. To rule out outliers, we repeat this experiment 10 times and average the resulting SNRs.

Furthermore, we analyze the MTD of Test Vector Leakage Assessment (TVLA) and CPA. For TVLA, we estimate the number of traces required to exceed the detection threshold and indicate leakage. To this end, we measure two sets of traces: First, a fixed set, in which the plaintext is held constant, and, second, the semi-fixed set, in which all plaintext bytes are chosen such that only ciphertext byte 1 is random. Additionally, we only analyze the TVLA during the last round, as we are only interested in the leakage caused by the effects of the first ciphertext byte.

The traces we record for the TVLA are also used for CPA experiments and attack byte 1 of the output. We assess how many traces are needed for the the correct key byte candidate to reach rank 1 in a CPA. For TVLA, we measure after how many traces the t-value (Equation 3) exceeds the threshold of 4.5. For all of these experiments, we run the evaluation 10 times to rule out outliers due to data specific leakage. Lastly, we specify the resource consumption of the countermeasures in terms of area, specifically as LUTs and slices on an FPGA, and power consumption.

Filter We consider adversaries that exploit the fact that RO designs may only leak in a specific frequency band. Consequently, our attacks are on preprocessed traces that retain only the frequency band in which the target design leaks. To determine this frequency band, we use the method by [SWL21]; see also Section 2, and analyse the SNR of the band-filtered unprotected AES. In Figure 3, we show the maximum SNR for different bands after applying a 10 MHz bandwidth bandpass filter; we start at 5 MHz (the lowest 5 MHz are filtered with a 5 MHz lowpass filter). We employ a 5th-order Butterworth bandpass filter. The filter is applied forward and backward (zero-phase), so it introduces no time shift of the leakage relative to the trigger. This doubles the effective attenuation, giving -6 dB at the band edges, about 60 dB of rejection half a bandwidth outside each

[4] We keep the partition with the AES design, but supply the algorithm with a fixed input and a fixed key.

edge, and a roll-off of 200 dB/decade. The SNR is maximal for the frequency band between 5 MHz and 15 MHz. Therefore, in our setup, we filter all measured traces with such a bandpass-filter. Note that these frequency-exploration results are specific to our setup and AES design; they do not carry over to other designs or platforms. I.e. the same AES design on a different FPGA, and/or a different setup, is likely to leak in a different part of the frequency spectrum. Furthermore, we confirmed that the frequency band does not shift even when ROs by rerunning the experiment with countermeasures applied.

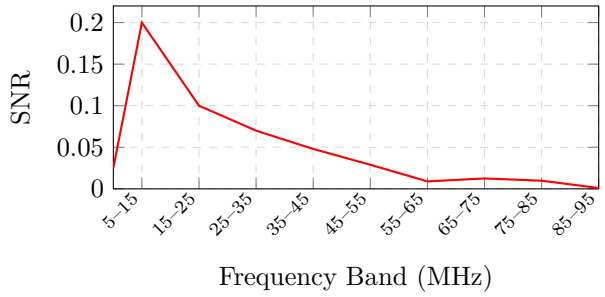


Figure 3: SNR of an unprotected AES without countermeasures by frequency band.

To demonstrate the fact that RO designs may contribute to only selected parts of the frequency spectrum, we show the results of using the Active Fences [KGS+19] design alongside the AES design. This RO design toggles large logic blocks; these blocks slowly start up after receiving their enable signal. Because of this, the strongest contribution of this design sits at very low frequencies, as the frequency plot in Figure 4 shows. The FPGA with the AES design and the RO design runs at 10 MHz. Therefore, the AES design, which runs synchronously with the clock, leaks mostly around 10MHz; the RO design does contribute there as well, but most of its contribution sits at the very bottom of the spectrum. An adversary using a bandpass filter will largely circumvent the RO design's effect. Figure 5 shows that before filtering, a large amount of noise with high variance is generated, but after applying the bandpass filter, only a tiny fraction of this noise is left. Consequently, we will conduct our analyses by mimicking such an adversary: we filter all measurements and focus only on the 5-15MHz frequency band where most of the information is concentrated.

Given these strong effects of filters on the resulting quantities, it is highly required that noise generators are evaluated under appropriate filtering. These filters need to be tailored to the target implementation and system, e.g., following the strategy discussed above by [SWL21]. Thus, we yield the results in Table 3, which are discussed next.

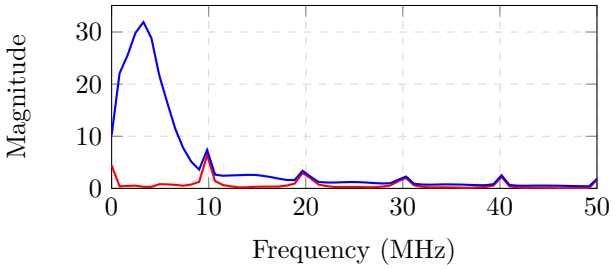


Figure 4: FFT of an unprotected AES (red) and an AES employing active fences (blue).

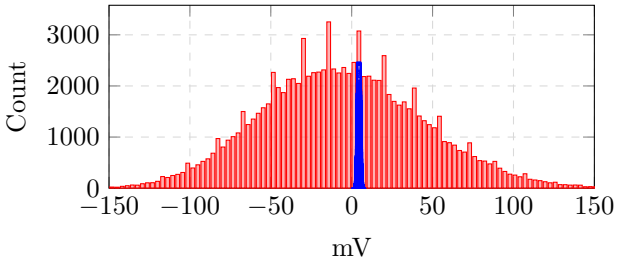


Figure 5: Noise distribution of active fences without filtering (red) and with filtering (blue).

4.2 Results

We conduct all experiments using the above presented setup under the same conditions. The implementation of the noise generators are based on the information provided by prior work, summarized in Table 2. This includes possible duplicate instantiations in order to match the amount of S-boxes in our target design or the overall area taken by our chosen AES implementation. As discussed above, to enable a fair comparison, we set the input variables to pseudorandom values generated by the PRNG, also if prior work specified secret-dependent values as the input. Lastly, in order to compare RO countermeasures to noise that might be generated by simply running on-board blocks in parallel, we implemented a Keccak core and evaluated the generated noise as well. The results are shown in Table 3.

Table 3: Evaluation Results of the Previously Described Proposals in the 5-15 MHz band.

Countermeasure		**Results**				**Area**		**Power**
		SNR	Noise Variance	# Traces for TVLA	# Traces for CPA	LUTs	Slices	mW
Unprotected AES		0.2	0.001	10	96	1414	415	87.12
Digital Controlled ROs	[LCL10, NYH13]	0.008	0.3	230	2712	446	142	27.92
FiGaRO	[MGZS17]	0.003	0.28	790	7747	656	192	138.83
Active Fences	[KGS+19]	0.001	24.41	13050	>20000	941	314	264.4
Xored High Consuming Module	[PGC+22]	0.11	0.003	20	176	22	7	3.48
Configurable RO chains	[NPBH23]	0.00098	70.68	>20000	>20000	4027	1146	213.2
Random Number of ROs	[Sch23]	0.06	0.018	40	180	329	100	5.49
Programmable RO	[YKS+23]	0.05	0.012	40	349	45	17	3.87
Keccak		0.022	0.02	110	804	1077	287	9.17

Aligned with the theoretic results in Subsection 2.3, our results show that, in general, the four quantities are approximately proportional to each other. An increase in the noise variance generally correlates with a decrease of the SNR and an increase in the number of required traces for TVLA and CPA. For comparability with SNR numbers reported in other side-channel literature, we note that our SNR results are this high primarily due to the filtering. The unfiltered SNR values, especially that for the unprotected AES implementation, are two orders of magnitude lower than the filtered values.

These results give us a comprehensive picture of the impact of different noise generators. Our methodology ensures a *like for like* comparisons between the designs. However, we do not provide a worst-case security analysis here. In other words, there are better attacks available, e.g. using profiling, using multiple intermediate values, etc. that would bring down the number of needed traces for a successful attack. In the following, we use the comparative capability of our method to compare the different designs in terms of security, resource utilization, and efficiency.

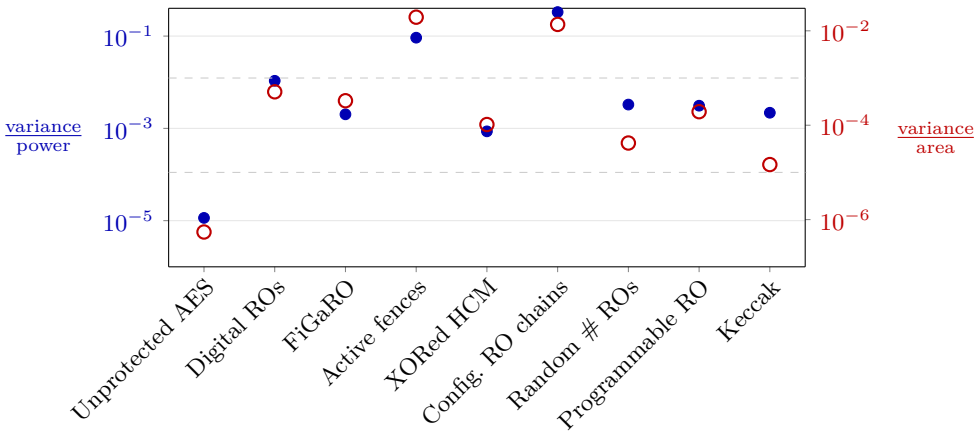


Figure 6: Efficacy of RO-based noise generators

Comparison. The required trade-off between security enhancement of the noise generator and its resource utilization is given by the requirements and constraints of the target system. In this work, we provide a rough estimate of this trade-off by computing the ratio of generated noise variance to area and to power. The resulting efficiency ratios are shown in Figure 6 together with the baseline of the noise efficacy of the unprotected AES.

It can be seen that the efficiency of the Digital Controlled ROs [LCL10, NYH13], the Xored High Consuming Module [PGC+22] and the random number of ROs [Sch23], lie around the common baseline. All of these noise generators only instantiate a certain amount of single-stage ROs that are randomly activated. This seems to limit their hiding effectiveness.

The Programmable RO [YKS+23] also shows no significant impact on the system noise level, but its very small area and low power consumption lead to a higher efficiency than the single-stage ROs. This effect might be explained by the noise amplification due to connecting the output to an IO-pin and dynamically varying the chain length. The impacts of these two factors on the security of hiding countermeasures are analyzed in detail in the next section.

FiGaRO [MGZS17] and Active Fences [KGS+19] are roughly an order of magnitude larger in area than the previously considered solutions. This pays off as the noise variance also increases by about an order of magnitude. However, the power consumption is roughly two orders of magnitude higher than that of the unprotected AES. As shown in Section 4.1 for Active Fences, the generated noise spectrum is broader-band than the secret signal and is therefore less efficient in the target band.

Configurable RO chains [NPBH23] again increase the noise variance by about an order of magnitude. However its size is approximately ten times larger than FiGaRO and Active Fences and approximately the same size of the AES. The power consumption is lower than that of active fences, better matching the secret frequency band. However, the power consumption remains roughly twice that of the unprotected AES.

By comparing these different designs, we can determine which design choices lead to more efficient noise generators. However, this evaluation does not explain the specific effects of the configuration options. If the target application has constraints that none of the designs meet, the strengths of the different designs need to be combined. In order to enable the design of more powerful noise generators, we provide a guideline for it in the next section.

5 Guideline

The different configuration options of RO-based countermeasures, which we describe in Section 3 affect the efficiency of the noise generator in different ways. In this section, we analyze these effects separately, before aggregating them to develop tailored noise generators for our setup. For the sake of readability, we only provide the numbers of the generated noise variance for the results in this section. All evaluations, except the output frequency measurements, are performed on filtered traces in the 5-15 MHz range at 20 °C.

Chain Length. Recall Equation 5 and that increasing the number of inverters n in the RO chain causes the output frequency to scale inversely with n (i.e., by $\frac{1}{n}$). We measure the output frequency by routing the RO output directly to an IO pin of the board. We then capture this signal and estimate its frequency. Figure 7 shows the measured output frequency as a function of RO chain length.

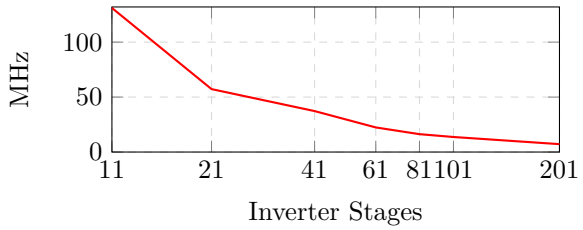


Figure 7: Output frequency of an RO measured through a IO pin.

The per-inverter switching frequency is set by the inverter delay t_p and, in contrast to the RO output frequency, does not change with the number of stages. If the RO output does not drive an external load, the dynamic power is dominated by the internal switching of the inverters. Thus, the dominant component in the power spectrum should be the stage switching frequency of a single-stage RO, given by $f_1 = \frac{1}{2t_p}$, no matter how long the chain length is. Equivalently, n stages produce n interleaved current spikes per period $T_{ro} = \frac{1}{f_{ro}}$, yielding an aggregated frequency at f_1, which is independent of n. In our case, $f_1 \approx 1.57$ GHz which is beyond the bandwidth of our laboratory instruments. If our lab equipment could measure it, the parasitic capacitances of board and probe would act as a low-pass filter, strongly attenuating this spectral component and making it unobservable.

Number of Ring-Oscillators. This effect also appears in our experiments that investigate the impact of increasing the number of ROs. The common assumption in related work, which we also adopt, is that the noise variance increases linearly with the number of ROs. However, neither measurements with varying numbers of 125-stage ROs ($f_{ro} \approx 12.6$ MHz) nor those with 201-stage ROs ($f_{ro} \approx 7.84$ MHz) show any change in the resulting noise variance. Enabling or disabling filtering does not change the results. Also the fact whether filtering of the data is enabled or disabled does not change the result. This further indicates that, without driving an external load, the noise is so high-frequent that it has no observable effect on the attack measurements.

A different picture emerges when instantiating varying numbers of ROs with randomized chain lengths such as the programmable RO countermeasure. We modified it to contain stage counts that fall within our 5 to 15 MHz range. It changes the frequency every cycle, while not driving an external load. While one instance only created a variance of 0.008, 50 instances yield a variance of 0.050 and 100 instances create a variance of 0.074. This indicates, that if a single active dynamic RO instance has an impact on the noise variance, increasing the number of instances results in an increase of the variance.

Amplifying the Output. In order to amplify the output signal of an RO, it can drive another circuit. Toggling at the RO output induces additional switching activity, which effectively amplifies the noise power consumption. The external loads used in the evaluated designs either employ `xor` logic circuits as in [MGZS17, KGS+19, PGC+22] or an IO-pin as in [KGS+19, YKS+23] as the external load.

To achieve an amplifying effect in the power consumption, the load must draw high current. This can be achieved by connecting high-current-drawing elements such as an IO-pin. We evaluated the effect of connecting an IO-Pin to the RO output by measuring the programmable RO design of [YKS+23] once without and once with the IO-pin connected. The results are shown in Table 4: Driving the IO-pin slightly the noise variance in our setup. The overall power consumption of the noise generator increases roughly by a factor of two, as it now drives the relatively large IO net. Similarly, when connecting an IO pin to a static 201-stage RO, the variance and power usage slightly increases.

The high current drawing effect can also be achieved by driving many parallel switching gates with the RO. This can be done by connecting the RO output to a large circuit. We evaluate this effect by connecting the RO output to various circuits. The results are shown in Table 4.

Table 4: Evaluation results of different external load experiments.

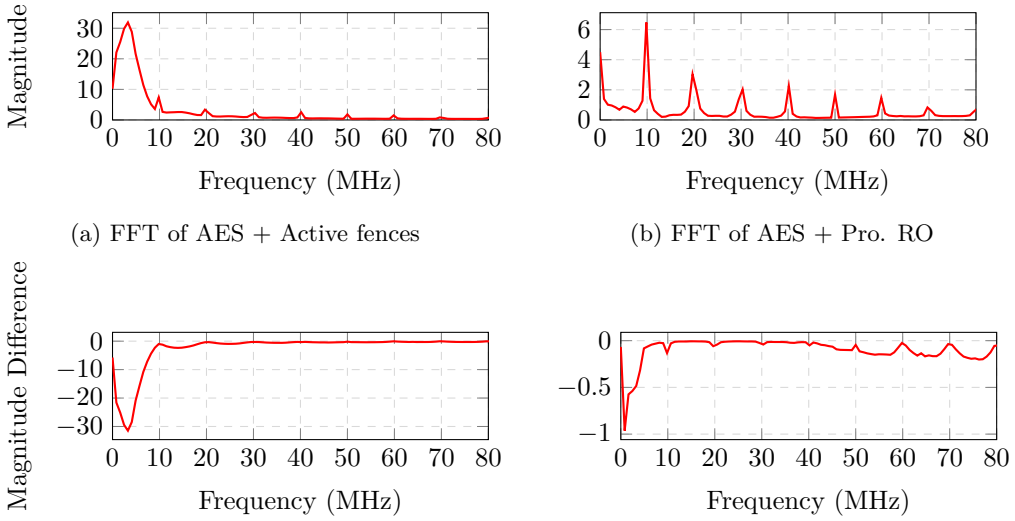
RO Type	Output Driver	Variance	Area		Power
			LUTs	Slice	mW
Static RO 201 Stages	-	0.0014	159	47	1.64
Static RO 201 Stages	IO Pin	0.0027	159	49	2.91
Static RO 201 Stages	Keccak Clock	4.81	1270	350	64.5
Static RO 201 Stages	LFSR32 Clock (SRL)	0.0017	229	68	1.75
Static RO 201 Stages	LFSR96 Clock (SRL)	0.0017	253	77	1.95
Static RO 201 Stages	LFSR1024 Clock (SRL)	0.0141	285	87	2.12
Static RO 201 Stages	LFSR1024 Data (SRL)	0.0018	210	63	2.06
Static RO 201 Stages	LFSR1024 Clock (FDRE)	0.169	248	274	6.15
Programmable RO	-	0.0008	45	15	1.89
Programmable RO	IO Pin	0.013	45	17	3.87

They show that driving a circuit using the RO can have an effect on the noise variance. For a LFSR, both the underlying implementation as well as the driven input matters. If the RO output drives the data input, we do not see any significant changes in variance as the RO only drives the first input stage of the LFSR. The remaining stages of the LFSR are switching at the system clock frequency. Instead of connecting the RO output to the data input of the LFSR, it can be connected to its clock input. Here the effects are more pronounced with larger LFSRs causing more variance. The variance is further increased by changing the synthesis settings. As Table 4 indicates, an RO-driven LFSR built from Shift-Register LUTs (SRLs) (a common FPGA building block) showed only a small increase in the noise variance, while the same configuration built with standard registers increased the variance significantly.

If a Keccak module or another on-board block is available, routing the RO output to its clock input yields even better results than driving an LFSR. This is likely because the combinational logic between the Keccak registers further increases its power consumption. However, as seen in the table, the power consumption rises significantly. As the RO output is not a proper clock signal, but instead a normal signal like any other gate output, it does not satisfy the usual requirements for a clock signal. In particular, routing one signal to hundreds of flip-flops introduces skew, which in term leads to glitching.

We conclude that connecting an external load to the RO output leads to an amplification of RO-generated noise. The larger the input net driven by the RO, the higher the amplification. Note that large circuits alone are not sufficient, as it is primarily the size and capacitance of the driven input net that determine the increase in noise power.

Dynamic ROs. As discussed above, dynamic ROs can be realized by either dynamically activating full RO instances, or by dynamically changing the chain length. We evaluate the effects created by dynamic ROs by analyzing the resulting frequency spectrum of the Active Fences countermeasure [KGS+19] in Figure 8 and compare it to the spectrum generated by the Programmable RO countermeasure [YKS+23]. Active fences use static ring oscillator blocks and toggle a random amount on and off to generate noise.This is in turn creates low frequency broadband noise as seen in the FFT spectrum, because the blocks slowly start drawing power as the ring oscillators start up. However, no significant noise is generated in other frequencies and the broadband noise becomes less powerful in higher frequency bands. This is especially noticeable when calculating the difference between an unprotected AES spectrum and spectrum of a AES protected by active fences. The frequencies clearly differ below 10 MHz, but equalize after 30 MHz. In contrast to this, programmable ROs only create minor low frequency noise, which is probably caused by activating chains within the oscillator or by the relatively large IO net. Calculating the FFT difference reveals that the design generates noise over a broad spectrum above 40 MHz. This is consistent with the the amount of stages which can dynamically range between 1 and 57 stages in the original design.



(a) FFT of AES + Active fences

(b) FFT of AES + Pro. RO

(c) Difference in frequencies between unprotected AES and AES + Active fences

(d) Difference in frequencies between unprotected AES and AES + Pro. RO

Figure 8: Effects of enable signal randomization (active fences) vs. chain length randomization (programmable RO).

Randomization of Dynamic ROs. In order to dynamically change between frequencies or activate RO blocks at random, the hiding module requires a PRNG. Here, the designer has to decide on the PRNG design and on the update rate. We assess the effect of the PRNG design by instantiating our tailored Keccak design (Subsection 5.1) with the previously used xorshift32 PRNG and with a simple 32-bit counter as an example of a bad PRNG. In this design, only the lower 6 bits of each PRNG output are used. Compared to xorshift32, the counter-driven design has a 23.4 % lower variance, which shows the importance of a good PRNG. We expect the variance to drop even further for designs that require more random bits but are driven by a counter, as the higher-order bits of a counter remain static for long periods.

The second choice is how often the PRNG updates the RO configuration. We assess

this by evaluating the original programmable RO countermeasure with different update intervals. Our measurements show that less frequent updates do not significantly influence the power consumption. Updating every cycle yields a variance of 0.012. Updating every 4, 9, and 49 cycles yields 0.0096, 0.0082, and 0.0064, respectively. We see that the noise variance thus decreases steadily with longer update intervals. This is expected as, for example, under ideal randomness, the active fences countermeasure always activates around 50 % of all RO blocks, regardless of the update interval. Our xorshift32 module only draws 1 mW and only consumes 73 LUTs and 23 registers, and is thus not a major concern when implementing a dynamic RO countermeasure. We can conclude that designers should generate new randomness as often as possible. In fact, we found in additional experiments that by increasing the update frequency above the clock rate further raises the noise variance.

Design Guideline We now condense these findings into concrete steps for system designers who aim to integrate an RO-based noise generator into a known target. First, the frequency band with the highest SNR must be determined. Adjacent bands may also contain an exploitable signal, so a designer may want to consider them as well. Afterwards, the delay of a single inverter should be characterized or extracted from the documentation of the technology node. Based on this information, appropriate chain lengths that cover the vulnerable frequency band can be chosen. Since we found side-channel leakage over a broad band, we generally advise using a dynamic RO configuration that covers multiple frequencies. Next, the RO output must be connected to drive a load, to amplify the switching behavior and generate effective noise in the vulnerable band. The clock input of a large register-based circuit, such as an existing on-board block or a dedicated LFSR, is an effective choice. The RO configuration should be updated by a good PRNG, in every clock cycle or even more frequently. Finally, the designer assesses the noise variance of filtered traces. If the noise is still insufficient, the load or the number of RO instances is increased and the evaluation is repeated until the requirements are met.

5.1 Building a Tailored RO-Based Noise Generator

Table 5: Evaluation Results of Different External Load Experiments.

Output Driver	Results				Area		Power
	SNR	Noise Variance	# Traces TVLA	# Traces CPA	LUTs	Slice	mW
Keccak	0.0009	40.64	>20000	>20000	1390	381	115.03
LFSR1024 x16 (SRL)	0.002	0.95	3170	9033	913	254	24.72
LFSR1024 x4 (FDRE)	0.001	8.66	>20000	>20000	309	805	81.70
LFSR1024 x1 (FDRE)	0.003	0.67	3120	11669	298	263	21.71
LFSR512 x1 (FDRE)	0.009	0.14	640	3575	323	165	17.27

Based on these guidelines, we design an RO-based noise generator tailored to our specific system parameters. As previously noted, the target AES core operates at 10 MHz, with the primary leakage occurring in the 5 to 15 MHz frequency band. Therefore, we configure a dynamic RO to operate within this specific range. The proposed design builds upon the programmable RO countermeasure, offering a compact footprint while enabling dynamic frequency switching. We instantiate an inverter chain with 105 stages, setting the base RO frequency at the upper limit of the target band. To achieve the lower end of the frequency band, we integrate additional delay stages (2, 6, 12, 24, 48, 98), allowing the RO length to be expanded up to 295 stages.

For noise amplification, we evaluated multiple approaches as seen in Table 5. If a Keccak module is available on the device, routing the RO output to its clock input proves highly effective. This configuration yields a noise variance of 40.64, which is almost double the variance generated by active fences as presented in Section 4. While the generated

variance is lower than the one generated by configurable RO chains, it requires less area and power to operate, while targeting the exact frequency spectrum. Alternatively, we consider driving the clock input of a long LFSR that either utilizes registers or SRLs. The data input is provided by the PRNG. Specifically, driving a single 512-bit LFSR produced a variance of 0.14, incurring only a moderate a hardware overhead. Scaling the LFSR approach improved the variance to 0.67 for a single 1024-bit LFSR, and up to 8.66 when deploying four parallel 1024-bit LFSRs.

6 Cautionary Notes

In the following we discuss several limitations that effect the operation of ROs and must thus be considered when integrating them into a target design.

Environmental Effects The operating environment of a circuit can affect its timing characteristics. The switching speed of a CMOS transistor is determined by its drain current I_D. At higher temperatures, the threshold voltage V_{th} decreases, which increases I_D and thus speeds up switching. At the same time, the carrier mobility decreases with temperature, which lowers I_D and slows down switching. Whether gate delay increases or decreases with temperature thus depends on whether the technology node is mobility-dominated. Nodes below 45 nm have been observed to be threshold-voltage-dominated [KK06, DBC+26], meaning the gate delay decreases with higher temperature. We confirm this effect on our 28 nm FPGA. We measure the frequency of simple ROs at 0, 20 and 40 °C in the climate chamber. The results are shown in Table 6.

Table 6: RO frequencies at different temperatures.

	Frequency [MHz]		
Stages	0 °C	20 °C	40 °C
201	7.02	7.05	7.08
101	14.84	14.91	14.99
61	22.58	22.68	22.79
21	62.02	62.30	62.61
11	112.36	112.45	113.78

While we observe a slight speedup at higher temperatures, the resulting drift of around 1 MHz is small compared to the 10 MHz bandwidth of our filters. We confirmed this by re-running our leakage experiments at different temperatures, which yielded the same results as described in Section 4. If the temperature cannot be manipulated, an adversary may instead undervolt or overvolt the circuit, which causes similar slowdown and speedup effects. However, the specifics of such an attack are out of scope for this work.

Placement and Routing. In this paper, we restrict the RO placement in order to prevent inadvertent coupling of secret-dependent data into the generated noise. However, under an expanded adversarial threat model, the placement strategy significantly impact the effectiveness of the RO-based noise generator. Assuming an adversary capable of accurately recording Electromagnetic (EM) emissions, placing a noise-generation block in close proximity to the hiding target becomes necessary. Conversely, research shows that the RO itself must remain physically isolated from the target. In [SMS20] it is demonstrated that if an RO is located close to gates or wires that transmit the signal, the secret can couple into the RO's switching behavior and, thus, get amplified. Lundberg [Lun25] uses this effect by inserting an RO as a hardware trojan and exploiting the increased signal

strength. Future work should investigate how these competing placement requirements influence the overall efficiency of RO-based noise generators. In particular, an evaluation on an Application-Specific Integrated Circuit (ASIC) could yield valuable insights, as its multi-layered architecture offers three-dimensional spatial flexibility, contrasting with the strictly two-dimensional routing plane of an FPGA.

Power Distribution Network. Another factor that can influence the effectiveness of an RO-based noise generator are the electrical characteristics of the Power Distribution Network (PDN) that could either amplify or attenuate the effect of the generated noise. Depending on the PDN impedance profile, spectral components of the generated noise may coincide with resonances or anti-resonances, leading to amplification or suppression, respectively. To investigate this effect, we measured the S_{11} parameter of our setup using a vector network analyzer. The measurement indicated two impedance minima at approximately 340 MHz and 614 MHz. Noise components in these frequency ranges would therefore be expected to be attenuated rather than amplified. However, we could not experimentally verify this behavior with our setup, as with $f_1 \approx 1.57$ GHz it is not possible to construct an RO with an oscillation frequency in these ranges. As discussed above, also an activation frequency of the individual ROs in this range does not allow us to show this effect as the relevant part of the power consumption is realized by amplifying the RO output signal, and is thus not in the range of our impedance minima.

Platform Dependency. This stage switching frequency f_1 is one of the reasons why an RO-based noise generator design cannot be transferred from one system to the other without re-evaluating the generated noise. A changing technology leads to a different transition time of the inverters which, in turn, changes f_1. Since also the synthesis itself is usually not deterministic, also designs for the same target might need to be re-evaluated after each synthesis run. To avoid unforeseen effects like this in our experiments, we used the AMD Vivado partial reconfiguration flow and only updated the noise generator partition far apart from the AES partition.

7 Conclusion

In this work, we addressed two main questions. First, we discussed how additive noise generators can be evaluated systematically to enable a fair comparison. Second, we investigated whether ring-oscillator-based noise generators, which are regularly proposed in research literature, can be designed effectively and in a tailored way. Regarding the comparability, we found that all considered evaluation metrics—noise variance, SNR, the number of traces required for CPA and TVLA—can give meaningful insights, if applied carefully. In particular, the frequency spectra of the secret-dependent signal and the generated noise must be considered, since noise outside the relevant frequency range may be filtered out and therefore provide little practical protection.

We then applied this evaluation strategy to existing RO-based noise-generator configurations and analyzed which designs best hide the secret-dependent signal of our AES implementation. We also compared these designs to a Keccak block running in parallel as an alternative noise source and showed that its efficiency trade-off lays in the same range as the RO-based designs from previous work. Based on this comparison, we analyzed the different RO configuration options in more detail and derived a guideline for designing RO-based noise generators tailored to specific systems. In particular, we observed that effective noise generation requires the RO output to be amplified, by connecting it to a power-wasting net such as the clock network of another circuit. Following our guideline, we designed a tailored RO-based noise generator for our experimental setup, achieving

an 3.8-fold improvement in power efficiency and an 52.6% improvement in area efficiency compared to previous RO-based designs.

Nevertheless, our results do not provide sufficient evidence to conclude that RO-based designs are generally the best choice for additive noise generation. They come with several limitations, including placement constraints, routing effects, and strong platform dependency, all of which must be considered when selecting a suitable countermeasure. Moreover, other hiding techniques, such as shuffling, the insertion of dummy operations, or clock-frequency modulation, may also be effective ways to increase the noise level. Therefore, RO-based noise generators can be a useful and efficient option when carefully tailored to the target system, but their effectiveness must always be evaluated with respect to the concrete system requirements.

References

[Chu16] Chunghwa Telecom Co., Ltd. HiKey PKI Token: FIPS 140-2 Security Policy. FIPS 140-2 Non-Proprietary Security Policy (module based on Renesas RS45C), September 2016. Section 8.3, Mitigation of Other Attacks.

[CVP24] Jack Connor, Nikita Veshchikov, and Lutz Pape. Method and circuit for protecting an electronic device from a side-channel attack. European Patent Application EP 4 307 155 A1, applicant NXP B.V., January 2024. Priority date 13 July 2022.

[DBC+26] Fatemeh Khojasteh Dana, Andrew Butler, Mehmet Ali Cetin, Yuval Yarom, Xinrui Wang, and Shahin Tajik. Chypothermia: Clock freezing for static side-channel attacks, 2026.

[DDF14] Alexandre Duc, Stefan Dziembowski, and Sebastian Faust. Unifying leakage models: From probing attacks to noisy leakage. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 423–440. Springer, Berlin, Heidelberg, May 2014.

[Del16] Vivien Delport. Randomizing current injection circuit to obscure gate noise for added security. US Patent 9,306,746 B2, assigned to Microchip Technology Inc., April 2016. Priority date 14 March 2013.

[DMN+17] Debayan Das, Shovan Maity, Saad Bin Nasir, Santosh Ghosh, Arijit Raychowdhury, and Shreyas Sen. High efficiency power side-channel attack immunity using noise injection in attenuated signature domain. In *IEEE HOST 2017*, pages 62–67. IEEE Computer Society, 2017.

[DS16] François Durvaux and François-Xavier Standaert. From improved leakage detection to the detection of points of interests in leakage traces. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part I*, volume 9665 of *LNCS*, pages 240–262. Springer, Berlin, Heidelberg, May 2016.

[GM11] Tim Güneysu and Amir Moradi. Generic side-channel countermeasures for reconfigurable devices. In Bart Preneel and Tsuyoshi Takagi, editors, *CHES 2011*, volume 6917 of *LNCS*, pages 33–48. Springer, Berlin, Heidelberg, September / October 2011.

[Grö07] Markus Grözing. Untersuchung des rauschens in komplementären metall-oxid-halbleiter-ringoszillatoren. 2007.

[Gün12] Tim Güneysu. Using data contention in dual-ported memories for security applications. *J. Signal Process. Syst.*, 67(1):15–29, 2012.

[Gun13] Kiran Kumar Gunnam. Hardware countermeasure against cryptographic attack. US Patent Application 2013/0111204 A1, assigned to Certicom (U.S.) Limited, May 2013. Filed 27 October 2011; granted as US Patent 8,627,131 B2 (7 January 2014).

[HFG+25] Dina Hesse, Jakob Feldtkeller, Tim Güneysu, Julius Hermelink, Georg Land, Markus Krausz, and Jan Richter-Brockmann. t-probing (in-)security - pitfalls on noise assumptions. Cryptology ePrint Archive, Report 2025/1202, 2025.

[KBG09] Najeh Kamoun, Lilian Bossuet, and Adel Ghazel. Correlated power noise generator as a low cost dpa countermeasures to secure hardware aes cipher. In *SCS 2009*, pages 1–6. IEEE, 2009.

[KBG11] Najeh Kamoun, Lilian Bossuet, and Adel Ghazel. A masked correlated power noise generator use as a second order dpa countermeasure to secure hardware aes cipher. In *ICM 2011 proceeding*, pages 1–5. IEEE, 2011.

[KGS+19] Jonas Krautter, Dennis R. E. Gnad, Falk Schellenberg, Amir Moradi, and Mehdi Baradaran Tahoori. Active fences against voltage-based side channels in multi-tenant fpgas. In David Z. Pan, editor, *Proceedings of the International Conference on Computer-Aided Design, ICCAD 2019, Westminster, CO, USA, November 4-7, 2019*, pages 1–8. ACM, 2019.

[KJJ99] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Michael J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 388–397. Springer, Berlin, Heidelberg, August 1999.

[KJJ01] Paul C. Kocher, Joshua M. Jaffe, and Benjamin C. Jun. Differential power analysis method and apparatus, December 2001.

[KK06] Ranjith Kumar and Volkan Kursun. Reversed temperature-dependent propagation delay characteristics in nanometer CMOS circuits. *IEEE Trans. Circuits Syst. II Express Briefs*, 53-II(10):1078–1082, 2006.

[KMSS18] Raghavan Kumar, Sanu K. Mathew, Sudhir K. Satpathy, and Vikram B. Suresh. Power side-channel attack resistant advanced encryption standard accelerator processor. US Patent 10,103,873 B2, assigned to Intel Corporation, October 2018. Priority date 1 April 2016.

[Kni06] Oliver Kniffler. Method for operating a microprocessor configuration and microprocessor configuration. US Patent 7,020,730 B2, assigned to Infineon Technologies AG, March 2006. Priority date 19 January 2000.

[LCL10] Po-Chun Liu, Hsie-Chia Chang, and Chen-Yi Lee. A low overhead DPA countermeasure circuit based on ring oscillators. *IEEE Trans. Circuits Syst. II Express Briefs*, 57-II(7):546–550, 2010.

[Lun25] Fanny Lundberg. Exploiting ring oscillators to leak secret information. 2025.

[Mar03] George Marsaglia. Xorshift rngs. *Journal of Statistical software*, 8:1–6, 2003.

[MGZS17] Jia-Wei Ma, Xuguang Guan, Tong Zhou, and Tao Sun. A new countermeasure against side channel attack for HMAC-SM3 hardware. In Yajie Qin, Zhiliang Hong, and Ting-Ao Tang, editors, *12th IEEE International Conference on ASIC, ASICON 2017, Guiyang, China, October 25-28, 2017*, pages 327–330. IEEE, 2017.

[MOP07] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power analysis attacks: Revealing the secrets of smart cards*. Springer, 2007.

[NEC04] NEC. V-WAY64 V3.0 (μPD79216000) Security Target Lite. Common Criteria Security Target Lite, version V1.00, Ref. 33-55N1-10001, French certification scheme (DCSSI), July 2004. Section 6.1.5, security function SF5 "Unobservability".

[NPBH23] Hassan Nassar, Simon Pankner, Lars Bauer, and Jörg Henkel. Late breaking results: Configurable ring oscillators as a side-channel countermeasure. In *60th ACM/IEEE Design Automation Conference, DAC 2023, San Francisco, CA, USA, July 9-13, 2023*, pages 1–2. IEEE, 2023.

[NYH13] Nianhao Zhu, Yujie Zhou, and Hongming Liu. Counteracting leakage power analysis attack using random ring oscillators. In *PROCEEDINGS OF 2013 International Conference on Sensor Network Security Technology and Privacy Communication System*, pages 74–77, Harbin, China, May 2013. IEEE.

[PCT+15] Luis Parrilla, Encarnación Castillo, Elias Todorovich, Antonio García, Diego Pedro Morales, and Guillermo Botella. Improvements for the applicability of power-watermarking to embedded ip cores protection: E-coreipp. *Digital Signal Processing*, 44:110–122, 2015.

[Ped14] Bruce B. Pedersen. Programmable logic device with improved security. US Patent 8,812,869 B1, assigned to Altera Corporation, August 2014. Priority date 3 December 2009.

[PG15] ProjectVault and Google Inc. orp: mselsoc AES crypto core (verilog rtl). https://github.com/ProjectVault/orp/tree/master/hardware/mselSoC/src/systems/geophyte/rtl/verilog/crypto_aes/rtl/verilog, 2015. GitHub repository, accessed April 16, 2026.

[PGC+22] Luis Parrilla, Antonio García, Encarnación Castillo, Salvador Rodríguez-Bolívar, and Juan Antonio López-Villanueva. Time-and amplitude-controlled power noise generator against spa attacks for fpga-based iot devices. *Journal of Low Power Electronics and Applications*, 12(3):48, 2022.

[PHF08] Thomas Plos, Michael Hutter, and Martin Feldhofer. Evaluation of Side-Channel Preprocessing Techniques on Cryptographic-Enabled HF and UHF RFID-Tag Prototypes. pages 114–127, 2008.

[Ren09] Renesas Technology Corp. AE57C1 Security Policy. FIPS 140-2 Non-Proprietary Security Policy, document RTA-AE57C1-BOS2.1-P0001, Rev. 2.04, December 2009. Section 10, Mitigation of Other Attacks.

[Sam09] Samsung Electronics Co., Ltd. Security Target Lite of Samsung S3CC91A 16-bit Secure RISC Microcontroller for Smart Card with optional Secure RSA Crypto. Library and specific IC Dedicated Software. Common Criteria Security Target Lite, version 1.0, November 2009. Section 2.1, paragraph 14.

[Sch23] Patrick Schaumont. You Can Hide but You Can't Verify: On Side-channel Countermeasure Verification. *Workshop on SSH-SoC, 2023*, 2023.

[SMS20] Zeinab Seifoori, Seyedeh Sharareh Mirzargar, and Mirjana Stojilović. Closing leaks: Routing against crosstalk side-channel attacks. In *Proceedings of the 2020 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, pages 197–203, 2020.

[SWL21] Dor Salomon, Amir Weiss, and Itamar Levi. Improved filtering techniques for single-and multi-trace side-channel analysis. *Cryptography*, 5(3):24, 2021.

[TPZA25] Erica Tena-Sánchez, Francisco Eugenio Potestad-Ordóñez, Virginia Zúñiga-González, and Antonio J Acosta. Low-cost full correlated-power-noise generator to counteract side-channel attacks. *Applied Sciences*, 15(6):3064, 2025.

[YKS+23] Yuan Yao, Pantea Kiaei, Richa Singh, Shahin Tajik, and Patrick Schaumont. Programmable ro (pro): A multipurpose countermeasure against side-channel and fault injection attack. In *Security of FPGA-Accelerated Cloud Computing Environments*, pages 297–325. Springer, 2023.