Design for Trust Utilizing Rareness Reduction

Aruna Jayasena and Prabhat Mishra
Department of Computer & Information Science & Engineering
University of Florida, Gainesville, Florida, USA

***Abstract*—Increasing design complexity and reduced time-to-market have motivated manufacturers to outsource some parts of the System-on-Chip (SoC) design flow to third-party vendors. This provides an opportunity for attackers to introduce hardware Trojans by constructing stealthy triggers consisting of rare events (e.g., rare signals, states, and transitions). There are promising test generation-based hardware Trojan detection techniques that rely on the activation of rare events. In this paper, we investigate rareness reduction as a design-for-trust solution to make it harder for an adversary to hide Trojans (easier for Trojan detection). Specifically, we analyze different avenues to reduce the potential rare trigger cases, including design diversity and area optimization. While there is a good understanding of the relationship between area, power, energy, and performance, this research provides a better insight into the dependency between area and security. Our experimental evaluation demonstrates that area reduction leads to a reduction in rareness. It also reveals that reducing rareness leads to faster Trojan detection as well as improved coverage by Trojan detection methods.**

***Index Terms*—Hardware security, Trojan Detection, Design-for-Trust, Design-for-Test, Rareness Reduction**

I. Introduction

The complexity of the hardware designs continues to grow over the years. To make matters worse, the hardware development life cycle has been shortened significantly. As a consequence, the designers do not have enough time to verify the functional behaviors as well as non-functional (e.g., security) requirements. This opens up opportunities for attackers to implant malicious circuits into the designs that can lead to serious security risks. This research utilizes rareness reduction techniques to improve the security verification process to enable trustworthy hardware systems.

A. Threat Model

We consider the threat model under supply chain vulnerability where the attackers (untrusted foundry, rogue designer, malicious CAD tool [1]) can insert stealthy hardware Trojans that can stay hidden during traditional functional validation and testing. Specifically, an attacker is likely to combine several rare signals with low activation probabilities as the trigger for the Trojan. Once the Trojan is activated, it may alter the functionality, leak sensitive data to the outputs, or perform other malicious activities. Figure 1 shows a simple hardware Trojan that is triggered by two rare signals of p and q, while it flips the design output as the payload.

This work was partially supported by the NSF grant CCF-1908131.

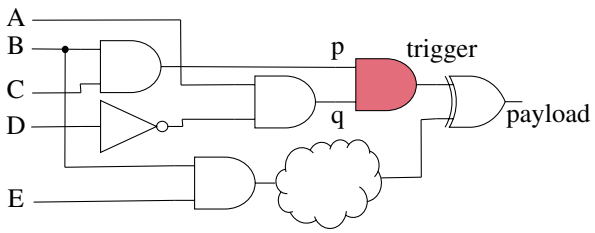


Fig. 1: Hardware Trojan triggered by two rare signals (p,q)

B. Limitations of Existing Methods

Hardware obfuscation (logic encryption) is a promising avenue to build a design-for-trust solution since it is hard for the attacker to figure out the functionality without the key, and therefore, hard to identify where to hide the Trojans [2]–[4]. However, it can lead to unacceptable area, power and performance overhead. Samimi et al. used logic obfuscation to reduce rareness of signals [2]. However, it faces three practical limitations. First, it inherits the disadvantages of obfuscation and leads to significant area (32%), power (9%), and performance (56%) overhead. Next, the requirement of a key itself can be under attack. Finally, the applicability is limited to small combinational designs (applied on simple designs with less than 4000 gates).

There are promising research efforts for efficient detection of hardware Trojans that can be broadly divided into the following categories: statistical test generation methods [5], [6], directed test generation methods [7], machine learning based techniques, side-channel analysis based techniques [8], self-referencing based techniques, and also equivalence checking based techniques [9]. The success of these methods heavily depend on the number of potential triggers for Trojans in the hardware designs. Specifically, if there are too many rare signals or a lot of rare signals with very low probabilities, it would be infeasible for the existing methods to detect any stealthy Trojans constructed from these rare signals. The proposed rareness reduction will be helpful for existing Trojan detection methods, as demonstrated in Section III-D.

C. Research Contributions

In this paper, we look at Trojan detection problem from an orthogonal perspective. We try to eliminate the hiding places for Trojans as much as possible during the design stage. This contributes to design-for-trust from two complementary avenues. (1) The reduced rareness can demotivate the attackers to introduce malicious implants in the design due to less number of potential triggers. (2) Trojan detection approaches can take the benefit of reduced rareness for faster and efficient

Trojan detection. Specifically, this papers makes the following major contributions.

- We perform a theoretical analysis of the root causes of rare signals that are likely to be exploited by adversaries to construct stealthy triggers in hardware Trojans.
- We explore various methods for rareness reduction, including design diversity and area optimization.
- To the best of our knowledge, this is the first attempt in formulating a theoretical relationship between design area and hardware security, and confirming with empirical results on real-world hardware designs.
- Experimental evaluation demonstrates the effectiveness of rareness reduction for Trojan detection using statistical test generation as well as maximal clique activation.

This paper is organized as follows. Section II describes our proposed methodology. Section III presents the experimental results. Section IV concludes the paper.

II. Rareness Reduction

In this section, we perform theoretical analysis as well as exploration of rareness reduction techniques. Specifically, this section is organized as follows. First, we define few terms that are used in the rest of the paper. Next, we introduce metrics to compare rareness between designs. Then, we perform a theoretical analysis of the root causes of the rare signals in hardware designs. We also explore several rareness reduction techniques. Finally, we discuss the effect of rareness reduction on two state-of-the-art Trojan detection techniques.

A. Definitions

We define three terms that are used in the rest of the paper.

Definition 1: *Rareness of a Signal* (S_ω)
Every signal has two possible values: high ('1') and low" ('0'). We define the rareness of a signal S as the minimum of the two probabilities as shown below. For example, if signal S^i is '0' 10% of the time ('1' for 90% of the time) during simulation, S^i_ω is 0.1.

$$S_\omega = \min(P(S \leftarrow 0), P(S \leftarrow 1)) \tag{1}$$

Definition 2: *Logic Probability Vector* ($P(\bar{S})$)
In order to represent the probabilities of a signal S having a value "Low" ("0") and "High" ("1"), we use the following vector and matrix representations.

$$P(\bar{S}) =< P(0), P(1) >= \begin{bmatrix} P(0) & 0 \\ 0 & P(1) \end{bmatrix} \tag{2}$$

Definition 3: *Ideal Transfer Matrix*
The Ideal Transfer Matrix (ITM) is used for the reliability evaluation of logic circuits [10]. In ITM, we express the truth table of a logic gate in matrix representation where rows represent the inputs combinations of the gate while two columns represent the output signal being the value of 0 and 1. ITM representation of primary logic gate types (AND,OR,NOT) is shown in Equation 3. ITM representation for other gates can be computed in a similar way.

$$ITM_{AND} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} ITM_{OR} = \begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} ITM_{NOT} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{3}$$

B. Metrics

We define the following three metrics to measure the rareness of the hardware designs.

Definition 4: *Rarest Rareness* in a Design (Ω)
In the following equation, Ω represents the rareness of the *rarest* signal in the design. Consider an example design that has only two rare signals: S^i and S^j where S^i_ω= 0.1 and S^j_ω= 0.05. Then Ω = 0.05 (smallest between S^i_ω and S^j_ω).

$$\Omega = \min(\{S^0_\omega, .., S^n_\omega\}) \tag{4}$$

Definition 5: *Average Rareness* ($\mu(\omega_n)$)
We define average rareness for most rare n signals as below. Clearly, *higher average rareness implies that the design is more resistant against malicious implants*. In other words, higher average rareness implies easier Trojan detection.

$$\mu(\omega_n) = \frac{\sum_{i=0}^{n} S^i_\omega}{n} \tag{5}$$

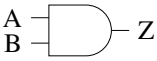
Definition 6: *Signal Count* less than a threshold ($\rho_{(<\tau)}$)
We count the number of rare signals with rareness less than a specific threshold τ in a design D as follows. Clearly, *lower $\rho_{(<\tau)}$ implies that the design is more resistant against malicious implants*. In other words, lower $\rho_{(<\tau)}$ indirectly implies easier Trojan detection for the given rareness threshold.

$$\rho_{(<\tau)} = |\forall S^i_\omega \in D : S^i_\omega \leq \tau| \tag{6}$$

C. Theoretical Analysis

Rareness of the signals in a hardware design depends on the type and the order of logic gates involved in the propagation path of the considered signal. In this section, we first show how to compute the logic probability vector of a signal. Next, we analyze the effects of various parameters on rareness, including the types of logic gates in a specific path, logic depth (number of logic gates in a path), as well as design area (total number of logic gates).

*1) **Calculating the Logic Probability Vector of a Signal**:*
We formulate the rareness calculation of a signal as a matrix multiplication problem. Figure 3 illustrates the example calculation for a fan-out of an AND gate. For this example we have to use the ITM corresponding to AND gate from Equation 3. Then we obtain Equation 7 by multiplying the Kronecker product of input probabilities ($P(A)$ and $P(B)$) of the gate with the ITM matrix of the AND gate. The column sum of the resultant matrix represents the P(0) and P(1) values of the fan-out (Z) signal.



$$P(A) = P(B) = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$$

$$X = P(A) \otimes P(B) \times ITM_{AND} \tag{7}$$

$$\begin{bmatrix} a_1 & b_1 \\ a_2 & b_2 \\ a_3 & b_3 \\ a_4 & b_4 \end{bmatrix} = \begin{bmatrix} 0.25 & 0 & 0 & 0 \\ 0 & 0.25 & 0 & 0 \\ 0 & 0 & 0.25 & 0 \\ 0 & 0 & 0 & 0.25 \end{bmatrix} \times \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$P(\bar{Z}) =< \sum_{i=1}^{4} a_i, \sum_{i=1}^{4} b_i >=< 0.75, 0.25 > \tag{8}$$

Fig. 3: Calculating the rareness probability for a signal (Z) when fan-in signals (A, B) propagate through an AND gate.

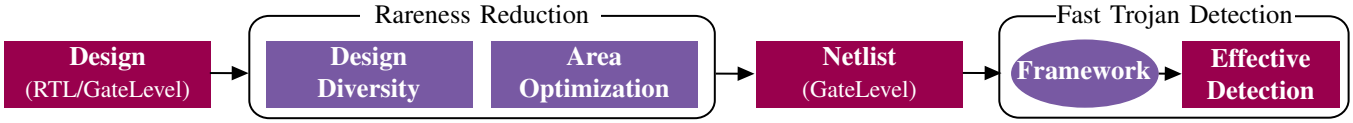
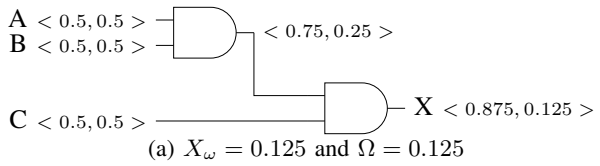


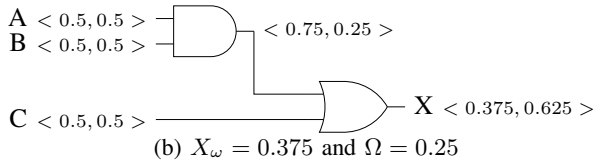
Fig. 2: Overview of proposed rareness reduction based design-for-trust improvement.

*2) **Effect of Logic Gate Types in the Path on Rareness**:* The types of logic gates involved in a circuit is a signature of the design. Since the ITM matrix (discussed above) is calculated based on the truth table of the logic gate, the rareness of a signal is affected by the type of logic involved in the circuit. In other words, P(1) is a lower value for the output of an AND gate, while an OR gate makes P(0) a lower value. Therefore, it is possible to obtain designs with lower rareness metric using different logic implementations. We will explore design diversity in Section II-D to verify this hypothesis.

*3) **Effect of Logic Depth on Rareness**:* Since the logic probability of a signal is always less than 1 ($P(\overline{signal}) \leq 1$), signal propagation through the same gate type will always reduce the probability. However, this phenomenon may not hold when gate types are interchanged in the propagation path. Figure 4 illustrates a counter-example to demonstrate this scenario. Figure 4a shows a circuit with a logic depth of two, with two AND gates. Figure 4b consists of a similar circuit except the last AND gate is replaced by an OR gate. In 4a, the fan-out signal is the rarest (Ω) signal, although in 4b the rarest (Ω) signal is not the fan-out signal. This demonstrates that the effect of logic depth on the rareness values depends on the design. Therefore, we will explore rareness reduction techniques in Section II-D.



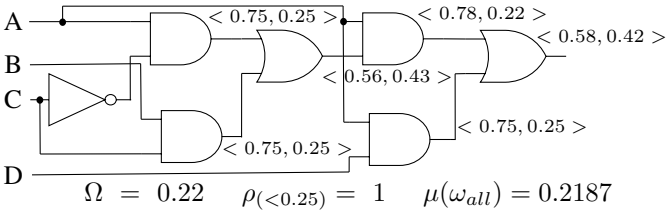
(a) $X_\omega = 0.125$ and $\Omega = 0.125$



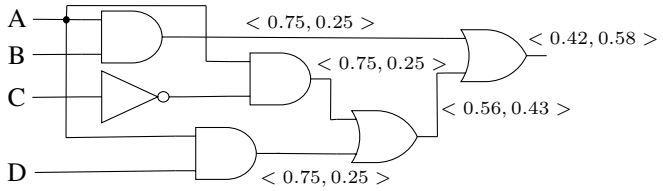
(b) $X_\omega = 0.375$ and $\Omega = 0.25$

Fig. 4: An example to illustrate the effect of gate type in rareness propagation through logic depth.

*4) **Effect of Area Optimization on Rareness**:* Logic optimization refers to reducing a complex logical equation to a simplified version without changing the indented behavior of the circuit. There are various logic optimization techniques for Boolean circuits such as Boolean algebra, graphical methods (e.g., Karnaugh maps, Quine–McCluskey algorithm, and Petrick's method), heuristic methods (e.g., Espresso heuristic logic minimizer), etc. During these optimizations, either logic gates get removed by gate sharing or a part of the circuit may get replaced with a simpler circuit. For example, Karnaugh map tries to identify repetitive patterns in the signals and eliminates them. Based on the intuition provided by logic optimization, we analyzed the relationship of rareness metrics ($\mu(\omega_n)$, $\rho_{(<r)}$) with logic area optimization. Results revealed that if the area reduction is occurred within the region that contributed towards the rareness metrics, then area optimization improves the rareness metrics. We have performed empirical analysis on real-world hardware designs with different synthesis area efforts, as demonstrated in Section III-C.



(a) X=(CB+AC̄)A+DA logic circuit before optimization



(b) X=AB+AC̄+DA logic circuit after area optimization

Fig. 5: An example scenario of rareness reduction of logic circuits through area optimization

Figure 5 presents an illustrative example to demonstrate the effect of area reduction on rareness metrics. Figure 5a shows the logic circuit representing the Boolean equation *X=(CB+AC̄)A+DA*. Corresponding $\Omega, \rho_{(<0.25)}$, and $\mu(\omega_{all})$ metric values are 0.2187, 1, and 0.3033 respectively. Then using Karnaugh maps we optimized the circuit *(CB+AC̄)A+DA* to equivalent Boolean circuit *(AB+AC̄)+DA*, which is represented in Figure 5b. From the metric values calculated for the optimized circuit, it can be observed that the rarest signal (Ω) has increased from 0.2187 to 0.25. The number of signals less than the threshold of 0.25 has reduced from 1 to 0 and the average rareness of all the signals has increased from 0.3033 to 0.3218 demonstrating that area reduction has positively impacted rareness reduction. Table I presents several illustrative examples for different logic circuit expressions.

TABLE I: Rareness metrics for different Boolean circuit expressions before and after area optimizations

Original Circuit				**Area Optimized Circuit**			
Logic	Ω	ρ	μ	*Logic*	Ω	ρ	μ
AB+BC(B+C)	0.18	1	0.26	*(A+C)B*	0.25	0	0.31
AC+AB̄C+ABC	0.12	2	0.25	*AB̄+AC*	0.25	0	0.31
ADC+ABD	0.12	2	0.19	*A(DC+BD)*	0.21	0	0.28

D. Rareness Reduction Techniques

Based on the theoretical analysis in Section II-C, we can conclude that two factors affect the rareness metrics in a hardware design. (i) nature of the design and (ii) area of the design. We propose two techniques to reduce the rareness of signals in hardware designs considering the above factors.

*1) **Design Diversity**:* In order to achieve a functionality, there can be multiple algorithms. There are multiple readily available implementations for most generic sub-components, such as adders, multipliers, dividers, sorting algorithms, search algorithms, hashing algorithms, etc. We explore different implementations for sub-components of the design. For example, if we need an adder, we can consider various adder choices (e.g., ripple-carry adder, carry lookahead adder, etc.) to select the implementation with the minimum contribution to the rareness. Similarly, if we need to implement sorting, we can consider diverse sorting algorithms, including bubble sort, insertion sort, quick sort, merge sort, etc. while we are trying to improve rareness, we also have to satisfy other design constraints, such as area, power, and performance.

*2) **Area Optimization**:* Our theoretical analysis revealed that area reduction leads to improved rareness. Therefore, a design-for-trust solution needs to select the implementation with the lowest area without violating other design constraints. Another way to reduce design area is by reducing the parallelism inside the design. Any hardware synthesis tool considers various avenues for area reduction including parallelism reduction (sharing components), simplified (bare-bone) implementation, and logic minimization. For example, let us consider a processor consisting of two ALU units. If we use a single ALU, it is expected to reduce the $\rho_{(<0.1)}$ value contribution of ALU's by $\frac{1}{2}$. Similarly, the bare-bone implementation of the required functionality is preferable for obtaining a verification-friendly design-for-trust solution. The logic minimization techniques are expected to reduce the area and improve the rareness.

E. Fast Detection of Trojans with Rareness Reduction

Existing Trojan detection techniques ([5]–[7], [11]–[13]) follows the threat model outlined in Section I-A. In other words, the Trojan detection time depends on the number of rare signals in the design. To evaluate the effects of rareness reduction on Trojan detection, we consider two complimentary test generation based Trojan detection techniques: statistical [5] and maximal clique activation [7].

*1) **Trojan Detection using Statistical Test Generation**:* Statistical test generation technique MERO [5] depends on N-detect [14] principle, where each rare signal is activated N times. First, it simulates the design with random test vectors while performing rareness calculations. Next, it identifies all the rare signals (potential trigger conditions) with rareness values less than a specific threshold (τ). Then using the initial set of random test vectors, the algorithm performs bit flips until the N criterion is satisfied for all the identified rare signals. Due to the statistical nature of the generated test set, if N is sufficiently large, a good Trojan coverage can be obtained. The authors demonstrate the results of the MERO framework on ISCAS'85 benchmarks. To achieve a good coverage of detecting Trojan triggers consisting of four triggers, the authors have used a N value of 1000. Rareness reduction is effective for statistical-based test generation in two ways. (1) It reduces the number of rare signals in the design. Assume that the number of rare signals that we can reduce is X. This reduces the initial rareness calculation time by reducing the signal value monitoring effort by X. This further reduces the test generation in the order of $X \times N$, (ii) Reducing the average rareness of the design improves the chances of signals getting activated during random simulations as well as during the execution of the underlying bit-flipping algorithm, yielding higher Trojan coverage from the generated test vectors. Section III-D demonstrates the effect of rareness reduction on statistical test generation.

*2) **Trojan detection using Maximal Clique Activation**:* Directed test generation technique of TARMAC [7] tackles the problem following a complementary approach to MERO. Similar to MERO, TARMAC first calculates the rare signals in the system with random test vectors. Let us assume that we have identified R number of rare signals. For all the rare signals, TARMAC creates a two-trigger connectivity graph by querying all pairs ($R \times \frac{R-1}{2}$) of rare signals using satisfiability solving. The complexity of the satisfiability graph construction is in the order of R^2. Next, maximal clique partitioning is employed on the satisfiability graph to identify the trigger cliques. SAT solver is used to generate test vectors to activate all the identified cliques in the design. Rareness reduction benefits TARMAC in two ways. (1) Suppose the number of rare signals that we can reduce is X. Then satisfiability graph construction complexity is reduced in the order of $(R-X)^2$. (2) Due to the reduction of average rareness, it is easier for the SAT solvers to activate the cliques. This significantly reduces the three major limitations of TARMAC, satisfiability graph construction, clique partitioning, and test generation using clique activation. Section III-D demonstrates the effect of rareness reduction on maximal clique activation.

III. Experiments

We have created several experimental scenarios to strengthen the analysis of rareness reduction techniques in Section II. First, we explain the experimental setup. Next, we conduct rareness reduction experiments using design diversity as well as area optimization. Finally, we evaluate the effects of rareness reduction on detecting randomly inserted Trojans.

A. Experimental Setup

All the experiments including the execution of state-of-the-art test generation methods were carried out on a server with Intel(R) Xeon(R) CPU E5-2640 v3 @2.60GHz processor and 64GiB Memory. For rareness and coverage analysis simulations, we have used *Synopsys VCS* simulator. For compiling the RTL designs to the gate-level netlist, *Synopsys DC Compiler* is used with *SAED90nm* CMOS technology. In order to calculate the rareness of the synthesized designs, we have obtained the

VCD dump of the synthesized designs. For validating the sampled Trojan triggers, *Synopsys TetraMax* was used. An overview of the experimental setup used for the evaluation is presented in Figure 6.

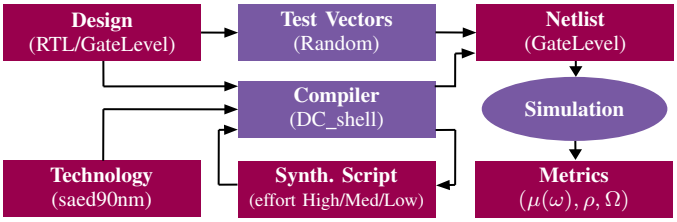


Fig. 6: Overview of our evaluation framework.

B. Design Diversity Experiment

For this experiment, we have selected 64-bit adder circuits of CarryRipple (CRA), CarrySkip (CSA), CarryLookAhead (CLA), CarrySelect (CSeA), Hybrid (HA) and Kogge-Stone (KSA). These circuits were synthesized in two area effort levels of high and low. Next, we simulated the synthesized circuits individually with 10,000 randomly generated test patterns. Then using the VCD dump, we calculated the average rareness of the 100 most rare signals in each circuit. Figure 7 shows the results of the experiment.

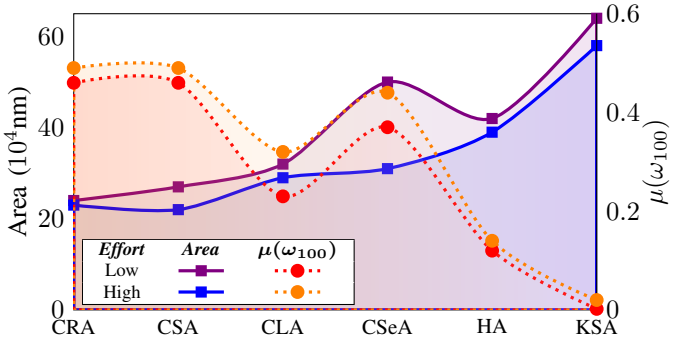


Fig. 7: Design diversity comparison for diverse 64-bit adder implementations synthesized with low and high area efforts versus average rareness for 100 most rare signals ($\mu(\omega_{100})$)

It can be observed that the average rareness for different algorithms that implement the same functionality are different. We analyzed the reason behind the drastically low $\mu(\omega_{100})$ values of the CarryLookAhead adder. In the synthesized design, it was observed that there are paths consisting of only NAND gates. This phenomenon negatively contributes to the rareness significantly, due to the involvement of the same type of gates in the propagation path of the signal, as discussed in Section II-C3. Therefore, it is important to consider diversity of algorithms and their rareness metrics to build a design-for-trust solution. Although the average rareness varies across the algorithms due to design diversity, the relationship between the area and the rareness metrics still holds for each algorithm. This can be observed by observing the rareness metrics at different area synthesis efforts. The designs with the lowest area have the highest average rareness, while the designs with the higher area have the rarest signals. This means that the signals in the area-optimized design are less rare, making it easier for Trojan detection.

C. Area Optimization versus Rareness Correlation

In order to empirically prove the hypothesis that we have outlined in Section II, we have created a correlation analysis experiment. For this we have selected diverse designs covering network-on-chip (NoC) routers, processors (Attiny), crypto cores (AES and ECDSA), error correcting (ECC) memory cores from *OpenCores* [15]. Figure 8 presents the correlation heat-map for the design physical features against the average rareness ($\mu(\omega)$) and number of rare nodes below the rareness threshold of 0.1 ($\rho_{(<0.1)}$). The results were obtained using the experimental setup illustrated in Figure 6. First, we have synthesized the designs with three different area effort levels of low, medium and high. Then we have simulated the synthesized designs with 10,000 test patterns to calculate the rareness metrics. Finally, we have calculated the correlation coefficient value for each design parameter against the signal rareness metrics. It can be observed that the design area is positively (■) correlated with $\rho_{(<0.1)}$ ($A \propto \rho_{(<0.1)}$) while design area is negatively (■) correlated with $\mu(\omega)$ ($A \propto \frac{1}{\mu(\omega)}$). This confirms that fact that the theoretical properties holds true on real-world designs. Further, it can be observed that the correlation between the rareness metrics against the logic levels varies depending on the design. This reflects the effect of gate type involved in the design for the signal rareness.

TABLE II: Percentage comparison of area reduction (A↓) , effect on rareness metrics ($\rho_{(<0.1)}$ ↓, $\Delta\mu(\omega_{all})$ ↑) and test generation time reduction for different hardware designs. The complexity of the designs in terms of number of logic gates is as follows: ECC memory (100K), Attiny processor (30K), NoC router (10K), AES (80K), and ECDSA (300K).

Design	A↓%	ρ ↓%	$\Delta\mu$ ↑	Test Generation Time ↓	
				MERO [5]	TARMAC [7]
ECC mem	10.1%	5.8%	0.007	8.9%	23%
Attiny	4.8%	3.4%	0.012	7.2%	19.8%
NoC router	7.3%	6.1%	0.010	10.3%	24.1%
AES	5.2%	11.8%	0.009	5.8%	17.9%
ECDSA	12.1%	9.7%	0.018	13.6%	28.4%

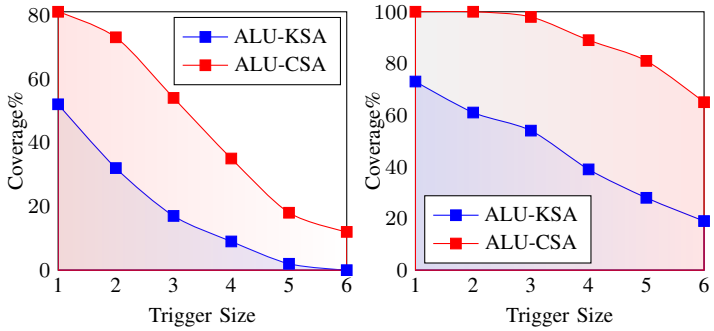
Table II presents the percentage reduction of area between lowest and highest area effort setting with the decrement of ρ and increment of ω for different benchmarks in this experiment.

D. Effectiveness of Rareness Reduction on Trojan Detection

For this experiment, we have used the MERO [5] and TARMAC [7] test generation-based Trojan detection algorithms. First, we have generated test vectors (τ= 0.2) using both methods on the design before and after rareness reduction. The test generation time reduction column of Table II illustrates the time saved during the test generation process by each method. For MERO, we have used N as 1000. Then we randomly insert Trojans into the design following the method outlined in [16] to evaluate the coverage improvement. Specifically, we compute the *Trojan coverage* as the ratio between the number of Trojans detected by the test vectors and the total number of inserted Trojans.

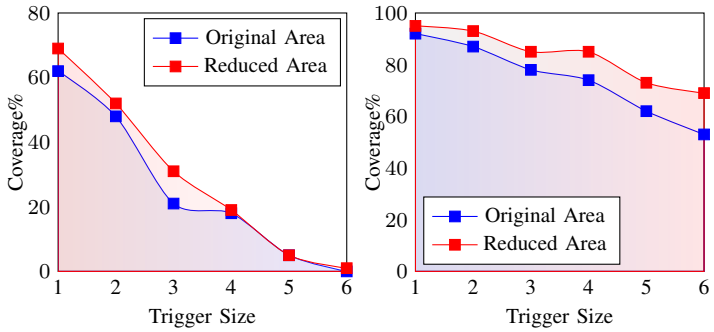
	Metrics	Logic Levels	Leaf Cells	Comb.Cells	Comb.Area	Net Area	Cell Area	Design Area	Total Nets
ECC	$\mu(\omega_{all})$	0.44	-0.68	-0.68	-0.85	-0.83	-0.85	-0.90	-0.64
memory	$\rho_{(<0.1)}$	-0.74	0.90	0.90	0.98	0.57	0.98	1.00	0.88
Attiny	$\mu(\omega_{all})$	0.80	-0.44	-0.56	-0.69	-0.31	-0.88	-0.87	-0.88
Core	$\rho_{(<0.1)}$	-0.29	0.79	0.78	0.82	0.82	0.91	0.92	0.80
ProNoC	$\mu(\omega_{all})$	0.00	-0.24	-0.85	-0.73	0.56	-0.61	-0.98	-0.85
router	$\rho_{(<0.1)}$	-0.19	0.09	0.92	0.69	-0.81	0.73	0.88	0.77
AES	$\mu(\omega_{all})$	0.57	-0.17	-0.97	-0.99	-0.17	-0.80	-0.83	-0.85
Core	$\rho_{(<0.1)}$	-0.27	-0.52	0.95	0.84	0.54	0.98	0.88	0.87
ECDSA	$\mu(\omega_{all})$	-0.37	0.42	-0.74	-0.99	-0.69	-0.88	-0.90	-0.98
Sign	$\rho_{(<0.1)}$	0.29	0.24	0.68	0.92	0.81	0.79	0.90	0.92

Fig. 8: Correlation analysis heat-map generated by analyzing synthesized design features vs rareness metrics ($\mu(\omega_{all})$,$\rho_{(<0.1)}$)



(a) MERO [5] (N=1000) (b) TARMAC [7]

Fig. 9: Design diversity coverage improvement for ALU-CSA and ALU-KSA designs with MERO and TARMAC.(τ = 0.2)



(a) MERO [5] (N=1000) (b) TARMAC [7]

Fig. 10: Area optimization coverage improvement for ECDSA module with MERO and TARMAC.(τ = 0.2)

In order to evaluate the effectiveness of rareness reduction due to design diversity, we have used two 64-bit ALU implementations. For the first ALU, we have inserted an instance of CarrySkip adder (ALU-CSA) and for the second ALU, we replaced it with a Kogge-Stone adder (ALU-KSA). Figure 9 illustrates the coverage results of the design diversity experiment. It can be observed that although the functionality of the two ALU's still the same, different implementations yields drastically different Trojan coverage results. In this experiment, the ALU-CSA implementation is more friendly toward security verification.

In order to demonstrate the effectiveness of area optimization based rareness reduction, we have selected the ECDSA core as the evaluation benchmark. Figure 10 demonstrates the coverage improvement results on the ECDSA benchmark before and after area optimization. It can be observed that coverage has been improved in both MERO and TARMAC methods on the most area optimized design.

IV. Conclusion

Design-for-trust is an important objective to develop secure and trustworthy systems. While obfuscation is a promising avenue, it can lead to unacceptable hardware overhead. In this paper, we explored the effectiveness of rareness reduction to design trustworthy systems. We performed a theoretical analysis of the root causes of rare signals that are likely to be exploited by adversaries to construct stealthy triggers in hardware Trojans. We also explored two techniques for rareness reduction, including design diversity, and area optimization. We performed empirical evaluation using real-world hardware benchmarks to demonstrate the validity of the theoretical analysis. We also conducted experiments to evaluate the effectiveness of rareness reduction for Trojan detection using statistical test generation as well as maximal clique activation. Experimental results demonstrated that our proposed rareness reduction techniques improved the Trojan detection efficiency in terms of reduction in test generation time as well as improved Trojan coverage.

References

[1] Polian *et al.*, "Trojans in early design steps—an emerging threat," 2016.
[2] M. S. Samimi *et al.*, "Hardware enlightening: No where to hide your hardware trojans!" in *IOLTS*, 2016, pp. 251–256.
[3] S. Dupuis *et al.*, "A novel hardware logic encryption technique for thwarting illegal overproduction and hardware trojans," in *IOLTS*, 2014.
[4] A. Marcelli *et al.*, "An evolutionary approach to hardware encryption and trojan-horse mitigation," in *DATE, 2017*, 2017, pp. 1593–1598.
[5] R. Chakraborty *et al.*, "MERO: A statistical approach for hardware trojan detection," in *CHES*, 2009, pp. 396–410.
[6] Y. Huang *et al.*, "MERS: Statistical test generation for side-channel analysis based trojan detection," in *CCS*, 2016, p. 130–141.
[7] Y. Lyu and P. Mishra, "Scalable activation of rare triggers in hardware trojans by repeated maximal clique sampling," *TCAD, 40(7)*, 2021.
[8] A. P. Fournaris *et al.*, "An fpga hardware trojan detection approach based on multiple parameter analysis," in *DSD*, 2018, pp. 516–522.
[9] S. Bhasin and F. Regazzoni, "A survey on hardware trojan detection techniques," in *ISCAS*, 2015, pp. 2021–2024.
[10] D. T. Franco *et al.*, "Signal probability for reliability evaluation of logic circuits," *MR*, vol. 48, no. 8-9, pp. 1586–1591, 2008.
[11] Z. Pan *et al.*, "Automated test generation for hardware trojan detection using reinforcement learning," in *ASP-DAC*, 2021, pp. 408–413.
[12] J. Cruz *et al.*, "Hardware trojan detection using atpg and model checking," in *VLSID*, 2018, pp. 91–96.
[13] A. Jayasena *et al.*, "Scalable Detection of Hardware Trojans using ATPG-based Activation of Rare Events," *IEEE TCAD*, 2023.
[14] I. Pomeranz *et al.*, "A measure of quality for n-detection test sets," *TOC*, vol. 53, no. 11, pp. 1497–1503, 2004.
[15] "Open source ips." [Online]. Available: https://opencores.org/projects
[16] J. Cruz *et al.*, "An automated configurable trojan insertion framework for dynamic trust benchmarks," in *DATE*, 2018, pp. 1598–1603.