




# Rethinking IC Layout Vulnerability: Simulation-Based Hardware Trojan Threat Assessment with High Fidelity

Xinming Wei   
School of Computer Science  
Peking University  
weixinming@pku.edu.cn

Jiayi Zhang   
School of Computer Science  
Peking University  
zhangjiayi@pku.edu.cn

Guojie Luo   
School of Computer Science  
Peking University  
gluo@pku.edu.cn

**Abstract**—Due to the escalating complexity of chip design and the exorbitant cost of building cutting-edge manufacturing facilities, outsourcing the fabrication of Integrated Circuits (ICs) is prevalent in modern semiconductor industry. However, significant security risks may arise because untrustworthy foundries can conduct insidious attacks without close supervision. Since prior works show the feasibility of implementing practical foundry-level Trojan attacks that circumvent post-fabrication detection, IC designers should protect their IC layouts before sending them to a third-party foundry, and such protections are known as design-time defenses. To this end, security metrics for layout vulnerability assessment are crucial to test the effectiveness of the proposed defenses. However, existing metrics are geometric-only and Trojan-oblivious, failing to capture the fundamental aspects of foundry-level Trojan insertion and the associated side effects.

To bridge the gap between real attacks and threat prediction, we present SiliconCritic, a simulation-based, extensible framework that leverages design-time techniques to simulate the blackbox foundry-level Trojan attacks and post-fabrication analysis. SiliconCritic encodes the difficulty of inserting a specific Trojan into a finalized physical layout by measuring the variation of side-channel parameters (timing, power) after the simulated Trojan insertion, where larger deviations denote better detectability and thus enhanced security. SiliconCritic allows IC designers to interactively refine defensive strategies against the objective Trojan based on the feedback of side-channel analysis. Through evaluations on real-world ASIC designs and reported hardware Trojans, SiliconCritic demonstrates the limitations of existing layout-level defenses and highlights the influence of Trojan properties on defensive efficacy. Our work refreshes the understanding of Trojan prevention and suggests future directions for defenses against untrustworthy foundries.

## 1. Introduction

The scaling trend of modern Integrated Circuits (ICs) has promised higher miniaturization, better performance, and lower power. However, the dramatic increase in chip complexity drives the cost of IC manufacturing sky-high, which is prohibitive for most IC designers. TSMC, the

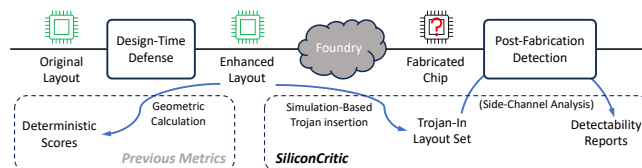


Figure 1. **Landscape of Layout-Level Defenses/Metrics.** SiliconCritic simulates the blackbox foundry attack with design-time Trojan insertion and analyzes the sensitivity of Trojan-in layouts to side-channel detection, while previous geometric-only metrics are agnostic to actual Trojans.

world’s largest foundry, has invested a staggering \$19.6B in their latest  $3nm$  fabrication plants, with an estimated cost of \$33.9B for the next-gen  $2nm$  node [1], [2]. Consequently, the IC industry relies heavily on outsourced chip fabrication with fully in-house supply chains being rare. Outsourcing IC fabrication exposes the chips to serious security risks. After finishing in-house design procedures, IC vendors send the finalized layout files to a third-party manufacturer (known as tapeout), losing control over their physical designs until the fabricated circuit dies are packaged and tested (Fig. 2). Therefore, untrustworthy foundries gain enhanced potential of *fabrication-time hardware Trojan* attacks. Prior studies demonstrate several ways a fabrication-time attacker can insert Trojans into an otherwise trusted IC [3]–[7].

Hardware Trojans injected in foundries are *stealthy* and *permanent*. The Trojan instances have varying forms, sizes, or mechanisms, triggered only by rare events [8]–[10]. Such stealthiness enables them to evade conventional post-manufacturing testing. Moreover, unlike software, IC products are unpatchable and cannot be restored even if malicious modifications are detected [11]. Accordingly, unique methodologies are necessary for Trojan prevention.

Prior research focuses on two main approaches to protect ICs against fabrication-time Trojans: *post-fabrication detection* and *design-time defense* (Fig. 1). Detection-based defenses are more commonly studied and applied [12] with two broad classes: 1) logic testing for Trojan activation, and 2) side-channel analysis. Logic testing [13]–[15] approaches are prone to complex Trojans that are hard to activate by test vectors. Alternatively, side-channel analysis [16]–[22] observes the change of physical attributes (power, delay, etc.)

after fabrication. For instance, adding unintended Trojan logic will consume extra power and thus increase the supply current from some power pins. Side-channel analysis has the advantage of detecting the presence of additional circuitry, even if the Trojan module does not cause any observable circuit malfunction during logic testing [20]. Unfortunately, post-fabrication approaches are deficient since small and stealthy Trojans can still evade the detection [4].

Recent design-time defenses [23]–[26] manage to address the insufficiency of post-fabrication detection schemes. They focus on layout-level protections, *i.e.*, enhancing the physical layout to frustrate subsequent Trojan insertion. Such defenses are designed based on the *observation*: fabrication-time Trojans require free spaces on IC layout to append additional logic gates and wires. In general, existing defenses increase the utilization of placement and routing (P&R) resources either of the whole design or around security-critical design components. Ideally, the defense exhausts available layout resources such that the attacker cannot integrate the Trojan with the victim design. In most cases, the introduced obstacles in the IC layout can increase the cost for attackers attempting to insert Trojans. Time, in particular, is a significant cost due to the limited turnaround time for IC fabrication [27].

Several sets of *security metrics* [27]–[30] have been proposed to measure the susceptibility of IC layouts to fabrication-time Trojans and quantify the coverage of design-time defenses. In fact, these metrics are homogeneous, based on the same observation as design-time defenses. While defined and computed individually, they estimate the amount of free placement sites or routing channels in specific regions on an IC layout, regardless of the properties of specific Trojans to be injected. The quantization results are deterministic, calculated by the geometric information on the physical layout (Fig. 1). Accordingly, we conceive such metrics as *geometric-only* and *Trojan-oblivious*. While they provide some evaluation of layout resilience, they are essentially first-order approximations, inadequate to capture the fundamental aspects of foundry-level Trojan insertion and the associated side effects crucial for post-Trojan analysis. The details and deficiencies of these metrics will be further investigated in §4.

To overcome the defects intrinsic in existing metrics and bridge the gap between actual Trojan insertion and threat estimation, we design and implement a simulation-based, extensible framework, *SiliconCritic*, which leverages *whitebox* design-time techniques to simulate the *blackbox* foundry-level Trojan attack and post-fabrication analysis (Fig. 1). Specifically, *SiliconCritic* is able to evaluate any finalized physical layout ready for foundry fabrication. It allows the IC designer to configure the specific Trojan that their layout under evaluation is supposed to handle. Existing metrics fail to account for the fact that there is no universal design-time defense capable of addressing all types of Trojans. Therefore, metrics should prioritize the coverage of specific Trojans to prevent the misleading estimations highlighted in §4.2. Given the Trojan circuit design and the Trojan-

free layout<sup>1</sup>, *SiliconCritic* annotates the security-critical signals in the design via either automatic analysis or user specification, and binds them with particular Trojan ports to form a collection of attack schemes. Subsequently, a Trojan insertion flow based on Engineering Change Order (ECO) efficiently transforms the attack schemes into Trojan-in layouts, to mimic the behavior of real fabrication-time attacks. Now we can assess the sensitivity of malicious modified layouts to side-channel analysis<sup>2</sup>, to determine the overall difficulty of inserting a given Trojan into the assessed layout. *SiliconCritic* focuses on timing and power analysis, including meticulous inspection of the variation of path delay and regional power. The diversity of attack schemes and fine-grained side-channel analysis enable *SiliconCritic* to feedback on a series of statistic-based reports and isolate Trojan effects from potential noises.

Using *SiliconCritic*, we evaluate four ASIC designs against four reported Trojans with varied sizes, complexity and mechanisms respectively. The designs include a microcontroller [31], an interconnection IP [32], an AES core [33], and an OR1200 processor [34]. Each design has four variants of layouts (original and enhanced by three representative design-time defenses). For each layout, *SiliconCritic* reports the coverage against the corresponding Trojan with a range of attack outcomes. *SiliconCritic* analysis reveals that no existing layout-level defense is “one-size-fits-all”; the Trojan properties largely influence the efficacy of one defense. Moreover, we find that well-crafted defenses centering security-critical signals prevail that exhausting global P&R resources, with better defensive effects and lower performance or power overheads. Lastly, *SiliconCritic* analysis indicates that current placement-centric defenses cannot address the stealthiest Trojans; future defenses can pay more attention to routing-level protections against Trojan attachment.

This paper makes the following contributions:

- We review the landscape of existing layout-level metrics (§4.1) and analyze the limitations of such geometric-only metrics with an elaborate case study (§4.2).
- We design, implement, and open-source<sup>3</sup> *SiliconCritic*, a simulation-based, Trojan-specific, extensible framework, to compute security metrics for a tapeout-ready layout (§5). It performs netlist-level Trojan/design integration (§5.1), conducts layout-level Trojan insertion to mimic fabrication-time attacks (§5.2), and captures Trojan perturbation with dedicated side-channel analysis (§5.3).

1. In the context of this paper, we use “Trojan-free” and “Trojan-in” to describe an arbitrary layout before/after Trojan insertion. We refer to “original design (layout)” as the layout not processed by design-time defenses, in oppose to “enhanced (strengthened) designs”.

2. The term “side-channel analysis” typically refers to the detection of potential Trojans in post-silicon ICs after fabrication. However, in *SiliconCritic*, we apply this concept to design-time IC layouts. To maintain simplicity, we do not differentiate the usage of this concept in our paper.

3. <https://github.com/xinming-wei/SiliconCritic>

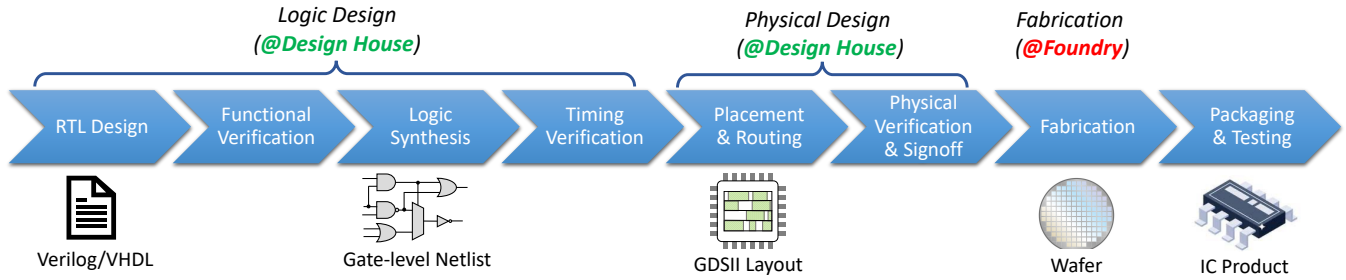


Figure 2. **IC Design Process.** In the scope of this work, the in-house logic design and physical design stages, together with the final packaging and testing, are considered trusted, while the fabrication carried out at the external foundry is untrusted.

- We use SiliconCritic to assess the effectiveness of mainstream design-time defenses over real-world IC layouts and Trojans. We analyze in-depth the discrepancy of defensive coverage between defenses, and between different Trojans one defense face (§ 6).
- We shed light on future untrusted foundry defenses (and equivalently, attacks) instructed by SiliconCritic simulation and interaction with post-fabrication analysis (§ 7).

## 2. Background

### 2.1. IC Design Process

The IC design process is a comprehensive and complex procedure that can be broadly categorized into three major stages: logic design, physical design, and fabrication, as shown in Fig. 2. The logic design stage commences with the description of circuit functionality at the Register-Transfer Level (RTL). This high-level abstraction is then translated into a *gate-level netlist* through a process known as *logic synthesis*. The netlist, a detailed description of the connectivity of the logic gates, bridges the gap between the logical and physical representations of the IC.

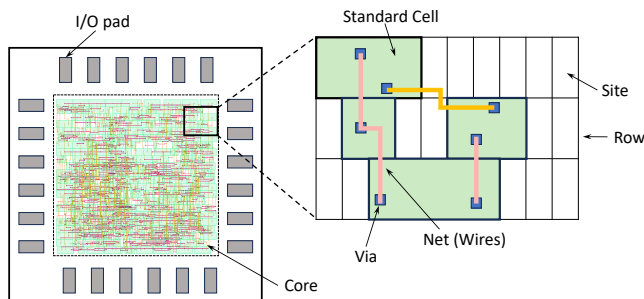


Figure 3. **Floorplan Illustration.** The plot annotates basic floorplan components while omitting macro blocks and power grid mesh for simplicity.

In the subsequent physical design stage, the synthesized netlist is transformed into a physical layout through *placement-and-routing (P&R)*. The first stage of P&R is to create a *floorplan*. Fig. 3 illustrates a simplified floorplan, which defines the size and position of the *core* (i.e., the

area for cell placement), and the location of the *I/O pads* around the periphery of the core. The core is comprised of a number of fixed height *rows*; each row consists of a number of *sites* which can be occupied by the circuit components (e.g., *standard cells*). The routing process then interconnects these standard cells based on the connectivity information provided by the netlist. Routing uses *metal layers* and *vias*, which are conductive paths for signals to travel through different mutually insulated layers, to form the interconnections, also known as *nets*. The output of P&R is the IC layout in Graphics Data System II (GDSII) format, a standard geometric representation of the implemented layout.

Physical verification and signoff ensure that the physical design meets all the foundry’s manufacturing rules and confirms the design’s readiness for fabrication. In signoff, *Engineering Change Order (ECO)* plays a crucial role in the final stages of physical design. Before the chip mask is made, ECO is used to implement minor logic modifications directly on the post P&R layout, without disturbing the overall design. This process allows designers to make incremental improvements or rectify issues that arise during the final verification stages, thereby ensuring the design’s functional correctness and manufacturability. Note that both logic design and physical design stages are conducted under the control of the chip designer, while the GDSII file is then sent to an external foundry for fabrication. The final stage, packaging and testing, ensure the IC’s functionality and reliability before it is shipped for use in electronic devices.

### 2.2. Hardware Trojan

**Trojan Model.** Hardware Trojans, a rising concern in the realm of electronic design automation (EDA) security, represent malicious modifications or inclusions within the hardware. These manipulations are often subtle and sophisticated, making them difficult to detect but potentially devastating in their impact. Each hardware Trojan is composed of two fundamental components: the trigger and the payload. The trigger, often dormant and inconspicuous, is the condition that activates the Trojan. It can be designed to respond to a specific event or a sequence of events, making the Trojan’s activation highly unpredictable and its detection even more challenging. The payload, on the other

hand, is the malicious function that is executed once the Trojan is activated. The damage inflicted by the payload can range from performance degradation to severe functional disruption or even system failure.

**Trojan Taxonomy.** Hardware Trojans can be categorized based on several characteristics as following [8], [35], [36]. 1) Trigger and payload types. The trigger and payload could be digital or analog. Digital Trojans can further be classified into combinational and sequential types. Sequential Trojans target the sequential elements of a circuit, such as flip-flops or registers, and aim to alter the circuit behavior based on specific triggers or conditions. These Trojans often manipulate the control signals or modify the state transitions within the sequential elements. On the other hand, combinational Trojans focus on the combinational logic blocks, such as gates or multiplexers. These Trojans aim to introduce subtle modifications to the logic equations or truth tables of these blocks, leading to unexpected outputs or triggering specific unintended behaviors. 2) Activation mechanism. Some are condition-based, activated by a specific sequence of inputs or a particular state of the system, while others are time-based, triggered after a certain period or at a specific time. 3) Insertion phase. The Trojan can be inserted at any stage of the IC production, including the specification, design, fabrication, testing, and packaging phases. Each phase presents unique challenges for Trojan insertion, detection, and mitigation. In this work, we focus on the fabrication-time Trojan. 4) Malicious effect. Some Trojans aim to change the functionality of the device, others seek to degrade performance or leak sensitive information, and still others aim to cause a denial of service.

Nowadays, advanced Trojans are small, stealthy, and controllable [27], which pose a significant and multifaceted threat to electronic systems, and underscore the importance of robust, effective Trojan prevention. The key to designing Trojan prevention strategies lies in, however, the unified metrics to evaluate how difficult an attacker can inject varied Trojans as mentioned into a protected design.

### 3. Threat Model

Our threat model mostly follows that adopted by prior art on foundry-level Trojan attacks and defenses [4], [27].

**Identity of the Attacker.** The design house and the IC design tools are trustworthy, together with the final packaging and testing parties. The untrusted entity is the IC foundry, as highlighted in Fig. 2. Our focus on fabrication-time is motivated by current IC supply chains, economic forces, and technology reasons that require heavy reliance on outsourced production.

**Capabilities of the Attacker.** The attacker inside the foundry has access to the tapeout-ready GDSII file and can extract the complete netlist information from the GDSII layout with reverse engineering [37], [38]. The adversary manipulates the layout by inserting additional cells or wires; they cannot remove, resize, or shift existing circuit components, which are practically infeasible due to the high risk of compromising functional equivalence, timing constraints,

or design rules. Even if these issues can be resolved with considerable effort, it would result in extended lead time for chip fabrication that exceeds the agreed contract limits. They cannot extend the layers, dimensions, or size of the current design, which can be easily discovered after manufacturing.

## 4. IC Layout Vulnerability Characterization: Metrics and Pitfalls

To measure the difficulty of inserting hardware Trojans into a given IC layout, security metrics that truthfully quantify the layout susceptibility are expected. In this section, we first introduce the layout-level security metrics in previous works. Then we discuss and illustrate these metrics' limited goodness of fit when characterizing layout vulnerabilities, drawing on an elaborate case study.

### 4.1. Existing Coverage Metrics

We term previous layout-level metrics as *geometric-only*. They generally calculate free P&R resources based on intrinsic GDSII geometric attributes. Salmani *et al.* [29] and Hossein-Talaei *et al.* [30] define the *vulnerability of a region* of circuit layout in a similar way, which is computed as the product of normalized whitespace and normalized unused routing tracks within the region. ISPD22 Contest [28] refines the analysis of whitespace. It defines *exploitable region* as where an attacker would be able to place-and-route their Trojans in the layout. An exploitable region satisfies: 1) it consists of spatial contiguous sites, whose sum exceeds a given threshold (for Trojan placement). 2) it is within a certain distance from security-critical instances (for timing closure). The number of sites and free routing tracks across the exploitable regions quantify the vulnerability. ICAS [27] divides Trojan insertion into three sub-tasks: 1) Trojan logic placement. 2) Victim/Trojan integration. 3) Intra-Trojan routing, and measures the difficulty of each sub-task with metrics *trigger space*, *net blockage*, and *route distance* respectively. The first metric reflects free placement resources, and the last two reflect routing.

### 4.2. Case Study: Limitations of Geometric-Only Metrics (Fig. 4)

The geometric-only nature of these metrics determines that, under many circumstances, they are inadequate of representing the authentic vulnerability of IC layout to actual Trojan insertion. We demonstrate their limitations with a toy, yet elaborate case study.

Suppose that an attacker would like to exploit a 4-bit carry look-ahead adder circuit. Specifically, the attacker expects to modify the most significant bit of the adder output, `sum[3]`, by simply inserting an XOR Trojan gate and XOR `sum[3]` with another signal, `add2[0]`. Obviously, this simple Trojan would be triggered if `add2[0]==1b'1`, and the value of `sum[3]` would be flipped.

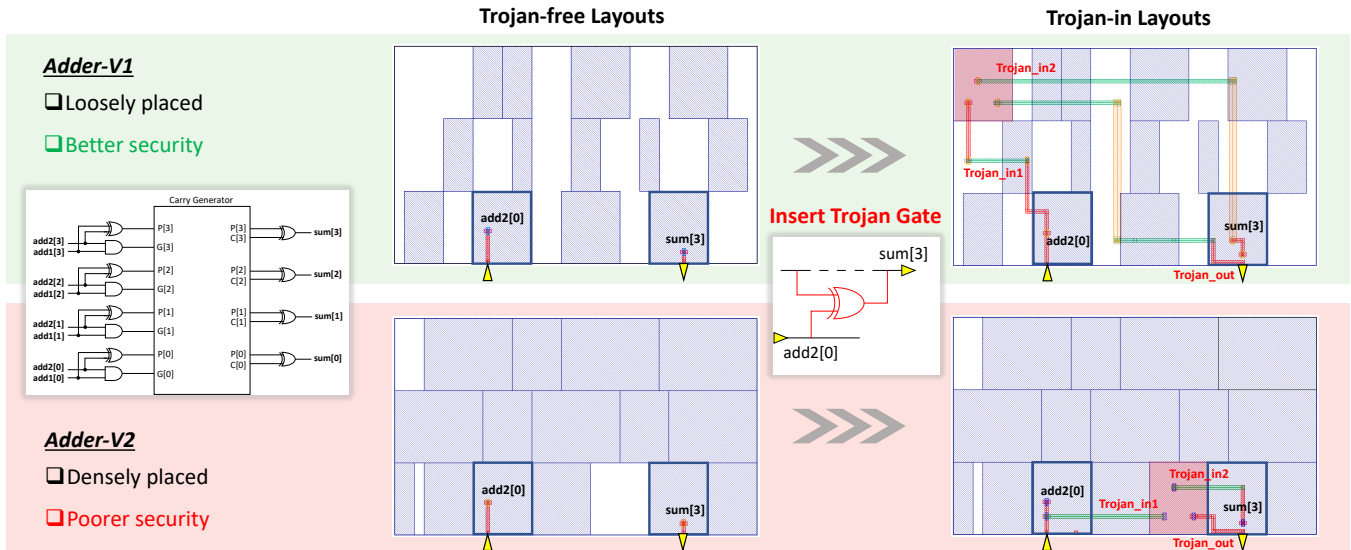


Figure 4. **Case Study of Layout Vulnerability.** We create an elaborate attack scenario where a 4-bit adder is targeted with the insertion of an *XOR* gate. The gate is designed to flip one output bit when activated. The adder schematic consists of two versions of layouts with different core utilization. For each layout, we demonstrate the placement of the Trojan cell in the available space and the routing of Trojan nets to connect victim signals. To emphasize the Trojan and victim logic, we hide unrelated nets and I/O ports of the design. Surprisingly, Adder-V1 proves to be more secure than Adder-V2 in relation to this specific Trojan, as it results in longer Trojan wire lengths, enhancing detectability for delay analysis. However, geometric-only metrics inaccurately characterize Adder-V2 with higher density as being more secure, disregarding the actual Trojan insertion.

Now we have the placed-and-routed layouts of the same adder schematic in two versions, termed as Adder-V1 and Adder-V2, as shown in Fig. 4. The former is loosely placed, while the latter evolves from the former by upsizing some existing gates to achieve a higher density. We simulate the Trojan insertion process with the commercial P&R tool. You can find that the valid position to place the Trojan cell is unique in both layouts of Adder-V1 and Adder-V2. Then the subsequent Trojan routing connects each pin of the Trojan cell with the corresponding victim net.

With the implemented Trojan insertion, we can conveniently assess the actual difficulty of the Trojan attack. The layout vulnerability can be measured by the total length of newly added Trojan routes, as longer wire length results in higher delay. Therefore, layouts with longer Trojan wire length are more detectable in post-fabrication analysis and pose greater challenges for attackers. Based on this, we can conclude that Adder-V2 is more susceptible to the constructed Trojan in this case compared to Adder-V1. This finding challenges the conventional belief that higher placement density offers better Trojan defense. In Tab. 1, we list the computed values of the metrics introduced in §4.1 on Adder-V1 and Adder-V2. For each metric item, we mark the superior result in the comparison in green, and it shows none of these metrics provides the correct prediction. We rethink these metrics and summarize their limitations as below.

**Inadequate Prediction Fidelity.** Why do these metrics fail in the simple case study? Will they behave worse in large, real world cases? The metrics are in essence, first-order estimation of the available P&R resources; these estimations

TABLE 1. COMPARISON OF ALL THE METRICS ON ADDER-V1 AND ADDER-V2. THE ADVANTAGEOUS RESULT FOR EACH METRIC IS MARKED IN GREEN.

Metric Set	Metric Item	Adder-V1	Adder-V2
Real Security	Trojan Net Length (um)	21.12	4.85
ISPD22 [28]	Exploit. Region: Place (# sites)	23	4
	Exploit. Region: Route (# tracks)	66	15
Regional			
Vulnerability	Vulnerability Factor (%)	41.2	6.8
[29], [30]			
	Trigger Space (# sites)	43	8
ICAS [27]	Net Blockage (%)	28.6	32.5
	Route Distance (# stdev)	5.9	7.1

apply unified rules to extract GDSII geometrical features locally or globally over the layout. In this case, Adder-V1 has significantly more free, continuous placement sites than Adder-V2, and thus the first-order based metrics give the rough, incorrect prediction that Adder-V1 is more vulnerable. For large layouts, the inaccuracy of such prediction will even scale due to its geometric-only nature.

**Trojan-Obliviousness.** Existing geometric-only metrics are not configurable for the specific Trojan to be injected. The difficulty of Trojan insertion depends on not only the layout landscape, but the Trojan pattern (e.g., number/size/type of Trojan gates, connections of Trojan routes). Here in this case, if the Trojan gate to be inserted is slightly larger, then Adder-V2 would defeat Adder-V1 since the gate is unplaceable. However, the geometric-only metrics cannot derive security values with respect to Trojan patterns.

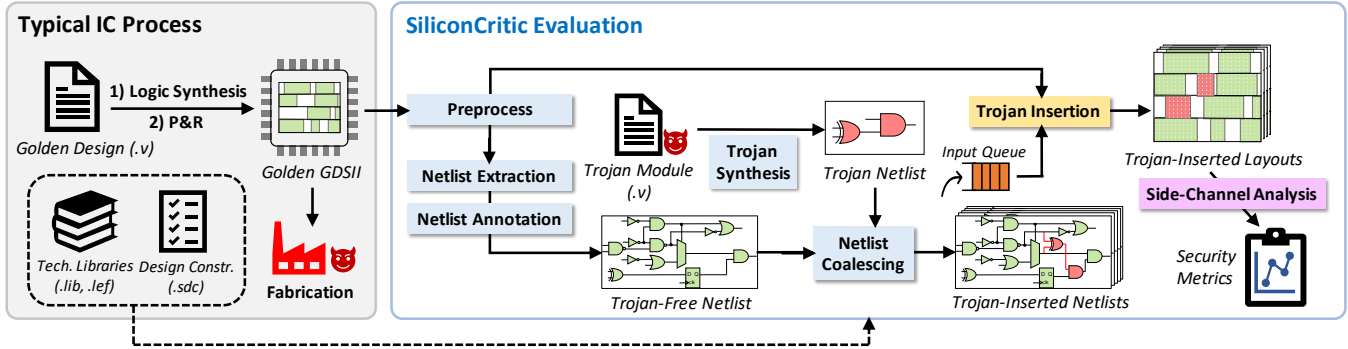


Figure 5. **SiliconCritic Overview.** SiliconCritic simulates the blackbox fabrication-time attack with design-time techniques available to IC designers. The flow begins with a finalized layout and the target Trojan RTL. First, *Netlist Preparation* identifies the security-critical signals in the extracted post-layout netlist and bind them with Trojan ports to form a variety of Trojan-inserted netlists. *Trojan insertion* implements the netlist-level insertion incrementally at layout-level. *Side-Channel Analysis* generates security reports based on the Trojan detectability observed on well-crafted timing/power metrics.

**Lack of Post-Trojan Analysis.** Existing geometric-only metrics include no analysis of side-channel effects (e.g., timing decay, power increase) after Trojan insertion; they merely focus on pre-Trojan assessment, i.e., the conceived difficulty of P&R a Trojan into a given layout. In real-world cases, it is trivial to find enough space to place small, stealthy Trojans into large designs. Instead, it is more practical to measure the side effects brought by Trojan insertion to track Trojan footprints. The geometric-based metrics do not involve the Trojan insertion process and are thus unable to undertake post-Trojan analysis.

## 5. SiliconCritic

Now that we have systematically analyzed and demonstrated the similarities of previous geometric-only metrics and their incapability of capturing layout vulnerabilities against realistic fabrication-time Trojans. To address the gap between actual Trojan insertion and layout susceptibility estimation, we propose and implement SiliconCritic, a Trojan-specific, customizable framework for GDSII vulnerability evaluation. The basic idea of SiliconCritic is to simulate the fabrication-time (post-design) Trojan attacks with design-time techniques. Then we assess the sensitivity of modified layouts to side-channel detection, in order to determine the overall difficulty of Trojan insertion into a given layout.

**Flow Overview.** As shown in Fig. 5, the SiliconCritic vulnerability analysis framework can be divided into three phases: *Netlist Preparation* extracts post-layout netlist from the layout before tapeout, retrieves security-critical signals, and appends synthesized Trojan to the extracted netlist to produce a collection of attack schemes. Based on these schemes and the Trojan-free layout, *Trojan Insertion* generates Trojan-inserted layouts, which then go through *Side-Channel Analysis* to obtain final security metrics.

**Inputs/Outputs.** SiliconCritic takes the following as inputs: 1) finalized layout (.def) of the Trojan-free design, 2) process technology libraries (.lib, .lef) and design constraints (.sdc), 3) Trojan module RTL (.v) and optionally the attack specifications (.json). The physical layout is a Design Ex-

change Format (DEF) file. The process technology libraries contain a Liberty Timing File (LIB), and a Library Exchange Format (LEF) file. LIB is a timing model with cell delays, transitions, etc., while LEF provides physical information of standard cells, layers, vias, pins and design rules, etc. The Synopsys Design Constraints (SDC) file specifies timing, power and area constraints throughout synthesis and P&R. The user also needs to provide the target Trojan RTL to measure layout resilience against. The output is a set of CSV reports as the security metrics.

**Implementation Overview.** We develop a fully automatic toolchain for SiliconCritic simulation and evaluation of modern IC designs. Our toolchain integrates frontend customized Python and Tcl<sup>4</sup> modules with commercial CAD tools as the backend.

### 5.1. Netlist Preparation

Netlist preparation consists of layout preprocessing and a series of netlist operations at the structural level. It complements gate-level information for physical-level layouts and generates potential attack schemes for the subsequent Trojan insertion flow. Initially, the tapeout-ready layout is preprocessed to facilitate netlist extraction. With the post-layout netlist, SiliconCritic identifies and annotates security-critical signals of the design, thus creating a set of aggressive attack combinations. Once the given Trojan RTL module is synthesized, SiliconCritic assigns each of its input/output to the target signals determined in the attack schemes and merges it into the design netlist. These Trojan-inserted netlists are pushed into a queue to the coming ECO flow.

**Layout Preprocessing.** We preprocess the Trojan-free layout to facilitate the insertion of Trojans and maximize the attacker’s capabilities, by developing specific Tcl scripts for P&R Tools. At first, we delete all the *physical-only cells* across the standard-cell rows in a layout, freeing tons of placement resources. These unfunctional physical-only

4. Tcl (Tool Command Language) is the standard programming interface to commercial CAD tools. IC designers develop Tcl scripts to configure and automate CAD processes with various APIs provided by the tools.

cells (e.g., filler cells) are used to maintain design stability. Hence, foundry attackers can remove these cells easily to create open spaces. Moreover, we identify and detach all the *dangling components* (i.e., functional but unconnected cells and wires) from the layout. These redundant circuit components are also trivial to recognize by attackers with reverse engineering extractions. With the GDSII preprocessing, we exaggerate the available P&R resources in the given layout and make more room for Trojan insertions.

**Trojan Synthesis.** To obtain the gate-level netlist of the Trojan module, we develop a customized Trojan synthesis flow by modifying the typical Tcl Scripts for logic synthesis. This flow takes the user-provided Trojan RTL as input and adopts the same technology libraries and constraints as the Trojan-free design. To maximize the stealthiness of the inserted Trojan, we assign the smallest possible standard cell instance to each Trojan gate. Logic synthesis maps the abstract logic gates to particular standard cells in technology libraries to optimize total area and delay. For example, in Nangate 45nm library [39], a 2-input NAND gate can be mapped to NAND2\_X1, NAND2\_X2 and NAND2\_X4, with ascending sizes. A larger gate can reduce path delay, though extra area is spent [40]. In our implementation, each Trojan gate is mapped to the smallest cell unless evident timing overhead is observed. In this way, we minimize both the total area and the delay of the synthesized Trojan.

**Netlist Extraction and Annotation.** A gate-level netlist is extracted from the preprocessed layout, with complete information on design components and connections. In this way, we assume that the attacker gains a full understanding of the functionality of the design effortlessly. Next, we assign input/output signals to Trojan trigger and payload ports. To determine the internal nets for Trojan ports to tap into, it is crucial to recognize and annotate *security-critical* signals from the extracted netlist. Since determining whether a signal is security-critical requires a certain understanding of the design usage, we provide two options for users: *manual configuration* or *autonomous identification*. Manual configuration requires the IC designer to specify all the candidate signal combinations, each of which states a mapping from security-critical signals to the Trojan ports. In this way, the designer can protect certain signals based on the contextual understanding of the design or human expertise.

In contrast, autonomous identification judges the probability of a victim net being attached by malicious Trojans. Prior art reaches the consensus that ideal signals for Trojan selection are *dormant* (i.e., rarely toggle) and *controllable* (i.e., easy to drive) [4], [30]. In our implementation, we select nets with low transition probability on non-critical paths as candidates for the input of Trojan trigger. In practice, we sample signals among the 10 lowest transition rates on non-critical paths with the aid of commercial synthesis tools. For the payload output, we follow the notion of Sandia Controllability/Observability Analysis Program (SCOAP) [41] to select the first 10 nets with the lowest testability scores as candidate payload victims. Additionally, SiliconCritic supports

the semi-autonomous mode that designers can specify some ports for autonomous victim selection, which the other for manual configuration. After acquiring all the  $\{(\text{Trojan port}[1], \text{signal}[1]), \dots, (\text{Trojan port}[n], \text{signal}[n])\}$  combinations, the last step is to eliminate the possible intra-signal data dependencies that can form a combinational loop. We transform the netlist to a directed acyclic graph (DAG) using Python and Pyverilog [42], and then remove the signal set that will cause a cycle with the added Trojan logic. Finally, we derive a collection of viable attacking schemes that promise the stealthiest Trojan injections given the complete information of the layout and Trojan.

**Netlist Coalescing.** Until now, we have at hand the design layout&netlist, a synthesized Trojan netlist, and a collection of attack schemes. The next step is to merge the original design netlist and the Trojan netlist into a set of malicious modified netlists, according to the connection relations in the attack schemes. We represent both design and the Trojan as DAGs, and the former has been completed when checking signal dependencies. Since the two DAGs obviously have no conflicting parts, we can 1) locate security-critical signals (represented as edges) in design DAG, 2) set the starting vertex of each critical edge as the starting vertex of a new edge, whose end vertex is the corresponding Trojan port (with no input edge initially), and 3) restore the resulting DAG to a netlist. In this way, we concatenate two netlists together without altering any other irrelevant logic of the design, which is essential for the subsequent Trojan insertion operation. The output of this flow is a set of Trojan-inserted netlists corresponding to the attack schemes.

## 5.2. Trojan Insertion

To mimic the foundry-level Trojan attack in reality, we leverage the *Engineering Change Order (ECO)* techniques to perform Trojan insertion into given layouts. In the context of a typical EDA process, an ECO flow is used to make further enhancements or fix functional bugs in a finalized layout. Attackers inside the foundry similarly edit the mask to introduce malicious layout modifications. Our design-time ECO simulation can provide strictly better Trojan integration than the actual attack in foundry because in-house simulation can sufficiently optimize the insertion, while the layout information and CAD tools are limited at fabrication-time. Since the objective of ECO P&R is to optimize overall PPA (timing, power and area) as well as basic P&R, ECO techniques promise minimized footprint of inserted Trojans.

We propose our Trojan insertion flow compatible with the whole SiliconCritic toolchain. Some recent works [43], [44] realize Trojan insertion with ECO approaches, but they target several hard-coded Trojans only and provide limited attack versatility. In our implementation, each round of ECO takes 1) the preprocessed layout, 2) a Trojan-inserted netlist, and 3) technology libraries and design constraints as the input, and returns a Trojan-inserted layout. We utilize commercial P&R tools and customized ECO control scripts, also written in Tcl, to drive an ECO flow. Since we have generated a set of attack schemes (Trojan-inserted netlists)

for a single layout, the ECO flow will be repeated for multiple rounds. To accelerate the whole insertion process, we maintain a process pool to enable ECO flows in batch mode. Each ECO insertion round undergoes the following steps: First, we anchor each instance of the original layout in place, in case that certain instances are moved or deleted during ECO optimizations. Then, the ECO operator differentiates the Trojan-inserted netlist with the schematic of the layout, and place-and-route the appended Trojan components with unused placement sites and routing tracks. The best defense outcome is to leave not enough spaces for Trojan placement and abort the ECO P&R. Note that it is intriguing, yet fallacious, to place-and-route the Trojan-inserted netlist from scratch to obtain a “Trojan-inserted” layout, because this layout has orthogonal geometric relationship with the Trojan-free one and fails to represent the additive behavior of fabrication-time Trojans.

### 5.3. Side-Channel Analysis

With a Trojan-free layout and a collection of Trojan-inserted layouts, we can finally analyze the vulnerability of the layout to the specific Trojan. We employ and refine classic side-channel detection methodologies to design SiliconCritic metrics, while targeting design-time measurements instead of post-fabrication. Commercial EDA tools can report various physical attributes such as timing, power, or circuit current, which are common options for side-channel analysis. We collect and observe the overall abnormality of a certain side-channel parameter across the attack schemes, compared with the pre-characterized golden reference of the Trojan-free layout, to derive the statistical variations as our security metrics. SiliconCritic focuses on metrics around two side-channel parameters: timing and power. To increase detection sensitivity and isolate Trojan effects from noise, we design fine-grained analysis on the rise of path delay and regional power. Ideally, the more difficult one layout is for Trojan insertion, the more pronounced discrepancy of side-channel parameters would be observed.

**Timing Analysis.** The reason behind timing-based analysis is that inserted Trojans will add fanout to original logic gates, introduce additional capacitive load, and thus cause a noticeable delay in the existing paths of the design [19], [22]. Longer timing paths can also create an accumulative and observable effect on the total timing of the design. For each Trojan-inserted layout ( $L_{Tj-in}$ ), SiliconCritic measures 1) Total Negative Slacks (TNS)<sup>5</sup>, 2) delay of the top-100K critical paths ( $p_t$ )<sup>6</sup>, and compares them with the respective values of the Trojan-free design ( $L_{Tj-free}$ ). TNS represents the overall timing performance of the design and is straight-

5. Total Negative Slack (TNS) measures the cumulative delay by which the actual arrival time of a signal falls short of its required arrival time in a digital circuit design. It is a key metric for IC designers to identify timing conditions and optimize circuit performance.

6. We query the delays of the top-100K critical (tight-slack) paths with manageable computational overheads, thanks to the efficient data structure (timing DAG) and implicit path representation powered by the commercial EDA tools.

forward to measure, while path delays contain detailed timing information. A timing path of a design can be extracted from an input/register pin to a register/output pin. We define *Maximum Path Delay Rise (MPDR)* of a given layout as the maximum path delay increase after Trojan insertion. MPDR magnifies the Trojan impact on the path delay, and provides higher detection sensitivity than TNS.

$$MPDR(L_{Tj-in}) = \max_{p_t \in \{critical\ paths\}} \frac{delay(p_t; L_{Tj-in})}{delay(p_t; L_{Tj-free})}$$

Since each layout has a TNS and an MPDR value, we can derive the histogram of TNS or MPDR for each design given all the Trojan-inserted layouts. Then we can determine the confidence in identifying the Trojan instance with the deviation of the TNS/MPDR from the Trojan-free reference (e.g., mean/max. change) in a statistical manner.

**Power Analysis.** A malicious Trojan inclusion is bound to have an impact on the power consumption [21], [35]. The Trojan trigger circuit is always actively monitoring the activation condition [4], [16]. Also, all Trojan gates dissipate extra static power due to the leakage current in idle mode, just like other standard cells. For each Trojan-inserted layout, SiliconCritic measures 1) total power, 2) regional power. Specifically, total power is the most elementary power-related metric while prone to noise or small Trojans. Regional power, on the other hand, measures the power supply of smaller functional regions ( $r$ ) within the chip core. Since the Trojan cells tend to be located around security-critical components, the region-based approach with measurements from clustered power ports can help localize the Trojan-infected regions [45]. We perform tile-wise partitions and adjust the tile size based on the core area. We define *Maximum Regional Power Rise (MRPR)* in a similar way as MPDR, to closely inspect the region with the largest growth in power dissipation.

$$MRPR(L_{Tj-in}) = \max_{r \in \{all\ regions\}} \frac{power(r; L_{Tj-in})}{power(r; L_{Tj-free})}$$

Likewise, we can perform power side-channel analysis by measuring the (mean, max.) variation of the power/MRPR histogram from the Trojan-free reference values. Enhanced layouts that hinder Trojans from meeting timing requirements after injection will also demonstrate larger power inflation, because attackers have to conduct additional fixes (e.g., gate upsizing, buffer-repeater insertion) to ensure timing closure and Trojan convergence [46], and such timing fixes would inevitably consume more power.

## 6. Evaluation

We evaluate the effectiveness of representative design-time defenses on four ASIC designs against four reported hardware Trojans, respectively. Corresponding to SiliconCritic’s side-channel analysis, we analyze the timing variations (§6.2) and power variations (§6.3) statistically. We specially inspect the Trojan impact of A2 [4], the stealthiest foundry-level Trojan to date [27], as a standalone case study (§6.4).



TABLE 2. ATTACK⇒DESIGN PAIRS

Trojan	# Std Cells	Trojan Properties	Design	# Std Cells	Trojan Footprint
Key Leak [47]	80	Sequential Digital	risc16f84 [31]	1290	<b>6.2016%</b>
Bus Hijacker [48], [49]	23	Combinational Digital	Conmax [32]	16537	<b>0.1391%</b>
Timebomb [48], [49]	33	Sequential Digital	AES [33]	189112	<b>0.0174%</b>
A2 [4]	2	Combinational Analog&Digital	OR1200 [34]	317296	<b>0.0006%</b>

## 6.1. Experimental Setup

**Designs and Trojans.** To comprehensively evaluate the aforementioned defenses with SiliconCritic, we select four open-source IC designs and four hardware Trojans with varied sizes, complexity and mechanisms; each kind of Trojan solely targets a design. We list the attack⇒design pairs in Tab. 2. Note that *Trojan footprint* refers to the division of # Trojan cells to # design cells, which denotes the Trojan’s stealthiness. In each attack⇒design pair, SiliconCritic flags victim signal candidates and binds them with specific Trojan trigger ports to form attack schemes via either automatic analysis (*e.g.*, counter-based trigger input) or user specification (*e.g.*, payload with specific targets). For each design, 150-300 attack schemes are formed. We expand on the attack details as below:

- **Key Leak ⇒ risc16f84:** *risc16f84* [31] is a 8-bit RISC PIC microcontroller. *Key Leak* [47] Trojan observes the number of execution of a specific instruction. Above a certain number of execution the Trojan is triggered, and it manipulates data lines to the external EEPROM.
- **Bus Hijacker ⇒ Conmax:** *WISHBONE Interconnect Matrix IP core (Conmax)* [32] supports interconnecting up to 8 Masters and 16 Slaves. *Bus Hijacker* [48], [49] is a combinational Trojan inserted in the master interface to change the destination slave under a specific user-defined combination of master data input.
- **Timebomb ⇒ AES:** *AES core* [33] supports 128-bit encryption per clock cycle. *Timebomb* [48], [49] Trojan’s trigger is a 4-bit asynchronous counter incremented when two victim signals in AES module are active. When the counter overflows, the Trojan is triggered and attacks AES by flipping the least important bit of the current encryption output.
- **A2 ⇒ OR1200:** *ORI200 processor* [34] is a 32-bit scalar RISC CPU with 5-stage pipeline, virtual memory support (MMU), and basic DSP capabilities. *A2* [4] Trojan uses OR1200 in its original studies. As the stealthiest hardware Trojan to date, *A2* has a digital payload consisting of as few as two standard cells and an analog trigger functionally similar to a digital counter by charge accumulation and leakage. Once activated, *A2* performs privilege escalation attacks by altering a privilege bit flip-flop.

**Assessed Defenses.** We evaluate the effectiveness of the recent representative design-time Trojan defenses [23]–[27] by running SiliconCritic on the original designs and their

security-strengthened versions. We classify the defense techniques and list them as below:

- **Layout Compression.** The lowest-cost approach to implement a design-time Trojan defense is to adjust the P&R parameters (*e.g.*, core density, clock frequency, slew rate) provided by commercial CAD tools and observe the effects [27]. Based on the results in [27] and our practice, increasing placement density to compress free resources on the layout has the most direct protection effect by creating probabilistic increases of congestion around security-critical logic and wires. In this work, we adopt the uniform 90% global density for layout compression defense, which is almost the highest tolerable density for timing closure.

- **Built-In Self-Authentication (BISA).** Unlike undirected layout adjustment measures above, BISA [23] introduces additional tamper-evident logic to replace the filler cells and occupy the unused sites. Ba *et al.* [24], [25] improves BISA by prioritizing the free spaces (*i.e.*, preferring empty spaces near the security-critical cells) to fill, since appending extra logic everywhere is unroutable on an implemented IC layout. Here we evaluate the optimized version and still call it BISA, as a class of defenses.

- **GDSII-Guard.** Another recent work GDSII-Guard [26] also performs defenses targeting critical areas. Instead of increasing placement resources, it rearranges existing instances and routes on the layout to eliminate spatially contiguous regions near critical cells that are most vulnerable [28]. In this way, GDSII-Guard minimizes the impact on timing and power while improving security.

**Tools, Environment, and Runtime.** We perform all experiments on a server with 2-way 24-core Intel Xeon Gold 6248R CPUs @3.0GHz and 512GB DDR4 RAM. To prepare the IC layouts for defenses and evaluations, we synthesize and place-and-route all four designs with Cadence Genus (version 18.12) and Cadence Innovus (version 18.12). All designs target 60% core utilization, while risc16f84 and Conmax target 1GHz clock frequency, AES and OR1200 target 200MHz. In the SiliconCritic evaluation flow, we synthesize the Trojan with Cadence Genus, perform netlist extraction and coalescing with Python3.8, finish ECO Trojan insertion, and export side-channel metrics with Cadence Innovus and customized Tcl scripts. We use Nangate FreePDK45 Open Cell Library [39] for all design implementations. We leverage multi-processing to provide attack-scheme-level parallelism for ECO Trojan insertion and accelerate the SiliconCritic evaluation process. For the largest design OR1200 and its protected versions we evaluate, the total runtime of each is as short as about 40min, thanks to the efficiency of ECO and the parallelism techniques.

## 6.2. Timing Analysis

Fig. 6 shows the TNS and MPDR variation after Trojan insertion over 12 layouts, *i.e.*, 3 designs with their original and design-time-enhanced variants, against their respective Trojan attack. For each plot, the TNS/MPDR histogram has a percent y-axis, and the x-axis denotes a change in TNS (*ps*) or MPDR (%). The annotated Ref. is the reference

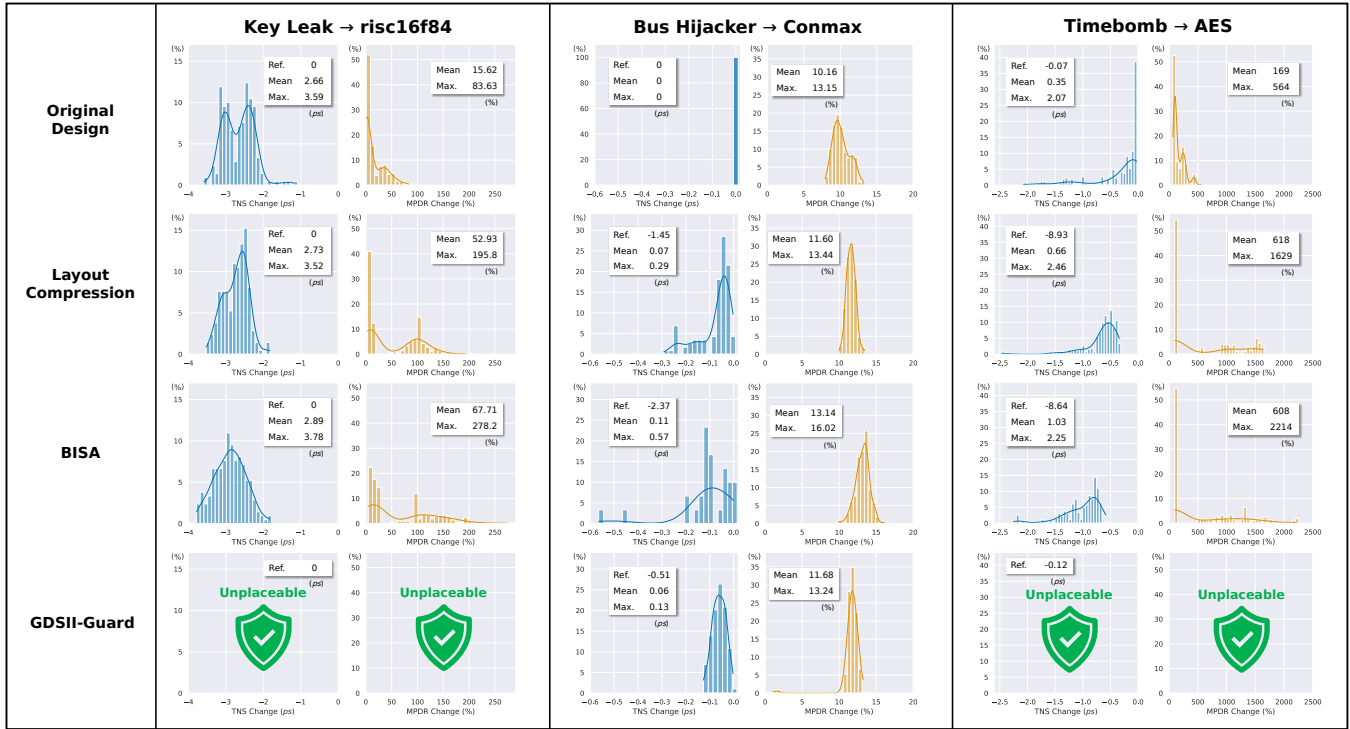


Figure 6. **Timing Analysis.** TNS and MPDR analysis of 3 attack⇒design pairs, and each design has 4 variants of layout (original, layout compression, BISA, and GDSII-Guard). We focus on the variation (Mean, Max.) of the TNS (*ps*) and MPDR (%) histograms from their respective Trojan-free reference. The base TNS of each Trojan-free layout is also annotated (Ref.), to help inspect the timing impact of each countermeasure on the original design.

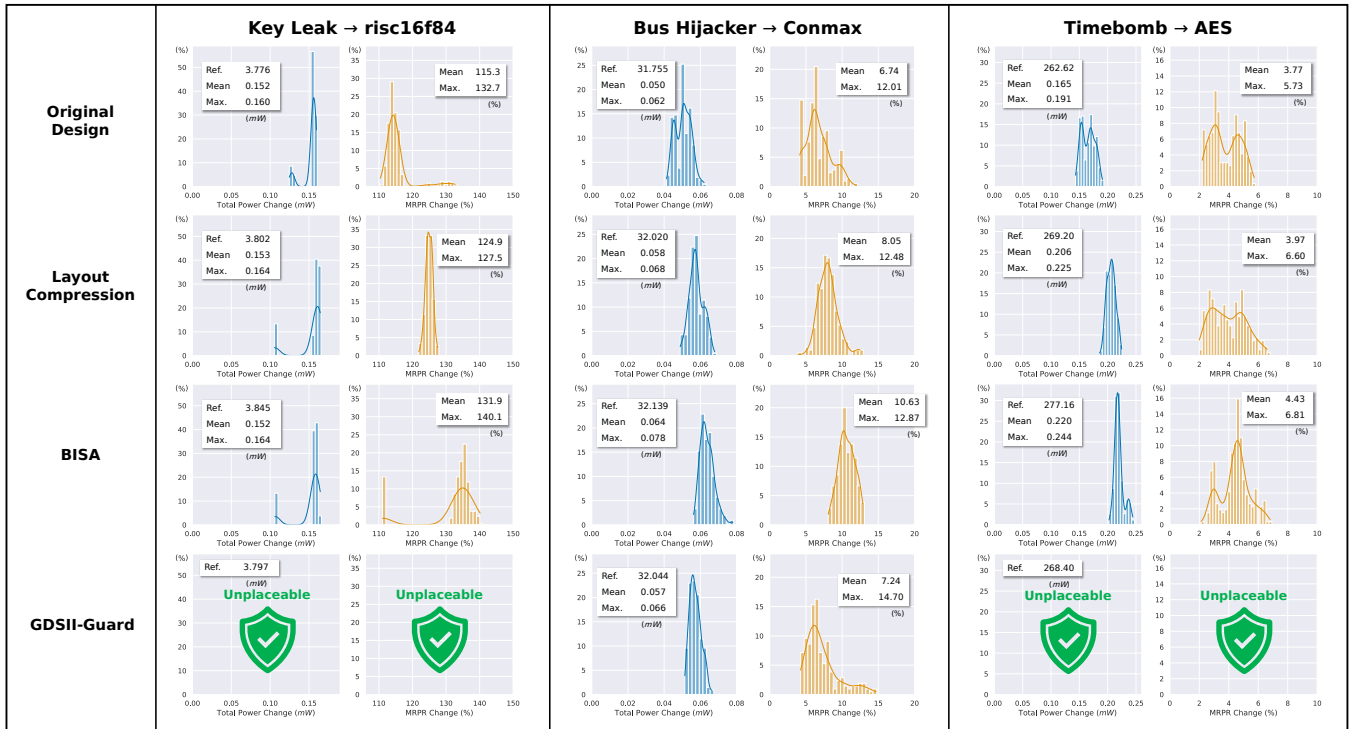


Figure 7. **Power Analysis.** Total power and MRPR analysis of 3 attack⇒design pairs, and each design has 4 variants of layout (original, layout compression, BISA, and GDSII-Guard). We focus on the variation (Mean, Max.) of the total power (*mW*) and MRPR (%) histograms from their respective Trojan-free reference. The base total power of each Trojan-free layout is also annotated (Ref.), to help inspect the power impact of each countermeasure on the original design.

TNS of the Trojan-free layout, while the MPDR reference is 100% according to its definition. The mean and maximum variation of the distributions from the reference data point is also marked. All the vertically aligned plots share the same x-axis and y-axis (in both scale and range) for the convenience of comparing the TNS/MPDR deviation among different enhanced layouts of the same design. The green “unplaceable” symbol indicates that the Trojan cannot be placed (and thus routed) on the given layout, and no further side-channel analysis can be performed.

We can simultaneously scan the results *vertically* to compare the effects of different design-time defenses, and *horizontally* to inspect the subtle discrepancy of the same defense’s effect caused by the different properties of the attack⇒design pairs. We make the following assumptions and interpretations. 1) Our experimental results affirm the prior notion [23], [27] that simply increasing placement density can frustrate Trojan insertion to some extent. For all three designs, compressed layouts (90% utilization) demonstrate larger average TNS/MPDR variation than the original ones. However, such “blind” layout filling manner provides low protecting efficiency and works at the expense of design timing with the worst TNS among all defenses. 2) Dedicated defensive strategies (BISA, GDSII-Guard) that manage free spaces around security-critical cells exhibit better Trojan detectability and lower timing overhead than layout compression. Without a core density as high as 90%, BISA has an advantage over layout compression on most metrics of the three designs by embedding filler logic near critical areas. GDSII-Guard shows the best overall timing, and makes Trojan placement impossible at two of three designs (explained next). 3) Sequential Trojans are easier to defend at design-time than combinational ones. This is because the D flip-flops contained in sequential logic are much larger than normal combinational gates. In 45nm technology, the minimal D flip-flop, DFF\_X1 occupies 17 sites, while the largest 2-input NAND gate, NAND2\_X4 occupies 9. From Fig. 6, we can see the sequential Timebomb Trojan shows much larger TNS and MPDR variation than the combinational Bus Hijacker, while the former has the minimum Trojan footprint. Moreover, considering that GDSII-Guard optimizes the layout with the goal of eliminating large spatially contiguous free spaces, if the threshold for the maximum # of sites in each contiguous region globally is 16, no room would be found to place even one single flip-flop. However, it is not viable to remove all the contiguous free regions that are as small as the size of Combinational Trojan gates since the overall cell density is restricted. 4) Path-level MPDR shows superior detection sensitivity than TNS. MPDR histogram contains much “outlier” data distributed far from the reference, which indicates that in such attack schemes the inserted Trojan is almost right located in the measured timing path, whereas the TNS histograms span across smaller ranges. When accumulated to TNS, the deviation between some negative paths can be easily masked.

To conclude, our timing analysis indicates that future design-time Trojan defenses should 1) perform meticulous, incremental layout optimization that focuses on critical

spaces, instead of striking high global utilization, 2) draws on the characteristics of the target Trojan and design to develop specific defense methodologies, and 3) preserve design timing while optimizing security.

### 6.3. Power Analysis

Similar to the timing analysis, Fig. 7 shows the total power and MRPR variations on the same combinations of designs and defenses. Our focus here is the variation (mean, max.) of the total power ( $mW$ ) and MRPR (%) histograms from their respective Trojan-free reference.

In general, the power analysis manifests similar variation behavior as the timing, when comparing each of the defenses. We will go through power-based observations and examinations. 1) Compared with timing, power increase after Trojan insertion is more uniform, namely, the mean variations among defenses are closer to each other, and the variance inside each histogram is smaller. Since Trojan gates constantly dissipate leakage power as any other gates, the expectation of power rise mostly depends on the Trojan composition. The defenses are differentiated indirectly by power, *i.e.*, they target damaging Trojan timing by making Trojan P&R difficult, while the power discrepancy results from different routing distances and added buffers for timing closure. 2) The degree of power increase is determined by both the Trojan composition and the Trojan footprint. For one thing, larger sequential Trojans consume more power than the smaller combinational ones. For another, attacks with smaller Trojan footprints exhibit less noticeable changes in TNS and MRPR. 3) Segmentation-based MRPR metrics effectively amplify the Trojan impact on power. Taking the original layout of Conmax against Bus Hijacker as an example, the total power increase on average is 0.16%, while the corresponding MRPR is 6.74%.

To conclude, power-based analysis shows overall similar results as timing, and relies on the actual Trojan composition and footprint. Future defenses need to supplement timing analysis with the information provided by power analysis or target power-centric detectability with the feedback of SiliconCritic post-Trojan power reports.

### 6.4. Case Study: A2

As the stealthiest hardware Trojan to date, A2 evades every known detection mechanism [27]. For conventional counter-based Trojans (*e.g.*, Key Leak, Timebomb), the sequential trigger requires an amount of large, digital logic cells, which prove to weaken the attack stealthiness in §6.2 and §6.3. A2 replaces the digital counter with analog components located in the analog layer, which are immune to delay extractions and not constrained by the timing requirements as their digital counterpart. The digital payload of A2 consists of simply two gates, NAND and INV with small area and timing overhead, creating a great chance to escape detection. Since the A2 Trojan is small enough to be inserted into the empty space of almost any design layout, it makes

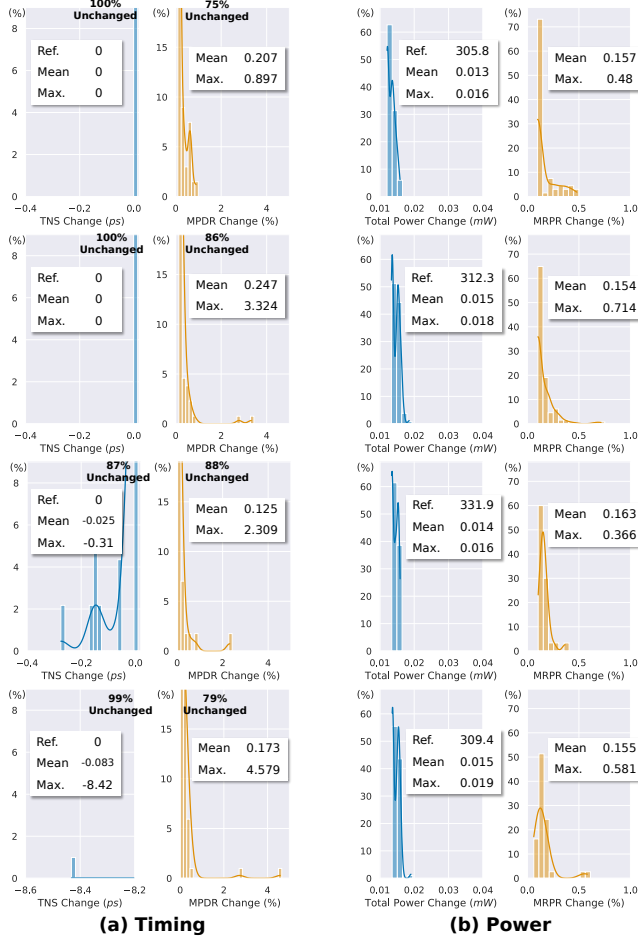


Figure 8. **A2 Analysis.** Side-channel analysis of OR1200 SoC attacked by A2 Trojan, with (a) timing (TNS, MPDR) and (b) power (total power, MRPR) variations. For both (a) (b), the subfigures are arranged in the same top-to-bottom order as Fig. 6, 7, representing the original layout and layouts processed by layout compression, BISA, and GDSII-Guard, respectively. In (a), we annotate the proportion of unchanged TNS (still  $0ps$ ) and MPDR (less than  $0.01\%$  increase) after Trojan insertion.

no sense for previous geometric-only metrics to estimate layout vulnerability without actual Trojan integration.

Due to the essential distinction of A2 from other evaluated Trojan attacks, we individually analyze the Trojan effects of A2 as a case study. Fig. 8 shows the timing (TNS, MPDR) and power (total power, MRPR) analysis of the original layout of OR1200 SoC, and those enhanced by layout compression, BISA, and GDSII-Guard. The format and arrangement are akin to Fig. 6 and Fig. 7. Compared with previously assessed conventional Trojans targeting smaller designs, here the timing and power perturbations are almost negligible, and distinguishing them from the fluctuations common to the delay and power measurements becomes extremely difficult. (a) Timing: For each of the defenses, unchanged TNS and MPDR are observed on most attacked layouts, while there exists a few outliers with large variations. (b) Power: The appended payload with two gates con-

tributes no more than  $0.015\%$  total power and  $0.16\%$  MRPR on average and can hardly be detected using power-only side-channel methods. Moreover, the defenses can instead lower the timing and power deviation because the defensive overheads on delay paths and power are likely to cancel the impact of small Trojans like A2.

To conclude, SiliconCritic analysis demonstrates that mainstream design-time defenses fail to impede A2 attacks by increasing the insertion cost. Future A2-centric defenses cannot concentrate only on placement since  $100\%$  core density is prohibitive for routing while the smallest free spaces can be exploited for A2 embedding. Future defenses can manage to block the integration of the analog and digital layers, protect critical signals from being tapped into with net shielding, or develop more sophisticated side-channel detection measures. In these ways, the defenses may provide high protection confidence as GDSII-Guard against sequential Trojans.

## 7. Discussions

### 7.1. SiliconCritic-Guided Design-Time Defense.

Classic IC design flow aims to optimize the performance, power, area, manufacturability, etc. of the chip. SiliconCritic opens up a new dimension: security against untrusted foundry attacks. SiliconCritic inspires design-time defenses in the following ways. 1) With the ability to configure the specific Trojan, SiliconCritic enables IC designers to tailor their protection approaches for the objective Trojans that are most adverse to the actual design. Our experiments imply that all-encompassing defenses rarely exist, while flexible defense strategies that deal with unique Trojan features (*e.g.*, size, digital logic type, domain) prevail general ones. 2) The simulation-based nature of SiliconCritic maximizes the prediction fidelity of Trojan threats and authentically reflects the coverage of assessed defenses. Existing design-time defenses are mostly *placement-centric* [23]–[26]. However, *routing* is another crucial aspect to thwart Trojan insertion, especially for the stealthiest, smallest Trojans like A2. The ECO Trojan insertion of SiliconCritic rests on the existing P&R landscape of the original layout and reflects the effects of defenses based on placement or routing. 3) The fast, parallel SiliconCritic evaluation flow enables IC designers to conduct efficient design space exploration over the parameters in the defense process to derive the optimal solution for the given layout and Trojan to tackle.

### 7.2. Cost for IC Designer and Attacker.

SiliconCritic unveils that the game between the IC designer and attacker won't be one-sided; *absolute* defense or attack is idealistic and impractical. For the designers, the goal ought to be increasing the difficulty for attackers to insert the target Trojan that shows good compatibility with the original design, or amplifying the detection sensitivity after insertion, instead of completely blocking Trojan insertion. Over aggressive defenses (*e.g.*, pursuing extremely high

placement density) are more likely to compromise common CAD metrics such as timing and power. The designers are supposed to strike a good balance between security and PPA, because the latter remains the primary objective of the modern CAD process. For the attackers, more baffling the Trojan is designed, more efforts will be spent to implement it within the foundry. Despite of the strength of A2 Trojan, analog-domain placement and analog-digital integration are more arduous than the pure digital.

### 7.3. Extensibility of SiliconCritic

**Extended to Customized Side-Channel Metrics.** SiliconCritic provides fundamental, pluggable timing/power side-channel metrics, which prove to be effective to spot Trojan impact in most cases. The user can also introduce more sophisticated side-channel measures (*e.g.*, multi-parameter analysis, calibration, test generation [35]) to further highlight Trojan presence and cancel out random noise. Since the side-channel evaluation backend of SiliconCritic is orthogonal to its Trojan insertion frontend, the users can customize their own side-channel methods seamlessly using our Python programming interfaces to manipulate post-Trojan layouts in batch. Note that SiliconCritic generates a family of post-Trojan layouts with various attack schemes, and this diversity in statistics is essential for advanced side-channel analysis to locate Trojans with minimized interference of noise [16].

**Extended to Imaging-Based Trojan Detection.** Optical imaging (*e.g.*, Scanning Electron Microscope (SEM)) is an alternative to side-channel analysis for post-fabrication Trojan detection [50], [51]. Imaging-based detection requires the acquisition at high magnification of each standard cell in the image of the fabricated chip, and conducts detection by correlation with a golden layout image / GDSII file / text CAD file (.def). SiliconCritic provides both Trojan-free and Trojan-in GDSII (and the corresponding CAD files), which can serve as golden references for real layout images. Just like converting vector graphics to pixel ones, the GDSII can be conveniently transformed into simulated SEM images with several levels of image noise. In this way, SiliconCritic can be smoothly extended to simulate the detection effect of chip imaging.

**Extended to Advanced Process Nodes.** SiliconCritic adopts the open-source, academic Nangate 45nm process technology to build the layouts and test Trojan insertion. However, SiliconCritic is independent of technology libraries; users can configure their private advanced process node by preparing the technology file. SiliconCritic accepts standard technology library files (.lef) that specify design rules for P&R and electrical data of cells and metal layers.

**Extended to Other CAD Tools.** SiliconCritic relies on commercial CAD tools (Cadence Genus and Innovus) for design/attack implementation and metric reporting. Nevertheless, the intermediate results (.def, .v) and control scripts (.tcl) are uniform, so it is trivial for the users to transfer to other open-source or commercial CAD tools.

**Extended to the Attacker’s Perspective.** In most of the context of this paper, we discuss that equipped with SiliconCritic, how IC designers can evaluate the layout vulnerabilities and optimize the design-time protections. Equivalently, attackers can also leverage SiliconCritic to facilitate the design of formidable Trojans against all variety of design-time defenses.

### 7.4. Limitations and Justifications

**Impact of Process Variation.** Since SiliconCritic is a pre-fabrication, simulation-based tool, there exist gaps between design-time predictions and golden results after fabrication. One major problem is that process variation in device parameters can obscure the impact of Trojans, leading to inconsistencies between simulated and post-fabrication results. Nevertheless, SiliconCritic allows for minimizing the impact of process variation in several ways. For one thing, most Trojans can result in evident timing or power variations that exceed the process noise margin, which designers can identify based on the actual process and foundry specifications. For another, the bundle of Trojan-inserted layouts SiliconCritic produce also allows designers to minimize noise with statistics-based calibrations that are prevalent in advanced side-channel analysis [16].

**Conservativeness of Metrics.** It is common practice to design relatively *conservative* security metrics to avoid false negatives when defending against real attacks. The side-channel verifier SiliconCritic assumes is more capable than an authentic one because the simulated timing/power information is more precise and noiseless than physical measurements, as stated above. Consequently, the actual post-fabrication side-channel detection can fail to detect a Trojan-inserted layout with a high MPDR/MRPR score in simulations. However, SiliconCritic implies improved conservativeness in the assumed attacker’s capabilities and the extensibility of metrics. SiliconCritic estimates the lower bound difficulty of Trojan insertion (*i.e.*, the attacks generated are the *stealthiest*), since the actual attack in foundries may not do it as well as in the simulation with complete circuit information. In effect, SiliconCritic overrates the capabilities of both the attacker side and the defense (post-fabrication detection) side. Additionally, the designer can determine “how much the side-channel metrics change would mean security to me” by configuring a larger noise floor on the simulated metrics to increase the confidence of Trojan detection.

**Range of Supported Trojans.** SiliconCritic deals with threat evaluation of *digital*, *additive* Trojans. It is not capable of handling analog-domain evaluations which require special tools and are incompatible with digital simulations. However, thanks to the extensibility of SiliconCritic to optical detection, it is promising to estimate the threat of less common subtractive Trojans such as dopant-level Trojans, which require exclusive SEM imaging techniques to identify [51], [52]. Although SiliconCritic can directly deal with ordinary Trojans, the unusual yet stealthy attacks emphasize the significance of SiliconCritic’s extensibility.

**Trade-Off between Customizability and Generality.** As a Trojan-specific assessment framework, SiliconCritic requires *known* Trojans to conduct the simulation and measurement. However, the previous geometric-only metrics can work for generic evaluations despite of their inaccuracies. Such generality may benefit IC designers when they need to gain a brief understanding of vulnerabilities of new designs. The distinction between these two styles of assessments is just like attack surface analysis as opposed to what fingerprint-based antivirus (AV) tools do, and both have pros and cons under different circumstances.

## 8. Related Works

### 8.1. Foundry-Level Trojan Attacks

**Fabrication-Time Implementations.** We cover several concrete Trojan designs in terms of structure and functionality in §6.1. Here we summarize previous works that show the practical feasibility of inserting such Trojans into finalized layouts by untrustworthy foundries. In 2009, Lin *et al.* [3] is the first to suggest the possibility of foundry-level attacks with a Trojan designed to leak the keys of AES covertly over the power side-channel, while they only implemented this attack on an FPGA. In 2016, Yang *et al.* [4] proposed A2 Trojan and fabricated an OR1200 SoC with the Trojan inserted, the first manufactured foundry-level attack on an ASIC. In 2020, Ghandali *et al.* [5] presented a Trojan introducing path delay faults to a side-channel protected block cipher. The attack was realized in 65nm and 90nm CMOS technologies. In 2022, Perez *et al.* [6] inserted a side-channel Trojan via ECO into a commercial 65nm chip. In the same year, Almeida *et al.* [7] developed a hardware-based ransomware Trojan and conducted its injection in a foundry-like manner with an increase of 0.7% in core density and 2% in static power.

**Design-Time Simulations.** In addition to the above works that focus on realizing Trojan insertions in foundry-like environments, another branch of research is concerned with performing design-time Trojan attacks that should have been carried out at fabrication time. Due to the huge cost of foundry operations, previous works with silicon demonstrations mostly target few specific Trojans, while those with design-time attacks are able to explore Trojan diversities. In 2021, Perez *et al.* [43] inserted a side-channel hardware Trojan based on ECO for the first time, and claimed that such process can be replicated in a foundry effortlessly. They also displayed some CAD parameters before and after Trojan insertion, such as core density, power, and timing slacks. In 2022, Hepp *et al.* [44] extended [43] by starting from an unknown layout, enabling flexible Trojan netlist generation and automatic signal selection for Trojan hooking. Nevertheless, SiliconCritic views Trojan insertion and evaluation from a different angle, compared with these works: 1) They target one/several hardcoded Trojans, while SiliconCritic accepts arbitrary user-provided ones. 2) They provide single attack possibility, while SiliconCritic evaluates potential attack schemes. 3) They solely observe side-channel parameters,

while SiliconCritic studies the variations of parameter distributions to their Trojan-free reference. 4) They serve as methods for Trojan attacks, while SiliconCritic provides an automatic framework as opposed to simply security metrics or attack/defense techniques.

### 8.2. Metrics for IC Layout Vulnerability

Several sets of geometric-only metrics have been proposed to estimate layout vulnerability against additive Trojans. In 2016, Salmani *et al.* [29] and Hossein-Talaei *et al.* [30] defined similar concepts called region vulnerability, which is proportional to the normalized unused spaces and routing channels in an arbitrary layout region. Then a vulnerability map can be built for the whole layout. In 2020, Trippel *et al.* [27] systematically profiled the Trojan insertion process and proposed dedicated metrics to predict the difficulty of each insertion step based on free P&R resources. In 2022, Johann *et al.* [28] proposed the concept of exploitable region, which identifies empty spaces with high risk based on the relative position of security-critical cells and the spatial connectivity of sites.

## 9. Conclusion

SiliconCritic is a simulation-based framework that bridges the gap between the actual foundry-level Trojan attacks and the estimation of Trojan threat. The SiliconCritic evaluation flow consists of three phases: 1) Netlist preparation extracts security-critical signals and attaches them to Trojan ports to generate gate-level attack schemes. 2) Trojan insertion performs transistor-level ECO based layout modification. 3) Side-channel analysis quantifies layout susceptibility with detectability. The experiments on multiple attack⇒design pairs with varied Trojan footprints highlight the importance of critical-area-centric protections and reveal the impracticality of one-size-fits-all defenses.

SiliconCritic is more than a layout assessment tool as previous metrics. Its simulation-based nature and integration of side-channel detection enable IC designers to recursively refine their layouts to improve the Trojan detectability over some post-silicon parameters. The extensibility of SiliconCritic paves the way for future upgrading of analysis backbone, process node, CAD tools, etc. We expect to witness more novel defenses against the stealthiest foundry-level Trojans with the aid of SiliconCritic.

## Acknowledgements

We thank the anonymous reviewers and our shepherd for their insightful feedback, which help us enhance the quality of this paper.

This work was partly supported by the National Natural Science Foundation of China (Grant No. 62090021), the National Key R&D Program of China (Grant No. 2022YFB4500500), and DeePoly Technology Inc.

## References

- [1] K. Everington, "TSMC starts work on US\$19.6 Billion 3nm fab in S. Taiwan," Taiwan News, Oct. 28, 2019. <https://www.taiwannews.com.tw/en/news/3805032>.
- [2] A. Friedman, "TSMC's new 2nm chip production fab will cost it how much?" PhoneArena, Jun. 7, 2022. [https://www.phonearena.com/news/tsmc-to-spend-fortune-on-2nm-production-fab\\_id140626](https://www.phonearena.com/news/tsmc-to-spend-fortune-on-2nm-production-fab_id140626).
- [3] L. Lin, M. Kasper, T. Güneysu, C. Paar, and W. Bursell, "Trojan side-channels: Lightweight hardware Trojans through side-channel engineering," in *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, 2009, pp. 382–395.
- [4] K. Yang, M. Hicks, Q. Dong, T. M. Austin, and D. Sylvester, "A2: analog malicious hardware," in *IEEE Symposium on Security and Privacy (S&P)*, 2016, pp. 18–37.
- [5] S. Ghandali, T. Moos, A. Moradi, and C. Paar, "Side-channel hardware Trojan for provably-secure SCA-protected implementations," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 6, pp. 1435–1448, 2020.
- [6] T. Perez and S. Pagliarini, "Hardware Trojan insertion in finalized layouts: From methodology to a silicon demonstration," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2022.
- [7] F. Almeida, M. Imran, J. Raik, and S. Pagliarini, "Ransomware attack as hardware Trojan: A feasibility and demonstration study," *IEEE Access*, vol. 10, pp. 44 827–44 839, 2022.
- [8] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor, "Trustworthy hardware: Identifying and classifying hardware Trojans," *Computer*, vol. 43, no. 10, pp. 39–46, 2010.
- [9] J. Knechtel, J. Gopinath, J. Bhandari, M. Ashraf, H. Amrouch, S. Borkar, S. K. Lim, O. Sinanoglu, and R. Karri, "Security closure of physical layouts ICCAD special session paper," in *International Conference on Computer-Aided Design (ICCAD)*, 2021, pp. 1–9.
- [10] K. Xiao, D. Forte, Y. Jin, R. Karri, S. Bhunia, and M. M. Tehranipoor, "Hardware Trojans: Lessons learned after one decade of research," *ACM Trans. Design Autom. Electr. Syst.*, vol. 22, no. 1, pp. 6:1–6:23, 2016.
- [11] S. Bhunia, M. Abramovici, D. Agrawal, P. Bradley, M. S. Hsiao, J. Plusquellic, and M. Tehranipoor, "Protection against hardware Trojan attacks: Towards a comprehensive solution," *IEEE Des. Test*, vol. 30, no. 3, pp. 6–17, 2013.
- [12] M. Tehranipoor and F. Koushanfar, "A survey of hardware Trojan taxonomy and detection," *IEEE Des. Test Comput.*, vol. 27, no. 1, pp. 10–25, 2010.
- [13] R. S. Chakraborty, F. G. Wolff, S. Paul, C. A. Papachristou, and S. Bhunia, "MERO: A statistical approach for hardware Trojan detection," in *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, 2009, pp. 396–410.
- [14] M. Banga, M. Chandrasekar, L. Fang, and M. S. Hsiao, "Guided test generation for isolation and detection of embedded Trojans in ICs," in *Great Lakes Symposium on VLSI (GLSVLSI)*, 2008, pp. 363–366.
- [15] A. Waksman, M. Suozzo, and S. Sethumadhavan, "FANCI: identification of stealthy malicious logic using boolean functional analysis," in *CCS*. ACM, 2013, pp. 697–708.
- [16] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar, "Trojan detection using IC fingerprinting," in *IEEE Symposium on Security and Privacy (S&P)*, 2007, pp. 296–310.
- [17] M. Potkonjak, A. Nahapetian, M. Nelson, and T. Massey, "Hardware Trojan horse detection using gate-level characterization," in *Design Automation Conference (DAC)*, 2009, pp. 688–693.
- [18] D. Rai and J. C. Lach, "Performance of delay-based Trojan detection techniques under parameter variations," in *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2009, pp. 58–65.
- [19] Y. Jin and Y. Makris, "Hardware Trojan detection using path delay fingerprint," in *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2008, pp. 51–57.
- [20] S. Narasimhan, X. Wang, D. Du, R. S. Chakraborty, and S. Bhunia, "TeSR: A robust temporal self-referencing approach for hardware Trojan detection," in *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2011, pp. 71–74.
- [21] S. Narasimhan, D. Du, R. S. Chakraborty, S. Paul, F. G. Wolff, C. A. Papachristou, K. Roy, and S. Bhunia, "Hardware Trojan detection by multiple-parameter side-channel analysis," *IEEE Trans. Computers*, vol. 62, no. 11, pp. 2183–2195, 2013.
- [22] D. Ismari, J. Plusquellic, C. Lamech, S. Bhunia, and F. Saqib, "On detecting delay anomalies introduced by hardware Trojans," in *International Conference on Computer-Aided Design (ICCAD)*, 2016, p. 44.
- [23] K. Xiao and M. Tehranipoor, "BISA: built-in self-authentication for preventing hardware Trojan insertion," in *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2013, pp. 45–50.
- [24] P. Ba, P. Manikandan, S. Dupuis, M. Flottes, G. D. Natale, and B. Rouzeyre, "Hardware Trojan prevention using layout-level design approach," in *European Conference on Circuit Theory and Design (ECCTD)*, 2015, pp. 1–4.
- [25] P. Ba, S. Dupuis, P. Manikandan, M. Flottes, G. D. Natale, and B. Rouzeyre, "Hardware trust through layout filling: A hardware Trojan prevention technique," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2016, pp. 254–259.
- [26] X. Wei, J. Zhang, and G. Luo, "GDSII-Guard: ECO anti-Trojan optimization with exploratory timing-security trade-offs," in *Design Automation Conference (DAC)*, 2023, pp. 1700–1705.
- [27] T. Trippel, K. G. Shin, K. B. Bush, and M. Hicks, "ICAS: an extensible framework for estimating the susceptibility of IC layouts to additive Trojans," in *IEEE Symposium on Security and Privacy (S&P)*, 2020, pp. 1742–1759.
- [28] J. Knechtel, J. Gopinath, M. Ashraf, J. Bhandari, O. Sinanoglu, and R. Karri, "Benchmarking security closure of physical layouts: ISPD 2022 contest," in *International Symposium on Physical Design (ISPD)*, 2022, pp. 221–228.
- [29] H. Salmani and M. M. Tehranipoor, "Vulnerability analysis of a circuit layout to hardware trojan insertion," *IEEE Trans. Inf. Forensics Secur.*, vol. 11, no. 6, pp. 1214–1225, 2016.
- [30] H. Hossein-Talae and A. Jahanian, "Layout vulnerability reduction against Trojan insertion using security-aware white space distribution," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2017, pp. 551–555.
- [31] OpenCores.org, "RISC16F84," <https://opencores.org/projects/risc16f84>.
- [32] —, "WISHBONE Conmax IP core," [https://opencores.org/projects/wb\\_conmax](https://opencores.org/projects/wb_conmax).
- [33] —, "AES core," [https://opencores.org/projects/tiny\\_aes](https://opencores.org/projects/tiny_aes).
- [34] —, "OpenRISC OR1200 processor," <https://github.com/openrisc/or1200>.
- [35] S. Bhunia, M. S. Hsiao, M. Banga, and S. Narasimhan, "Hardware Trojan attacks: Threat analysis and countermeasures," *Proc. IEEE*, vol. 102, no. 8, pp. 1229–1247, 2014.
- [36] R. S. Chakraborty, S. Narasimhan, and S. Bhunia, "Hardware Trojan: Threats and emerging solutions," in *IEEE International High Level Design Validation and Test Workshop (HLDVT)*, 2009, pp. 166–171.
- [37] R. Torrance and D. James, "The state-of-the-art in IC reverse engineering," in *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, 2009, pp. 363–381.

- [38] R. S. Rajarathnam, Y. Lin, Y. Jin, and D. Z. Pan, "Regds: a reverse engineering framework from gdsii to gate-level netlist," in *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. IEEE, 2020, pp. 154–163.
- [39] J. E. Stine, I. Castellanos, M. Wood, J. Henson, F. Love, W. R. Davis, P. D. Franzon, M. Bucher, S. Basavarajaiah, J. Oh, and R. Jenkal, "FreePDK: An open-source variation-aware design kit," in *IEEE International Conference on Microelectronic Systems Education (MSE)*, 2007, pp. 173–174.
- [40] S. K. Karandikar and S. S. Sapatnekar, "Technology mapping using logical effort for solving the load-distribution problem," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 27, no. 1, pp. 45–58, 2008.
- [41] L. H. Goldstein and E. L. Thigpen, "SCOAP: Sandia controllability/observability analysis program," in *Design Automation Conference (DAC)*, 1980, pp. 190–196.
- [42] S. Takamaeda-Yamazaki, "Pyverilog: A Python-based hardware design processing toolkit for Verilog HDL," in *International Symposium on Applied Reconfigurable Computing (ARC)*, 2015, pp. 451–460.
- [43] T. D. Perez, M. Imran, P. Vaz, and S. Pagliarini, "Side-channel trojan insertion - a practical foundry-side attack via ECO," in *International Symposium on Circuits and Systems (ISCAS)*, 2021, pp. 1–5.
- [44] A. Hepp, T. Perez, S. Pagliarini, and G. Sigl, "A pragmatic methodology for blind hardware Trojan insertion in finalized layouts," in *International Conference on Computer-Aided Design (ICCAD)*, 2022, pp. 69:1–69:9.
- [45] C. Lamech, R. M. Rad, M. Tehranipoor, and J. Plusquellic, "An experimental analysis of power and delay signal-to-noise requirements for detecting Trojans and methods for achieving the required detection sensitivities," *IEEE Trans. Inf. Forensics Secur.*, vol. 6, no. 3-2, pp. 1170–1179, 2011.
- [46] F. Wang, Q. Wang, B. Fu, S. Jiang, X. Zhang, L. Alrahis, O. Sinanoglu, J. Knechtel, T.-Y. Ho, and E. F. Young, "Security closure of IC layouts against hardware Trojans," in *International Symposium on Physical Design (ISPD)*, 2023, pp. 229–237.
- [47] S. T. King, J. Tucek, A. Cozzie, C. Grier, W. Jiang, and Y. Zhou, "Designing and implementing malicious hardware," in *USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET)*, 2008.
- [48] H. Salmani, M. Tehranipoor, and R. Karri, "On design vulnerability analysis and trust benchmarks development," in *International Conference on Computer Design (ICCD)*, 2013, pp. 471–474.
- [49] B. Shakya, T. He, H. Salmani, D. Forte, S. Bhunia, and M. Tehranipoor, "Benchmarking of hardware Trojans and maliciously affected circuits," *Journal of Hardware and Systems Security*, vol. 1, pp. 85–102, 2017.
- [50] F. Courbon, P. Loubet-Moundi, J. J. Fournier, and A. Tria, "A high efficiency hardware trojan detection technique based on fast sem imaging," in *Design, Automation, and Test in Europe (DATE)*. IEEE, 2015, pp. 788–793.
- [51] E. Puschner, T. Moos, S. Becker, C. Kison, A. Moradi, and C. Paar, "Red team vs. blue team: A real-world hardware Trojan detection case study across four modern CMOS technology generations," in *IEEE Symposium on Security and Privacy (S&P)*, 2023, pp. 56–74.
- [52] T. Sugawara, D. Suzuki, R. Fujii, S. Tawa, R. Hori, M. Shiozaki, and T. Fujino, "Reversing stealthy dopant-level circuits," in *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, 2014, pp. 112–126.

## Appendix A. Meta-Review

The following meta-review was prepared by the program committee for the 2024 IEEE Symposium on Security and Privacy (S&P) as part of the review process as detailed in the call for papers.

### A.1. Summary

This paper proposes SiliconCritic, a framework for quantifying the risk of a given layout design against hardware Trojan (HT) insertion by a malicious foundry. The paper first shows the limitation of the conventional geometric-only metric and proposes SiliconCritic as a substitute. The idea behind SiliconCritic is inserting synthesized HT and measuring the side channel difference through simulation. The proposed method is evaluated with a combination of the open-source IC designs, HT, and the representative design-time HT defenses using timing and power analyses.

### A.2. Scientific Contributions

- Creates a New Tool to Enable Future Science.
- Provides a Valuable Step Forward in an Established Field.

### A.3. Reasons for Acceptance

- 1) Pre-silicon hardware trojan assessment is an important area of research, and the proposed approach, using simulation-based side-channel analyses as a metric for HT resistance, is significant and offers a leap for future research.
- 2) The feasibility and effectiveness of SiliconCritic are evaluated with a well-selected combination of target designs, HTs, and HT defenses.

### A.4. Noteworthy Concerns

- 1) SiliconCritic is as correct as the underlying simulation, and there are known differences between the simulated and actual circuits, such as the ones caused by process variation.
- 2) The proposed metric is tied with the easiness of side-channel-based HT detection, and the distinction between the simulated and measured side-channel information can cause false negatives, i.e., easy to detect in simulation but difficult to detect in real measurement.
- 3) The proposed method is more specific than the conventional geometric-only metrics, potentially narrowing the target HTs.
- 4) The evaluation is limited to small examples, and the scalability to practical, large-scale circuit designs is not verified.