International Journal of Computer Engineering and Technology (IJCET)
Volume 14, Issue 1, January-April 2023, pp. 167-189, Article ID: IJCET_14_01_015
Available online at https://iaeme.com/Home/issue/IJCET?Volume=14&Issue=1
ISSN Print: 0976-6367; ISSN Online: 0976-6375; Journal ID: 5751-5249
Impact Factor (2023): 18.14 (Based on Google Scholar Citation)
DOI: https://doi.org/10.34218/IJCET_14_01_015

HARDWARE TROJAN DETECTABILITY ENHANCEMENT USING LOGIC LOCKING AND POWER DISCREPANCY ANALYSIS

Santosh Appachu Devanira Poovaiah
Electrical and Computer Engineering
University of Southern California, Los Angeles, California, USA.

ABSTRACT

Most semiconductor companies outsource the manufacturing of their chip designs to third-party fabrication foundries. However, untrusted foundries pose significant security risks, including intellectual property (IP) piracy and the insertion of hardware Trojans (HTs)—malicious modifications intended to sabotage circuits. These HTs can lead to increased power consumption, but due to process variations and the minimal footprint of small HTs in large-scale circuits, traditional power consumption analysis (PCA) often fails to detect them. To address this challenge, circuits can be partitioned into smaller sub-circuits, with PCA applied individually to each. In this paper, we propose a novel logic locking [2] technique designed to enhance the effectiveness of PCA-based HT detection. Building on the "RESTORATION" approach introduced in [1], our method increases the power contribution of HTs relative to the overall circuit power, thereby improving their detectability. Experimental results at both the gate level and post-synthesis demonstrate that the proposed locking scheme facilitates more efficient and reliable HT detection compared to existing logic locking [2] techniques.

Keywords: Hardware security, logic Locking, Hardware Trojan detection, AND/OR locking, Power side channel analysis, Power consumption Analysis (PCA).

Cite this Article: Santosh Appachu Devanira Poovaiah. (2023). Hardware Trojan Detectability Enhancement Using Logic Locking and Power Discrepancy Analysis. *International Journal of Computer Engineering and Technology (IJCET)*, 14(1), 167-189. DOI: https://doi.org/10.34218/IJCET_14_01_015

1. Introduction

With the increasing complexity and cost of semiconductor manufacturing, most companies now rely on third-party fabrication foundries for the production of integrated circuits (ICs). However, outsourcing to untrusted fabs introduces significant security risks, including intellectual property (IP) piracy and the insertion of malicious components into the design. Hardware Trojans (HTs) are a major concern in this context. These are malicious circuit modifications deliberately inserted into ICs to compromise their functionality or integrity. What makes HTs particularly dangerous is their ability to evade detection during conventional post-silicon verification processes. Once embedded, HTs can be triggered under rare conditions, potentially causing system failure or data leakage. This poses serious threats to sensitive sectors such as military defense, financial systems, and critical infrastructure like transportation. In addition to the threat of HTs, untrusted fabs may engage in reverse engineering or overproduction of proprietary IC designs, leading to IP theft and unauthorized distribution. To counteract these risks, logic locking [2] has emerged as a promising design-time solution. Logic locking protects chip designs by embedding secret keys into the logic, thereby preventing unauthorized access, tampering, and overproduction. Given the growing reliance on third-party manufacturing and the associated security threats, robust mechanisms for HT detection and IP protection are imperative. This paper focuses on enhancing HT detection using logic locking [2] in conjunction with power consumption analysis (PCA), a technique that leverages anomalies in power usage to identify malicious insertions.

A class of HT detection methods is based on side-channel analysis [4]. Side-channels are circuitry parameters e.g. power consumption, performance, etc. Any abnormality in the side channels measured from a circuit under Trojan test (CUTT) warns the presence of an HT. However, HT effects on side channels may be undetectable among the effects of process and environmental variations. Process variations happen during IC manufacturing and cause variations in some transistor characteristics such as channel length and oxide-thickness. Environmental variations happen while a circuit is working and induce changes in the operating environment of the circuit such as temperature and supply voltage [1].

Logic locking is a preventive DfTr technique initially introduced to combat IP piracy. This method obfuscates the functionality of a circuit, rendering it unusable by attackers or pirates unless the correct key is applied. If an incorrect key is provided, the circuit will fail to operate as intended, thereby preventing unauthorized duplication or resale. Moreover, while HT attackers may attempt to insert a Trojan into a locked circuit, such Trojans might only become active when incorrect keys are used—further complicating their efforts and reducing the likelihood of successful exploitation [1]. In this work, we expanded on the DfTr approach facilitating HT detection methods based on power consumption analysis (PCA). The work was built upon the article “Restricting Switching Activity Using Logic Locking to Improve Power Analysis Based Trojan Detection” [1]. We propose a novel AND/OR based logic locking technique and compare its effectiveness against traditional MUX- and XOR-based locking schemes. In addition to offering the general security advantages of logic locking [2] as a preventive measure, our method introduces a key benefit: switching activity localization. Specifically, it increases switching activity in a targeted region of the circuit while reducing activity in the rest of the design. This results in an overall decrease in total power consumption, as only the subcircuit under test is actively engaged. If the targeted region contains a hardware Trojan (HT), its switching activity—and consequently, its power consumption—will increase. By comparing the circuit’s power profile with and without the HT, we can effectively detect the presence of the Trojan.

2. Problem Definition

According to Moore’s Law, the number of transistors on a chip doubles approximately every two years. With the continuous scaling of technology nodes into the nanometer regime, parametric variations have become increasingly significant. These variations pose a serious challenge to power consumption analysis (PCA)-based hardware Trojan (HT) detection methods, as they obscure the subtle power differences introduced by small-scale Trojans. To address the scalability limitations of PCA-based HT detection, the circuit under test (CUTT) is partitioned into multiple sub-circuits, each of which is individually analyzed using PCA. A promising approach to support this strategy is Switching Activity Localization (SAL). SAL offers two main advantages: (1) if a Trojan is present in the targeted sub-circuit, its switching activity—and thus its power consumption—is amplified, making it more detectable; and (2) by confining activity to a single region, the overall power consumption and corresponding variations across the CUTT are minimized. This improves the signal-to-noise ratio for detecting HTs in complex, modern ICs.

3. Proposed Method and Comparison with Related Work

Our proposed method involves dividing the circuit into sub-circuits using logic locking, with the number of sub-circuits equal to the desired number of key-gates. Corresponding key-gates are then inserted in a way that maximizes their impact on Switching Activity Localization (SAL). Unlike the reference work, which uses MUX-based locking, we employ AND/OR-based logic locking. This approach is demonstrated using the ISCAS85 C17 benchmark circuit. The dotted lines in Fig.1 surround two sub-circuits SC1 and SC2 are obtained by a partitioning algorithm.

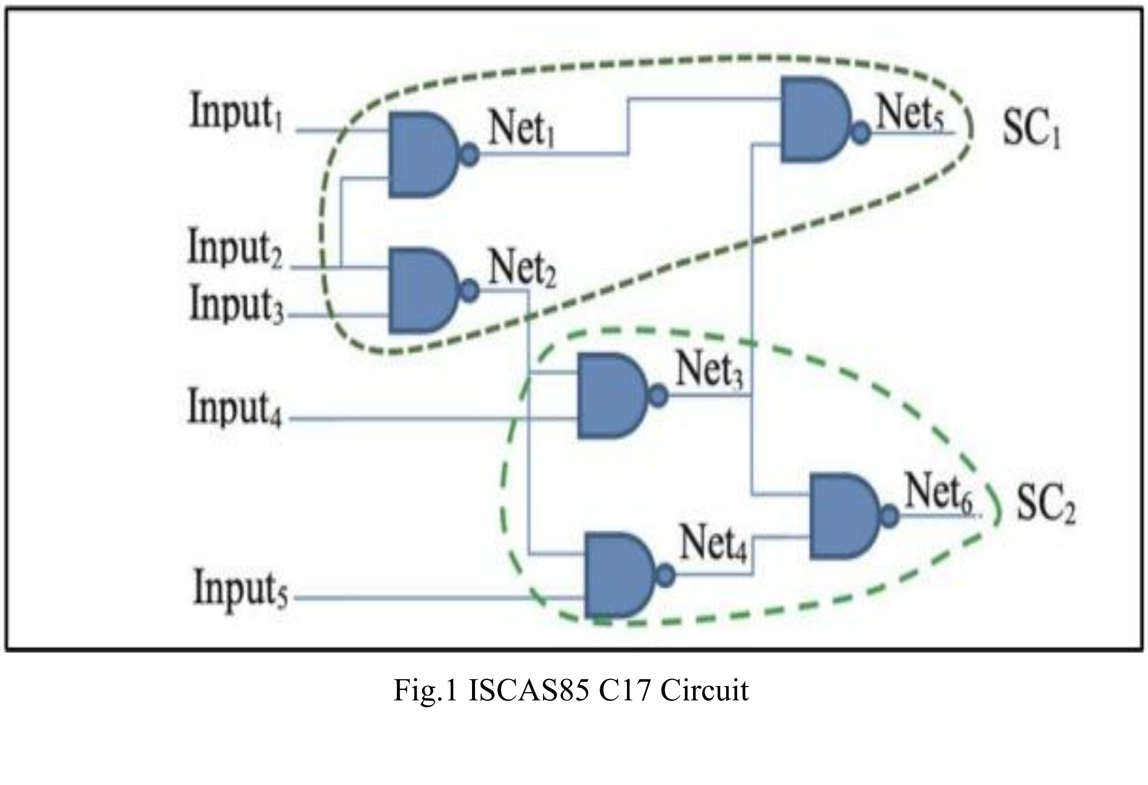


Fig.1 ISCAS85 C17 Circuit

3.1 Partitioning the circuit

Every digital circuit can be partitioned into *cones of logic*, where each cone represents a region of the circuit influenced by a specific input or a set of inputs. As illustrated in Fig. 2, inputs **ni** and **nj** activate logic within their respective cones. With each input transition, not all elements of the circuit switch states—only those within the relevant cone are affected. However, due to overlap between these cones, some switching activity can propagate across cone boundaries. Our objective in partitioning is to minimize this cross-cone propagation and confine switching activity to the specific sub-circuit under test. This localization improves the precision of power-based hardware Trojan (HT) detection by isolating the region of interest and reducing noise from unrelated circuit regions. Such partitioning techniques are widely used in the industry to facilitate Design-for-Test (DFT) processes.

To perform this partitioning, we employ the Kernighan–Lin (K–L) algorithm [3], a foundational heuristic widely used in graph partitioning problems, particularly in VLSI design and circuit optimization. The K–L algorithm models the circuit as an undirected graph, where nodes represent individual logic elements (such as gates or modules), and edges represent interconnections or signal paths between them. The primary objective of the algorithm is to divide the graph into two roughly equal-sized partitions in a way that minimizes the number of edges (also known as the "cut size") crossing between them. The algorithm begins with an initial partition—often chosen randomly or based on structural hierarchy—and then iteratively improves this partitioning through a series of node pair swaps. In each iteration, it identifies and exchanges pairs of nodes from opposite partitions in a manner that results in the greatest reduction in cut size. After several iterations, the algorithm converges to a locally optimal solution that offers a significantly improved partitioning layout compared to the initial configuration. One of the strengths of the K–L algorithm lies in its ability to maintain partition balance, ensuring that the two resulting groups are of approximately equal size. This is crucial in hardware security applications, as unbalanced partitions can skew power distribution and undermine switching activity analysis. In our context, using the K–L algorithm enables the division of the circuit into localized sub-circuits that minimize cross-partition activity. By reducing interconnectivity between these sub-circuits, we effectively confine the switching activity to targeted regions. This is essential for Switching Activity Localization (SAL) and enhances the sensitivity of power consumption analysis (PCA) in detecting hardware Trojans. Both our proposed method and the reference work utilize the K–L algorithm to achieve efficient, well-isolated circuit partitions, which form the basis for robust and scalable HT detection.

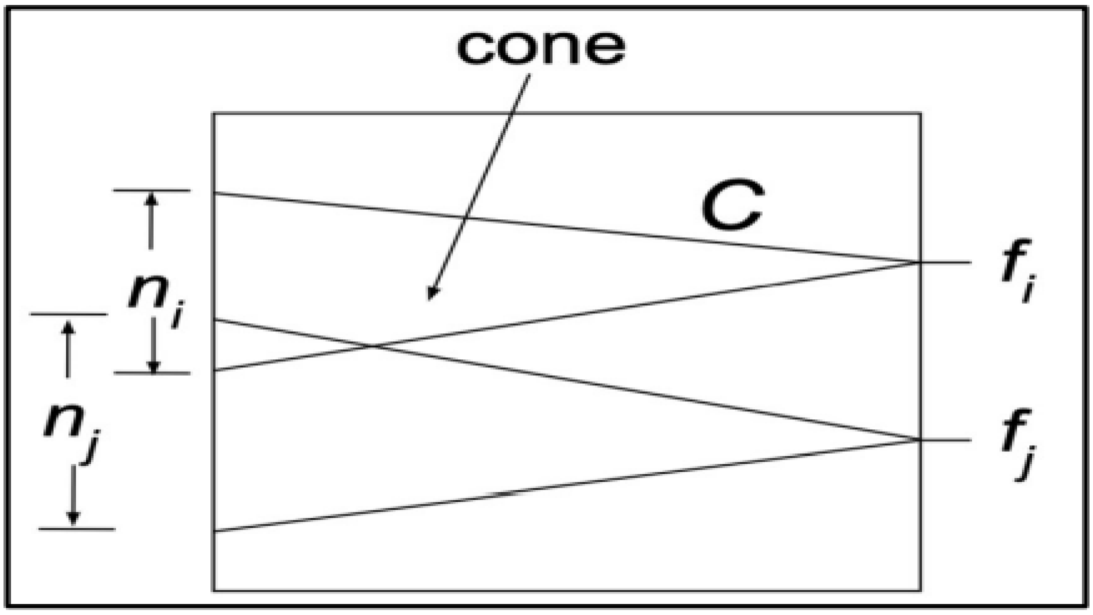


Fig.2 Cones of logic

In order to filtrate possibly-propagated transitions from a targeted subcircuit to other one’s, key-gates on wires that have high switching activity ratio and go form the targeted partition. In Fig.3 AND gates are inserted on Net2 and Net3 that connect subcircuits SC1 and SC2 such nets are called pseudo inputs or outputs (PSI and PSO).

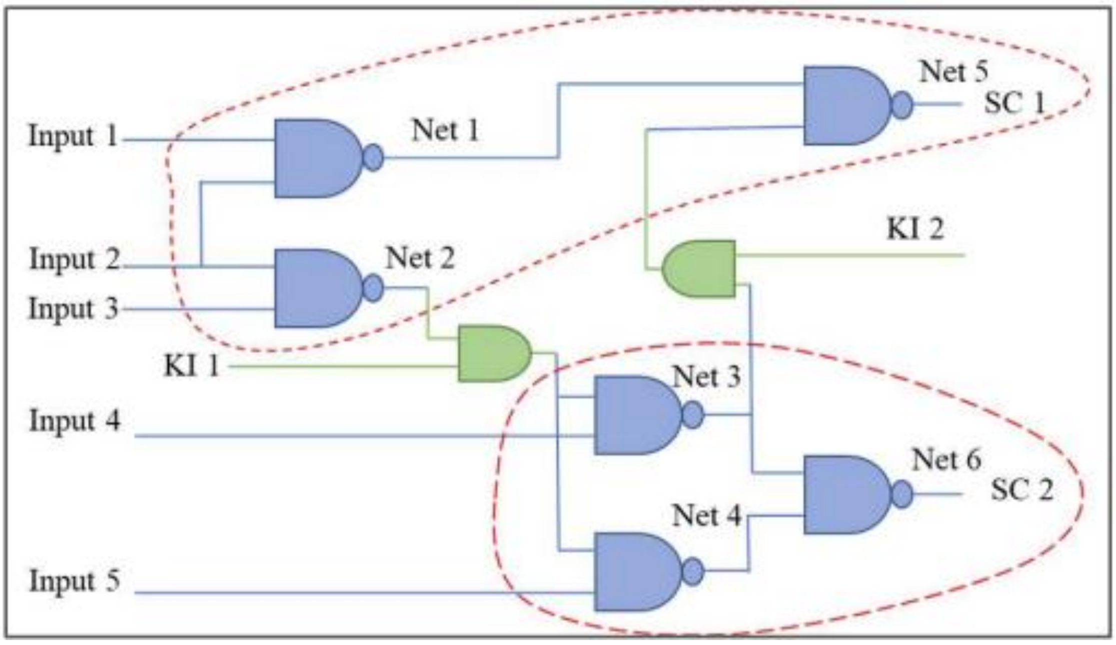


Fig.3 Partitioned ISCAS85 C17 circuit

In other words, PSI (PSO) of a partition is a net coming from (going to) a cell in other partitions. Inserting a AND/OR keygate on a PSI gives full controllability to this net. Hence, it is useful when a partition is the subject of SAL and especially if the partition does not have any primary input. Likewise, inserting an AND (NAND) / OR (NOR) key-gate on a PSO allows us to lock its output to '0(1)' or '1(0)' respectively, by applying the key inputs as 0 or 1 respectively. It does not let the switching activity of a targeted partition pervades to other partitions which must have low switching activity. SC1 with inputs 1-3 can be seen as one cone, and SC2 with inputs 4-5 as another. Where, net 2 and net 3 are those overlapping logic that we are trying to isolate. TABLE I shows the combinations of key inputs and circuit operation in Fig 3. For example, when K1=0 and K2=1 switching activity is locked to 0 for the NAND gate whereas, the switching activity from Net 3 is allowed to pass. When K1=K2=1 the circuit is in normal operation mode.

TABLE I. C17 modes of operation depending on K1 and K2

K1	K2	Mode
0	0	Logic Locked
0	1	SC1 Switching
1	0	SC2 Switching
1	1	Normal Operation

3.2 Vector Generation

To perform Switching Activity Localization (SAL) effectively, input vectors must be carefully generated and applied to the primary inputs of the circuit. These vectors are tailored to maximize the switching activity within the sub-circuit under test, while simultaneously minimizing activity in the rest of the design. This selective activation is critical for enhancing the contrast in power consumption, which is essential for detecting the presence of hardware Trojans (HTs) within a specific region of the circuit. Each sub-circuit requires a unique set of input vectors, as the logic and path sensitization differ across regions. Our proposed AND/OR-based logic locking technique supports this process by structurally constraining the circuit, thereby ensuring that switching activity remains localized during vector application. This confinement enhances the effectiveness of PCA by isolating the power signature of the targeted sub-circuit. The reference work employs the MERS (Multiple Excitation of Rare Switching)

[4] algorithm for vector generation. MERS is designed to repeatedly activate rare switching nodes—signals that seldom toggle during normal operation—which makes it particularly effective for exposing anomalous behavior introduced by HTs. By stimulating such nodes, the algorithm increases the chances of triggering hidden malicious logic.

In the broader context of design verification and test, vector generation for SAL shares objectives with Design-for-Test (DFT) methodologies. Industry-standard tools such as Tessent [5] by Siemens EDA (formerly Mentor Graphics) and TestKompress [5] by Synopsys already incorporate advanced automatic test pattern generation (ATPG) algorithms to maximize fault coverage and control internal switching. These tools often use fault simulation and observability-driven techniques to target rarely activated paths, which aligns well with the needs of HT detection through localized activity analysis. In summary, the generation of tailored input vectors, guided by SAL principles and supported by our locking strategy, is key to isolating HT-related power anomalies. Leveraging existing techniques like MERS and tools developed for DFT provides a strong foundation for scalable and automation-friendly HT detection.

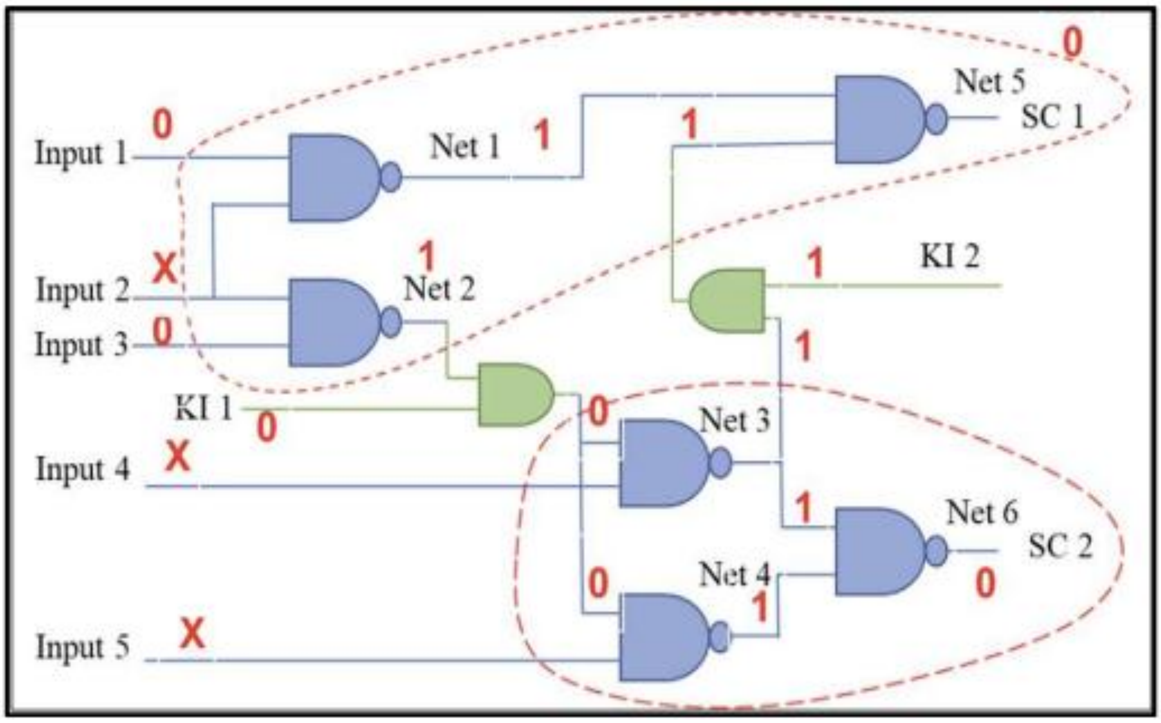


Fig.4 Logic Locked C17 circuit when V1 is applied

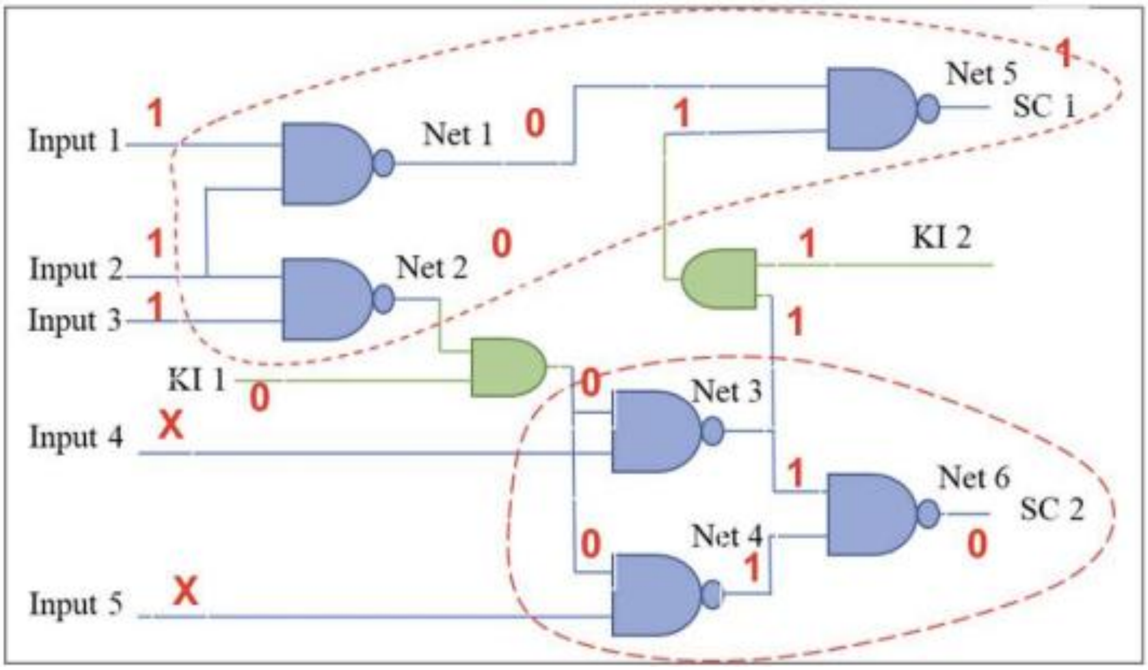


Fig.5 Logic Locked C17 circuit when V2 is applied

Here, V1 = {0, X, 0, X, X, KI1=0, KI2=1} and V2 = {1, 1, 1, X, X, KI1=0, KI2=1} were generated for c17 circuit shown in Fig 3. Fig 4 and Fig 5 show the effect of each vector respectively. It can be observed that the values for Net 3, Net 4 and Net 6 are not switching as SC2 has been isolated by K1 input and Net 1, Net 2 and Net 5 are switching as our vector pair targets SC1. Thus, we see that our vector pair minimizes the activity in the targeted sub-circuit under test SC1, while minimizing it in SC2.

3.3 Trojan Insertion

Fig. 6 illustrates the Trojan-infected C17 circuit, where the orange AND gates represent the inserted hardware Trojan (HT). In our approach, when the sub-circuit SC1 is targeted by the test vectors, the switching activity within this specific sub-circuit increases significantly due to the applied stimuli. This elevated switching activity directly translates into a measurable increase in power consumption localized to SC1. To detect the presence of the HT, we compare the power consumption profile of the Trojan-infected circuit against that of a clean, Trojan-free reference circuit under identical test conditions, specifically when SC1 is activated. The difference in power consumption between the two circuits provides a clear signature of the HT's presence, as the additional gates introduced by the Trojan contribute to higher switching activity and thus higher dynamic power in the infected circuit. This localized power analysis is made possible by our proposed logic locking and partitioning methods, which confine switching activity to the sub-circuit under test and minimize interference from other parts of the circuit.

By enhancing the signal-to-noise ratio in power consumption measurements, our method improves the sensitivity and reliability of HT detection, even for small Trojans that would otherwise be masked by process variations or noise in large circuits. This technique is particularly valuable for complex ICs where HTs represent a small fraction of the overall gate count but pose significant security threats.

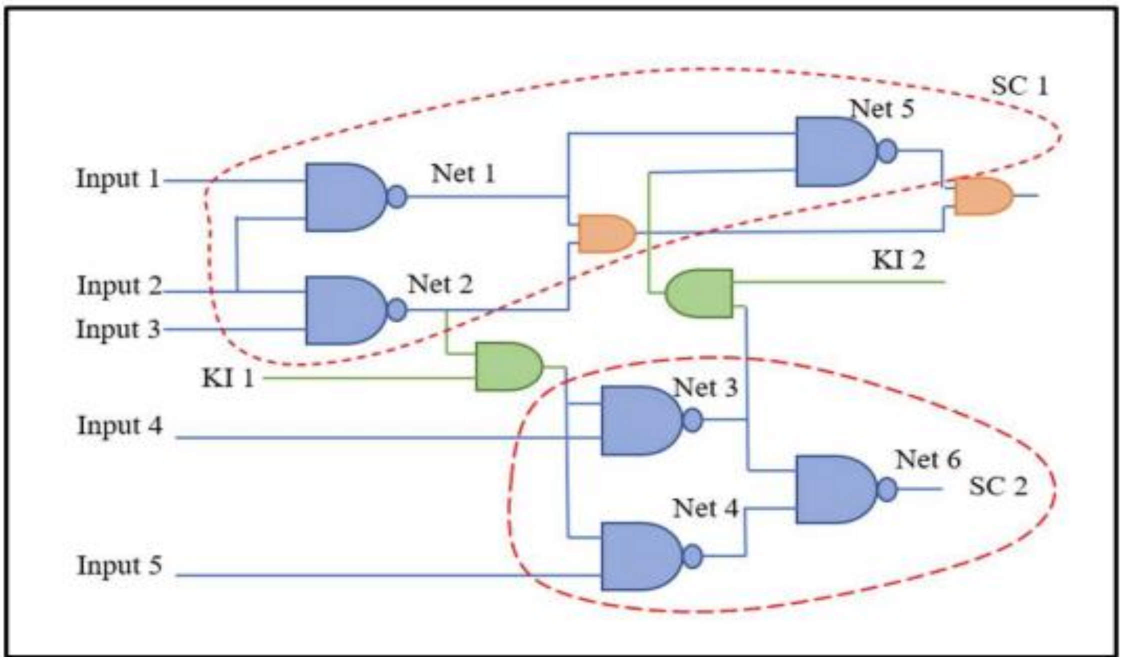


Fig.6 C17 logic locked circuit infected with HT

Increasing the number of partitions in the circuit generally leads to a greater observable power difference between the Trojan-infected and Trojan-free versions. This is because finer partitioning enables more precise localization of switching activity, thereby amplifying the relative power impact of any inserted hardware Trojan. However, this improvement comes with a tradeoff: a larger number of partitions requires the insertion of more key-gates, which can increase design complexity, area overhead, and potentially impact circuit performance.

Determining the optimal number of partitions is therefore critical to balancing detection accuracy with implementation cost. While the following section provides an analytical approximation for the number of partitions needed, the precise optimal value depends heavily on the specific circuit architecture, Trojan characteristics, and process variations. Consequently, these parameters must be validated and fine-tuned through extensive experimental evaluation and simulation.

3.4 Formal Approximation For Number Of Partitions In The Circuit

To approximate the number of partitions in the circuit we are assuming that the trojan introduced is concentrated in one part of the circuit and the test vector applied is such that each net is switched at least once. To detect the trojan we also assume that we need X% power difference in power between Trojan Free and Trojan Inserted circuit, The power generated would be highly dependent on technology node, and PVT variations. Now, assuming we have N gates in a circuit, H gates in a trojan and we need to create P partitions of the circuit. The trojan free subcircuit will have N/P total gates and trojan infected will have N/P + H. So, we have the below approximation.

$$X\% = \frac{\left[\frac{N}{P} + H\right] - \frac{N}{P}}{\frac{N}{P}} \tag{1}$$

$$X\% = \frac{P \times H}{N} \tag{2}$$

If we rearrange the terms for number of partitions we get,

$$P = \frac{N \times X}{100 \times H} \tag{3}$$

The above equation (3) shows us that the number of partitions is directly related to the number of gates in the circuit and power difference desired for detection. While it is inversely proportional to total gates in the inserted Trojan. The reference paper had also demonstrated experimentally that increasing partitions indeed increases the detection capability until a certain point after which it reaches a saturation point and further increase in number of partitions does not provide benefit.

4. Experimental Flow

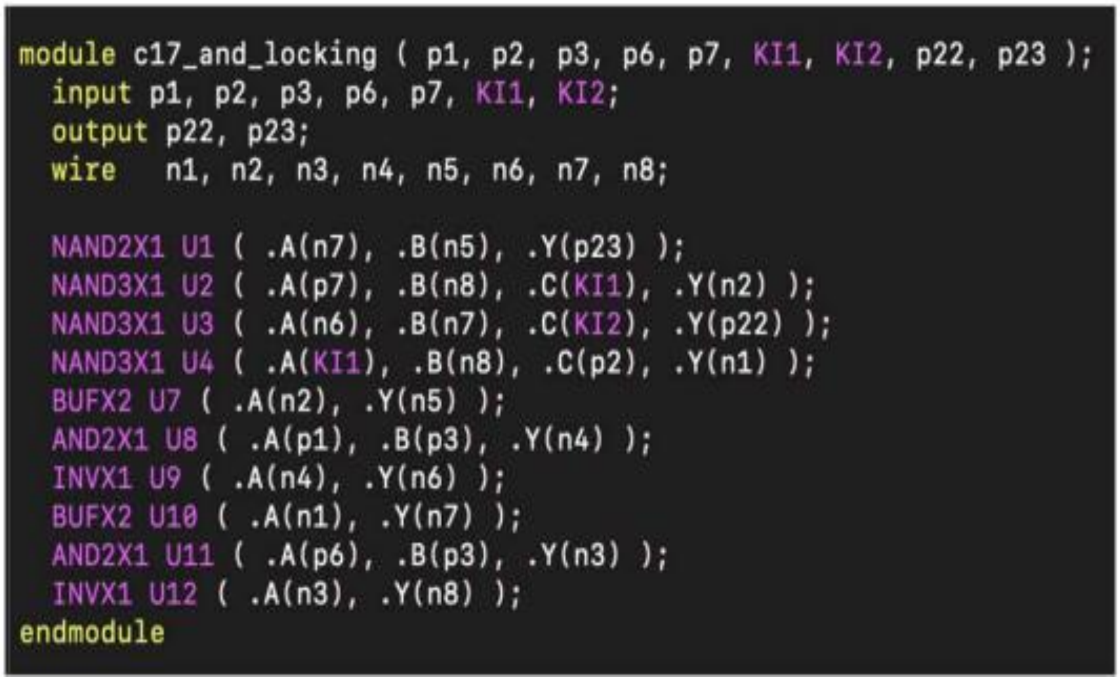
- **Implement AND/OR logic locking to the circuit based on partition subcircuits (Neos Tool and ABC tool):**
  Begin by partitioning the original circuit into multiple sub-circuits based on switching activity localization goals. Apply AND/OR logic locking by inserting key-gates strategically within each sub-circuit to maximize control over the switching behavior. Use the Neos tool [6] for initial logic transformations and the ABC tool [9] for optimization and synthesis to ensure the locked circuit meets design constraints without significant overhead.
- **Generate vectors for circuit based on partition (Manual):**
  Manually develop test vectors that specifically target each sub-circuit partition. The goal is to induce maximum switching activity in the sub-circuit under test while suppressing activity in other regions, thereby isolating power consumption changes to localized areas. This requires careful analysis of the circuit's functional paths and logic dependencies.
- **Synthesize the netlist (DC compiler):**
  Use Synopsys Design Compiler [7] to convert the RTL or locked netlist into a technology-specific gate-level netlist. The synthesis process includes logic optimization, technology mapping, and timing closure to produce a netlist suitable for detailed power and functional analysis.
- **Add HT at gate-level netlist generated (Manual):**
  Insert hardware Trojans manually into the synthesized netlist at the gate level, simulating real-world attack scenarios. The HT is designed to be stealthy, affecting only a small portion of the circuit and activating under rare conditions, making its detection challenging without localized analysis.
- **Validate the manual addition of HT using Logic Equivalence Check:**
  Perform a formal logic equivalence check between the original (Trojan-free) netlist and the HT-inserted netlist to ensure that the Trojan insertion does not unintentionally alter the overall functionality or introduce errors. This step confirms that the HT is logically transparent under normal operating conditions.

- **Simulate the netlist based on the vectors and generate the SAIF file using NC Sim for each circuit:**
  Conduct gate-level simulations using NC Simulator [5] with the previously generated input vectors. Capture switching activity during simulation and produce Switching Activity Interchange Format (SAIF) files, which provide detailed toggling information essential for accurate power estimation and dynamic analysis.
- **Perform dynamic power analysis for the circuits using Power Compiler to get power reports (part of DC compiler):**
  Leverage Synopsys Power Compiler [8] to analyze the SAIF files and calculate dynamic power consumption for both Trojan-infected and clean circuits. This analysis includes switching power, internal power, and leakage power estimates, providing a comprehensive profile for comparison.
- **Compare and analyze the report with the Trojan-free circuit to detect presence of Hardware Trojan:**
  Carefully compare the dynamic power reports of the Trojan-infected circuit against the reference circuit. Look for localized increases in power consumption that correlate with the activated sub-circuit and the presence of the HT. This comparison serves as the basis for detecting and confirming hardware Trojans through power analysis.

5. Technical Challenges

- **Manual Insertion of Locking Gates Due to Tool Limitations:**
  The Neos tool [6] was initially used to insert logic locking gates; however, it places them at random locations, which does not align with the requirements of our proposed method. For our technique to be effective—especially in enhancing switching activity localization—the key-gates must be inserted at carefully selected, high-impact locations within specific sub-circuits. This precise placement is crucial for maximizing the power difference during analysis. Due to this limitation, the insertion process had to be performed manually, which was both time-consuming and error-prone. As a result, we were constrained to work with smaller benchmark circuits, as applying the same manual process to larger circuits would not be scalable or practical within our timeframe.

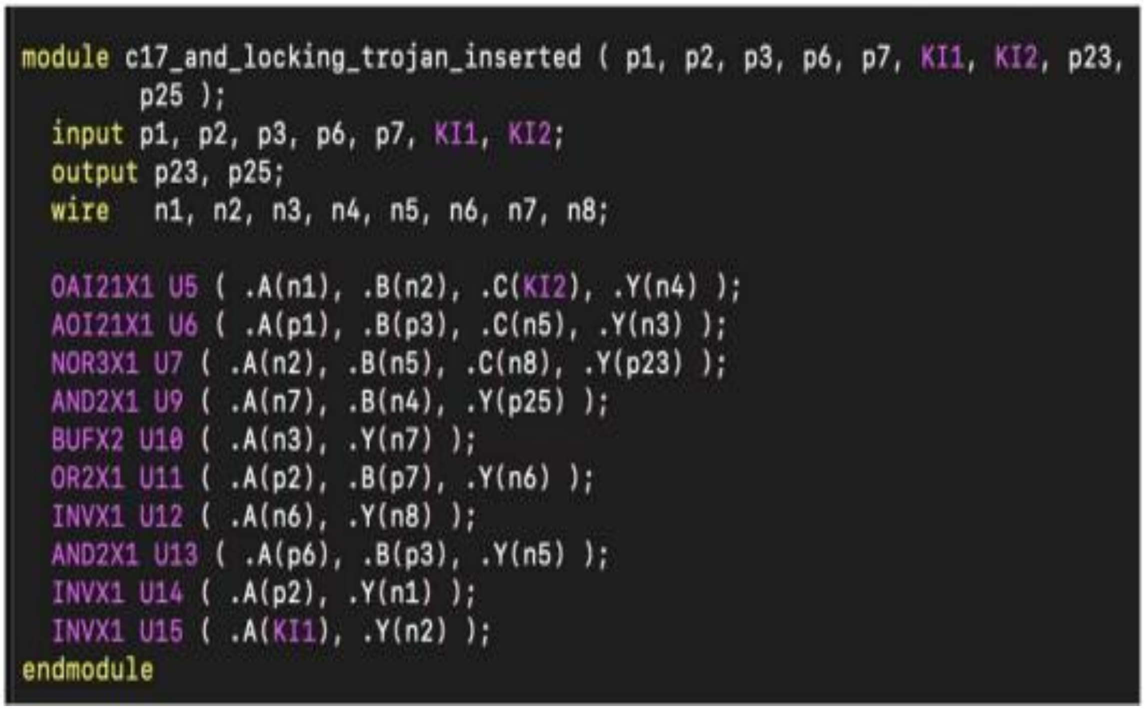
- **Lack of Open-Source Vector Generation Tools:**
  Another significant challenge we encountered was the absence of publicly available tools for generating test vectors that support our Switching Activity Localization (SAL) strategy. Most commercial ATPG (Automatic Test Pattern Generation) tools used in the industry—such as Tessent or TestKompress—are proprietary and not accessible in academic environments. Consequently, we had to generate the input vectors manually for each partitioned sub-circuit. This limitation, once again, restricted our ability to scale the experiment to larger, more complex circuits. Moreover, manual vector generation introduces subjectivity and increases the chances of missing edge-case behaviors necessary for effective HT detection.
- **Unintended Synthesis Optimization Masking Trojan Impact:**
  During the initial phases of our experiment, we encountered unexpected results where the power profiles of the Trojan-infected and Trojan-free circuits were nearly identical. After detailed debugging, we identified a critical flaw in our flow: the synthesis tool (Synopsys Design Compiler) [7] automatically optimizes the entire circuit during the synthesis process. As a result, when we added the Trojan at the RTL level, the tool minimized redundant or unused logic, effectively neutralizing the structural differences introduced by the HT. Both the clean and infected versions of the C17 circuit were synthesized down to 10 gates, as shown in Fig. 7 and Fig. 8. Since dynamic power is directly proportional to the number of gates switching during operation, this gate count similarity led to nearly identical power reports. We later realized that for accurate HT detection through power analysis, the Trojan must be added *after* synthesis—directly into the gate-level netlist—to preserve its physical and functional footprint. So, we started adding trojan in the synthesized gate level netlist instead of RTL level as shown in Fig 9.



```
module c17_and_locking ( p1, p2, p3, p6, p7, KI1, KI2, p22, p23 );
  input p1, p2, p3, p6, p7, KI1, KI2;
  output p22, p23;
  wire   n1, n2, n3, n4, n5, n6, n7, n8;

  NAND2X1 U1 ( .A(n7), .B(n5), .Y(p23) );
  NAND3X1 U2 ( .A(p7), .B(n8), .C(KI1), .Y(n2) );
  NAND3X1 U3 ( .A(n6), .B(n7), .C(KI2), .Y(p22) );
  NAND3X1 U4 ( .A(KI1), .B(n8), .C(p2), .Y(n1) );
  BUFX2 U7 ( .A(n2), .Y(n5) );
  AND2X1 U8 ( .A(p1), .B(p3), .Y(n4) );
  INVX1 U9 ( .A(n4), .Y(n6) );
  BUFX2 U10 ( .A(n1), .Y(n7) );
  AND2X1 U11 ( .A(p6), .B(p3), .Y(n3) );
  INVX1 U12 ( .A(n3), .Y(n8) );
endmodule
```

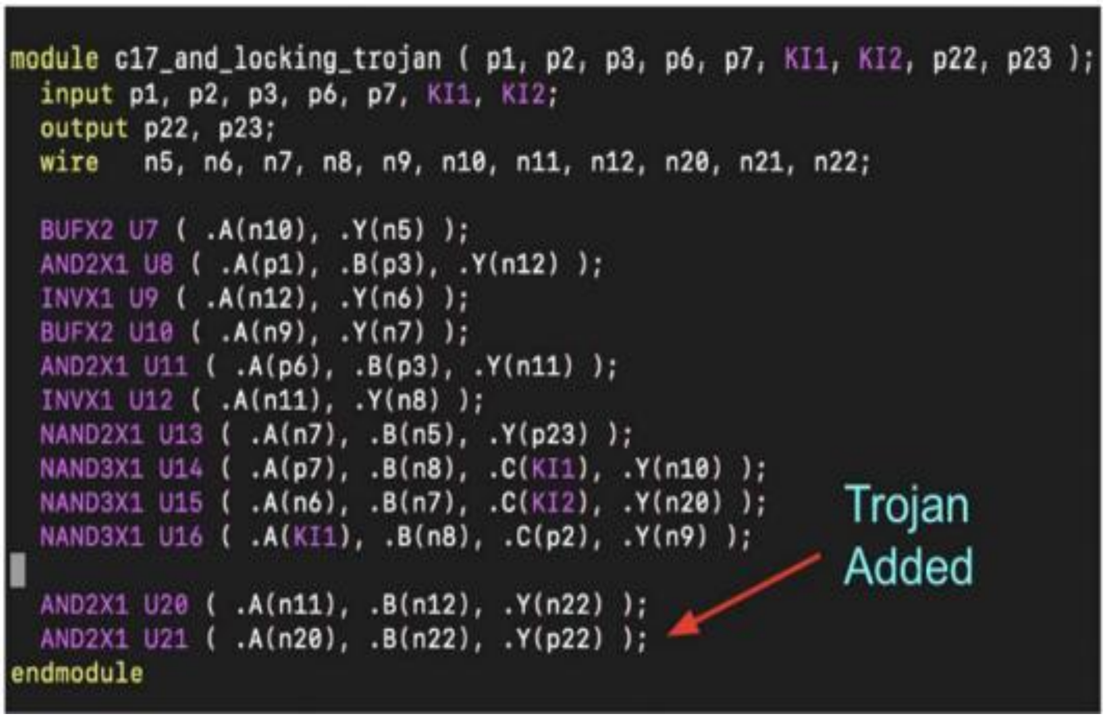
Fig. 7 Trojan Free circuit’s Synthesized netlist



```
module c17_and_locking_trojan_inserted ( p1, p2, p3, p6, p7, KI1, KI2, p23,
        p25 );
  input p1, p2, p3, p6, p7, KI1, KI2;
  output p23, p25;
  wire   n1, n2, n3, n4, n5, n6, n7, n8;

  OAI21X1 U5 ( .A(n1), .B(n2), .C(KI2), .Y(n4) );
  AOI21X1 U6 ( .A(p1), .B(p3), .C(n5), .Y(n3) );
  NOR3X1 U7 ( .A(n2), .B(n5), .C(n8), .Y(p23) );
  AND2X1 U9 ( .A(n7), .B(n4), .Y(p25) );
  BUFX2 U10 ( .A(n3), .Y(n7) );
  OR2X1 U11 ( .A(p2), .B(p7), .Y(n6) );
  INVX1 U12 ( .A(n6), .Y(n8) );
  AND2X1 U13 ( .A(p6), .B(p3), .Y(n5) );
  INVX1 U14 ( .A(p2), .Y(n1) );
  INVX1 U15 ( .A(KI1), .Y(n2) );
endmodule
```

Fig. 8 Trojan Inserted circuit’s Synthesized netlist



```
module c17_and_locking_trojan ( p1, p2, p3, p6, p7, KI1, KI2, p22, p23 );
  input p1, p2, p3, p6, p7, KI1, KI2;
  output p22, p23;
  wire   n5, n6, n7, n8, n9, n10, n11, n12, n20, n21, n22;

  BUFX2 U7 ( .A(n10), .Y(n5) );
  AND2X1 U8 ( .A(p1), .B(p3), .Y(n12) );
  INVX1 U9 ( .A(n12), .Y(n6) );
  BUFX2 U10 ( .A(n9), .Y(n7) );
  AND2X1 U11 ( .A(p6), .B(p3), .Y(n11) );
  INVX1 U12 ( .A(n11), .Y(n8) );
  NAND2X1 U13 ( .A(n7), .B(n5), .Y(p23) );
  NAND3X1 U14 ( .A(p7), .B(n8), .C(KI1), .Y(n10) );
  NAND3X1 U15 ( .A(n6), .B(n7), .C(KI2), .Y(n20) );
  NAND3X1 U16 ( .A(KI1), .B(n8), .C(p2), .Y(n9) );

  AND2X1 U20 ( .A(n11), .B(n12), .Y(n22) );
  AND2X1 U21 ( .A(n20), .B(n22), .Y(p22) );
endmodule
```

Fig 9. Trojan Added manually in the netlist

6. Experimental Results

To demonstrate the effectiveness of the proposed method, we implemented the described experimental flow on two benchmark circuits from the ISCAS suite: C17 and S27. These circuits were chosen due to their manageable complexity, which allowed for manual manipulation and detailed analysis. The designs were synthesized using the NCSU 45nm OpenCell Library, which provided the standard cell technology needed for both logic synthesis and power analysis. This technology node enabled realistic estimation of gate-level power consumption and switching activity, aligning our experimental setup with modern semiconductor design practices.

6.1 Results and Analysis for the C17 Circuit

TABLE II shows the experimental results for the c17 circuit. It shows the dynamic power for trojan free and trojan inserted circuit reported by the DC compiler for both locked and unlocked circuit. We can calculate the difference we can observe in a trojan free and trojan infected circuit. The table clearly shows the percentage difference in dynamic power using the PCA method for detection when using the proposed logic locking method. Thus, the chance of HT detection becomes higher as the difference in dynamic power would be much more apparent.

TABLE II. Results for C17 Circuit

	Dynamic Power (μW)	Power Difference (μW)	Dynamic Power Difference
Circuit	14.4990	1.2008	8.282 %
Circuit + Trojan	15.6998		
Circuit + AND locking	14.2905	9.4996	66.475 %
Circuit + AND locking + Trojan	23.7901		

6.2 Results and Analysis for the S27 Circuit

Similarly, TABLE III presents the experimental results for the ISCAS'89 S27 circuit, which, unlike the C17 benchmark, is sequential in nature and not purely combinational. This introduces additional complexity due to the presence of flip-flops and internal states, making Trojan detection inherently more challenging. Despite this, the proposed technique demonstrates its effectiveness in improving the likelihood of detecting hardware Trojans. The results show a noticeable difference in power consumption between the Trojan-infected and Trojan-free versions of the circuit, reinforcing the applicability of our approach to both combinational and sequential designs.

TABLE III. Results for S27 Circuit

	Dynamic Power (μW)	Power Difference (μW)	Dynamic Power Difference
Circuit	13.6648	1.2675	9.276 %
Circuit + Trojan	14.9323		
Circuit + AND locking	9.3473	3.6828	39.40 %
Circuit + AND locking + Trojan	13.0301		

6.3 Comparison with MUX based Logic Locking

The Mux based logic locking technique, proposed in the reference paper, was also implemented for the c17 circuit, as described in the paper [1], with our flow, and compared with our AND/OR technique. The partitions, vector, and place of trojan insertion was kept the same, only the and gates used to lock the circuit were replaced with MUX's. The results for the same are described below. From TABLE IV, we observe that the dynamic power difference between the Trojan-free and Trojan-infected circuits is significantly greater when using the AND/OR-based logic locking technique compared to the traditional MUX-based locking method. This indicates that AND/OR locking is more effective in amplifying the switching activity localized to the sub-circuit under test, thereby making the presence of a hardware Trojan more distinguishable through power analysis. The increased sensitivity to power variations enhances the detectability of small-scale Trojans that might otherwise remain hidden in large circuits. This result highlights the advantage of our proposed locking scheme in improving the reliability and accuracy of PCA-based Trojan detection methods.

TABLE IV. Comparison of power for C17 with MUX vs AND locking

	Dynamic Power (μW)	Power Difference (μW)	Dynamic Power Difference
AND locking	14.2905	9.4996	66.475 %
AND locking + Trojan	23.7901		
MUX locking	11.5	3.1708	27.572 %
MUX locking + Trojan	14.6708		

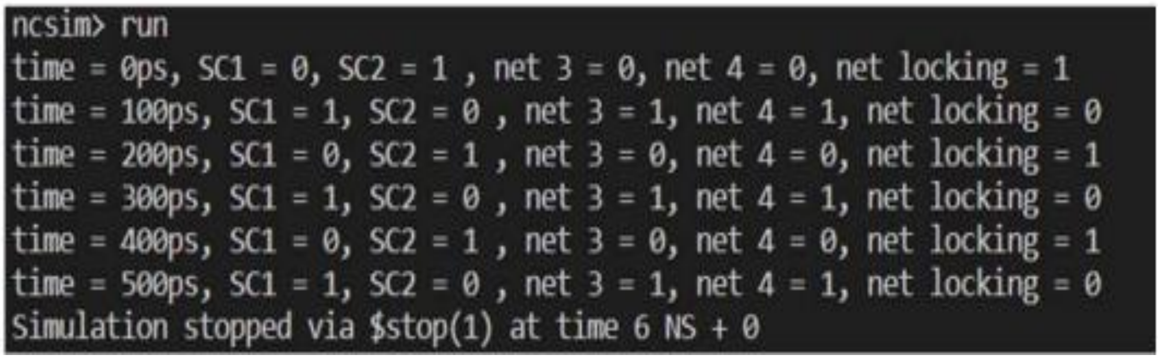
We can also observe from TABLE V that the AND-based logic locking used in our proposed method results in lower area overhead compared to the traditional MUX-based locking technique. This observation is supported by our area analysis performed post-synthesis, where the gate count and overall cell usage were evaluated using standard cell mapping under the NCSU 45nm technology library. The AND/OR gates, being simpler in structure and requiring fewer transistors than MUX elements, contribute to a more compact and resource-efficient implementation. This reduction in area is especially beneficial for cost-sensitive or resource-constrained applications, where minimizing silicon real estate is crucial. Thus, in addition to offering better Trojan detection capability through enhanced power signature localization, the AND/OR-based logic locking technique also proves to be more area-efficient. This makes it a superior alternative to MUX-based locking in terms of both security and design resource optimization.

TABLE V. Comparison of area for C17 with MUX vs AND locking

Circuit Type	Area (µm sq)
AND locking	21.1185
MUX locking	25.8115

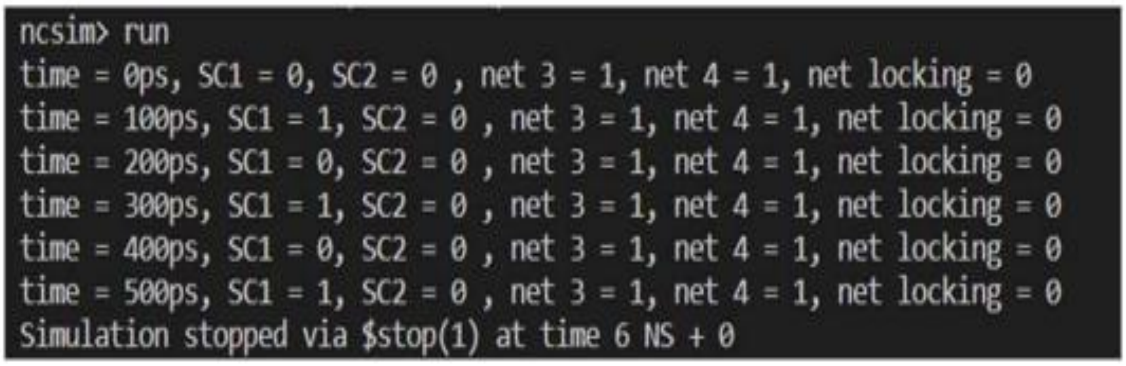
6.4 Comparison with XOR based Logic Locking

We also experimented with the XOR-based logic locking technique as part of our proposed method to evaluate its effectiveness in localizing switching activity. However, as illustrated in Fig. 10 and Fig. 11, this approach did not perform as expected. In the case of the C17 circuit, even when only SC1 was intended to be active (i.e., SC2 was locked), we observed unintended switching in other parts of the circuit—specifically at SC2, net3, and net4. This unintended switching indicates that XOR-based locking lacks the isolation capability required for effective Switching Activity Localization (SAL). The propagation of switching activity across multiple sub-circuits not only dilutes the power signature but also introduces noise, making it more difficult to detect hardware Trojans based on localized power anomalies. Furthermore, XOR gates introduce symmetrical logic behavior, which may not effectively block transitions under incorrect key values, reducing their usefulness in enforcing strict partition-level control. Therefore, while XOR-based locking remains popular in traditional logic obfuscation schemes, our results demonstrate that it is less suitable for applications requiring high granularity in power-based HT detection, especially when compared to the more targeted behavior of AND/OR-based locking. Thus, even though XOR based logic locking is preferred for HT prevention it would not be useful for HT detection.



```
ncsim> run
time = 0ps, SC1 = 0, SC2 = 1 , net 3 = 0, net 4 = 0, net locking = 1
time = 100ps, SC1 = 1, SC2 = 0 , net 3 = 1, net 4 = 1, net locking = 0
time = 200ps, SC1 = 0, SC2 = 1 , net 3 = 0, net 4 = 0, net locking = 1
time = 300ps, SC1 = 1, SC2 = 0 , net 3 = 1, net 4 = 1, net locking = 0
time = 400ps, SC1 = 0, SC2 = 1 , net 3 = 0, net 4 = 0, net locking = 1
time = 500ps, SC1 = 1, SC2 = 0 , net 3 = 1, net 4 = 1, net locking = 0
Simulation stopped via $stop(1) at time 6 NS + 0
```

Fig. 10 XOR locking simulation



```
ncsim> run
time = 0ps, SC1 = 0, SC2 = 0 , net 3 = 1, net 4 = 1, net locking = 0
time = 100ps, SC1 = 1, SC2 = 0 , net 3 = 1, net 4 = 1, net locking = 0
time = 200ps, SC1 = 0, SC2 = 0 , net 3 = 1, net 4 = 1, net locking = 0
time = 300ps, SC1 = 1, SC2 = 0 , net 3 = 1, net 4 = 1, net locking = 0
time = 400ps, SC1 = 0, SC2 = 0 , net 3 = 1, net 4 = 1, net locking = 0
time = 500ps, SC1 = 1, SC2 = 0 , net 3 = 1, net 4 = 1, net locking = 0
Simulation stopped via $stop(1) at time 6 NS + 0
```

Fig. 11 AND locking simulation

7. Future Work

- **Automation of Manual Steps for Scalability:**
  Several steps—including logic locking gate insertion and input vector generation—had to be carried out manually due to the lack of available open-source or customizable tools. Automating these steps using custom scripts or integrating with existing EDA tools would significantly improve the scalability of the methodology. This automation would enable the application of the proposed technique on larger and more complex benchmark circuits, helping to further validate its effectiveness in real-world scenarios.
- **Experimental Study on Partition Count vs. Detection Probability:**
  Further research is needed to experimentally analyze how the number of partitions affects the probability of hardware Trojan detection. As partition count increases, switching activity can be more localized, potentially improving detection resolution. However, it also increases area and control complexity. An empirical study across

various circuit sizes and types could help determine optimal partitioning strategies that balance detection accuracy, design overhead, and implementation feasibility.

- **Integrating XOR-Based Locking for Enhanced Security:**
  Although XOR-based locking showed limited ability to localize switching activity, it remains one of the more secure logic locking schemes due to its resistance to certain SAT and removal attacks. Future work could explore hybrid locking approaches that combine XOR-based logic with AND/OR locking. Such integration could potentially offer the benefits of both robust IP protection (prevention) and power-based Trojan detection, resulting in a comprehensive hardware security solution.
- **Leveraging Design-for-Test (DFT) Techniques for Integration:**
  As previously discussed, many aspects of our method, such as vector generation and switching activity analysis, are conceptually aligned with Design-for-Test (DFT) practices. A more in-depth study of existing DFT methodologies and tools—such as scan chain insertion, ATPG, and built-in self-test (BIST)—could pave the way for seamless integration of our approach into the standard VLSI design and validation flow. This would help facilitate adoption in industrial settings without requiring significant changes to existing workflows.
- **Power Side-Channel Analysis in Physical Silicon:**
  While our experiments are conducted at the simulation and synthesis level, future work could involve implementing the proposed method on FPGA or silicon prototypes to evaluate its effectiveness under real-world conditions. Performing side-channel power measurements on physical hardware would provide insights into noise, thermal effects, and environmental variations—factors often not captured in gate-level simulation.
- **Tool Development for Academic Use:**
  Finally, the creation of a modular, open-source toolchain that incorporates logic locking, partitioning, vector generation, and power analysis tailored for HT detection would greatly benefit the academic and research community. Such a tool could help standardize research in hardware security and provide a platform for collaborative benchmarking and innovation.

8. References

[1] A. Nejat, Z. Kazemi, V. Beroulle, D. Hely and M. Fazeli, "Restricting Switching Activity Using Logic Locking to Improve Power Analysis-Based Trojan Detection," 2019 IEEE 4th International Verification and Security Workshop (IVSW), 2019, pp. 49-54, doi: 10.1109/IVSW.2019.8854402.

[2] Kamali, Hadi Mardani, Kimia Zamiri Azar, Farimah Farahmandi, and Mark Tehranipoor. "Advances in Logic Locking: Past, Present, and Prospects." Cryptology ePrint Archive (2022)

[3] Hendrickson, Bruce, and Robert W. Leland. "A Multi-Level Algorithm For Partitioning Graphs." SC 95, no. 28 (1995): 1-14.

[4] Huang, Yuanwen, Swarup Bhunia, and Prabhat Mishra. "MERS: statistical test generation for side-channel analysis based Trojan detection." In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, pp. 130-141. 2016.

[5] Cadence NCsim user Guide

[6] Neos, [Online]: https://bitbucket.org/kavehshm/neos/src/master/

[7] Synopsys design Compiler, [Online] : https://www.synopsys.com/implementation-and-signoff/rtl-synthesis-test.html
http://www.synopsys.com/Tools/Implementation/RTLSynthesis/DesignCompiler/Pages/default.aspx

[8] SynopsysPowerCompiler, [Online]: https://www.synopsys.com/implementation-and-signoff/rtl-synthesis-test/power-compiler.html

[9] ABC tool, [Online] https://people.eecs.berkeley.edu/~alanmi/abc/

[10] Cadence Design Systems, *"Incisive® Enterprise Simulator User Guide,"* Product Version 15.2, 2015. https://support.cadence.com

Citation: Santosh Appachu Devanira Poovaiah. (2023). Hardware Trojan Detectability Enhancement Using Logic Locking and Power Discrepancy Analysis. International Journal of Computer Engineering and Technology (IJCET), 14(1), 167-189.

Abstract Link: https://iaeme.com/Home/article_id/IJCET_14_01_015

Article Link:
https://iaeme.com/MasterAdmin/Journal_uploads/IJCET/VOLUME_14_ISSUE_1/IJCET_14_01_015.pdf