

Partition based Approaches for the Isolation and Detection of Embedded Trojans in ICs

Mainak Banga

Thesis submitted to the Faculty of
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Engineering

Dr. Michael S. Hsiao, Chair

Dr. Sandeep K. Shukla

Dr. Chao Huang

September 1, 2008

Blacksburg, Virginia

Keywords: State-space, Trojans, Power profile, Side-Channel Analysis

Copyright © 2008, Mainak Banga

Partition based Approaches for the Isolation and Detection of Embedded Trojans in ICs

Mainak Banga

ABSTRACT

This thesis aims towards devising a non-destructive testing methodology for ICs fabricated by a third party manufacturer to ensure the integrity of the chip. With the growing trend of outsourcing, the sanity of the final product has emerged to be a prime concern for the end user. This is especially so if the components are to be used in mission-critical applications such as space-exploration, medical diagnosis and treatment, defense equipments such as missiles etc., where a single failure can lead to a disaster. Thus, any extraneous parts (Trojans) that might have been implanted by the third party manufacturer with a malicious intent during the fabrication process must be diagnosed before the component is put to use.

The inherent stealthy nature of Trojans makes it difficult to detect them at normal IC outputs. More so, with the restriction that one cannot visually inspect the internals of an IC after it has been manufactured. This obviates the use of side-channel signal(s) that acts like a signature of the IC as a means to assess its internal behavior under operational conditions.

In this work, we have selected power as the side-channel signal to characterize the internal behavior of the ICs. We have used two circuit partitioning based approaches for isolating and enhancing the behavioral difference between parts of a genuine IC and one with a sequence detector Trojan in it. Experimental results reveal that these approaches are effective in exposing anomalous behavior between the targeted ICs. This is reflected as difference in power-profiles of the genuine and maligned ICs that is magnified above the process variation ensuring that the discrepancies are observable.

To my Family

Acknowledgments

It gives me immense pleasure to thank my advisor Dr. Michael S. Hsiao for his continued guidance and valuable suggestions throughout the duration of my research. His insight in my field of research enabled me to comprehend and approach the target problem in a very methodical way. His continuous encouragement always inspired me to strive for the best.

I would also like to thank Dr. Sandeep Shukla and Dr. Chao Huang for their consent of being a part of my Masters Committee and valuable suggestions to improve the thesis composition. I am thankful to Sanjay Sengupta and Surendra K. Bommur for providing me an opportunity of internship at Intel in the related area of ATPG.

I am thankful to my friends in the PROACTIVE Lab. viz. Shirang Yardi, Maheshwar Chandrasekar, Vishnu Vimjam, Lei Fang, Ankur Parikh, Karthik Channakeshava, Bin Li, Weixin Wu, Xueqi Cheng, Nannan He, Anupam Shrivastava, Harini Jagdeeshan, Patrick Cowhig, Huy Lam and Ravindran Ramnathan for their timely suggestions in my research work and constructive amendments that helped in shaping up the solution in a more complete way.

I am grateful to my roommate Sumit Ahuja for his encouragement and support. Finally, I would like to extend my heartiest thanks to my parents and all my relatives and friends whose good wishes always motivated me towards my endeavor.

Mainak Banga

1st September 2008.

Contents

1	Introduction	1
2	Background	7
2.1	Hardware Trojan	7
2.1.1	Classification of Trojans	8
2.1.2	Trojan Characteristics	13
2.1.3	Trojan Detection Challenges	14
2.2	Side Channel Analysis	17
2.3	Power Profile	18
2.4	Hamming Distance	19
3	State-space Partition with Hamming Distance Maximization	20
3.1	Motivation	20
3.2	Our Approach	21
3.2.1	Circuit Partitioning	21
3.2.2	Activity Magnification	23

3.2.3	Implementation	25
3.3	Experimental Results	27
3.3.1	Circuit partitioning Results	27
3.3.2	Activity Magnification Results	29
3.4	Summary	32
4	Region based Partition with Toggle Count Maximization	35
4.1	Motivation	35
4.2	Our Approach	36
4.2.1	Region Based Partitioning	36
4.2.2	Relative Toggle Count Magnification	39
4.2.3	Implementation	40
4.3	Experimental Results	43
4.3.1	s444	43
4.3.2	s1196	44
4.3.3	s1423	46
4.3.4	s3271	47
4.4	Summary	48
5	Conclusion & Future Work	50
	Bibliography	52

List of Figures

2.1	Taxonomy of Trojans	8
2.2	Physical Characteristic of Trojans	9
2.3	Activation Characteristic of Trojans	11
2.4	Action Characteristic of Trojans	12
2.5	A Trojan circuit and its FSM	16
3.1	Overall flow of the Trojan identification process	26
3.2	Relative increase in Trojan circuit activity by our approach vs. the random approach for s1196	29
3.3	Relative increase in Trojan circuit activity by our approach vs. the random approach for s3330	30
3.4	Relative increase in Trojan circuit activity by our approach vs. the random approach for s5378	30
3.5	Relative increase in Trojan circuit activity by our approach vs. the random approach for s38584	31
3.6	Ratio of relative magnification of Trojan circuit activity over the actual circuit activity for different circuits	31

3.7	Ratio of relative magnification of Trojan circuit activity over the actual circuit activity for different circuits	32
3.8	Activity Magnification for s1196, (group 2) between our approach vs. random approach	33
3.9	Activity Magnification for s5378, (group 5) between our approach vs. random approach	33
3.10	Activity Magnification for s15850, (group 27) between our approach vs. random approach	34
4.1	Illustration of the concept of <i>region</i> and <i>radius</i> in a circuit	38
4.2	TCM(Radius 2, flip-flop Count 3) for s444	44
4.3	TCM(Radius 2, flip-flop Count 4) for s444	45
4.4	TCM(Radius 3, flip-flop Count 3) for s444	45
4.5	TCM(Radius 3, flip-flop Count 2) for s1196	46
4.6	TCM(Radius 4, flip-flop Count 2) for s1196	46
4.7	TCM(Radius 4, flip-flop Count 5) for s1423	47
4.8	TCM(Radius 3, flip-flop Count 3) for s3271	48
4.9	TCM(Radius 4, flip-flop Count 5) for s3271	49

List of Tables

3.1	Table showing the selection criteria for input vectors	24
3.2	Circuit Partitioning Statistics	27
4.1	Functions of Algorithm 1	42

List of Abbreviations

IC	Integrated Circuit
SoC	System On Chip
MB	Megabyte
MSB	Most significant Bit
LSB	Least significant Bit
R&D	Research and Development
IP	Intellectual Property
PUF	Physically Unclonable Functions
ATPG	Automatic Test Pattern Generation
FSM	Finite State Machine
LFSR	Linear Feedback Shift Register
BIST	Built-In Self-Test
CUT	Circuit Under Test
I/O	Input-Output

Chapter 1

Introduction

Since its first inception four decades ago, *Moore's Law* has been the driving factor behind the growth of the modern VLSI industry. Although Moore's Law was initially coined as an observation and forecast, with the passage of time it has become a goal for an entire industry. This drove both marketing and engineering departments of semiconductor manufacturers to focus enormous effort and energy aiming for the specified increase in processing power that it was presumed one or more of their competitors would soon actually attain. In this regard, it can be viewed as a self-fulfilling prophecy [32].

Modern semiconductor industry trends show two major perspectives for the transistors viz. shrinking size and increasing speed. A direct implication of *Moore's law* is the rapid improvement in computing performance per unit cost, because increase in transistor count is also a rough measure of improvement of computer processing performance. Stated alternatively, along with performance, cost per component is one of the major factors that determine the success of a semiconductor industry. To remain competitive in the market, the producer strives to deliver the ICs at the lowest possible cost. Since the complexity of the state-of-art System-on-Chips (SoCs) have been steadily increasing to meet the high end requirements of the customer, this directly implies that the expenditure incurred on R&D, design, verification, fabrication, testing and validation of such SoCs have increased considerably as

well.

While decreasing feature size allows etching out more components within the same wafer, this is not a major reparation factor. Instead, IC producers rely on high volume production to compensate for the increased design and manufacturing costs. The increasing use of embedded systems in our daily life creates the demand for large number of ICs that drive these systems. Hence, the IC producer aims to sell larger and larger quantities of ICs at a reasonable cost to remain profitable - the process termed as “outsourcing” in the contemporary industry jargon.

Outsourcing is the process of employing a third party to accomplish the desired work [32]. Factors like the work competencies along with optimum utilization of land, labor, capital, technology and resources play a major role in defining the framework of an outsourced process. Outsourcing came into existence around 1980’s. Outsourcing can involve a part or whole of the process to be managed by the third party. Socioeconomic factors, cultural differences, economic stability of the country and existing organizational behavior also play a major role in outsourcing.

There are multiple reasons for outsourcing a process. Organizations that outsource a part or whole of a process seek to reap benefits or address the following issues:

- * Cost Reduction - Reducing the overall cost of the service to the business. This involves reducing the scope, defining quality levels, re-pricing, re-negotiation.
- * Cost Restructuring - Operating leverage is a measure that compares fixed costs to variable costs. Outsourcing changes the balance of this ratio by offering a move from variable to fixed cost and also by making variable costs more predictable.
- * Improve Quality - Achieve a step change in quality.
- * Knowledge - Access to wider experience and knowledge.
- * Operational Expertise - Access to operational best practice that would be too difficult or time consuming to develop in-house.

- * Staffing - Access to a larger talent pool and a sustainable source of skills.
- * Capacity Management - An improved method of capacity management of services and technology where the risk in providing the excess capacity is borne by the supplier.
- * Reduced time to market - The acceleration of the development or production of a product through the additional capability brought by the supplier.
- * Modularization of business structure - The trend of standardizing business processes, IT Services and application services enabling businesses to intelligently buy at the right price. This allows a wide range of businesses access to services previously only available to large corporations.

High volume production demands a substantial work force cost required to operate the fabrication units. This factor has urged IC producers to explore the option of searching skilled labor at a reasonable expense so that the profit margin on the finished product can be maximized. Consequently, they have set up off-shore fabrication units in countries with emerging economies where trained manpower is readily available at a judicious price.

At a glance, outsourcing seems to be an obvious and lucrative solution for IC producers to ensure their competitive edge in the semiconductor market. However, a closer look on the process reveals several disadvantages. To begin with, it creates a dependency of the IC producer on the third party manufacturer. Since the manufacturing unit is located overseas, direct vigilance of the fabrication processes and practices are not feasible. Therefore, the producer has to rely on the adeptness of the manufacturer to meet the deadlines and quality requirements for the end product. Any deviation or delay in the production schedule directly affects the time to market for the components.

Secondly, there is always a probability of Intellectual Property (IP) Rights violation. The manufacturer can reverse engineer the designs under production to fabricate and sell the same parts at lower cost. This creates a direct impact on the net consumption of the product in the target market thereby reducing the profit margin. Social and economic factors,

organizational behavior also play a major role in the success of an outsourced process.

The third and most critical issue with outsourcing is that of security and trust. A manufacturer with an malicious intention can make small alternations in the design before manufacturing it that can result in drastic consequences. While a faulty product is likely to fail, making the device in which it is used to be nonoperational, a tampered design can make the device act to the contrary. This thought has raised the question of reliability on the finished products imported from the fabrication centers. Design errors and manufacturing defects are detectable at the post-silicon validation process but intentional tampering can be almost impossible to detect using established techniques. Moreover, as the process geometries are shrinking down the nanometer scale, leakage and process variation continue to increase making the detection even more difficult.

Methods have been proposed to ensure security and verify genuineness of manufactured ICs. Of these, the most common are - *Watermarking*, *PUFs (Physically Unclonable Functions)*, *Scan-chain encryption* and *Security Engineering* are most common. *Watermarking* is a technique in which information is embedded directly and imperceptibly into digital data (e.g., image, video, or audio signals), also called host data, to form watermarked data. Applications of digital watermarking include copyright protection, distribution tracing, authentication, and authorized access control [28]. Attacks on watermarked systems can be removal attacks, geometric attacks, cryptographic attacks and protocol attacks. A mechanism to detect Watermarking based on error control codes is described in [29]. While testing and diagnosis is intended to uncover the on-chip faults and defects, *Security Engineering* and *Scan-Chain Encryption* are methods to prevent access to the original system. A way to analyze the security of the underlying scan architecture has been proposed in [31]. In this work scan-chain scrambling technique has been used to improve the security of the protected device.

Recently, PUF based structures have been proposed [22, 23] to characterize individual IC security key. Physical unclonability is very hard because exact control over the manufacturing

process is very difficult. Different sources of physical randomness can be used in PUFs, the most prominent being the process variations present under the manufacturing conditions. The inputs fed to the PUFs are termed as challenges and the outputs that they produce are termed as responses. PUFs inherit their unclonability property from the fact that every PUF has a unique and unpredictable way of mapping challenges to responses. Two PUFs that were manufactured with the same process will still possess a unique challenge-response behavior. While PUF based schemes are very effective in preventing external attacks to extract out the internal information from the ICs, Trojan attacks are on-chip intrusions and hence require a different approach.

So it becomes a problem of paramount importance to ensure the sanity of the finished product lot before embedding the components into a real application. Otherwise, in case of an operational failure consequences can be disastrous. Since the option of overseas manufacturing is unlikely to be eliminated from contemporary IC production road map, devising effective non-destructive techniques to distinguish the tampered chips from the genuine ones is a priority call. In this thesis, we have discussed this problem in detail and provided some innovative approaches to address the issue.

Contributions of this thesis:

This thesis outlines ways to characterize ICs based on their side-channel signal behavior specifically the power profile. In the process, we develop non-destructive testing methodologies to diagnose undesirable implantations in the fabricated ICs. The effectiveness of the diagnosis depends on the extent of difference that can be created and observed between the devices under test. Hence the **first contribution** of our approaches is the magnification such difference and project it to a level where it is unlikely to be suppressed by process variation. This thesis presents two separate partition-based approaches that prove to be successful in realizing this objective. The experimental results show that the proposed methodologies produce variations much higher than those produced using existing approaches. The **second contribution** of this thesis is on the diagnosis of the specific locations within the fabricated

chip that the embedded Trojan may reside. This is valuable information for post-silicon diagnosis. Existing approaches are not helpful in indicating such locations.

Organization of this thesis:

The rest of the thesis is organized as follows:

- Chapter 2: This chapter presents a detailed classification of Trojans based on different parameters. It illustrates the terms and concepts used throughout the rest of the text along with the previous related works in the field.
- Chapter 3: This chapter describes our first partition-based approach to solve the problem.
- Chapter 4: This chapter describes our second partition-based approach to solve the problem.
- Chapter 5: This chapter concludes the work, discusses the exceptional scenarios where the proposed algorithms could not produce desired results, reasons for such behaviors and future enhancements that can add value to the proposed solution.

Chapter 2

Background

This chapter details the concepts and theories that we have used to formulate our theoretical analysis and explain the results obtained. We have defined the specific terms which will be used throughout the rest of the thesis. A discussion on the relevant previous works and their implications on our approach is included in the appropriate sections as well.

2.1 Hardware Trojan

The tiny circuits implanted in the original design to make it work contrary to the expected way in certain rare and critical situations are called as *Trojans*. It is usually an aggregate of few gate(s) or even some wire(s) connected in an intelligent way to realize the desired behavior. In its most basic structure, Trojans are mainly categorized as:

- **Combinational Trojan:** A combinational circuit that becomes active when a specific condition arises in the internal signals and/or circuit flip-flops or a portion of it.
- **Sequential Trojan:** An FSM that monitors a portion of the internal circuit signals and triggers the output upon the occurrence of specific sequence(s).

2.1.1 Classification of Trojans

Apart from the broad classification stated earlier, depending on the point of view, Trojans are classified in many different ways. A nice explanation of such a hierarchical distribution has been presented in [18] and a part of it is reproduced here for a quick overview. As shown in Fig 2.1, Trojans can be distinguished based on their *physical characteristics*, *activation characteristics* or *action characteristics*. *Physical characteristics* can be further based on *type*, *size*, *distribution* and *structure*.

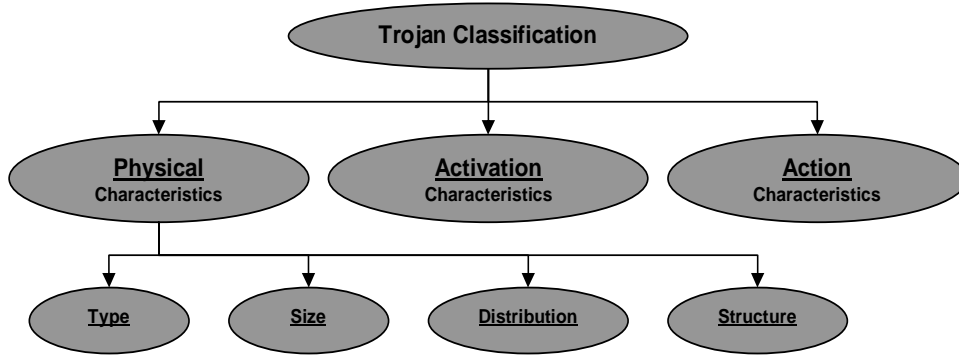


Figure 2.1: Taxonomy of Trojans

I. Classification based on Physical Characteristics

The detailed classification of Trojan based on its physical characteristics has been shown in Fig 2.2.

- **Type** : There are two sub categories in *type* section. The *functional type* of Trojan includes Trojans that are framed by manipulating the original structure of the design. This includes addition or deletion of gates from the circuit. *Parametric type* of Trojan involves modifications of existing wires and logic. This can include thinning of interconnect wires, the weakening of transistor strength by varying the length-width ratio.

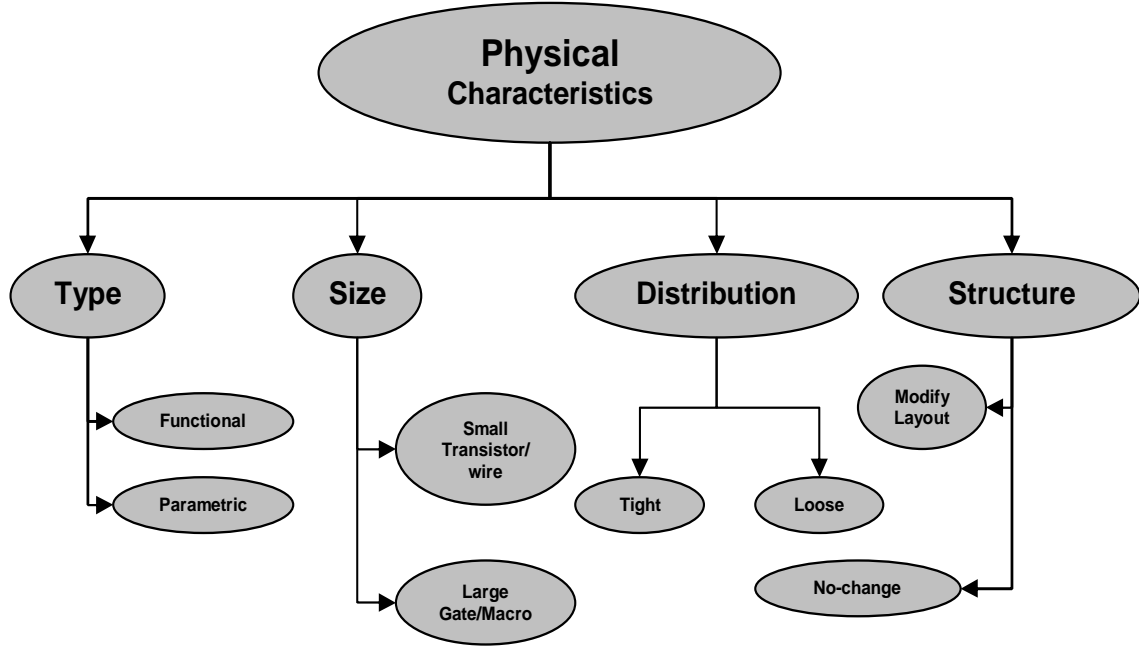


Figure 2.2: Physical Characteristic of Trojans

- **Size** : This accounts for the overhead incurred in implanting the Trojan. Size can be an important factor for Trojan activation and detection. *Large/macro* Trojans are relatively easier to detect at lower frequencies because the leakage current consumed by Trojans is directly proportional to its size and hence a substantial Trojan will have a considerable leakage power consumption and the variations are more easily observed at a lower operating frequency [1]. On the other hand, *small transistor/wire* Trojans may be easier to activate because they require less constrained conditions to trigger them.
- **Distribution** : This refers to the location(s) of Trojan on the IC. While a *tight* Trojan consists of a few gates topologically coalesced together in a localized area, *loose* Trojan consists of gates distributed all over the circuit or portions of a circuit. Flexibility of Trojan fabrication depends on the available space in the original layout. Hence a malicious third party manufacturer has to choose a proper distribution of the required components to design

the Trojan.

- **Structure** : Changing the structure of the IC affect the power, delay characteristics of the device. Structural changes especially that leading to a *layout modification* is very difficult to achieve. Thus to incorporate such changes which inevitably requires the layout to be reconfigured, the adversary is likely to use a Trojan with a very small physical *footprint*. *Physical footprint* is measured in terms of the area, power consumption, delay characteristics and other such factors. Since for a functional Trojan size and distribution have significant impact on the original footprint of the Trojan, for larger Trojan sizes, distributing the components across the layout can assist in reducing the impact on the power and delay characteristics thereby making it more stealthy. Trojans which require *no-change* in the original layout uses the existing spare cells in the original design to fabricate the Trojan circuit in which case there will be *no change* in the original circuit structure.

II. Classification based on Activation Characteristics

Trojans can be classified according to their activation characteristics. This has been shown in Fig 2.3. Activation characteristics refer to the conditions which triggers the Trojan towards its objective.

Broadly speaking, there are two types of Trojans based on activation -

- **Externally Activated Trojans** - These Trojans are those whose triggering conditions are controlled using the input pins of the IC. Thus it is on the part of the operator as to when he/she wants to trigger the Trojan. This can be done by monitoring the device conditions using a side channel signal that transmits the internal information of the device and using that information to appropriately trigger the device.
- **Internally Activated Trojans** - These Trojans monitor the internal configuration of the system for its operation i.e. they derive their condition for activation from the

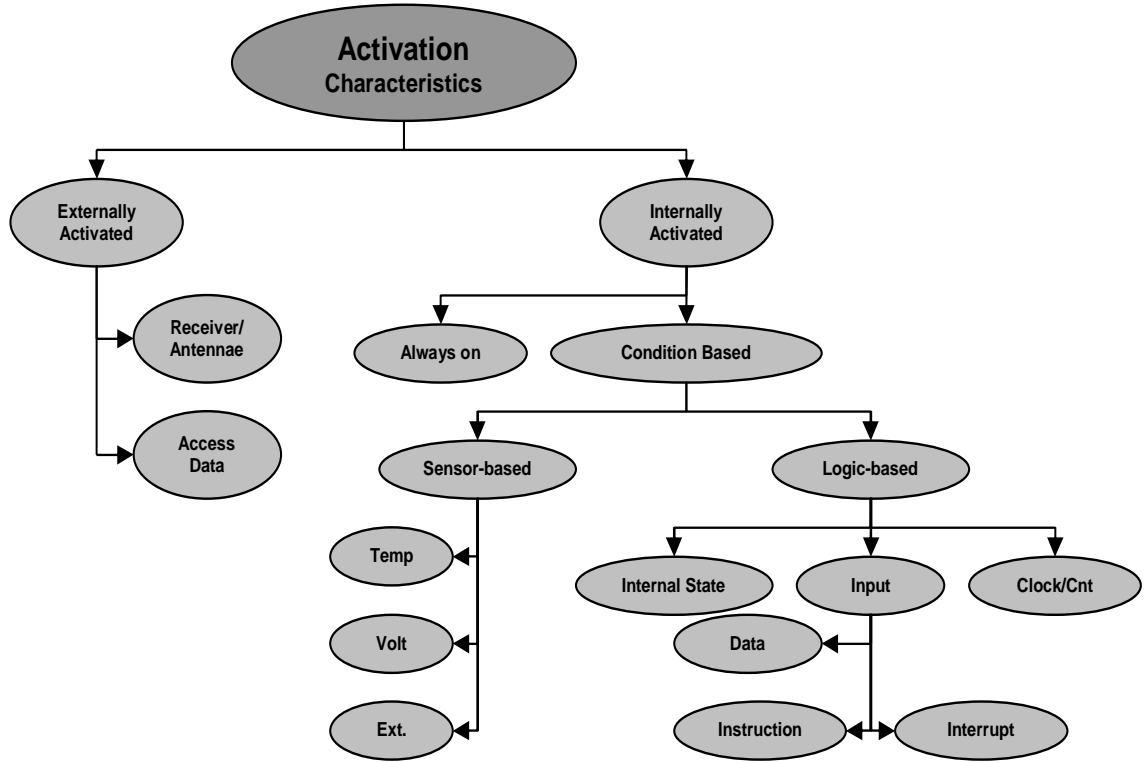


Figure 2.3: Activation Characteristic of Trojans

existing internal environment. *Internally Activated Trojans* are subdivided into two categories:

- *Always-on Trojans* are perennially active and can get triggered any time inside the operating device. This includes Trojans like thinning of transistor wires, changing the drive strength of the transistors by tampering their length-width ratio etc. A device with such change may start working exactly as a normal device but it will fail whenever the internal conditions exceed the physical threshold supported by the fabricated device. Thus their failure cannot be accurately predicted but only a statistical probability of occurrence of such a failure can be estimated.
- *Condition based Trojans* are much more intelligent. They wait for the circuit to enter a specific configuration or the state bits to attain a specific value to

get activated. *Condition based* Trojans are further classified based upon the conditions that triggers them. Thus they can be -

- * *sensor based Trojans* whose activation depends on values/thresholds of some physical parameters like temperature, voltage or any types of external environmental conditions like pressure, humidity, electromagnetic interference that is monitored by the sensor.
- * *logic based Trojans* are those where the logic inside the Trojan intelligently monitors the internal circuit environment to get triggered. Example of *logic based Trojans* are counter Trojans, sequence-detector Trojans etc.

III. Classification based on Action Characteristics

The third classification of Trojans is on the basis of their *action characteristics*. This has been shown in Fig 2.4. *Action characteristics* describe the effect of triggering of the Trojan on the underlying design. These are of three types:

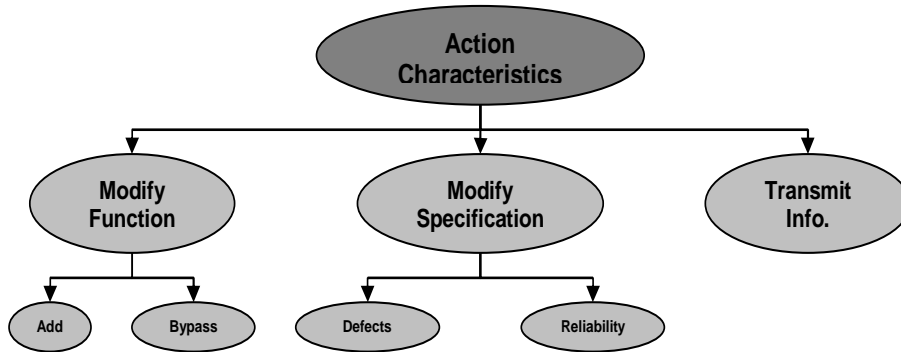


Figure 2.4: Action Characteristic of Trojans

- **Modify Function** - These Trojans change the original functionality of the logic. This can imply removal of a portion of the logic to remove a property, disable some functionality to cause an operational failure or addition of extraneous logic to realize something additional to what is intended.

- **Modify Specification** - This class of Trojans change the properties of the chip such as delay to realize their intended objective. These are similar to *parametric Trojans* discussed earlier. *Parametric Trojans* tamper the strength like the fanout supporting capability of an output of a gate, ability to supply a desired current through a wire etc. Changing the strength of the wires or gates affects the delay of the combinational path and hence fall within this category.

- **Transmit Info** - This type of Trojans don't interfere with the operation of the device. It has been proved that side-channel signals can be decrypted to reveal important internal information embedded within the device. Trojans under this class emit signals containing such key information. This information can be misused by an adversary.

2.1.2 Trojan Characteristics

In general the type of tampering subjected to a device greatly depends on the underlying application that the device performs. Irrespective of the category a Trojan belongs to, they share a number of common features. These are:

1. They occupy a very small area on the IC, small enough that implanting a Trojan does not change the chip dimensions neither do they change the pin count of the original IC.
2. They are inherently stealthy implying that they do not manifest their behavior during normal activity of the device at the chip outputs. This makes traditional ATPG ineffective in uncovering them.
3. They are dormant for most part of their operation which means that they remain inactive for most part of the operation of the device. They require specific rare conditions to appear inside the system to turn them on.
4. They are malicious. Once triggered the consequences are dire.

5. Trojans are embedded deep in the circuit. So they are not connected to gates that are either highly controllable or highly observable. Stated alternatively, they are inserted away from inputs or outputs or gates nearer to inputs or outputs.

2.1.3 Trojan Detection Challenges

Trojans are hard-to-detect using conventional testing mechanisms. In hardware domain, cryptographic algorithms based on *public and private key* concept and approaches based on LFSR and Logic BIST [6–8] have been proposed to monitor the proper operation of the internal hardware. However, Trojans can be intelligently built to deter the advantages of such vigilant approaches. In addition, Trojans can be selectively implanted and its absence in one IC does not guarantee its absence on any other. So destructive testing is also not a viable option. On one hand, destructive testing incurs a yield loss where a chip that has been cut open for analysis has to be discarded, while on the other it cannot guarantee genuineness of the other parts not subjected to such testing. Trojans can have varying spatial locations on the IC and different logical behaviors (counter-based Trojans, sequence-detector Trojans etc.) [18] which complicates the detection mechanism. In software, Trojans (a class of viruses) have been prevalent and many software solutions (anti-virus) exist for their detection. In [12] the authors propose a method for identifying the Trojan software running on a microprocessor. This method uses a digital signature to validate the authenticity of the software before running it on the machine. But there is a difference between a software virus and the hardware Trojan. *Viruses* are necessarily malicious and interfere with the normal operation of the host on which they reside, whereas *Trojans* are passive monitors for most part of their operational life cycle until they are triggered.

An intelligent adversary can insert the Trojans such that their detection probability using test patterns (functional or random) turns out to be extremely low. Assume a Trojan with n inputs. Also assume p_i is the probability of justifying a 0 or 1 on the i^{th} input of the Trojan circuit. In case of deeply embedded Trojans p_i will be extremely small. The probability

of activating this n input Trojan and propagating its effect to an observable point is given by [18]:

$$\mathbf{P} = \mathbf{P}_{activation} \cdot \mathbf{P}_{propagation} \quad (2.1)$$

Also the probability of activation is given by:

$$\mathbf{P}_{activation} = \prod_{i=1}^n p_i \quad (2.2)$$

Assuming $p_i=10^{-3}$ and $n=10$, P turns out to be 10^{-30} . In addition to this, $P_{propagation}$ can further worsen the value of P . This clearly indicates that test patterns can assure a very low reliability in detecting embedded Trojans.

Trojans used in our experimentation can be classified according to the aforementioned taxonomy. From the *Physical characteristics* standpoint, these are *functional Trojans* with a *small size, tight distribution* and *doesn't change the existing internal structure* of the circuitry. From *Activation characteristic* viewpoint, they are *internally activated*. The activation is dependent on a *logical condition* derived from the *internal state* of the FSM. The *Action characteristic* of these Trojans are to *modify* the existing behavior under triggering circumstances. The structure of a sequential Trojan used in our work has been shown in Figure 2.5 (a). Structurally and behaviorally all the Trojans used in our experimentation are similar (sequence-detector) but the triggering scenario varies as per the achievable state space of the circuit on which it is implanted. The motivation behind selecting a sequence-detector Trojan is that they are intelligent, hard to detect and can have profound effect on the system once triggered. Random tampering may affect the circuit functionality but that it will deter the performance cannot be ascertained. The four inputs to the Trojan are the state bits of the flip-flops in the original circuit. The finite state machine (FSM) for the Trojan circuit is shown in Figure 2.5 (b). Here the sequence that the Trojan is trying to detect is **1011**, **0001** and **0010** in this order. This sequence triggers the output of the Trojan that affect one or more internal signals. The flip-flop outputs for FF1, FF2, FF3 and FF4 in Figure 2.5

(a) are represented by \mathbf{a}_1 , \mathbf{a}_2 , \mathbf{a}_3 and \mathbf{a}_4 respectively. The states \mathbf{S}_0 , \mathbf{S}_1 and \mathbf{S}_2 are encoded as **00**, **01** and **10** respectively as shown in Figure 2.5 (b). The two bits represent $\mathbf{Q}_0\mathbf{Q}_1$ in that sequence. The next state and the output equations are given below. \mathbf{Q}_0^+ represents the next state of the MSB in the state encoding, \mathbf{Q}_1^+ represents the next state of the LSB in the state encoding and **OUTPUT** gives the value of the OUTPUT signal in the circuit. A bar on any signal represents the complement of that signal and the \wedge represents a logical **AND** operation.

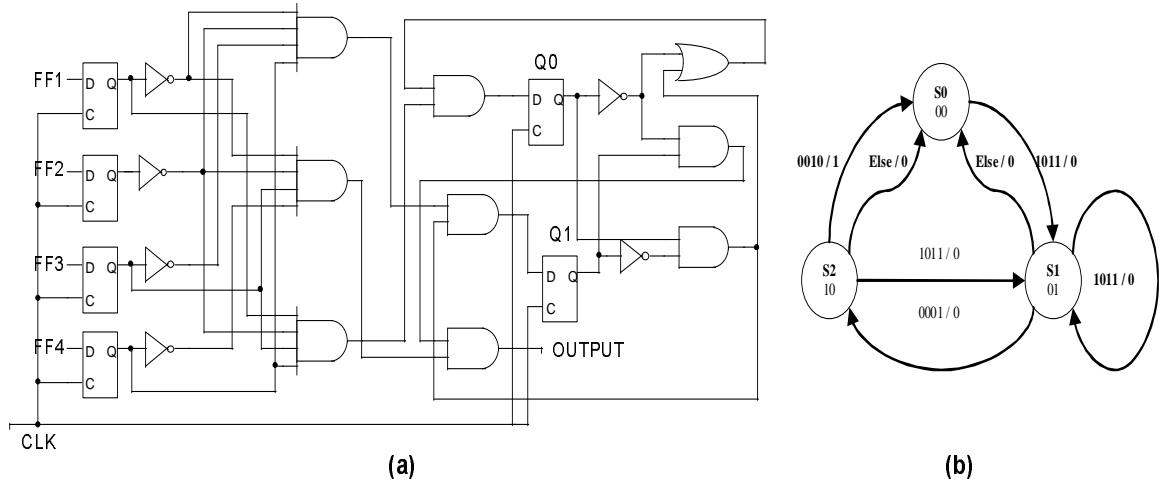


Figure 2.5: A Trojan circuit and its FSM

$$\mathbf{Q}_0^+ = \bar{\mathbf{Q}}_0\mathbf{Q}_1 \wedge \bar{\mathbf{a}}_1\bar{\mathbf{a}}_2\bar{\mathbf{a}}_3\mathbf{a}_4 \quad (2.3)$$

$$\mathbf{Q}_1^+ = (\bar{\mathbf{Q}}_1 + \bar{\mathbf{Q}}_0\mathbf{Q}_1) \wedge \mathbf{a}_1\bar{\mathbf{a}}_2\mathbf{a}_3\mathbf{a}_4 \quad (2.4)$$

$$\mathbf{OUTPUT} = \mathbf{Q}_0\bar{\mathbf{Q}}_1 \wedge \bar{\mathbf{a}}_1\bar{\mathbf{a}}_2\mathbf{a}_3\bar{\mathbf{a}}_4 \quad (2.5)$$

In our work, all the Trojans used are less than 3% of the gate count for small circuits and 1% of the gate count for larger circuits (which equivalently translates to chip area). We have ensured that once triggered, the Trojan affects one or more parts of the circuit impairing the normal internal functionality. Furthermore, we authenticate that the Trojans are difficult-to detect by confirming that the output generated by simulating a set of 1000 random vectors

exactly match for both the genuine and affected circuits. Otherwise, if one could easily detect the Trojan at a primary output, any other analysis would not have been required at all.

2.2 Side Channel Analysis

In manufactured ICs, we normally do not have access to the internal signals within the circuit. Therefore, to assess the internal behavior of such a device during operation, one can analyze parameters like electromagnetic radiation, I/O timing behavior or power profile of the overall system. Such parameters that act like a signature for the device are commonly known as the *side channel signals*. The method of using *side channel signals* to extract internal information of a device is known as the *side channel analysis*, and *side channel analysis* have been effectively used to detect the anomalies in the behavior of a circuit [9,10]. For our approach, we compute the power profile of the genuine CUT. The dynamic power for an IC is proportional to the operating frequency f , switching capacitance \mathbf{C} , and supply voltage \mathbf{V} , shown in the following expression [5]:

$$\mathbf{P} = \mathbf{C}\mathbf{V}^2\mathbf{f} \quad (2.6)$$

Total power consumed in a circuit is the sum of the dynamic power (given by equation 2.6) and the leakage power. Leakage power consumed by the Trojan depends on its size. Since dynamic power depends on clock frequency, a lower frequency will result in a lower dynamic power consumption. In such a case, if the leakage power is high enough, it will be reflected as a discrepancy in the power numbers between the two CUTs. This was illustrated in [2] by an experiment in which a large Trojan could not be detected when the circuit was operated at 100 MHz, whereas it was detected at 500 KHz. But Trojans are mostly small and their leakage power consumption is negligible submerging it within the process variation and so their response to the clock frequency change is not practically observable. Hence we need

a different approach to uncover their presence. All the parameters except the switching capacitance C in the Trojan circuit is same under normal operating condition of an IC. This means that variation in the switching capacitance should be the distinguishing parameter for analyzing two distinct ICs - one genuine and the other malicious. Switching capacitance is directly proportional to the total number of gates toggling in a circuit for a particular vector pair. Thus our ultimate goal is to induce maximum toggles in the Trojan portion of the circuit.

2.3 Power Profile

A power profile represents the pattern of power consumption in a system. Power consumption for any pair of vectors is dependent on the total number of gates that switch which accounts for the changing switching capacitance (other factors in Equation 2.6 remaining the same). In our work we have used the terms activity profile or power profile interchangeably because number of gate switches in a circuit is directly proportional to the dynamic power consumed by it.

Dynamic power profile indicates the variation in the power consumed by a circuit. Variations in power profile may arise because of multiple reasons. Of these, the switching gates and process variation are noteworthy. This is a common observation that power profile of identical design will not be exactly be the same for two different chips. That is, they will differ within a certain range which we call as the *process variation*. In order for the extraneous activity generated by the Trojan to be observable, we must be able to highlight it above the process variation. In [1], they have assumed three distinct values of process variation, viz. 2%, 5% and 7.5%. In our work we have assumed the process variation to be 5% although in some cases, as our results will show, we are able to generate power profile differentials in excess of 7.5%.

2.4 Hamming Distance

For two states in a circuit, the Hamming distance between them is defined as the number of bits that differs in between those states. For instance, if we have a circuit consisting of four flip-flops then for two arbitrary successive states 0001 and 0110 the Hamming distance is 3 because all the 3 bits except the MSB differ.

In our approach, we have used both maximum and minimum Hamming distance concepts. By increasing the Hamming distance, we try to explore different parts of the state-space extensively. On the other hand, minimizing the Hamming distance (but disallow it to be in a sleep state), we try to ensure that the activity within the Trojan can contribute to a greater portion of the total power. Since the power consumed in the circuit is directly proportional to the amount of switching activity occurring in it [13,14], by minimizing the switching activity, we actually try to minimize the total power consumption.

Chapter 3

State-space Partition with Hamming Distance Maximization

This chapter outlines the first *divide and conquer* approach that we devised for improved isolation and detection of Trojan. The ‘Motivation’ section describes the rationale behind the methodology used. ‘Our Approach’ section explains the techniques used to formulate the methodology in detail. We have included the results obtained using this approach in the ‘Experimental Results’ section followed by a summary section in which we discuss the pros and cons of the formulated methodology along with an explanation on observed results and future scope of enhancement.

3.1 Motivation

As the authors in [1] have identified that the power signature difference must exceed process variation to be statistically significant, some intelligent Trojan’s may hide the discrepancy in signatures within process variation levels. Such Trojans are difficult to detect. Furthermore, the authors in [1] employ a (random) non-redundant set of tests. The non-discriminate

nature from random test patterns is not ideal in maximizing the discrepancy in the power signatures. This is because the random patterns keep the circuit activity to an average value and not to the lowest, which means that the activity of the Trojan is more likely to be overridden by the entire circuit activity. Random vectors would traverse the state space aimlessly.

3.2 Our Approach

In this chapter, we propose a test generation technique that targets at magnifying the discrepancy between the CUT (*Circuit Under Test*) and genuine design waveforms. Our approach is a two-staged process. This first test set intelligently and quickly sweeps through the state space in a controlled manner and generates activity within subsets of flip-flops while keeping the activity of the rest of circuit low. After analyzing this power profile, we identify possible subsets of state signals that may feed the Trojan in the circuit. In the second step, we focus on those regions identified in the first step and generate a new test suite to further increase the relative difference in the power profiles between the actual circuit and the Trojan counterpart. In this second step, if we observe a sustained increased activity over the expected behavior, it clearly indicates anomalous behavior that strongly indicates the presence of a Trojan. we call these two steps as *Circuit Partitioning* and *Activity Magnification* respectively, and they are detailed below.

3.2.1 Circuit Partitioning

Trojans usually constitute a tiny fraction of the total chip area. It is intuitive that during normal functional operations, the activity in the overall circuit could be several orders of magnitude greater than the activity of the Trojan. Hence, the relative increase in the circuit activity due to the presence of the Trojan may not be projected above the process variation, and consequently it is difficult to make any inference about its presence. Therefore, in order

to detect a Trojan circuit, we need to increase the activity within the Trojan portion of the circuit while simultaneously minimizing the activity for the rest of the circuit. Noting that we should not decrease the power so low such that the CUT enters some sleep mode. If the Trojan also enters the *sleep mode*, we will not be able to observe discrepancies in the power signatures. Here *sleep mode* refers to a state where the circuit is totally inactive. In such a situation, no gates in the circuit toggle and the circuit power is at a very low level. Since our intention is to keep the circuit at a least possible active state, we should make sure that we avoid sleep state.

Since we cannot predict the location of the Trojan in the circuit, we use a *divide and conquer* approach as an attempt to isolate it. In general, one can broadly classify the flip-flops in a circuit into different groups depending on the functionality with which they are associated. Trojans being intelligent monitors, their triggering condition is likely to be associated with one or more such functional groups. Hence, it is better to focus on a smaller portion of the state space than the complete set of flip-flops considered together. Consider a circuit with N flip-flops. In our approach, we want to traverse through the state space by visiting and exercising different partitions.

Given a subset of flip-flops, G , the signals and gates that lie in the fanout cone of G defines a region of interest. Our algorithm partitions the circuit into small regions. Each region may contain 5, 10 or 20 flip-flops depending on the total number of flip-flops in the CUT. At any point during test generation, we try to increase the activity in the corresponding region of interest while keeping the rest of the circuit at low activity. For this, we maximize the Hamming distance between any two successive states in the subset G , while simultaneously minimizing the Hamming distance for the rest of the state variables. This is important because we do not want the power from the non-Trojan part of the circuit to drown out the power from the Trojan. By minimizing the Hamming distance for the rest of the flip-flops, the gates in the transitive fanout cone of these flip-flops undergo little activity thereby reducing the overall circuit activity. We calculate the per flip-flop increase in the Hamming distance for the group G as well as for all other flip-flops that are not in G . The difference between

these two quantities serves as the selection parameter for an appropriate input vector from a list of available vectors.

Let S be the entire set of flip-flops in the circuit. Again, let G be a group of flip-flops for which we are maximizing the Hamming distance. Let d be the Hamming distance for the flip-flops in G and d' be the Hamming distance for the rest of the flip-flops. Then, we define our objective function F as the following:

$$\mathbf{F} = \mathbf{max}(d/g - d'/g') \quad (3.1)$$

where g is the number of flip-flops in the group G and g' is the number of flip-flops in the rest of the circuit apart from those in G .

An example of evaluation of the objective function F has been shown in Table 3.1. From Table 3.1 it is clear that the sequence corresponding to the vectors 10 and 11 in the table maximizes the function F defined earlier. Therefore, we select this vector-pair from the current vector set and append it to our existing test set.

We generate k random input vectors and select the best vector-pair from within it. We repeat this until we have obtained enough vectors for the concerned set of flip-flops. We note that a large value of k ensures that we get a good vector-pair. On the other hand, k should be small enough so that we do not incur a major runtime penalty. In our experiments, we limit k to be less than 20 for each subset, and we repeat the process for all the subsets of flip-flops in the circuit.

3.2.2 Activity Magnification

Based on the comparison of the relative difference in the power profiles for the genuine and Trojan circuits using the vector sequence generated in *Circuit Partitioning* stage, we identify the regions (set of flip-flops) that exhibited increased relative activity. These regions are the

Table 3.1: Table showing the selection criteria for input vectors

Vector Number	HD diff/FF in the Group	HD diff/FF outside the Group	Net HD diff/FF
1	0.15	0.0288462	0.2211538
2	0.15	0.0096153	0.1403846
3	0.25	0.0144231	0.2355769
4	0.25	0.0288462	0.2211538
5	0.15	0.0288462	0.1211538
6	0.15	0.0144231	0.1355769
7	0.15	0.0144231	0.1355769
8	0.2	0.0336538	0.1663462
9	0.1	0.0144231	0.0855769
10	0.15	0.0288462	0.1211538
11	0.25	0.00961538	0.240385
12	0.1	0.0288462	0.0711538
13	0.15	0.0144231	0.1355769
14	0.15	0.0096153	0.1403846

HD diff: Hamming Distance difference
FF: flip-flop

ones that are most likely to influence and feed the Trojan as indicated by the increased activity when compared to the activity profile for the genuine circuit. The advantage of using our proposed method is that it focuses on a small portion of the circuit at a time and hence we can explore the regions more elaborately to confirm the presence of a Trojan.

In this stage, we generate more vectors for the specific region(s) marked as possible regions of the Trojan using the same test generation approach as discussed in *Circuit Partitioning* stage. We repeat this process for all the targeted regions. Results show that our method significantly magnifies the relative activity difference between the Trojan and the genuine circuit. Our aim is to maximize the discrepancy in the CUTs. It is relatively easier to focus on a smaller portion of the circuit rather than the entire circuit as a whole. Our first step intended to narrow down the regions of search. The second step takes the advantage of spotted areas in the circuit and intensifies the search process. If the Trojan is indeed in the target region then it will show greater discrepancies with a long sequence of vectors than with a shorter one. So in this stage we increase the number of vectors for the regions chosen from stage 1. Keeping a very long sequence of vectors for each of the regions is an overkill on runtime.

The flow of our overall approach is shown in Flowchart 3.1.

3.2.3 Implementation

In our approach, we start with an initial reset state of 0s and generate a set of 20 random vectors for the first flip-flop group. We simulate all the vectors independently on the circuit and based on the heuristic defined by equation 3.1. We select one of the input sequences. For groups with 5 flip-flops we generate 20 patterns per group while for groups with 10 or 20 flip-flops we generate 10 patterns per group. We compare the power profiles of the Trojan and genuine circuits, and our results show that we can identify regions that show relatively high activity as compared with the random power profile. Note that since random vectors do not distinguish areas in the entire circuit, no specific information about the region of the

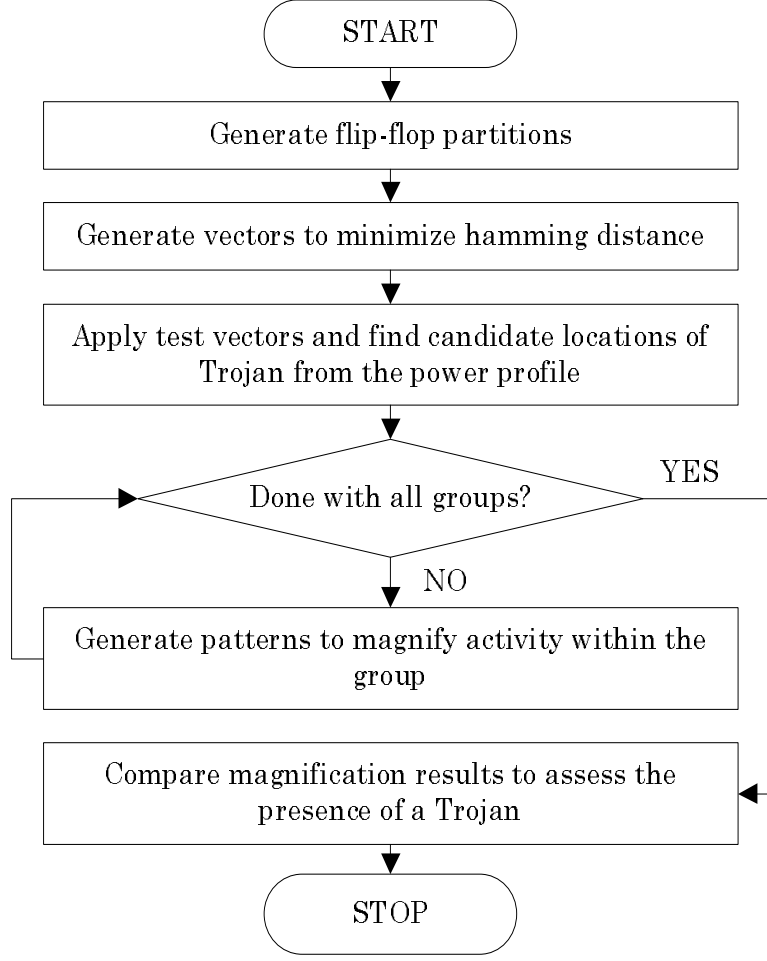


Figure 3.1: Overall flow of the Trojan identification process

Trojan can be deduced from the power profile. Next, we probe into these identified regions in *Activity Magnification* stage. Our analysis assesses the extent of extraneous activity more elaborately and confirms whether process variation alone can account for the discrepancies observed.

The Trojans used in our experiments are *Sequence Detection Trojans*. The general structure of the Trojans are similar to the one discussed in the previous chapter. Each of the Trojan was tested for stealthiness where a sequence of 1000 random vectors could not distinguish them at the outputs. Also the area occupied by the Trojans was less than 3% for small circuits and less than 1% for the larger circuits.

3.3 Experimental Results

In this section, we present results obtained using our proposed two-stage test generation approach. This section is divided into two parts - the first analyzes the result for the circuit-partitioning step, the second for the activity maximization stage within the identified regions.

3.3.1 Circuit partitioning Results

In each of the graphs shown below, a rectangular legend (corresponding to the blue curve) represents the plot for the results obtained by our method while the graph with a diamond legend (corresponding to the brown curve) shows the plot for the random patterns. The X-axis denotes the index for vector numbers, while the Y-axis denotes the percentage difference in activity between the Trojan circuit and the genuine circuit.

Table 3.2 given below summarizes the total flip-flop count, number of groups that the circuit is partitioned into, the group(s) to which the Trojan is connected in our experiments and the total number of vectors generated for each ISCAS89 benchmark circuit.

Table 3.2: Circuit Partitioning Statistics

Circuit	Flip-flop Count	Total Groups	Trojan Group	Vector Length
s444	21	5	3rd & 4th	100
s1196	18	4	2nd	80
s3271	116	12	8th	120
s3330	132	14	5th	140
s5378	179	18	5th & 6th	180
s9234	228	12	4th	120
s15850	597	30	27th	300
s38584	1452	73	72nd	730

For s1196, the blue (rectangular legend) curve in Fig 3.2 shows that the percentage activity difference between actual circuit and the Trojan circuit is amplified in the regions covered by our generated vectors 15 to 20 and between vectors 23 to 34. The Trojan is indeed connected

to the 2nd group excited by vector numbers 21-40. The difference in the magnification obtained by our approach as compared to the random clearly separates out the 2nd region for further magnification.

In Fig 3.3 for circuit s3330, we can separate out regions corresponding to flip-flop groups 3, 4, 5, 7, 9, 12 and 13 as the portions with distinct increase in the percentage circuit activity. In our experiment, we have associated the Trojan with a portion of the flip-flops in group 5 which we could isolate as a target region for further analysis in Stage 2.

Fig 3.4 is for circuit s5378. The behavior of the curves suggest that the infected portion of the circuit may map to any of the regions covered by vector sets 5, 6, 9, 10, 11, 12 and 13. It turns out that this circuit has a Trojan split over groups 5 and 6 which were included as candidate regions.

Among these graphs, there are portions where the difference in percentage activity between the genuine and the Trojan circuits for random patterns exceed our approach. However, the random patterns cannot narrow down the search region. On the other hand, with our two-stage scheme, we can isolate the regions that allow us to probe further.

Fig 3.5 shows that the random method hardly shows any difference in the percentage activity between the genuine and the Trojan circuits. However, our approach separates out distinct regions viz. 5, 6, 12, 61, 67, 71 and 72 where the extraneous activity in the Trojan is high enough to produce a difference as high as 8% from the actual circuit. This is the graph for circuit s38584 where the Trojan is embedded into the 72nd group represented by the vectors 711 to 720.

Fig 3.6 and Fig 3.7 give the ratio by which our method magnifies the Trojan circuit activity as compared to that of the random method. We observe that our method magnifies the Trojan to actual circuit activity by 4 to 20 times in the portions which are identified as candidate regions. Fig 3.6 contains the relative magnification information for the circuits s444, s1196 and s3271 while Fig 3.7 contains the relative magnification information for circuits s3330, s5378 and s9234. Primarily, the division of the circuits into different groups is based on

the circuit size. These two figures show that our *Circuit Partitioning* stage can consistently locate the regions most responsible for the Trojan.

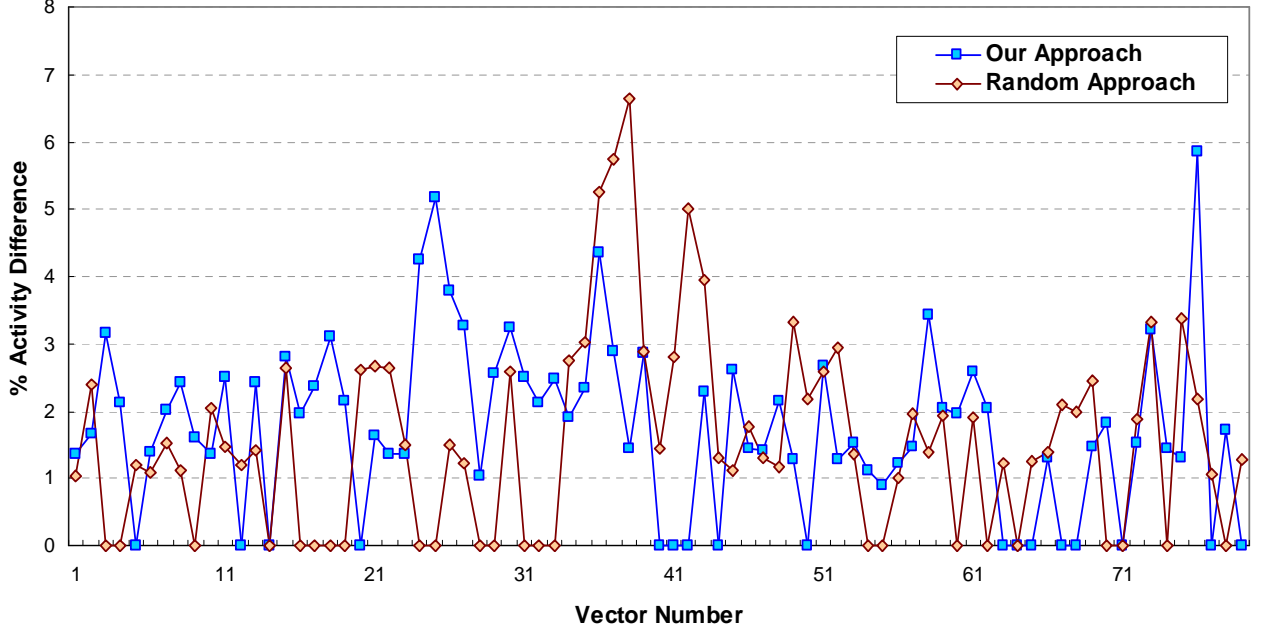


Figure 3.2: Relative increase in Trojan circuit activity by our approach vs. the random approach for s1196

3.3.2 Activity Magnification Results

Our attempt to magnify the activity for target groups in circuit s1196 shows that when we zero in on those regions most responsible for the Trojan, the magnification of the power dissipation ratios is significant when compared with the random vectors. Fig 3.8 shows these results. The legends are the same as in the previous section. An important observation is that, at times, we are able to achieve a magnification in the activity of the Trojan from the actual circuit in excess of 6% (which is normally greater than the process variation) and this trend is not observed in the graph obtained at the first stage.

For s5378, activity magnification plot for group 5 is shown in Fig 3.9. Although the disparity in the activity between the Trojan and actual circuit is less pronounced in the later portion

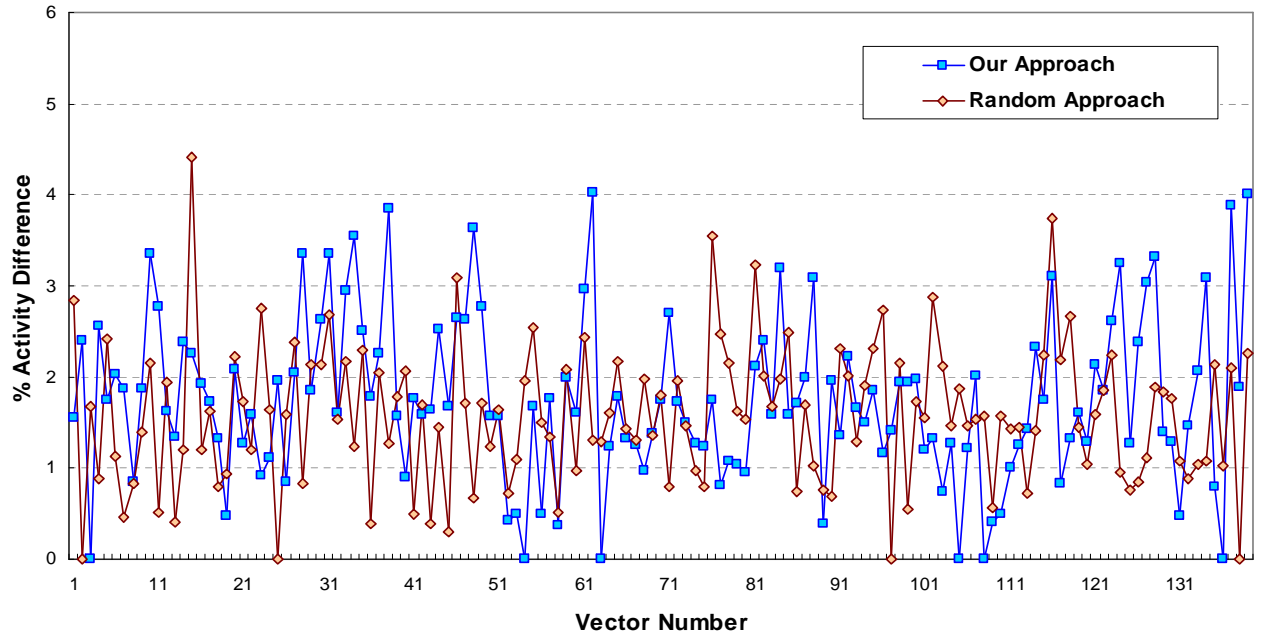


Figure 3.3: Relative increase in Trojan circuit activity by our approach vs. the random approach for s3330

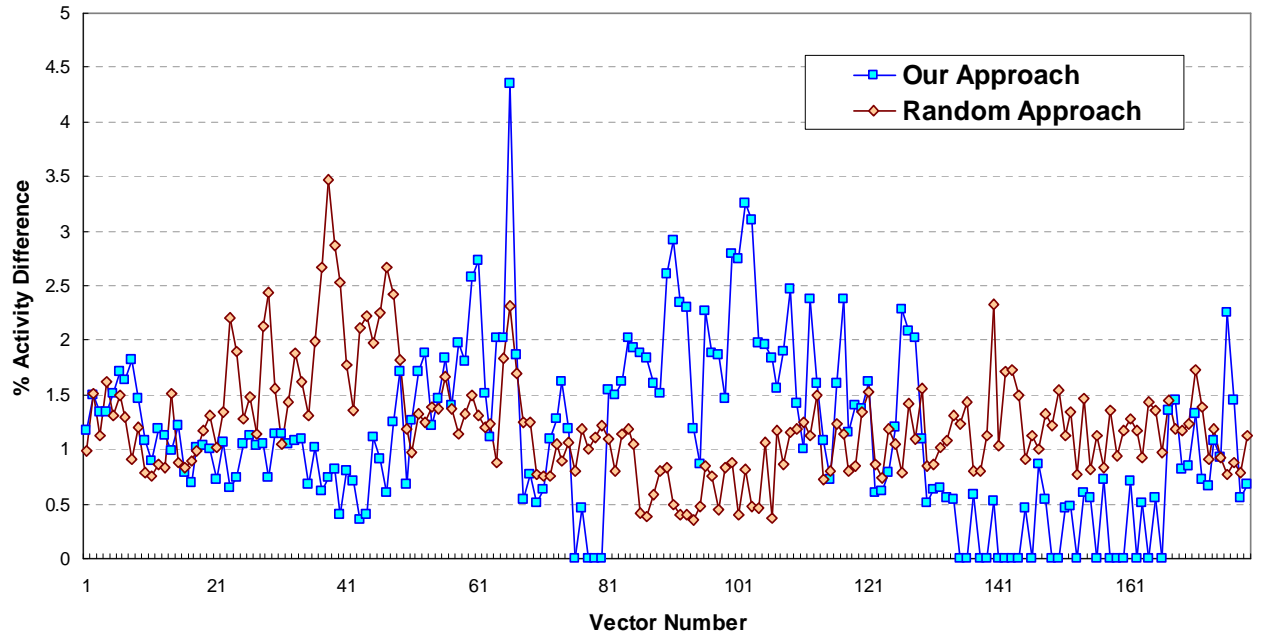


Figure 3.4: Relative increase in Trojan circuit activity by our approach vs. the random approach for s5378

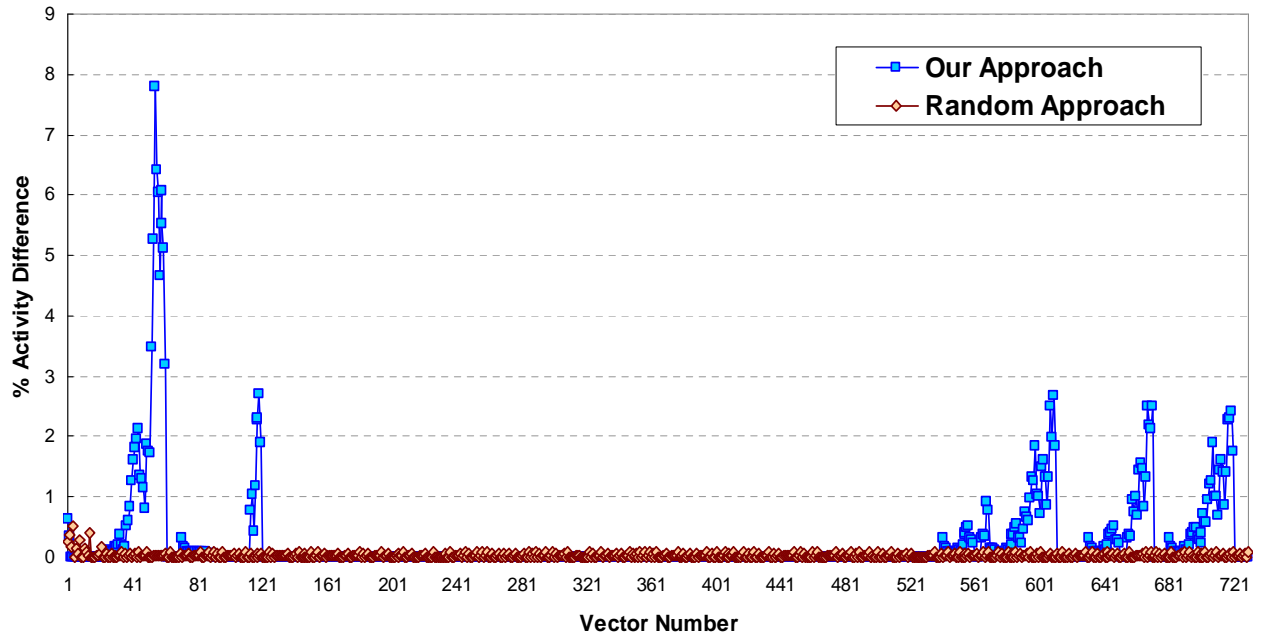


Figure 3.5: Relative increase in Trojan circuit activity by our approach vs. the random approach for s38584

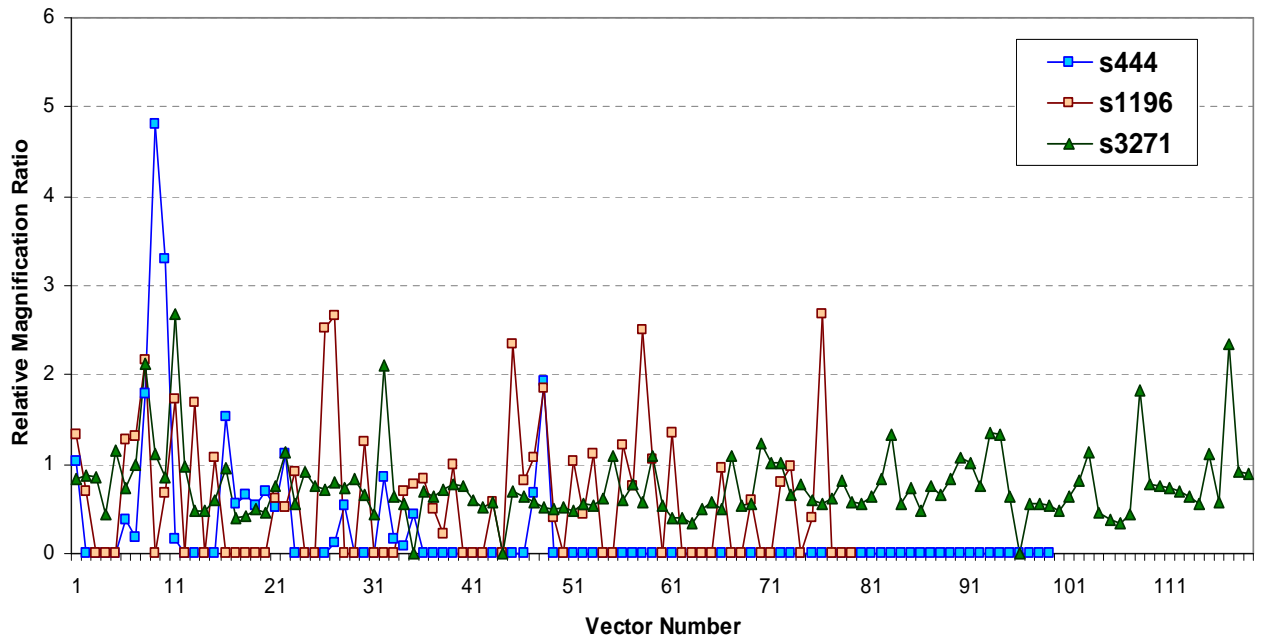


Figure 3.6: Ratio of relative magnification of Trojan circuit activity over the actual circuit activity for different circuits

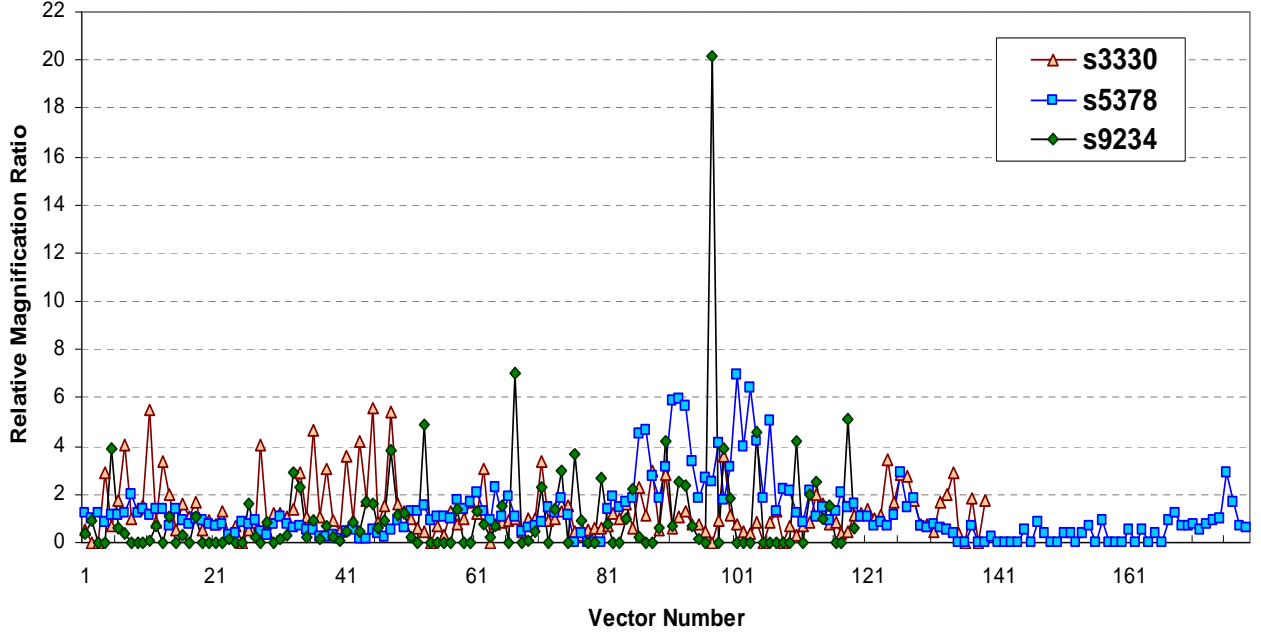


Figure 3.7: Ratio of relative magnification of Trojan circuit activity over the actual circuit activity for different circuits

of the plot, the degree of variation in between vectors 41-51 is wide enough to indicate a clear anomaly.

s15850 shows one of the best results for the activity magnification step, shown in Fig 3.10. In this circuit, the Trojan was connected to 27th group and when we attempted to zero-in on the power numbers for the 27th group, it clearly indicated that the targeted group produces noticeable extraneous activity as compared to the random vectors because plot corresponding to our approach almost always exceeds the random curve in magnitude.

3.4 Summary

We have presented a two-stage approach to generate a set of effective test cases that is able to detect the presence of a Trojan in a given design. Experiments showed that our method is able to provide a 4 to 20 times magnification in the circuit activity for the circuit with a Trojan over a genuine circuit. Moreover, in circuits like s38584 our method points the

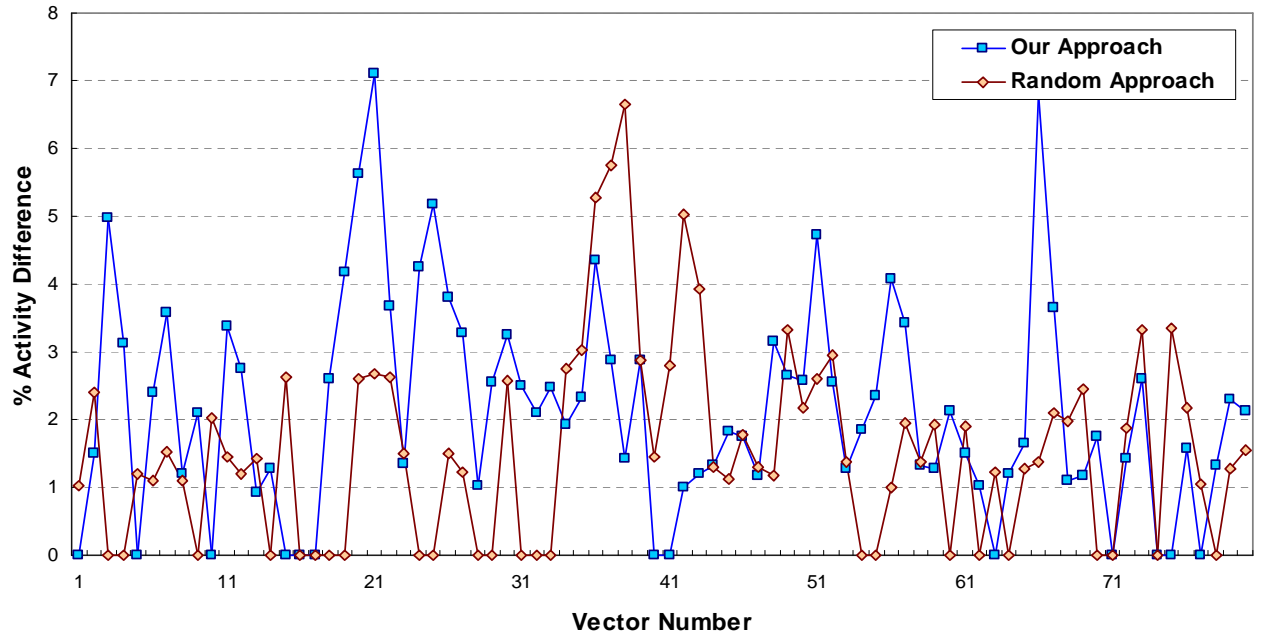


Figure 3.8: Activity Magnification for s1196, (group 2) between our approach vs. random approach

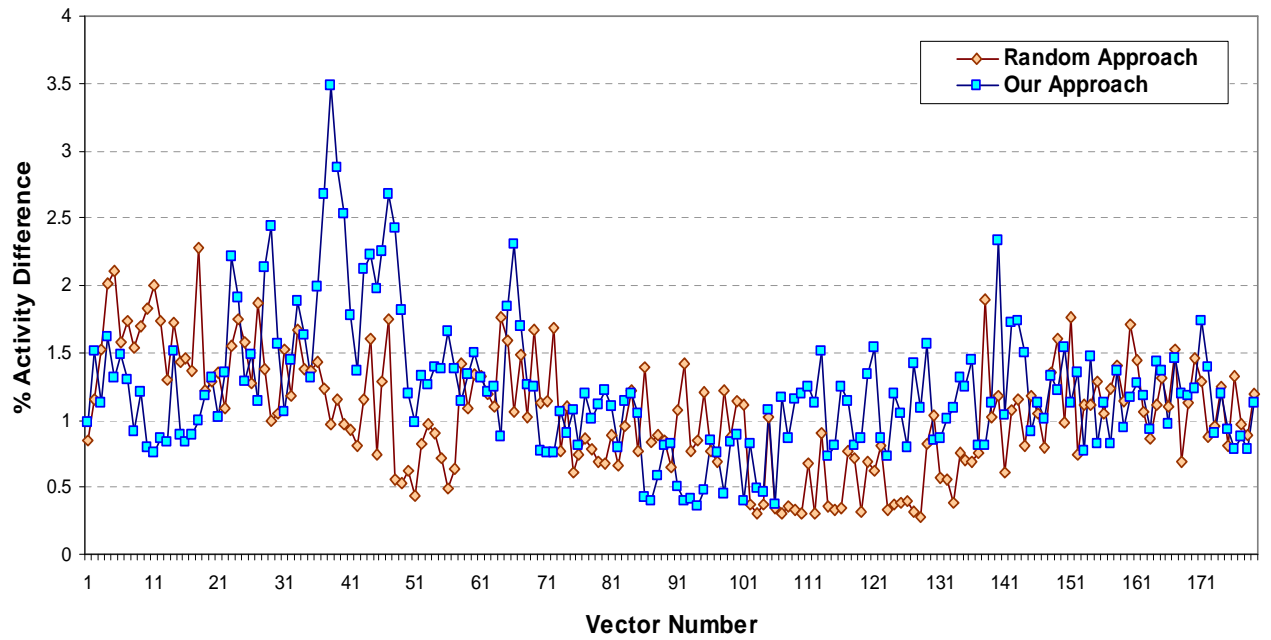


Figure 3.9: Activity Magnification for s5378, (group 5) between our approach vs. random approach

target areas distinctly where the conventional random patterns fail to make any distinction. The first step in the two-staged approach helps to narrow down the target regions effectively

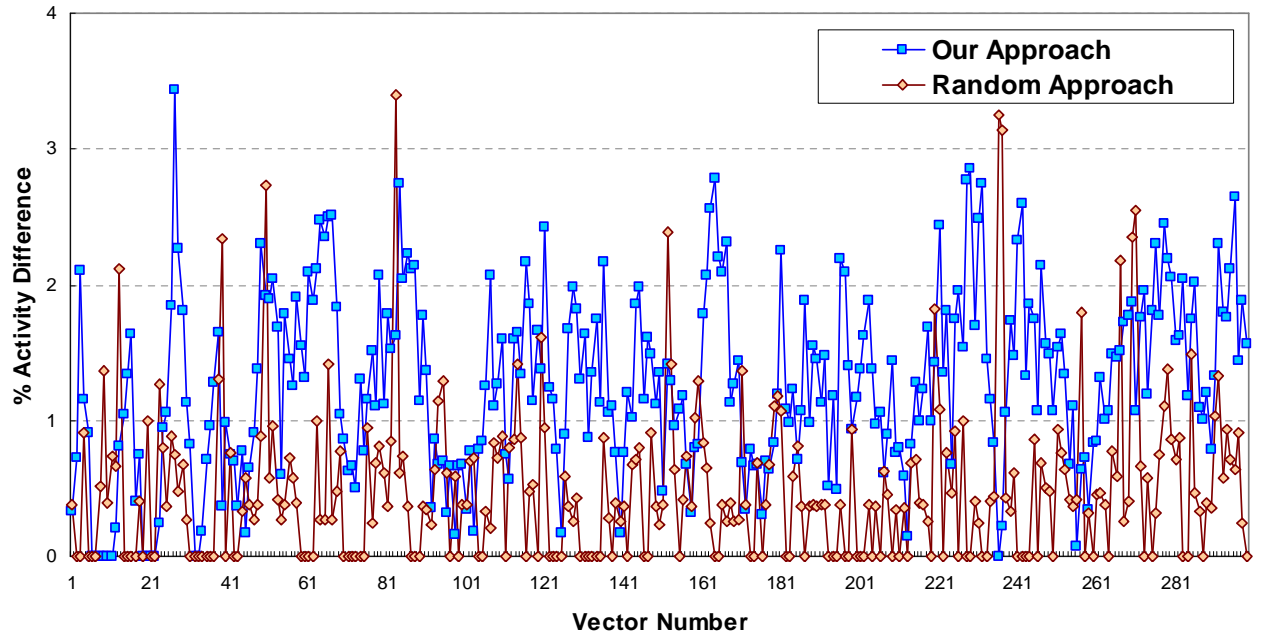


Figure 3.10: Activity Magnification for s15850, (group 27) between our approach vs. random approach

and the second step magnifies the anomalous behavior. In many circuits, the distinction in the power profile for the targeted groups is prominent enough to observe the behavioral discrepancy.

Chapter 4

Region based Partition with Toggle Count Maximization

4.1 Motivation

In *Chapter 3* we proposed an approach for creating a partitioned circuit in terms of the state elements to enhance the effectiveness of the search for a Trojan-infected part. This approach focuses on exercising a subset of state-elements at a time and hence provides a better indication of the location of the embedded Trojan. The maximization of the Hamming distance among the state variables in the targeted state partition is used to differentiate between the genuine and the tampered parts. Nevertheless, maximizing the Hamming distance need not necessarily ensure the increased circuit activity in the region where the Trojan is present. Likewise, minimizing the Hamming distance also need not necessarily facilitate reduced circuit activity in the parts that are not targeted for reasons explained later in the chapter. In addition, Trojans are intelligent circuits so that they are most likely to be attached to a set of internal signals that are logically related to a particular function. So a more judicious selection of the groups of flip-flops can help us excite the Trojan in a more effective way.

In this chapter, we propose a region based partition and excitation approach for circuit designs that makes accurate estimate of the Trojan location(s). A region is defined as a structurally connected set of gates. In context of our discussion, a set of gates are said to be structurally connected if they have a common successor. The predecessors are connected via wires to this successor. Experimental results show that our approach not only separates out the possible location of the Trojan(s) but also, in many cases, provides robust indication of the anomalous behavior in circuit parts that confirms its presence.

4.2 Our Approach

Our approach consists of two steps. The first step is to compute and select appropriate regions for analysis within the circuit, and the second step is to generate a suitable input vector set that maximizes the partial relative power consumed in each of the selected regions. We name these steps as - *Region Based Partition* and *Relative Toggle Count Magnification* respectively.

4.2.1 Region Based Partitioning

In our methodology, we partition the circuit into smaller sub-circuits that we call as *regions*. A circuit consisting of five *regions* is shown in Figure 4.1(a). *Region* based partitioning has been used earlier in error diagnosis and detection [11]. Its *radius* defines the extent of a *region*. For a gate, the *region* around it comprises of all the transitive fanin and fanout gates that are within the defined *radius*. Thus, a single gate constitutes *region* of *radius* zero (G1 in Figure 4.1(b)), immediate fanin and fanout gates along with the original gate constitutes *region* with *radius* one (G1, G2, G3, G4, FF1 G6 and G7 in Figure 4.1(b)) and so on. The *regions* are restricted across clock boundaries i.e. no gates crossing flip-flops are included in a *region* (G11 is not included in a *region* of *radius* 2 around gate G1 in Figure 4.1(b)). Clearly, for any given circuit with a specified *radius*, the total number of *regions* is equal to

the number of gates, as each gate can serve as a center of a *region*.

For large circuits, we needed to define a suitable selection criterion that allows us to intelligently select a subset of the *regions* that are most important for analysis. Considering the fact that Trojans are mute spectators for most part of the operational cycle of the circuitry, it is intuitive that they act as state monitors. This implies that they are most likely to be associated with signals related to the circuit state elements (i.e., flip-flops). Even then, the number of flip-flops can still be large enough to consider them individually. More so, analysis of individual flip-flop regions may not be sufficient to affect a substantial portion of the Trojan that ensures a noticeable disparity in the side-channel signal behavior, which in our case is the power profile.

To handle this issue we need to cluster the flip-flops into groups that are most likely to be associated with a Trojan. As stated earlier, since Trojans are intelligent circuits, they are most likely associated with a particular logical functionality in the chip. Groups of flip-flops that are structurally connected through a combinational logic determine the signal behavior of any signal in its fanout cone based on the current input and the value of the state bits on the flip-flops. Therefore, it is worthwhile to consider only those *regions* that contain a certain number of structurally related flip-flops. We call this bound as the *Flip-Flop Threshold*. Consider Figure 4.1(b) again using radius 1. The *region* centered at gate G1 contains a flip-flop FF1. On the other hand, the *region* centered at gate G2 does not contain any flip-flop. All the *regions* that contain a *Flip-Flop Threshold* number of flip-flops will be selected for our analysis. For larger circuits we need to start with a larger value of the *radius* and the *Flip-flop Threshold* otherwise the computation and selection of required regions can take a substantial amount of runtime. Also the approach discussed in *Chapter 3* can be used as the first step to isolate the portions of the circuits potentially infected with the Trojan and then create out *regions* from those portions for further investigation.

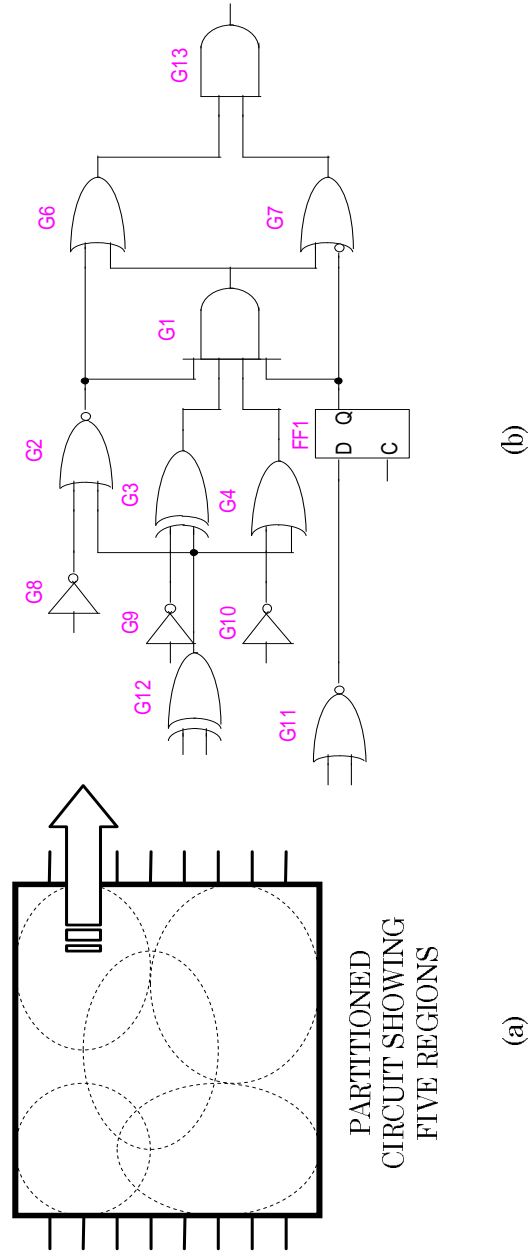


Figure 4.1: Illustration of the concept of *region* and *radius* in a circuit

4.2.2 Relative Toggle Count Magnification

Once we have identified the *regions* of interest, we attempt to create an activity peak on a *per-region* basis. For this, we simulate the circuit with vectors that maximize the switching activity within the *region* of interest while simultaneously minimize the switching activity for the rest of the circuit. Thus, if *in-region activity* and *out-region activity* represent the amount of switching activity for the gates within the *region* of interest and for the rest of the circuit respectively; then our objective function is defined by:

$$\mathbf{F} = \mathbf{max}(\textit{in-region activity} - \textit{out-region activity}) \quad (4.1)$$

The behavior of a Trojan is perceptible only if the difference in the activity of the Trojan-infected chip and the genuine chip (without Trojan) is above the process variation. Since the circuit power is directly proportional to the switching activity, which in turn translates into the number of toggles in the circuit, the function F mentioned in Equation 4.1 ensures that the power consumed in each of the *regions* are individually exaggerated with respect to the entire circuit. If the Trojan is connected to portions of one or more such *regions*, the circuit activity in the genuine chip will very likely to be different from the tampered one owing to the extra activity of the Trojan portion. This, in turn, is projected as the difference in the power profiles obtained from the two chips at the infected *regions*. This idea of maximizing the difference in the toggle count is better than our previous approach of maximizing the Hamming distance because maximizing Hamming distance need not always increase the power consumed and vice versa. To illustrate this, consider a case in which a flip-flop not in the targeted *region* that feeds to a high fanout gate. A single toggle in this flip-flop may not add much difference to the Hamming distance but it will certainly make many other gates in the circuit to toggle thereby increasing the total circuit power. On the other hand, the approach discussed in the previous chapter is very effective is quickly sweeping through the circuit and narrow down the search to few locations. This method take more preprocessing time before the actual vectors are applied on the CUT.

4.2.3 Implementation

In our implementation, we used an iterative approach as discussed above to isolate the regions that are most likely to be associated with the Trojan. We start with an initial *radius* of 2 and a *Flip-Flop Threshold* of 2. We increase the threshold for the flip-flop count until there are no *regions* within the specified *radius* with the given *Flip-Flop Threshold* count. Then we increase the *radius* and reset the *Flip-Flop Threshold* to the original value of 2 again. We continue this process until a certain upper bound on the *radius* is achieved. During the *region* creation, we define the maximum number of *regions* for any given *radius* and *Flip-Flop Threshold* as 1000 crossing which we abort the combination and move over to the next iteration. At the end of this step, we have obtained a number of sets of *regions*, each of which contains at least *Flip-Flop Threshold* number of flip-flops.

After marking the *regions* of interest for a given combination of *radius* and *Flip-Flop Threshold*, we generate test vectors for each one of the *regions*. For this, we start by generating a set of 20 random vectors. We simulate each of these vectors individually followed by computing the value of the difference in the switching activities on the targeted circuits for each one of them. From this set, we select a single vector according to the function defined by Equation 4.1. For each *region*, we repeat this process 10 times and collect 10 vectors to have a visible effect in the power profile for that *region*. Note that other vector generation methods can be used to derive the vector set.

In our experimental setup, we insert a number of hypothetical Trojans in a number of circuits. We simulate the generated vectors on both the genuine circuit and the Trojan circuit and compute the switching activity for each one of them separately. We plot the percentage difference in the activity of the Trojan circuit as compared to genuine circuit for the generated vector set against a random vector set. From the plot, we collect all the *regions* that show enhanced difference in the switching activity profile for our approach as compared with the random simulation. Our experimental results reveal that the actual flip-flops feeding the Trojan appears in high frequency count within the *regions* collected from

the power profile analysis. If $\mathbf{freq}(G)$ represent the count the flip-flop G appears in the selected *regions*, i be the total number of *regions* and ***Frequency Threshold*** represent the minimum count to qualify for a Trojan associated flip-flop, then the gates accountable for the Trojan is given by:

$$\mathbf{Trojan} = \Pi_0^i(G : G \in \text{Gate in a selected region} \wedge \mathbf{freq}(G) > \text{Frequency Threshold}) \quad (4.2)$$

We can formulate the entire procedure in the form of Algorithm 1. The functions used in the algorithm has been explained in Table 1.

Algorithm 1 Generate differential power profile plots for genuine and Trojan circuits

Require: *FFThreshold, RadiusThreshold, GenuineCkt, TrojanCkt, InRegionFFCount*

Ensure: *Power profile plots for Radius & Flip-Flop Combinations*

```

1: Radius  $\Leftarrow$  2
2: FFCount  $\Leftarrow$  2
3: VectorSet  $\Leftarrow$   $\emptyset$ 
4: Regions  $\Leftarrow$   $\emptyset$ 
5: while Radius < RadiusThreshold do
6:   Regions  $\Leftarrow$  ComputeRegions()
7:   while FFCount < FFThreshold do
8:     for all Regions do
9:       if InRegionFFCount > FFCount then
10:        VectorSet  $\Leftarrow$  GenerateVectors(Region)
11:      end if
12:    end for
13:    SimulateVectors(GenuineCkt)
14:    SimulateVectors(TrojanCkt)
15:    ComputeActivityDifference()
16:    IncrementFFCount()
17:  end while
18:  IncrementRadius()
19:  Reset(VectorSet, Regions, FFCount)
20: end while

```

Table 4.1: Functions of Algorithm 1

Function	Purpose
ComputeRegions()	Compute all <i>Regions</i> with given <i>Radius</i>
GenerateVectors(<i>Region</i>)	Generate vectors to maximize <i>Toggle Count Difference</i> in selected <i>Region</i>
SimulateVector(<i>Ckt</i>)	Simulate a <i>VectorSet</i> on given <i>Ckt</i>
ComputeActivityDifference()	Calculate <i>Activity Difference</i> of Trojan and genuine circuits for <i>VectorSet</i>
IncrementFFCount()	Increment the flip-flop count by 1
IncrementRadius()	Increment the radius by 1
Reset(<i>VectorSet</i> , <i>Regions</i> , <i>FFCount</i>)	Reset the <i>VectorSet</i> to , <i>Regions</i> to and <i>FFCount</i> to 2

4.3 Experimental Results

The results are reported in the form of activity-profile graphs. In each graph, the abscissa refers to the vector count, which can be mapped to a suitable flip-flop group. The ordinate shows the percentage difference in the circuit activity of the Trojan circuit over the actual circuit for the random vector set (shown by the blue curve and a square legend) and the vector set generated by our approach (shown by the brown curve and a diamond legend). The experimental circuits are from a subset of ISCAS'89 sequential benchmark circuits. The abbreviation *TCM* in the Figures stand for *Toggle Count Magnification*.

4.3.1 s444

Results for the Toggle Count Magnification process for circuit s444 for a varying set of selected *radii* and varying count of flip-flops included in the target *regions* are displayed in Figure 4.2, 4.3 and 4.4. From the plots, it is evident that the relative percentage difference in the activity is way above the process variation (which is around 5% in average case or may be even lower in certain cases). Also there are clear *peaks* corresponding to specific vector sets which in turn indicate specific *regions* in the circuit. In Figure 4.2, there is a sustained activity for vectors 20-30 and vectors 290-300. This is to note here that the magnification peaks in these *regions* are comparable to many other *regions* in the same graph but the sustained nature is missing for other regions. The vectors mentioned above correspond to the *regions* (8, 9 and 10) and (8, 9 and 10) respectively. If we focus on Figure 4.3, the peaks of the *regions* defined by vector sets 50-60 and 70-80 are elevated compared to others. These are the *regions* containing the flip-flops (8, 9 and 10) and (12, 13, 14, 15 and 17) respectively. For Figure 4.4 also, it is clear that sustained toggle difference is seen in *regions* between vectors 270-280 (corresponding to flip-flops 8, 9, 10) and vectors 340-350 (corresponding to flip-flops 8, 9, 10 and 11). As per our observation the flip-flops 8, 9 and 10 have the maximum frequency of occurrence in the selected *regions* and indeed our Trojan is

fed from the flip-flop group 8, 9, 10 and 11. It is possible that at certain points the random vectors may give a very high peak (one reason for such behavior is accidentally triggering the Trojan) but it cannot refer to any particular location in the circuit. More so, chances of triggering the Trojan are rare so that such behavior may be rarely observed.

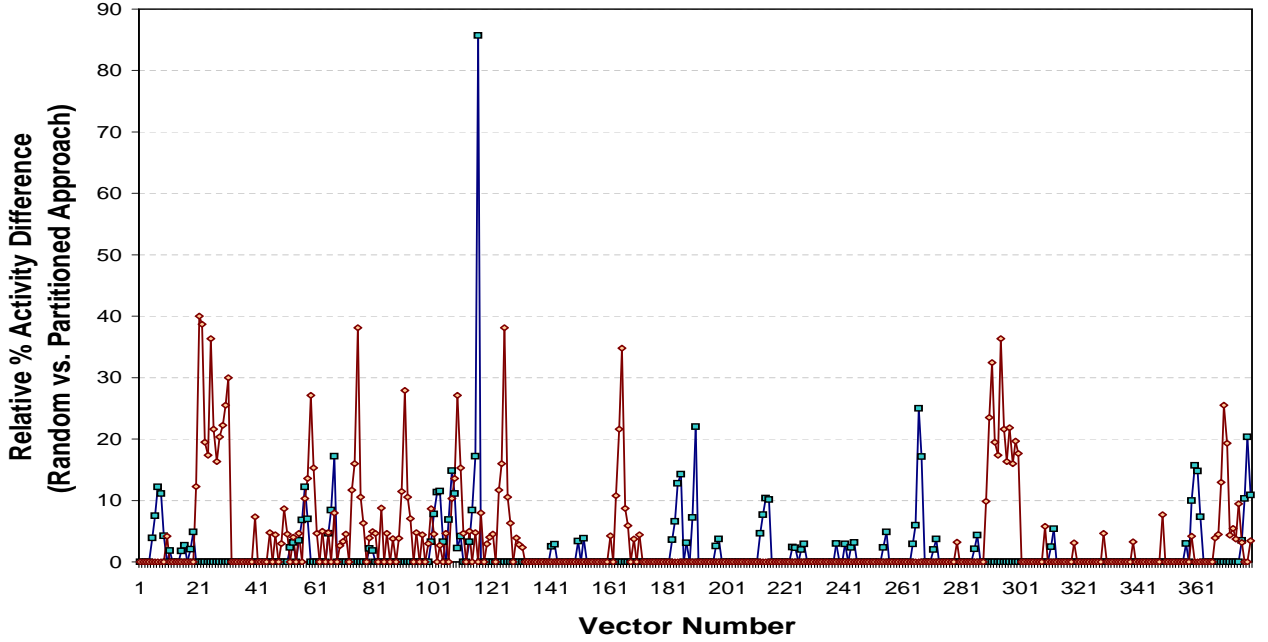


Figure 4.2: TCM(Radius 2, flip-flop Count 3) for s444

4.3.2 s1196

Results for Toggle Count Magnification for s1196 are plotted in Figure 4.5 and Figure 4.6. Figure 4.5 is for a partition *radius* of 3 while Figure 4.6 is for a partition *radius* of 4 with a maximum flip-flop count of 2 in both. Vectors 10-20, 20-30, 40-50 and 50-60 cover the *regions* of prominence in Figure 4.5. These refer to the *regions* containing the flip-flops (16 and 17), (16 and 17), (20 and 28) and (23 and 24). In Figure 4.6, vectors 50-60, 90-100, 100-110 and 120-130 cover the target *regions*. The corresponding groups of flip-flops are (16, 17 and 32), (20 and 28), (23 and 24) and (23 and 24). It is clear from the selection that flip-flops 16, 17, 23 and 24 are among the top runners in terms of frequency of count and

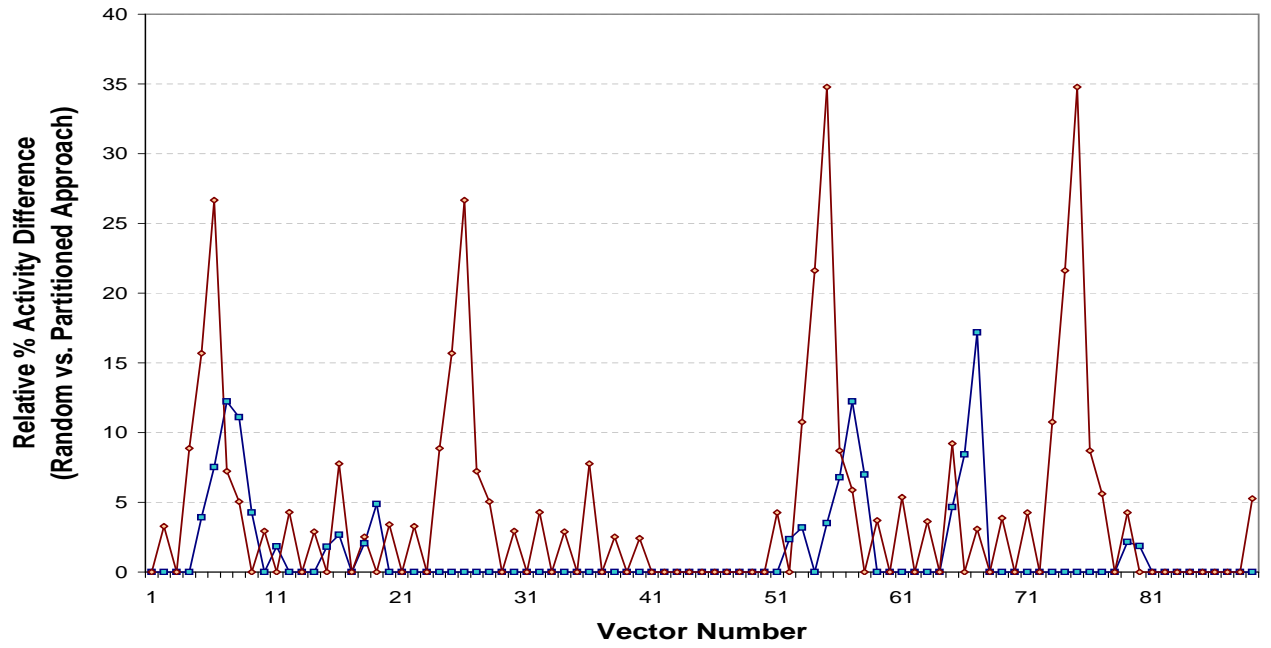


Figure 4.3: TCM(Radius 2, flip-flop Count 4) for s444

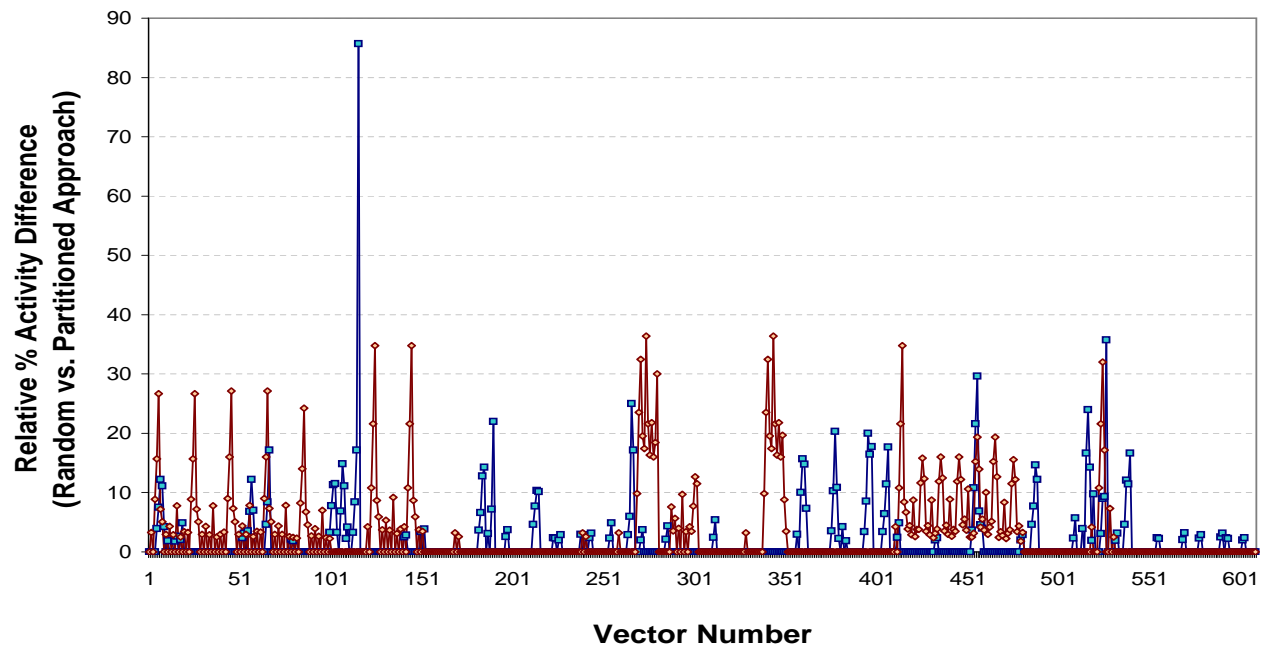


Figure 4.4: TCM(Radius 3, flip-flop Count 3) for s444

truly enough our Trojan circuit derives its input from the set (16, 17, 23 and 24).

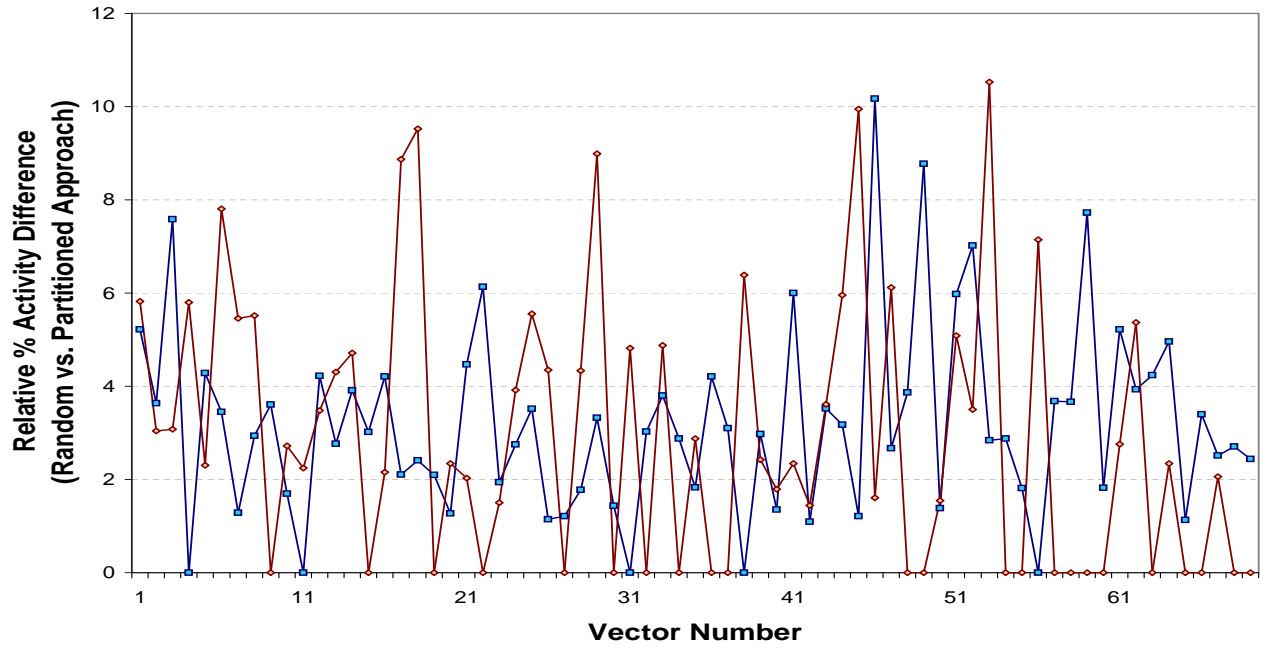


Figure 4.5: TCM(Radius 3, flip-flop Count 2) for s1196

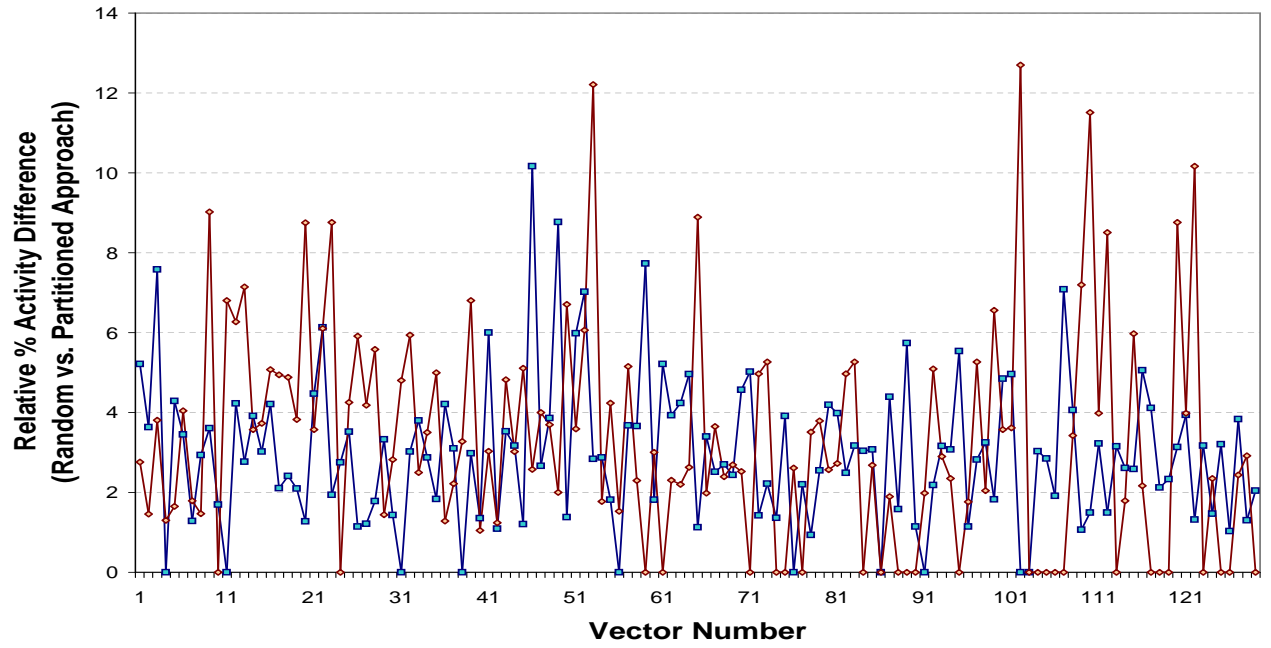


Figure 4.6: TCM(Radius 4, flip-flop Count 2) for s1196

4.3.3 s1423

For circuit s1423, the target vector sets selected for observation pertain to *regions* 10, 15 and 20 for the peaks in the Figure 4.7. The corresponding flip-flops accountable for the regions

are (70, 71, 72, 73, 74, 75, 76, 77 and 78), (18, 19, 43, 44, 45 and 46) and (38, 19, 43, 44, 45 and 46). Clearly, the flip-flops 43, 44, 45 and 46 are frequently observable and are actually connected to the Trojan part.

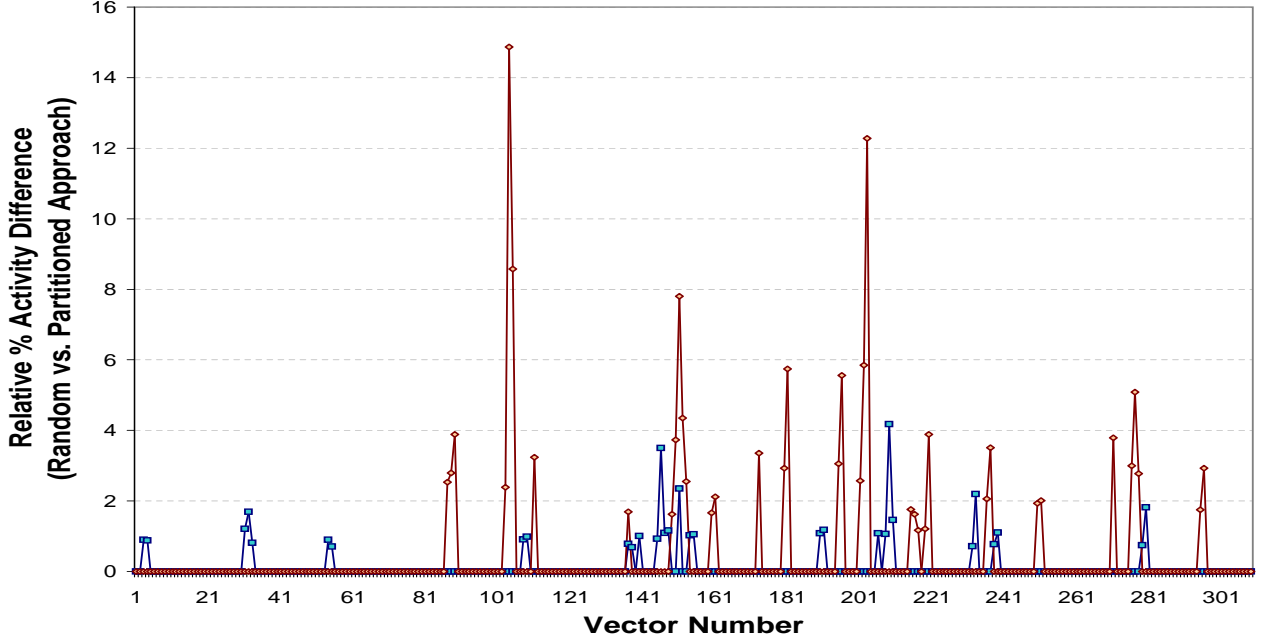


Figure 4.7: TCM(Radius 4, flip-flop Count 5) for s1423

4.3.4 s3271

Plots for s3271 shows a marked difference in the Toggle Count Difference Profile for the two (actual and Trojan infected) circuits under consideration. While the random vectors uniformly distribute the toggle difference over the entire vector sequence, our approach clearly mark out *regions* that potentially cannot contain the Trojan. Any *region* until vector 280 (in Figure 4.8) and after vector 230 (in Figure 4.9) does not give any difference in the Toggle Count between the actual and the Trojan circuit indicating that these *regions* are most likely not to contain the infected part. There is a sustained Toggle Difference count at 1.5% in Figure 4.8 after vector 320 that is a little better than the random one. For Figure 4.9, this difference is approximately 2%. In these cases, we could not pinpoint the Trojan location

because the Toggle Difference behavior is similar for many other *regions* also. In addition, the Toggle Count Difference is low as compared to the process variation so that it is not guaranteed that these effects can be surely visible under actual testing conditions. Unlike s1423, this is one of the high toggling circuits (in which there are many toggles between any vector pair) and so the relative toggle alleviating effect for Trojan portion proposed in our method is suppressed. A future work in this context can focus on ways to minimize the circuit activity of such hyperactive circuits.

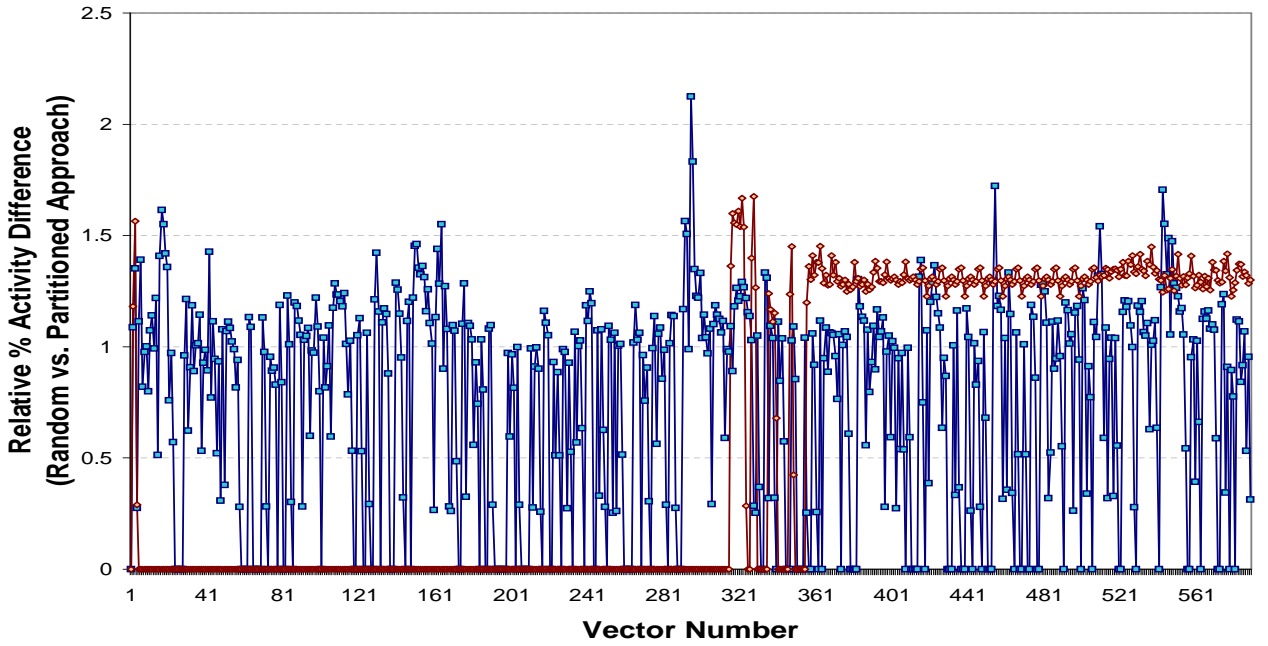


Figure 4.8: TCM(Radius 3, flip-flop Count 3) for s3271

4.4 Summary

In this work, we have presented a simple yet effective approach for isolating and distinguishing circuit portions accountable for embedded Trojans. In the process, we have devised an algorithm for non-destructive testing of ICs that have the risk of being tampered by the third party manufacturer. Experimental results show that our method utilizes the candidate region search to give a very close approximation of the infected regions. Further, the switching

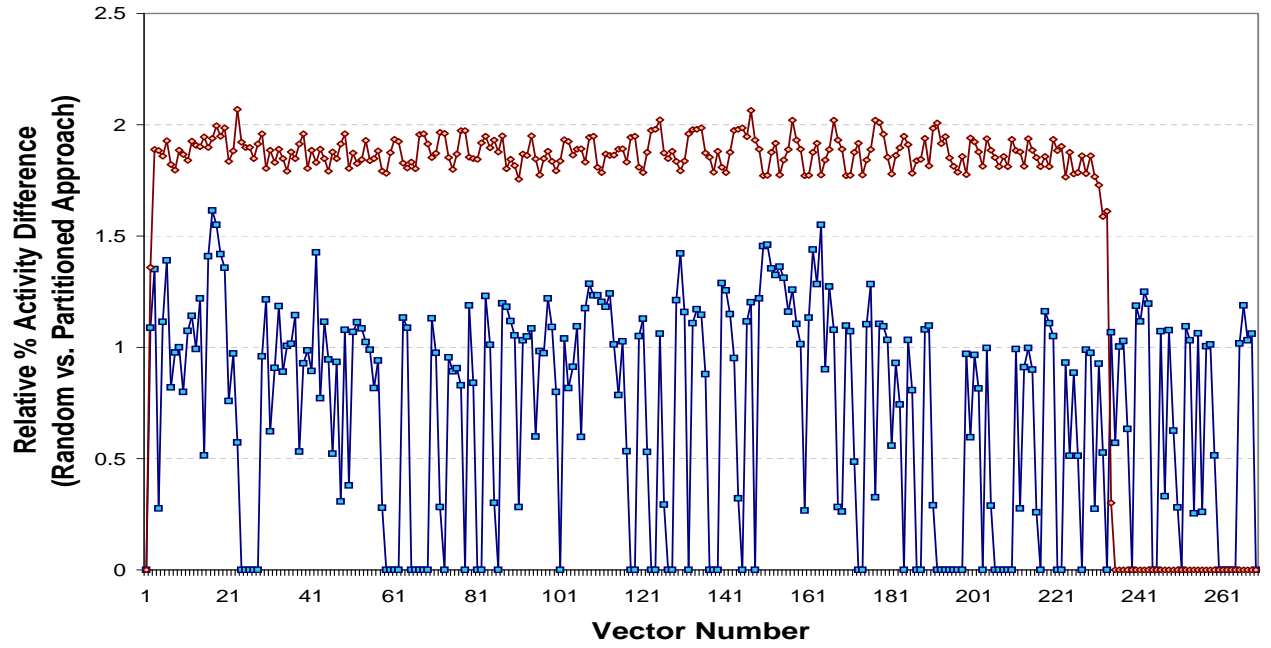


Figure 4.9: TCM(Radius 4, flip-flop Count 5) for s3271

activity based analysis results in creating the difference between the actual and the Trojan circuit which is above the process variation and hence easily observable. Our method works for circuits with moderate gate counts having a sequence detector Trojans embedded in it. Future work in this area will be to devise an approach to handle circuits that have inherent nature of being highly active so that the activity difference in those circuits can be projected above the process variation.

Chapter 5

Conclusion & Future Work

In this thesis, we have addressed one of the most recent challenges faced by the semiconductor companies - the question of trust and integrity of an IC that has been imported from an overseas fabrication center. The low cost components trend in the embedded market have forced the design organizations to outsource fabrication units to third parties. However, at the same time they have realized that the integrity of the overall system needs to be ensured because the parts can be used in systems where the security cannot be compromised. We have outlined the existing security measures that are employed to enhance the trust in a chip. Encryption schemes, watermarking, PUFs, scan-chain encryption and security engineering have been talked about in *Chapter 1*. In *Chapter 2* we have defined *Trojan*, the extraneous parts that are maliciously embedded within the genuine parts to deter their performance. We have discussed in detail about the classification of Trojans, their characteristics, side channel analysis and other tools used for non-destructive characterization of ICs.

In *Chapter 3* we proposed our first partition based approach to isolate and detect the Trojans embedded in an IC. This method is based on a state-space partition scheme and the two-staged approach is successful in isolating the behavioral difference between a genuine IC and a malignant one. The heuristic based on hamming distance maximization and minimization has been effectively used to traverse the state space locally as well as globally thereby exercising

all portions of the design logic. Experimental results on the ISCAS'89 benchmark circuits show that this methodology exaggerates the power difference in the original and Trojan infected circuit to an extent in excess of process variation.

In *Chapter 4* we proposed our second scheme of partition based mechanism for improving the isolation and detection of malicious parts within an embedded design. This idea is based on the logical integrity of a Trojan. The intelligent and vigilant nature of the Trojan was exploited to probe out the affected parts from the circuit. *Region based partitioning* backed up by an intelligent set of stimuli proved very effective in differentiating the behavioral difference and project it above the process variation that is reflected in the results where we were able to get more *peaks* in the differential power profile plots.

This problem is still an ongoing area of research. Methods of minimizing the activity in certain *hyperactive* circuits remain an open question. Combinatorial explosion is also another problem. For very large circuits, *Region based approach* can incur a substantial runtime penalty. To avoid such time penalties, one can choose to use a crude partitioning approach as discussed in *Chapter 3*, stage 1 and then refine it further to magnify the difference in the Trojan behavior. But this is only one way of looking at it. There can be potentially more research possible in this direction. This thesis concentrates on sequence-detector Trojans because these are the most intelligent and hard-to-detect kind of Trojans. Other Trojans which are solely intended to disrupt the normal activity in the SoCs can be investigated. Last but not the least, in face of decreasing geometries and rising process variations there is a call for even more enhanced techniques that can further raise the behavioral difference which makes it an interesting and challenging issue.

Bibliography

- [1] D. Agarwal, S. Baktir, D. Karakoy, P. Rohatgi and B. Sunar, *Trojan Detection using IC Fingerprinting*, IBM Research Report, 2006.
- [2] K. Nowaka, G. Carpenter, F. Gebara, J. Schaub, D. Agarwal, P. Rohatgi, W. E. Hall, S. Baktir, D. Karakoyunlu and B. Sunar; *IC Fingerprinting and Stable IC Sensors for Enhanced IC Trust*, Government Microcircuit Applications and Critical Technology Conference, March 2007.
- [3] M. Banga, M. Chandrasekar, L. Fang and M. Hsiao, *Guided Test Generation for Isolation and Detection of Embedded Trojans in ICs*, ACM Great Lake Symposium on Very Large Scale Integration, 2008, pp - 363-366.
- [4] M. Banga and M. Hsiao, *A Region Based Approach for the Detection of Hardware Trojans*, IEEE Int. Wkshop on Hardware-Oriented Security and Trust, 2008, pp 43-50.
- [5] S. Pilli and S. Sapatnekar, *Power estimation considering statistical IC parametric variations*, ISCAS 1997, pp. 1524 - 1527, vol.3.
- [6] C. Fagot, O. Gascuel, P. Girard and C. Landrault, *On Calculating Efficient LFSR Seeds for Built-In Self Test*, Proc. Of European Test Workshop, 1999, pp 7-14.
- [7] G. Hetherington, T. Fryars, N. Tamarapalli, M. Kassab, A. Hassan and J. Rajski, *Logic BIST for large industrial designs: real issues and case studies*, ITC, 1999, pp. 358-367.

- [8] W. T. Cheng, M. Sharma, T. Rinderknecht and C. Hill, *Signature Based Diagnosis for Logic BIST*, ITC 2006, Oct. 2006, pp. 1 - 9.
- [9] L. J. Kohout, A. Yasinsac and E. McDuffie, *Activity profiles for intrusion detection*, Fuzzy Information Processing Society, 2002. pp. 463 - 468.
- [10] D. Agarwal. et al, *The EM side-channel(s)*, CHES 2002, Lecture Notes on Computer Science, Springer-Verlag, pp. 29-45, 2002.
- [11] A. L. D'Souza and M. Hsiao, *Error diagnosis of sequential circuits using region-based model*, Proceedings of the IEEE VLSI Design Conference, January, 2001, pp. 103-108.
- [12] M. A. Williams, *Anti-Trojan and Trojan Detection with In-Kernel Digital Signature testing of executables*, Technical report, Security Software Engineering: NetXSecure NZ Limited, April 2002.
- [13] W. Li, S. M. Reddy and I. Pomeranz; *On reducing peak current and power during test*, Proc. IEEE computer society annual symposium, 2005, pp. 156 - 161.
- [14] F. N. Najm, *Transition density: a new measure of activity in digital circuits*, IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems, Vol 12, Issue 2, Feb. 1993 pp. 310 - 323.
- [15] C. H. Kim and J. J. Quisquater, *How can we overcome both side channel analysis and fault attacks on RSA-CRT?*, Workshop on Fault Diagnosis and Tolerance in Cryptography, 2007, pp. 21 - 29.
- [16] O. X. Standaert, E. Peeters, G. Rouvroy and J. J. Quisquater, *An overview of power analysis attacks against field programmable gate arrays*, Proc. IEEE, Vol 94, Issue 2, Feb. 2006, pp. 383 - 394.
- [17] D. P. Vallett, *An overview of CMOS VLSI failure analysis and the importance of test and diagnostics*, Proc. International Test Conference, 1996, pp. 930.

- [18] X. Wang, M. Tehranipoor and J. Plusquellic, *Detecting Malicious Inclusions in Secure Hardware: Challenges and Solutions*, International Workshop on Hardware Oriented Security and Trust, 2008, pp. 15-22.
- [19] R. Rad, J. Plusquellic and M. Tehranipoor, *Sensitivity Analysis to Hardware Trojans using Power Supply Transient Signals*, International Workshop on Hardware Oriented Security and Trust, 2008, pp. 3-7.
- [20] J. Li and J. Lach, *At-Speed Delay Characterization for IC Authentication and Trojan Horse Detection*, International Workshop on Hardware Oriented Security and Trust, 2008, pp. 8-14.
- [21] Y. Jin and Y. Markis, *Hardware Trojan Detection Using Path Delay Fingerprint*, International Workshop on Hardware Oriented Security and Trust, 2008, pp. 54-60.
- [22] J. Guajardo, S. S. Kumar, G.-J. Schrijen and P. Tuyls, *Physical Unclonable Functions and Public-Key Crypto for FPGA IP Protection*, Int. Conf. on Field Programmable Logic and Applications, Aug. 2007, pp. 189 - 195.
- [23] B. Gassend, D. Clarke, M. van Dijk and S. Devadas, *Controlled physical random functions*, 18th Annual Proceedings of Computer Security Applications Conf., Dec. 2002, pp. 149 - 160.
- [24] E. Ozturk, G. Hammouri and B. Sunar, *Physical unclonable function with tristate buffers* IEEE International Symposium on Circuits and Systems, May 2008, pp. 3194 - 3197.
- [25] F. Wolff, C. Papachristou, S. Bhunia and R. S. Chakraborty, *Towards Trojan-Free Trusted ICs: Problem Analysis and Detection Scheme*, Design, Automation and Test in Europe, Mar 2008, pp. 1362 - 1365.

- [26] N. Wu, Y. Qian and G. Chen, *A Novel Approach to Trojan Horse Detection by Process Tracing*, IEEE International Conference on Networking, Sensing and Control, Apr 2006, pp. 721 - 726.
- [27] D. Real, C. Canovas, J. Clediere, M. Drissi, F. Valette, *Defeating classical Hardware Countermeasures: a new processing for Side Channel Analysis*, Design, Automation and Test in Europe, Mar 2008, pp. 1274 - 1279.
- [28] S. Voloshynovskiy, S. Pereira, T. Pun, J. J. Eggers and J. K. Su, *Attacks on digital watermarks: classification, estimation based attacks, and benchmarks* Communications Magazine, IEEE Volume 39, Issue 8, Aug. 2001, pp. 118 - 126.
- [29] P. Loo and N. Kingsbury, *Watermark detection based on the properties of error control codes*, IEE Proceedings on Vision, Image and Signal Processing, Volume 150, Issue 2, Apr 2003, pp. 115 - 121.
- [30] A. Cui and C. H. Chang, *Intellectual property authentication by watermarking scan chain in design-for-testability flow*, IEEE International Symposium on Circuits and Systems, May 2008, pp. 2645 - 2648.
- [31] D. Hely, M. L. Flottes, F. Bancel, B. Rouzeyre, N. Berard and M. Renovell, *Scan design and secure chip [secure IC testing]*, IEEE International On-Line Testing Symposium, Jul 2004, pp. 219 - 224.
- [32] <http://en.wikipedia.org> : *Wikipedia*
- [33] <http://www.intel.org> : *Intel Website*