October 2018

Gate Level Probabilistic Simulation Based Hardware Trojan Susceptibility Analysis of Combinational Circuits

Venkata Lakshmi Bhargavi Gurram
University of South Florida, gurramv@mail.usf.edu

Gate Level Probabilistic Simulation Based Hardware Trojan Susceptibility Analysis of Combinational Circuits

by

Venkata Lakshmi Bhargavi Gurram

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Computer Science
Department of Computer Science and Engineering
College of Engineering
University of South Florida

Major Professor: Srinivas Katkoori, Ph.D.
Robert Karam, Ph.D.
Hao Zheng, Ph.D.

Date of Approval:
October 19, 2018

Keywords: Logic Simulation, Hardware Security, Trojan Trigger, Very-Large-Scale-Integration

DEDICATION

I dedicate this work to my family and all my beloved ones who helped and supported me.

ACKNOWLEDGMENTS

I would first like to formally express my deep gratitude to Dr. Srinivas Katkoori for providing the opportunity to work on this project. I always cherish his support and express my gratitude that he has helped me identify my strengths and use them in a best possible way. I am forever grateful to him for bringing me success with his guidance and constant support. I would like to thank my parents, whose love and support are with me in whatever I pursue. I would like to thank Dr. Hao Zheng and Dr. Robert Karam for spending their precious time to serve as members on my thesis committee. I would like to thank Sheikh Ariful Islam for his assistance throughout this work. I would like to thank God for shielding me to pursue a fruitful path. I would also like to thank my friends and colleagues I have associated with for their help and assistance.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Today multi-million gate integrated circuits are being commonly used in many critical (eg., health care) and sensitive (eg., military) applications. Therefore, they are susceptible to malicious modifications, namely, Hardware Trojans (HTs), with the intent of leaking sensitive information, denial-of-service etc. There are many ways of inserting Hardware Trojans in ICs. Thus, HT detection and mitigation are very important tasks. In this thesis, we propose a novel probabilistic simulation based approach to estimate the susceptibility of a combinational circuit to a HT. One of the common ways is to simulate the netlist with typical input sequences and identifying low activity nets in the design that could be exploited for Hardware Trojans. This approach has the drawback of excessive simulation time. Probabilistic simulation is very efficient as the input sequences are condensed into probabilistic waveforms which are then propagated through the netlist to identify low activity nets. The main advantage of the proposed technique is rapid analysis of a netlist for HT susceptibility. We experimented with eleven MCNC benchmarks and demonstrate significant speedup (approximately 2 x 10^3 to 5 x 10^3) over traditional simulation based approach.

CHAPTER 1: INTRODUCTION AND MOTIVATION

Identifying hardware trojans (HTs) in a circuit has become a huge challenge for the designers. The conventional simulation approaches are not sufficient to address this problem effectively, as an typical HT has a small footprint in the circuit and is designed such a way that it is rarely triggered. In this thesis, we propose a new and promising approach that employs probabilistic simulation to quickly analyze the susceptibility of a design to HTs designed with low activity nodes in the circuit.

1.1 Thesis Motivation

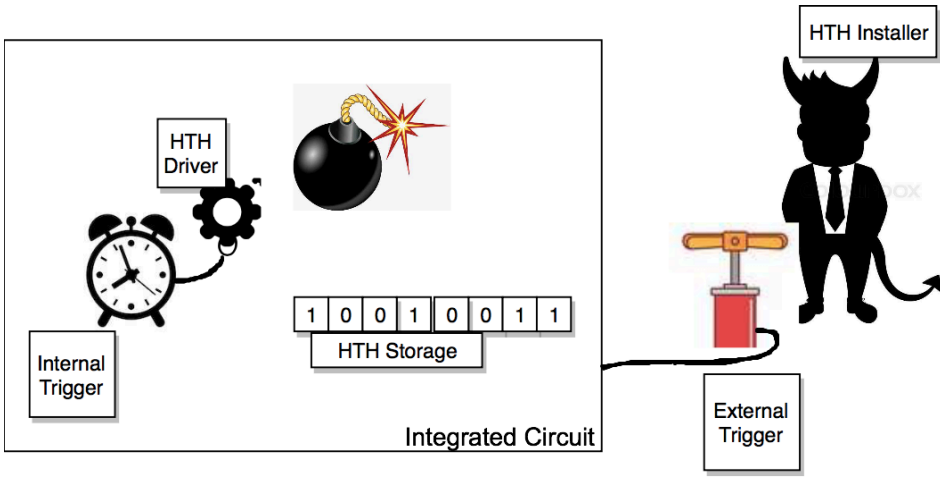

Figure 1.1: Components of a Hardware Trojan Horse

The changes made by antagonist/adversary to abuse the hardware system to achieve access to the information or the design flow in the circuits are termed as Hardware Trojans [1]. A smart antagonist conceals the manipulation of the integrated circuit. This makes it very hard to identify

Hardware Trojan during traditional testing phase of the system design. These adversaries will be very careful to hide this behavior and thereby can be provoked only in unusual conditions of the nodes. Therefore, these are very uncertain to be identified during traditional testing. Therefore, there is a need to develop a faster technique to test the design that helps in consistent testing of the system to avoid Hardware Trojan injection in the circuit.

Usually the Hardware Trojans are of two types [2], *combinational* and *sequential*. The combinational trojan circuit as shown in Figure 1.2 relies upon the existence of unusual states in different nodes concurrently. Whereas in sequential trojan, prior to provoking a fault, a sequence of state transfers occurs as shown in Figure 1.3.

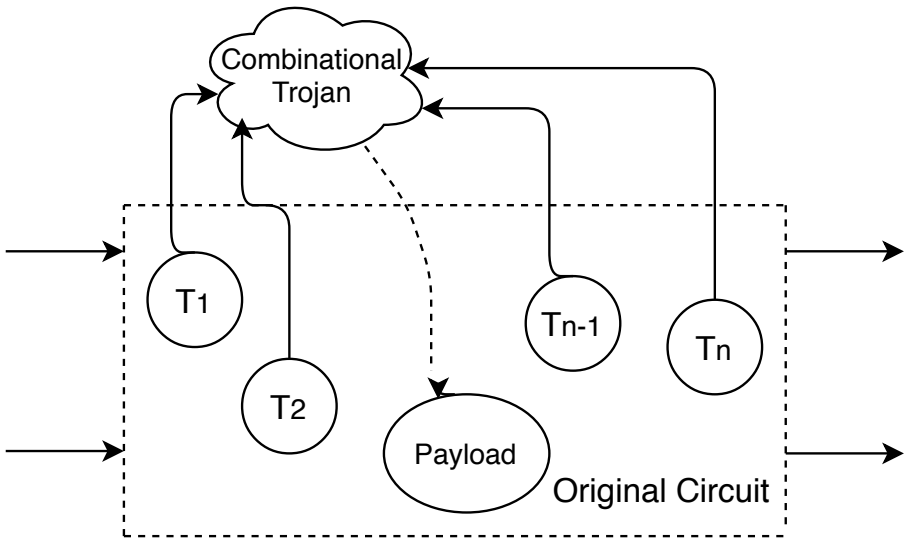

Figure 1.2: Combinational Hardware Trojan

1.2 Proposed Approach Overview

In this thesis, we propose a novel probabilistic simulation based approach to estimate the susceptibility of a combinational circuit to a HT. One of the common ways is to simulate the netlist

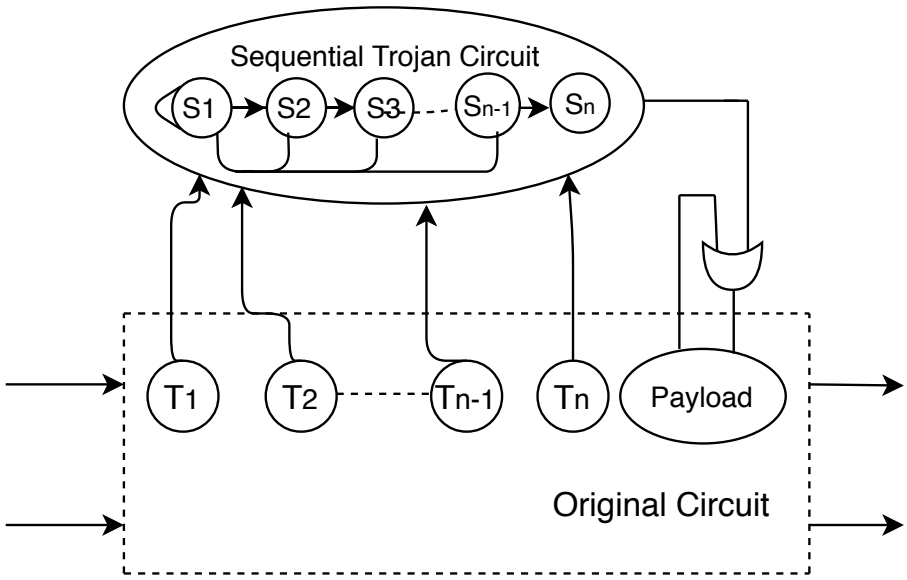

Figure 1.3: Sequential Hardware Trojan

with typical input sequences and identifying low activity nets in the design that could be exploited for Hardware Trojans. This approach has the drawback of excessive simulation time. Probabilistic simulation is very efficient as the input sequences are condensed into probabilistic waveforms which are then propagated through the netlist to identify low activity nets. The main advantage of the proposed technique is rapid analysis of a netlist for HT susceptibility.

1.3 Experimental Results Overview

We experimented with eleven (11) MCNC benchmark suite and demonstrate significant speedup over traditional simulation based approach. The probabilistic simulation was implemented in C. Three different lengths of random vector sequences are applied, namely, 10K, 50K, and 100K and the average speed ups obtained are 5516, 4050, and 2277 respectively.

1.4 Thesis Organization

The rest of the thesis is organized as follows. Chapter 2 presents the concepts of signal and transition probabilities, simulation techniques, hardware trojans and various techniques used to resist reverse engineering at different levels. Chapter 3 describes in detail the gate-level probabilistic simulation to estimate hardware trojan susceptibility. Chapter 4 reports experimental results of the proposed approach. Finally, Chapter 5 draws conclusions and outlines future directions.

1.5 Chapter Summary

In this Chapter, we introduced the concept of hardware trojan, its types, and the difficulty of detecting them in circuits with traditional simulation. We gave an overview of the proposed approach and the experimental results obtained.

CHAPTER 2: BACKGROUND AND RELATED WORK

In this Chapter, we analyse Hardware Trojans and review different methods for detection of Hardware Trojans. We provide the background on probability waveforms and introduce probabilistic simulation related terminology.

2.1 Hardware Trojans

In the recent times, security and protection of Integrated Circuits (ICs) has become a great challenge because of obtaining the fabrication of IC's from foundries that are unreliable. The IP Cores designed in off-shore foundries and third party CAD tools have become threat to logical design due to potential malicious functionality in them. This kind of malicious behavior in the design is known as Hardware Trojan. These hardware Trojans are injected in the circuit to disturb the normal functionality [3] of the circuits. A hardware trojan horse to be inserted in a logic circuit has three components, *Trigger*, *Driver*, and *Storage*.

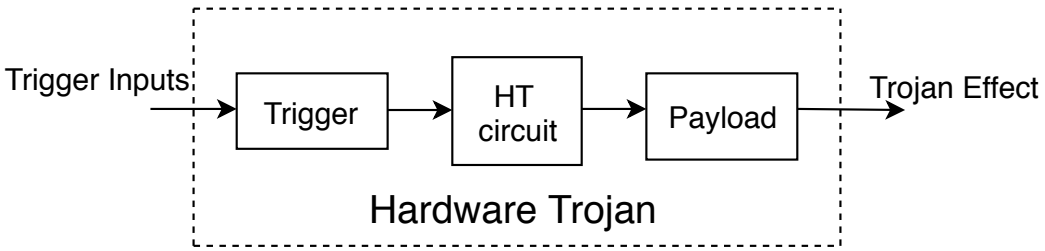

Figure 2.1: Typical Structure of Hardware Trojans

The Trigger in the design encourages the activities of a hardware Trojan horse by attaining externally by using the stream of inputs or internally through a clock. The activities after existence

of trigger are saved in a storage component externally or with a serial circuit [4] embedded internally to a logical circuit. All these activities initialized by trigger are performed by the driver component. Most of the time these Trojan activities [5] are not visible to ordinary users that connect with the system. Therefore creating suitable test cases in order to identify hardware Trojans in the circuit is more challenging because these hardware Trojans are fabricated to be provoked at unusual constraints of the inputs to the circuit.

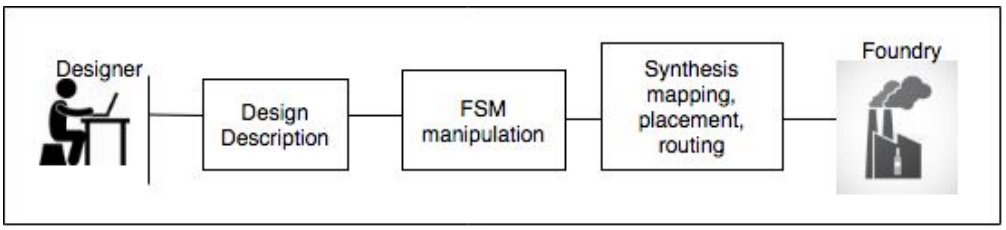

Figure 2.2: Design Process for Hardware Trojan

Figures 2.3 and 2.4 represents the Hardware Trojans having trigger conditions with four and eight primary input nodes. The combinational circuit as in Fig. 2.3 activated only when its input is “0101" and the combinational circuit as in Fig. 2.4 activated only when its input is “10011100."

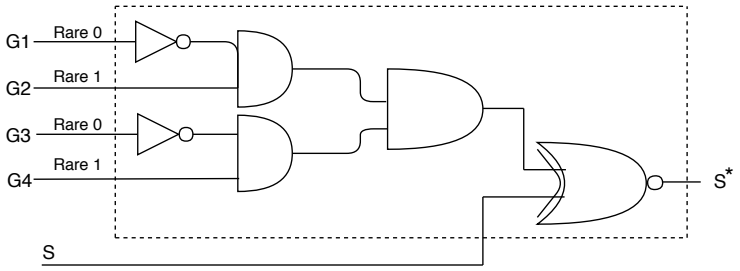

Figure 2.3: Four Trigger Hardware Trojan

2.1.1 Hardware Trojan Threat and Susceptibilities

The segment of circuit in addition with the original logic circuit used for harmful causes is defined as a hardware Trojan. The figure 2.1 describes hardware Trojan. It carries payload,

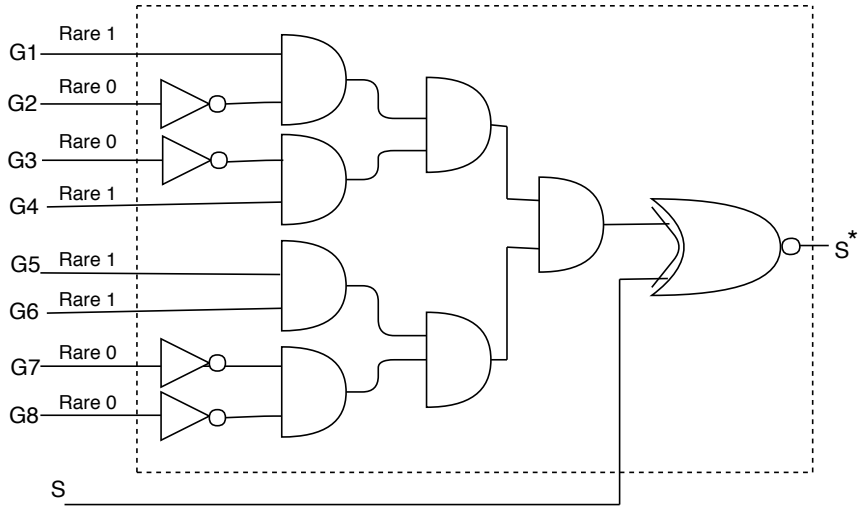

Figure 2.4: Eight Trigger Hardware Trojan

Hardware Trojan Circuit and a trigger. Usually these hardware Trojan circuits are maintained to be calm for caching in the circuit and these are remained to be active when ever a particular signal or event is triggered. Then the payload circuit applies the attacks [6] caused by hardware Trojans in the circuit. These attacks cause undesirable effects such as sensitive data leakage, DoS (Denial of Service). In real time, there are many hardware Trojan models that have varied actuation procedures.

- Integrated Circuit commerce model: Five different parties are included in this Integrated Circuit model in its fabrication, design and flow of its implementation. Figure 2.5 depicts the responsibilities shared between these parties through their interaction. Here SoC manufacturers fabricates and designs enterprise merchandise that has different Internet Protocol (IP) addresses. The central components such as DSP cores, memory storage, etc., for IC's are provided by IP vendors and foundries combines with these Soc manufacturers to design Integrated Circuits (ICs). The EDA used by Soc manufacturers and IP suppliers to assist the modeling of extensive IC's are provided by EDA vendors. Finally, the institutions or individual persons

that leverage merchandise from SoC manufacturers are the end users of Integrated Circuits. Normally all the parties involved here provide their machinery and design to all other parties in this commerce model. In particular, the SoC manufacturers contain links with all the other parties in the Integrated circuit commerce model.

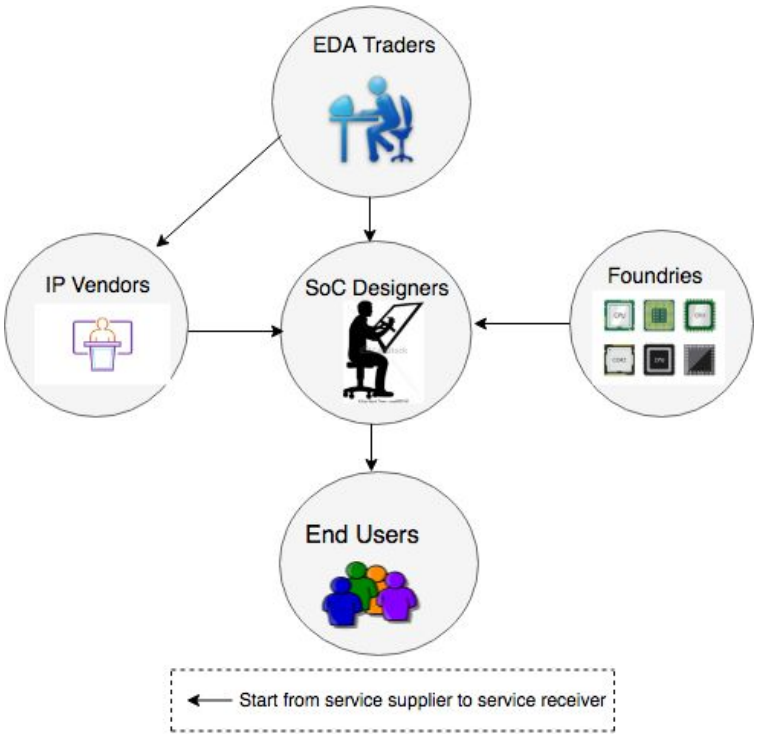

Figure 2.5: Prototype for Modeling Integrated Circuit

- Hardware Trojan threats in Integrated Circuit model: The issues caused by Hardware Trojans during the communication among different parties in IC commerce model are discussed here. There is no assurance that Hardware trojan will not be placed in circuit by foundries while manufacturing IC's during the interaction between the SoC manufacturers and foundries. These hardware Trojans can be inserted by any malicious workers or the hackers from other third parties consciously or unknowingly. These can also be injected during the interaction between IP suppliers and SoC designs or IP suppliers and EDA traders. These are many

chances of various categories of hardware Trojan injection into these stages. This can be achieved in IP suppliers and SoC manufacturers by the suspicious individual in the IP providers team. It it very simple for adversaries in the team to exploit RTL (Register Transfer Level) and perform malign activities like modifying the arrangement in the chip, injecting harmful codes etc. These adversaries hide these activities during fabrication and there by it will be hard for anyone to suspect these in future.

- Hidden channels: The drivers of Hardware Trojan circuits are hidden from the end users. These Hidden (Covert) Channels are designed to be placed into the resultant path of a changed circuit in order to get the confidential information. The general method to execute these hidden channels is to change the specific features of a commonly shared path in a more unusual way. By employing such changes through hidden channels, the data is received by the adversary by not letting the modification noticed by anyone.

2.1.2 Measures to Avoid Hardware Trojan

Here we discuss various steps involved to assist for the risks caused by Hardware Trojans as we mentioned. Figure 2.6 delineated three steps involved in Hardware Trojan identification and prevention. The principal steps involved are the Trojan Detection, diagnosis and Prevention of Hardware Trojans. Hardware Trojan Detection method identifies whether there is any Hardware Trojan present in the circuit or not and the Diagnosis phase decides the actual position where the Hardware Trojan is present. This helps in eliminating the Hardware Trojans from the circuit or dissembles them from the logical circuit. The prevention phase encourages to use techniques to avoid Hardware Trojans at the initial stages of design of the logical circuits.

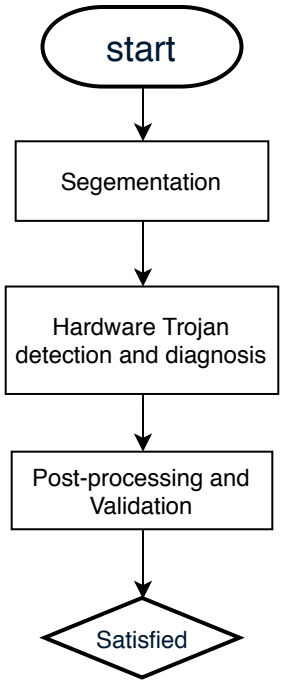

Figure 2.6: Hardware Trojan Detection and Diagnosis Flow

1. Hardware Trojan Detection: Detecting Hardware Trojans in modern circuits is really a big challenge where Integrated Circuit permits are misused by the adversaries through low-level procedures. Though there are different approaches to overcome this problem by helping the end users. The detection procedures of hardware Trojans are classified into two fundamental categories:

    - Destructive Method: Here, the manipulations are detected by injecting micro photographs of levels of chip through de-metallizing all these levels in a fabricated Integrated Circuit (IC). But this technique works by believing that there is a malicious attack for a small amount of arbitrary fragments in the scope of fabrication. The choice of perfectly fabricated Integrated Circuit design is the way to recognize the Trojans. But the main drawback of this kind of approach is more costly in scope of time and price. Verifying a

single Integrated Circuit also takes months. Therefore it became important to examine non-destructive techniques for identifying the Trojans.

- Non-destructive Method: Generally these non-destructive methods [7] are mainly classified into two categories.

a. Logical test case: This logical testing in contrast with the Side-Channel Evaluation are completely dependent on changes in processes and noise impacts. The complex problem in this kind of test is an opponent can use an excessive amount of Trojans. Therefore, logical testing can detect few Trojans in order to solve the problems comparably. The design approach for identifying the infused Trojans in the circuit by logical testing can be achieved by inserting a significant circuitry in the Integrated Circuit. Though the circuits that does not have this kind of design pattern can not be used for identifying the Trojans in the circuit.

b. Side-Channel Evaluation: Where as in Side-Channel Evaluation it is concerned more about the changes caused in architectural pattern or substantial components like passage obstruction, power consumption in Integrated Circuit model. We can notice the variations in waveforms at primary outputs rather than eliciting at malicious actions. In contrast with all the other side-channel analysis techniques, hardware Trojan detection [8] can be attained better by using path delay method. Where each path in the circuit are different and independent with each other, the delay in the passage can be measured computed distinctly. But the drawback of this kind of technique is that we can observe huge malicious alterations whereas tiny Trojan insertions can not be determined using side-channel analysis. The high-level flow chart of MERO approach to identify Hardware Trojans is described in Figure 2.7. This figure illustrates the identification of unusual events and its evaluation using test-cases.

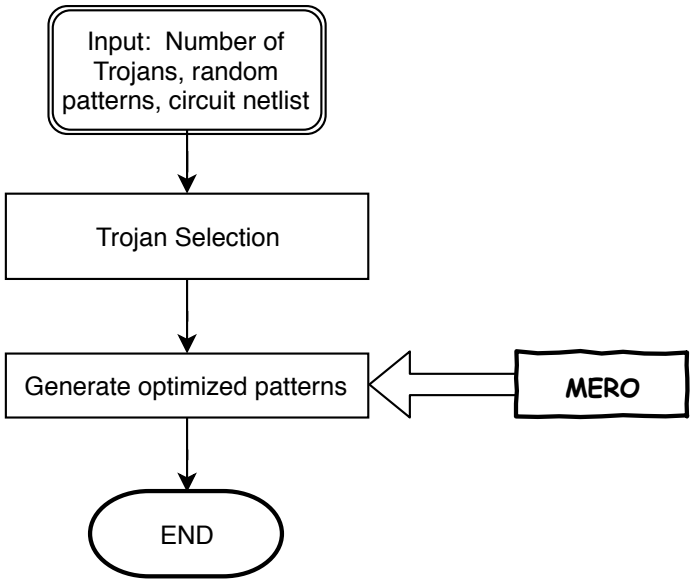

Figure 2.7: MERO Approach for Detecting Rare Events (Trojan Insertions)

2. Hardware Trojan Diagnosis: There are many different Hardware Trojan detection techniques available today. But the particular data regarding the hardware Trojans like the category and region of Hardware Trojans and also the triggers of Hardware Trojans are needed by the SoC manufacturers or the IP suppliers that helps in discarding the Hardware Trojans from the logical circuits.

   Here we discuss the techniques that are used for diagnosing Hardware Trojans. This kind of diagnosis technique is dependent of GLC (Gate Level Characterization) and analysis, segmentation of the circuit. Hardware Trojan Segmentation or analysis, Hardware Trojan Detection, and Hardware Trojan Diagnosis are the three main stages involved in this method as depicted. The representation of analysis is designed to be first prepared by the ratios of association, controllability and also by the precision of Gate Level Characterization. Then, the next step is to partition the large logical circuit into smaller circuits. Then, it is simpler for detection and diagnosis of Hardware Trojans in these smaller sub-circuits. The analytical logic is imple-

mented here as the next step in this procedure. All the steps in this system can be iterated as many number of times as required.

The hardware Trojan diagnosis can also be done using another procedure that is more focused on the detailed examination of uniformity, regularity and segmentation of features, and properties at gate-level. By using this approach, by calculating the properties at gate-level anyone can detect the Hardware Trojans with converging gates. The leakage of capacity can be revealed in these gates. Here, another fragment is initiated to specify the location of the Hardware Trojan.

3. Hardware Trojan Prevention: Though the procedures discussed in the above sections- detection and diagnosis of Hardware Trojans are favorable, they have some problems that includes the recognition of the infrequent node, the fluctuations in the process flow and the divergence of the computations. So, in order to upgrade these procedure for increasing their advantages, it is important to manufacture the Integrated Circuits that maintain self-defense. Today Trojan prevention is introduced in many models such as design or layout-filler, obfuscation, dummy inclusions in logical circuits and split fabrication.

   BISA (Built in Self Authentication) model can also be used for Hardware Trojan prevention by prohibiting the inclusion of auxiliary Trojan gates into the design of logical circuit. The method of Layout filling assists the prevention of Hardware Trojan by minimizing the probability of malicious inclusions in the circuit by occupying the free space in the circuit with some useful logic. The other way is recognizing the extra stock at the level of design inside the device. We can also achieve prevention of Hardware Trojans by the model that multiplies the infrequent node transition probabilities by placing duplicate flip-flops in the logical circuit.

The nodes that has transition probability below the threshold value are recognized initially. Then after the duplicate flip-flops are included to raise the probabilities of transition for the nodes.

2.1.3 Time Estimation for Activation of Hardware Trojans

Prior data regarding the category of the Trojans or the site of the Trojan in the circuit for that helps in authenticating are not available. Therefore, it is important to examine either the full activation time or partial activation time [9] of Trojans. The Trojans full activation time is indicated as the design pattern that affects the output of a logical circuit and that produces glitches by initiating Trojans. Whereas the partial activation is indicated by inducing one or more transitions in the logical Trojan circuit, by which we can enhance the efficacy of the methodologies. The Trojan constitutes Trigger and payload.

These Trojans are initiated under specific circumstances, there by an error is infused by the payload. The transitions occur in a logical circuit are dependent on its previous nodes. These transitions are established through the primary inputs. Generally a Trojan contains many gates, but here we consider only one Trojan gate. Logical circuit that is coupled with Trojan gate inputs is defined as a Trojan cone. The time needed to produce transitions in gate of a Trojan is regulated by this Trojan cone by determining the Trojan cone shape, the type of gate and number of gates.

2.2 Probability Waveforms

When two input vectors are considered during varied time intervals, then logic quantity for every gate may be varied for each time interval based on the factor of signal propagating paths from

initial inputs of the circuit to that particular gate. This results in the possibility for generation of huge number of discrete output waveforms at each logic gate.

The methodology of waveform probabilities extracts the input stream provided by the user along with the circuit to a specific gate in order to generate the output of the gate by considering assigned probabilities for the circuit inputs. Probability waveform representation is exhibited in the following diagram. The CREST methodology uses probability waveforms to denote the random action at each node.

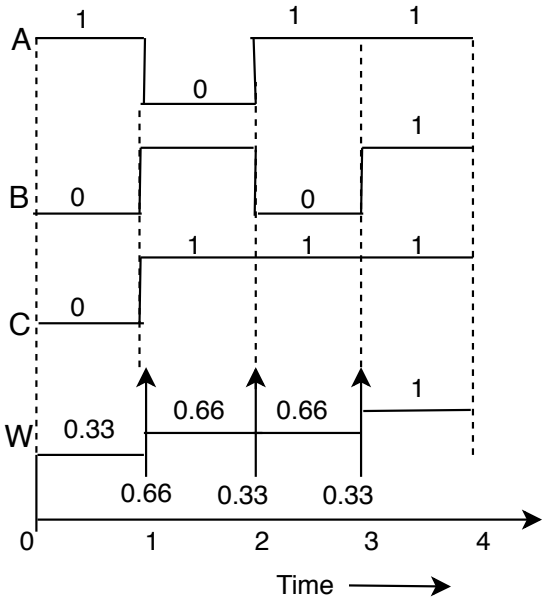

Figure 2.8: A Probabilistic Waveform (Bottom) for Three Input Logical Waveforms (Top)

Therefore, the Probability Waveforms [10] designated by the probabilistic computations along with the logical waveforms of every gate in the circuit. These probabilistic waveforms are the series of the transition events (or edges) for a particular time frame, where occurrence probability represents the events of the circuit. Each probabilistic waveform is combined with both signal probability and transition probabilities obtained by a series of transition events or edges. This waveform will known

to be as simple logic waveform when the signal probability and transition probability are defined with 1's and 0's. Therefore every probability waveform 'w' has two quantities: a) Signal Probability b) Transition Probability

2.2.1 Signal Probability

The probability assigned to a logic signal at input gate-level is defined as a *signal probability* [11]. Every input node will be assigned with some probability value that specifies the maximum occurrence of the event at that node.

- Case1: The logic signal probability for signal A can be stated as (n= Probability of N=1)
- Case2: The signal probability when N=0 is stated as P(n=0)=1-(Probability of N=1)=1-n

Here, we consider the upper case letters as the names of the logic signals and the lower case letters indicates the probabilities with respect to the logical signals.

2.2.2 Transition Probability

The probability of transition in logic signal by a node n during the time interval t is defined as a *transition probability*. The transitions in logic signals can be either upward or downward. If there is transition in probabilities from logic zero to logic one, which is known as upward transition probability tun(t) and the change of transition probability from logic one to logic zero is known as downward probability tdn(t). The upward transition probability an also be called as rising transition probability and the downward transition probability is defined as falling transition probability.

The transition probability location in a logical waveform is also called as transition edge, where this transition edge plays a crucial role in probability waveforms as it generates values through

Table 2.1: Two Input AND Gate Forcing Set Table

Output Tags	Input Tags
00	(00,00), (00,01), (00,10), (00,11), (01,00), (01,10), (10,00), (10,01), (11,00)
01	(01,01), (01,11), (11,01)
11	(11,11)
10	(10,10), (10,11), (11,10)

the circuit. These transition edges are known as events or also probabilistic events. Three different probabilities are depicted by each and every event at a node X in a logical circuit:

- $P_{X,h}(t-)$
- $P_{X,lh}(t)$
- $P_{X,h}(t+)$

These statistics help in calculating probabilities at other nodes in the circuit. For example, at node X, ability of high to low transition can directly be derived from the probabilistic identity:

$$P_{X,h}(t+) - P_{X,h}(t+) = P_{X,lh}(t) - P_{X,hl}(t)$$

2.3 Simulation Algorithm

The simulation algorithm CREST is used to disseminate the initial inputs described by the user into the circuit to obtain the predicted current waveform. This algorithm focuses and works on probabilities and does not consider the logical signals. Therefore, it is a probabilistic simulation algorithm and more deterministic itself. The strategy of integrating joint tagged waveforms is represented in Table 2.1. The state values of every input of the gate can be used to anticipate the values at the corresponding output of the gate.

The inputs of probability waveforms can be produced in two different ways. Entering the inputs as a series of absolute numbers in the range between 0 and 1 to the waveforms is one way. In a different way the former simulations logic [12] for that design is considered in a way to generate the inputs. Probability waveforms equivalent to these inputs can be acquired by considering the average of all the possible logical waveform values for each node and this final value should be ranged between 0 and 1. If there is no variation in signal at the input node, then the acquired value at that node is considered as the essential value of the probability waveform for that time interval. If there is a variation in the signal from high to low or from low to high, then the fraction that is considered by the logic waveforms for the transition at that time is considered as the essential value of the probability waveform for that node at that time interval.

The event handling simulation approach is utilized to propagate the given input waveform probabilities throughout the circuit. Splitting the circuit into gates and considering the occurrence of the event and estimating the current pulse caused by the event on each input to a gate and thereby calculating the appropriate probability at the gate output. Finally, the waveform at the output of the circuit can be generated by considering all the estimated streams from discrete gates are accounted.

2.4 Signal Dependence

Recording the dependence or correlation of the probability values is a big challenge in these kind of tools like CREST. So, the correlation in a circuit is divided into two various types in CREST model. They are:

1. Temporal Dependence: The correlation among the values of a corresponding node of a circuit.

2. Spatial Dependence: The correlation among the signals of the nodes that rely on equivalent fan-out path in the logical circuit.

Most commonly, the dependence among the signals can be both or either of these correlations. The restriction for calculations of temporal dependence by CREST is by considering the correlation among the the signal transition edge that discrete two signal edge values. So, this is the main cause for the introduction of the transition probabilities. So, to account the temporal correlations among two nodes at a waveform is same as that of the full delineation of both the random process. So keeping track of these temporal correlations can be more expensive.

Here, we consider that the inputs to the gates of a logical circuit are independent with each other. This can be contradicted when there are re-convergent fan outs in the logical circuit. So in order to control this situation we use the idea of supergates [13]. The segment of a logical circuit that contains independent inputs is known as a Supergate. The input waveform stream that has re-convergent fanout is known as re-convergent fanout inputs(RFI) for the nodes in the logical circuit. All the inputs to the nodes of the circuit are independent and the propagation through the circuit would become simple when logical circuits are allocated to the input nodes.

The notion of a *supergate* controls the spatial correlations. By diminishing this super gate, we can achieve propagation through the gates with the help of independent inputs gate solver, that has independent inputs to the gates. This gate can be propagated with the strategy of first evaluating the gate that has its inputs with particular events to generate the equivalent output. This way of producing the outputs by considering the corresponding input events is easy and fast. To do this we need to take some part p of the gate and then generate a graph. Then the gate probabilities of the transistor describes the edges of the generated graph.

2.5 Chapter Summary

In this chapter, various traditional approaches for hardware Trojan susceptibility that are being used in Integrated circuits are discussed. We presented in detail the concept of probabilistic simulation.

CHAPTER 3: PROPOSED APPROACH

In this Chapter, we describe the proposed approach for detecting Hardware Trojans by probabilistic simulation that uses gate-level probability waveforms. This process determines the gate delays in the circuits by considering signal and transition probabilities [13]. The implementation and principal features of proposed approach is explained here.

3.1 VLSI Synthesis Design Flow

Currently, as many electronic designs are using Very Large Scale Integrated (VLSI) circuits that has millions of transistors and their interconnections inside a little area. So in order to manage the complexity of circuits we follow the hierarchy illustrated in Figure 3.1. The different levels in this VLSI synthesis design flow hierarchy are explained as follows.

3.1.1 Behavioral Specification

This level describes the design of actual performance by not considering the precise time frame. The specification of this behavioral level can be assisted by various programming languages. We have implemented behavioral model for the simulator using C programming, as the simulation is faster using C by proving excellent domain for programming. Then this behavioral model has to be developed to an register-transfer level (RTL) model to implement finally as an Integrated Circuit.

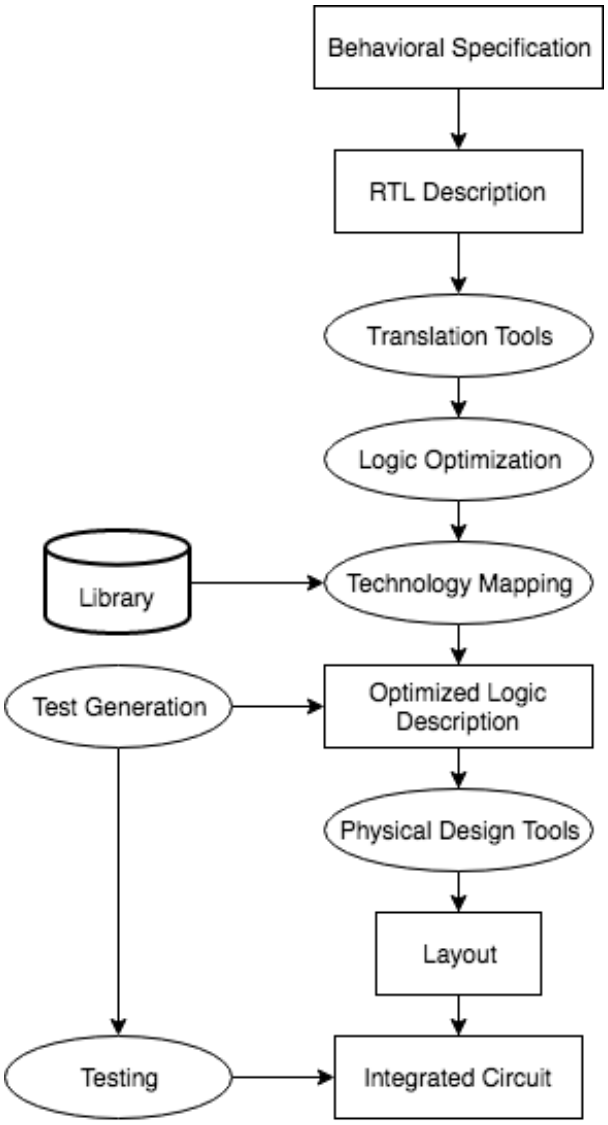

Figure 3.1: VLSI Synthesis Flow

3.1.2 RTL Description

The micro architecture of the circuit is depicted by the Register-transfer-level (RTL) [14]. We use Verilog hardware design language for explaining the register-transfer-level in our model. Here, the performance is measured by considering the functional unit transformation such as register files and arithmetic-logic units in the sequential circuit happening concurrently.

3.1.3 Logic Optimization

This level is responsible for transforming to nearer optimal performance from gate-level description obtained from RTL, focused in optimizing the space and rate of the implementation. There are two ways for implementing logic optimization. It can be two level or multi level logic optimization. The two level logic optimization applied as well organized programmable logic array (PLA) for simpler logic synthesis. Whereas in multi level logic optimization we use the transformation for boolean and algebra by producing more efficient circuits.

3.1.4 Technology Mapping

As the optimized logic net list is produced at gate-level the following step is Technology mapping, where we effectively maps obtained logic net list to a semiconductor library consisting of multiple gates from the trader. This level reviews the most of the difficulties encountered in logical optimization at high level.

3.1.5 Physical Design Tools

Here we use several physical designer tools like module generators to produce the final layout by making use of programming and layout languages. This tool also implements the data path for high level performance of ICs. These physical design tools helps in generating prominent circuits with high quality.

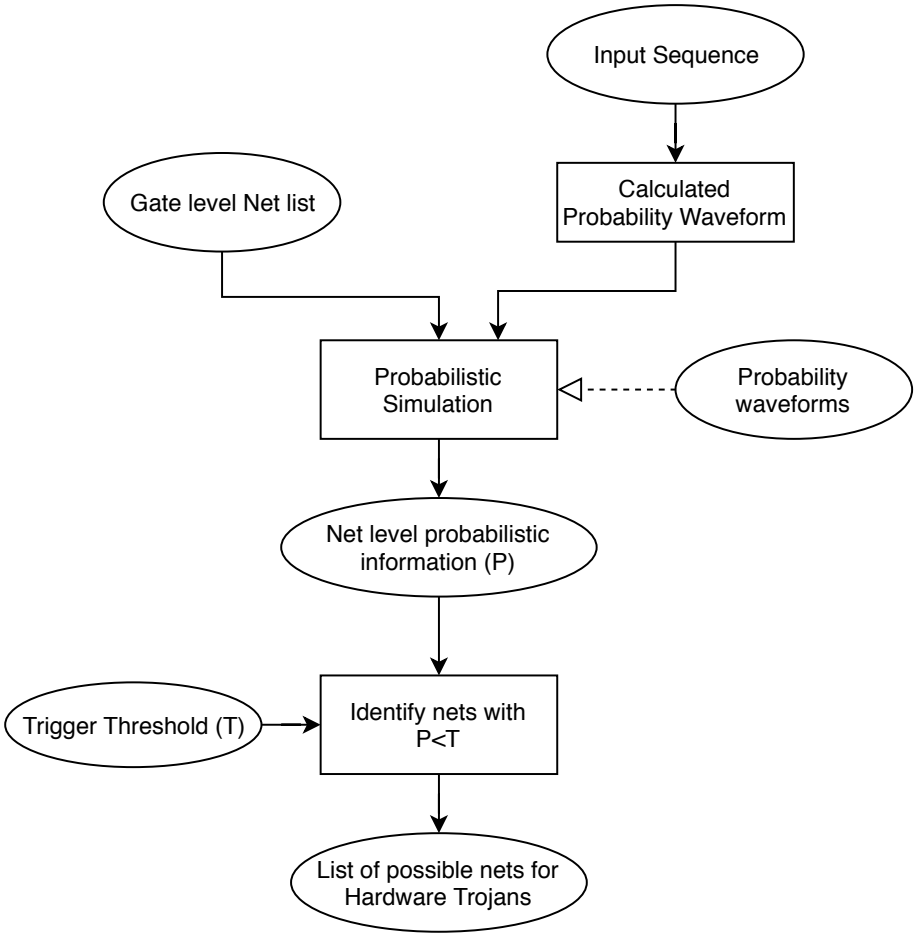

Figure 3.2: Proposed Hardware Trojan Susceptibility Analysis Tool

3.2 Proposed Framework

3.2.1 Probabilistic Simulation Based Hardware Trojan Susceptibility Tool

The actual procedure that we have followed in developing the proposed Hardware Trojan Susceptibility tool is shown in Figure 3.2. The procedure is explained in detail as follows:

1. The gate-level circuit in verilog format and sequence of inputs are given to the Hardware Trojan Susceptibility tool.

2. Then the input sequences are condensed into probabilistic waveforms. The waveform calculation varies depending on the type of gate. For example formulas for AND and OR as follows.

   AND gate:

   Signal probability:

$$P_{e_s}(t+) = P_{w1,1}(t+) + P_{w2,1}(t+) \tag{A1}$$

   Transition probability:

$$P_{e_t,01}(t) = P_{w1,01}(t)P_{w2,1}(t+) + P_{w2,01}(t)P_{w1,1}(t+) - P_{w1,01}(t)P_{w2,01}(t) \tag{A2}$$

   OR gate:

   Signal probability:

$$P_{e_t}(t+) = P_{w1,1}(t+) + P_{w2,1}(t+) - P_{w1,1}(t+)P_{w2,1}(t+) \tag{A1}$$

   Transition probability:

$$P_{e_s,01}(t) = P_{w1,1}(t-)P_{w2,01}(t) + P_{w2,1}(t-)P_{w1,01}(t) - P_{w1,01}(t) \tag{A2}$$

3. The probabilistic information (P) at circuit-level are obtained.

4. We set a threshold value (T). The Hardware Trojan in the netlist is triggered when its probabilistic information value 'P' is less than the trigger threshold value (T).

5. Then all the nets that satisfies Step 5 are notified by the tool.

The obtained nets from Step 6 has high possibility of Hardware Trojans. This helps in detecting the existence of Hardware trojans in the circuit and also diagnosis the exact location where HTs may occur. Therefore, obtaining all this information before designing the final product or prior to selling helps in preventing Hardware Trojans in the commercial products and improves the trustworthiness of Integrated circuits.

3.2.2 Illustrative Example

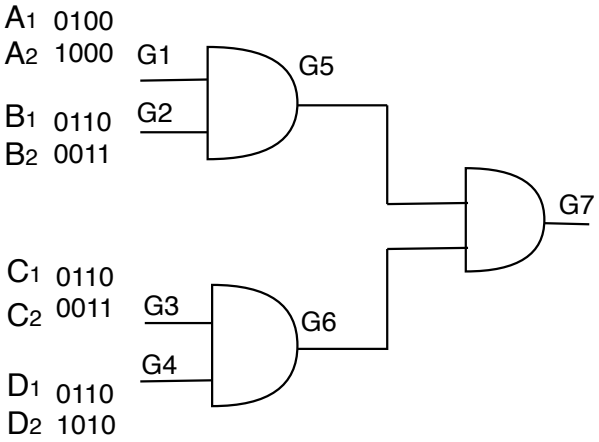

Figure 3.3: Two Level Circuit

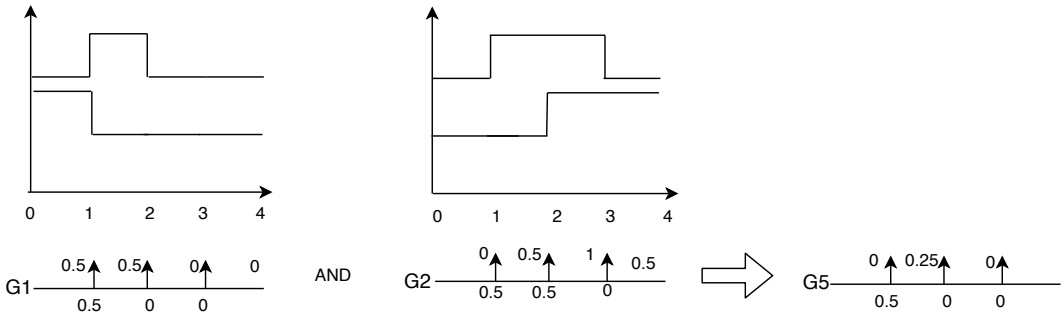

Figure 3.4: Probabilistic Waveforms on Primary Inputs G1 and G2 and Internal Line G5

When we pass input waveforms into each gate of a 2-level AND circuit as shown in Figure 3.1, the probabilistic waveforms can be computed by using formulas (A1) and (A2) shown above.

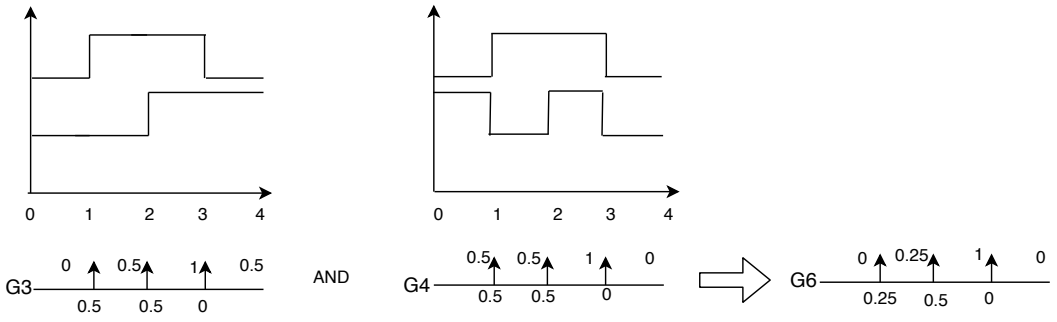

Figure 3.5: Probabilistic Waveforms on Primary Inputs G3 and G4 and Internal Line G6

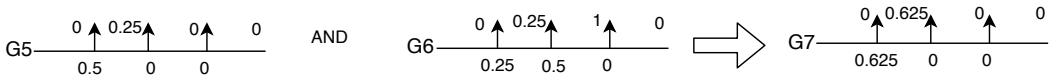

Figure 3.6: Probabilistic Waveforms on Internal Lines G5 and G6 and on Primary Output G7

In order to achieve this, we need to pass the circuit through the proposed synthesis tool as shown Figure 3.2. The outputs generated from the tool gives data related to probabilistic waveforms that helps in determining the occurrence possibility of Hardware Trojans in the circuit at each net. However we have compared the data generated from tool with that of actual values. The results are exactly the same. This test proves the accuracy and speedup of the proposed tool.

3.3 Chapter Summary

We have demonstrated proposed approach in detail. The primary steps involved in the process flow of the proposed susceptibility tool are analyzed and explained with an example. Most of the techniques we represented in the literature has the drawback of excessive simulation time. This proposed approach is very efficient and overcome this problem as the input sequences are condensed into probabilistic waveforms which are then propagated through the net-list to identify low activity nets.

CHAPTER 4: EXPERIMENTAL RESULTS

In this chapter, we demonstrate actual flow of our proposed approach for Hardware Trojan Detection and we report the experimental results performed on various benchmark circuits.

4.1 Experimental Flow

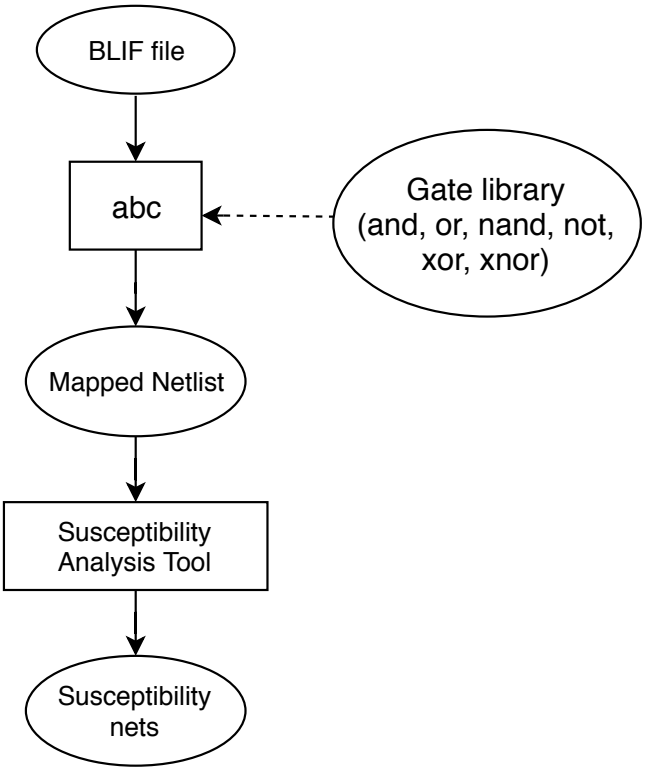

Figure 4.1: Experimental Flow

To perform probabilistic simulation, we need to follow the flow as shown in Figure 4.1. The first step is to generate the mapped Verilog net list from ABC synthesis tool by using corresponding

Table 4.1: MCNC Benchmark Circuits and the Netlist Data

Benchmark	No. of Primary Inputs	No. of Primary Outputs	Nets	Gates
c17	5	2	4	6
c432	36	7	157	164
c499	41	32	179	211
c880	60	26	260	286
c1355	41	32	179	211
c1908	33	25	240	265
c2670	233	140	513	653
c3540	50	22	937	959
c5315	178	123	1306	1429
c6288	32	32	1407	1439
c7552	207	108	1369	1477

BLIF file. In the next step, we pass the generated net lists through our proposed Susceptibility Analysis Tool in order to generate the susceptibility nets by producing information regarding the mapped netlist, which thereby helps in identifying Hardware Trojans in the circuits.

4.2 Probabilistic Simulation Execution Times

Table 4.1 shows eleven benchmark models that represents number of primary input nodes, number of primary output nodes, number of nets and number of actual gates present in the verilog netlist. Tables 4.2, 4.3, and 4.4 report the simulation times for eleven MCNC benchmarks for 10K, 50K and 100K input vector streams by comparing the results of proposed probabilistic simulations with that of conventional simulation.

4.3 Chapter Summary

In this Chapter, we reported results for eleven (11) MCNC benchmarks. Compared to conventional simulation, the probabilistic simulation approach can provide a speedup of up to 5000x.

Table 4.2: Comparison of Execution Times and Speedup - 10K Test Vector Sequence

Benchmark	Conventional Simulation (sec)				Probabilistic Simulation (sec)				Speedup T1/T2
	real	user	sys	user + sys T1	real	user	sys	user + sys T2	
c17	42.49	6.87	29.94	36.81	0.002	0.001	0.000	0.001	36811
c432	44.39	7.88	31.67	39.55	0.008	0.006	0.002	0.008	4943.75
c499	37.28	7.49	29.12	36.61	0.008	0.007	0.001	0.008	4576.25
c880	38.05	8.04	29.26	37.30	0.017	0.010	0.006	0.016	2331.25
c1355	39.19	7.70	30.88	38.58	0.007	0.006	0.002	0.008	4822.50
c1908	37.01	7.50	28.75	36.25	0.011	0.010	0.000	0.010	3625
c2670	40.05	9.65	29.48	39.13	0.033	0.029	0.004	0.033	1185.76
c3540	42.45	9.38	32.31	41.69	0.054	0.045	0.008	0.053	786.60
c5315	40.81	9.87	29.51	39.38	0.084	0.076	0.007	0.083	474.46
c6288	39.40	9.12	29.40	38.52	0.067	0.064	0.002	0.066	583.64
c7552	43.64	10.96	31.94	42.90	0.082	0.068	0.012	0.080	536.25
Average	40.43	8.59	30.21	38.79	0.033	0.029	0.004	0.033	5516.04

Table 4.3: Comparison of Execution Times and Speedup - 50K Test Vector Sequence

Benchmark	Conventional Simulation (sec)				Probabilistic Simulation (sec)				Speedup T1/T2
	real	user	sys	user + sys T1	real	user	sys	user + sys T2	
c17	209.42	6.94	29.94	36.87	0.002	0.000	0.001	0.001	36875
c432	209.71	7.83	29.15	36.98	0.031	0.027	0.003	0.003	1232.67
c499	210.00	7.93	29.18	37.11	0.026	0.024	0.001	0.025	1484.40
c880	214.36	14.69	30.13	44.82	0.061	0.052	0.008	0.060	747
c1355	203.61	8.09	29.46	37.55	0.025	0.024	0.001	0.025	1502
c1908	206.75	11.61	30.36	41.97	0.038	0.036	0.001	0.037	1134.32
c2670	217.93	13.58	30.76	44.34	0.112	0.095	0.016	0.111	399.47
c3540	228.36	24.64	30.36	55.00	0.173	0.148	0.024	0.172	319.77
c5315	216.64	17.99	30.14	48.13	0.256	0.221	0.034	0.255	188.75
c6288	275.14	70.03	31.15	101.18	0.204	0.196	0.009	0.205	493.56
c7552	215.71	12.61	29.73	42.34	0.237	0.205	0.032	0.237	178.65
Average	218.88	17.81	30.03	47.84	0.105	0.093	0.012	0.103	4050.51

Table 4.4: Comparison of Execution Times and Speedup - 100K Test Vector Sequence

Benchmark	Conventional Simulation (sec)				Probabilistic Simulation (sec)				Speedup T1/T2
	real	user	sys	user + sys T1	real	user	sys	user + sys T2	
c17	205.49	8.88	31.81	40.69	0.003	0.002	0.000	0.002	20345
c432	213.39	16.08	31.49	47.57	0.060	0.052	0.007	0.059	806.27
c499	217.32	17.30	31.23	48.53	0.050	0.049	0.000	0.049	990.41
c880	229.17	22.67	30.88	53.55	0.123	0.097	0.026	0.123	435.37
c1355	212.13	8.93	29.97	38.9	0.051	0.050	0.001	0.051	762.75
c1908	210.32	16.06	31.19	47.25	0.076	0.074	0.002	0.076	621.71
c2670	219.09	18.96	29.76	48.72	0.217	0.184	0.033	0.217	224.53
c3540	243.64	42.78	31.97	74.75	0.336	0.276	0.060	0.336	222.47
c5315	233.86	28.57	31.49	60.06	0.469	0.413	0.055	0.468	128.33
c6288	346.81	138.7	33.92	172.62	0.392	0.374	0.017	0.391	441.48
c7552	220.77	17.36	17.36	34.72	0.466	0.408	0.057	0.465	74.67
Average	231.99	30.57	30.09	60.66	0.204	0.180	0.023	0.203	2277.54

CHAPTER 5: CONCLUSION

We have proposed a tool to estimate the Hardware Trojan susceptibility at gate level in combinational circuits. Our proposed approach is more time efficient and can be used to apply rapidly for improving the security of Integrated circuits. This approach is a systematic way in which we can deal with complex circuits. We have verified proposed tool with eleven benchmark circuits.

REFERENCES

[1] Mohammad Tehranipoor and Farinaz Koushanfar. A survey of hardware trojan taxonomy and detection. *IEEE design & test of computers*, 27(1), 2010.

[2] Rajat Subhra Chakraborty, Francis Wolff, Somnath Paul, Christos Papachristou, and Swarup Bhunia. Mero: A statistical approach for hardware trojan detection. In *Cryptographic Hardware and Embedded Systems-CHES 2009*, pages 396–410. Springer, 2009.

[3] Kan Xiao, Domenic Forte, Yier Jin, Ramesh Karri, Swarup Bhunia, and M Tehranipoor. Hardware trojans: Lessons learned after one decade of research. *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, 22(1):6, 2016.

[4] Richard Burch, Farid N Najm, Ping Yang, and Timothy N Trick. A monte carlo approach for power estimation. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 1(1):63–71, 1993.

[5] F Alkabani Koushanfar. Designer's hardware trojan horse. In *Proc. of IEEE International Workshop on Hardware oriented Security and Trust. Anaheim, USA: IEEE Press*, 2008.

[6] Swarup Bhunia, Michael S Hsiao, Mainak Banga, and Seetharam Narasimhan. Hardware trojan attacks: threat analysis and countermeasures. *Proceedings of the IEEE*, 102(8):1229–1247,2014.

[7] Yuanwen Huang, 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, pages 130–141. ACM, 2016.*

[8] He Li, Qiang Liu, Jiliang Zhang, and Yongqiang Lyu. A survey of hardware trojan detection, diagnosis and prevention. In *Computer-Aided Design and Computer Graphics (CAD/Graphics), 2015 14th International Conference on*, pages 173–180. IEEE, 2015.

[9] Hassan Salmani, Mohammad Tehranipoor, and Jim Plusquellic. A novel technique for improving hardware trojan detection and reducing trojan activation time. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 20(1):112–125, 2012.

[10] Chih-Shun Ding, Chi-Ying Tsui, and Massoud Pedram. Gate-level power estimation using tagged probabilistic simulation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 17(11):1099–1107, 1998.

[11] S Ercolani, M Favalli, M Damiani, P Olivo, and B Ricco. Estimate of signal probability in combinational logic networks. In *European Test Conference, 1989., Proceedings of the 1st*, pages 132–138. IEEE, 1989.

[12] Farid N Najm. Transition density: A new measure of activity in digital circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 12(2):310–323, 1993.

[13] Farid N Najm, Richard Burch, Ping Yang, and Ibrahim N Hajj. Probabilistic simulation for reliability analysis of cmos vlsi circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 9(4):439–450, 1990.

[14] S Gayathri and TC Taranath. Rtl synthesis of case study using design compiler. In *Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT), 2017 International Conference on*, pages 1–7. IEEE, 2017.