A Region Based Approach for the Identification of Hardware Trojans

Mainak Banga and Michael S. Hsiao
Bradley Department of Electrical and Computer Engineering
Virginia Tech., Blacksburg, Virginia - 24061
Email: {banga, mhsiao}@vt.edu

***Abstract*—Outsourcing of SoC fabrication units has created the potential threat of design tampering using hardware Trojans. Methods based on side-channel analysis exist to differentiate such maligned ICs from the genuine ones but process variation in the foundries limit the effectiveness of such approaches. In this work, we propose a circuit partition based approach to detect and locate the embedded Trojan. Results show that our approach is effective in separating out candidate Trojans in the circuit. In addition, we provide a power profile based method for refining the candidate regions that may contain a Trojan. In many cases, such an isolation method leads to noticeable manifestation of the anomalous behavior of the circuit due to the presence of the Trojan thereby enhancing chances of their detection.**

I. Introduction

The competitive market advantages of providing quality semiconductor products at the lowest possible cost have compelled the design companies to outsource the manufacturing process. However, this poses a potential threat of tampering the original design by the third party manufacturer. While a defective part makes an application running on it to fail, intentional tampering can make the device act contrary to the expected where the results can be catastrophic. Therefore, testing the originality of the finished parts is of paramount importance, especially if the parts are intended to be used in mission-critical applications.

The parts in the embedded IC responsible for altering the normal behavior are commonly referred to as *Trojans*. Within the design, they behave like a hidden monitor waiting for certain special events to occur that triggers them to disrupt the normal functionality of the IC. Hardware Trojans usually comprise of a negligible fraction of the circuit area and hence do not affect physical dimensions of the chip. They are stealthy by nature as they remain dormant for most part of their life cycle making it is very difficult to assess their presence in a chip using conventional testing methods. Destructive testing of a portion of the finished product via the cutting open of the chip is not a feasible solution because it cannot assure the sanity of the rest of the chips not subjected to such testing. More so, not every chip in a lot may be tampered which means that destructive testing can only be beneficial after a preliminary filtering has been applied to the whole lot to ensure that the yield loss is minimized.

In software, Trojans (a subclass of viruses) have been prevalent and many software solutions (antivirus) exist for their detection. The distinction between Trojans and viruses is that the latter necessarily deteriorates the normal functionality of the host on which it attacks whereas Trojans remain passive for most part of the operation of the device and doesn't interfere with its normal functionality. In [11] 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.

In the hardware domain, cryptographic algorithms based on *public and private key* concept and specialized structures like Built-in-Self-Test (*BIST*) exist to ensure enhanced security of the manufactured device. Researchers have proposed approaches based on Linear Feedback Shift Register (*LFSR*) and Logic BIST [5], [6], [7] to monitor the proper operation of the internal hardware. However, one can build the Trojan intelligent enough to deter the advantages of such vigilant approaches. Recently, there have been attempts to devise ways to sieve out tampered chips by using side-channel based signal analysis. In [1], [2] the authors have used random sequence of test patterns to generate a noticeable difference between the power profile of the genuine IC and the Trojan counterpart but their effectiveness is limited in terms of the manufacturing processes, behavior and size of the Trojans. However, while their approach can qualitatively tell the presence of a Trojan, they cannot predict any spatial location for such structures in the chip.

In [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 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 needs not necessarily ensure the increased circuit activity in the region where the Trojan is present. Likewise, minimizing the Hamming distance also needs not necessarily facilitate reduced circuit activity in the parts that are not targeted. 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 judicious selection of the groups of flip-flops can help us excite the Trojan in a more effective way.

1-4244-2401-6/08/$20.00©2008 IEEE

In this work, 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. 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. Other choices exist that can be explored and a few of them are worth mentioning. The use of multiple voltage rails to keep the regions separate seems to be a good option but if the circuit is large enough (which in turn means that there would be large number of regions to start with), constructing so many voltage rails can be an additional cost or may not be feasible at all. Instead a mixed approach of activating the circuits using specific voltage rails followed by the region based approach might prove to be a better approach. More so our method is aimed to work for any general IC. There is a pre-silicon requirement if we want to fabricate additional voltage rails to assist us in post-silicon diagnosis. Statistical techniques might be useful in averaging out the variations thereby giving a better chance for isolation of the parts with behavioral difference but that requires access to actual product lot and test equipments. Leakage current is also a parameter that can be considered. But since the Trojan size is really small as compared with the original circuitry, it is unlikely that the Trojan will consume a noticeable amount of leakage power. With the process technology decreasing down the nanometer scale, the overall leakage current is increasing thereby reducing the chances that a small leakage current variation from the Trojan will be observable.

II. Preliminaries

A. 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 such internal information of a device is known as the side channel analysis, and side channel signals have been effectively used to detect the anomalies in the behavior of a circuit [8], [9].

For our approach, we compute the power profile of the genuine circuit. The total power for an IC is proportional to the operating frequency f, switching capacitance C, and supply voltage V, shown in the following expression [4]:

$$\mathbf{P = CV^2f} \tag{1}$$

As the overall power consumption will reduce if the circuit is operated at a low frequency, it was shown in [2] that simple Trojans could be more easily detected when operating at a lower frequency, since the power consumed by Trojan will make up a greater portion in the total consumed power. This was illustrated in [2] by an experiment in which a simple Trojan could not be detected when the circuit was operated at 100 MHz, whereas it was detected at 500 KHz.

B. Trojan Types

We use the following terms for our subsequent discussion:

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: A finite state machine (FSM) that monitors a portion of the internal circuit signals and triggers the output upon the appearance of a specific occurring sequence(s).

Generally, Trojans are sequential sub-circuits(which is the case in our experimentation as well). But combinational Trojans can be used if a hard property for the system is targeted.

C. Power Profile

The total power consumed in the circuit over a set of vectors constitute the power profile of the circuit for that vector set and the individual power value for any particular vector-pair is called the power number for that vector-pair. Frequently, we estimate the power numbers by parameters such as the switching activity of the circuit. In course of our discussion we shall use the terms *power profile* and *circuit activity* interchangeably because circuit activity has a linear relationship with the power numbers.

III. Our Approach

Our approach consists of two major 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 each of the selected regions. We name these steps as - *Region Based Partition* and *Relative Toggle Count Magnification* respectively.

A. Stage 1: Region Based Partition

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 1(a). Region based partitioning has been used earlier in error diagnosis and detection [10]. 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 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 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 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 need to define a suitable selection criterion that allows us to select a subset of the regions

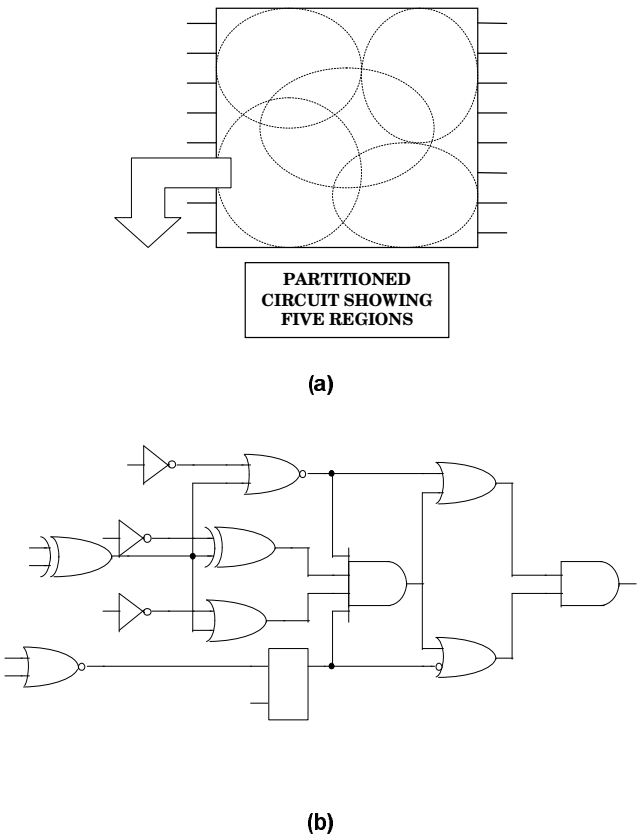


Fig. 1. Illustration of the concept of *Region* and *Radius* in a circuit

intelligently 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 those 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 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.

B. Stage 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 minimizing 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:

$$\mathbf{F} = \mathbf{max}(\textit{in-region activity} - \textit{out-region activity}) \qquad (2)$$

The behavior of a Trojan is perceivable only if the difference in the activity of the Trojan-infected chip and the genuine chip (without Trojan) is above the process variation. This means that the Trojan is most detectable when the power consumed in the genuine circuit is kept low, but non-zero. Thus, we aim to stimulate a small region while keeping the rest at low or zero activity. 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 2 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 region(s). This idea of maximizing the difference in the toggle count is better than the previous approach of maximizing the Hamming distance because maximizing Hamming distance need not necessarily increase the power consumed and vice versa. To illustrate this, consider 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.

C. 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 a *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 the single vector according to the function defined by Equation 2. 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 plots, 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 ***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:

$$\mathbf{Trojan} = \mathbf{\Pi_0^i}(G : G\epsilon \textit{Gate in a selected region} \wedge \textit{freq (G)} > \textit{Frequency Threshold}) \quad (3)$$

The method can be applied to the large circuits in the same way. As stated earlier, we shall have more regions because of the increased circuit size which will result in a bigger initial set of vectors, but once we start analyzing the regions with enhanced activity difference between the golden and the Trojan affected circuits, we shall narrow down to a small number of regions. We can formulate the entire procedure in the form of Algorithm 1. The functions used in the algorithm has been explained in Table 1.

IV. Trojan Description

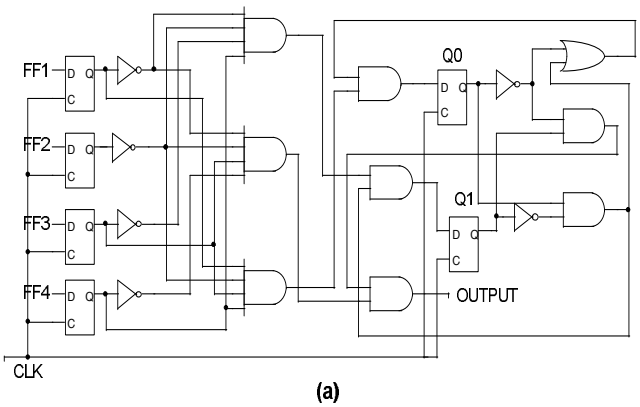
Now we shall illustrate the construction of a typical Trojan used in our experimentation. The Trojan consists of the sequential circuit as shown in Figure 2(a). The four inputs to the Trojan are the state bits of the flip-fops in the original circuit. The finite state machine (FSM) for the Trojan circuit is shown in Figure 2(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(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

Algorithm 1 Generate Vector Set to Maximize Toggle Count Difference for Different Groups

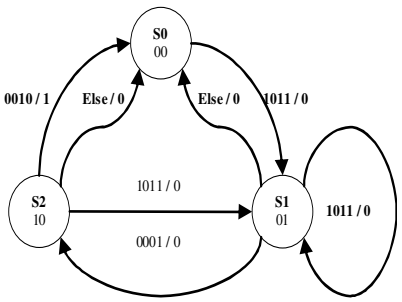
Require: *FFThreshold*, *RadiusThreshold*, *GenuineCkt*, *TrojanCkt*, *InRegionFFCount*

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

```
Radius ⇐ 2
FFCount ⇐ 2
VectorSet ⇐ ∅
Regions ⇐ ∅
while Radius < RadiusThreshold do
    Regions ⇐ ComputeRegions()
    while FFCount < FFThreshold do
        for all Regions do
            if InRegionFFCount > FFCount then
                VectorSet ⇐ GenerateVectors(Region)
            end if
        end for
        SimulateVectors(GenuineCkt)
        SimulateVectors(TrojanCkt)
        ComputeActivityDifference()
        IncrementFFCount()
    end while
    IncrementRadius()
    Reset(VectorSet, Regions, FFCount)
end while
```



(a)



(b)

Fig. 2. Example of a Trojan circuit and its associated FSM

TABLE I
FUNCTIONS OF ALGORITHM 1

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

00, **01** and **10** respectively as shown in Figure 2(b). The two bits represent $\mathbf{Q_0Q_1}$ 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.

$$\mathbf{Q_0^+ = \bar{Q_0}Q_1 \wedge \bar{a_1}\bar{a_2}\bar{a_3}a_4} \quad (4)$$

$$\mathbf{Q_1^+ = (\bar{Q_1} + \bar{Q_0}Q_1) \wedge a_1\bar{a_2}a_3a_4} \quad (5)$$

$$\mathbf{OUTPUT = Q_0\bar{Q_1} \wedge \bar{a_1}\bar{a_2}a_3\bar{a_4}} \quad (6)$$

Note that the Trojan is not a random circuit, but that we need to select the sequence carefully that activates the Trojan. The foremost condition is that such sequence should be rarely occurring for the set of circuit flip-flops under consideration. This stems from the fact that Trojans probe for special conditions in the circuit that are rarely occurring. In our experiments, we have simulated a random set of 1000 vectors on the circuit and have selected a particular sequence that occurs only once within the simulation results. Furthermore, we need to ensure that the behavior of the Trojan is not exposed at the circuit outputs because in that case the Trojan circuit will no longer be stealthy! To ensure this we feed the output of the Trojan to a gate for which its normal value is non-controlling for the gate. In our case, the Trojan remains zero for most part of the simulation and so we feed the Trojan output to either an OR gate or a NOR gate. To ensure that the Trojan is really hidden, we simulate the random set of 1000 vector on both the actual and the Trojan circuit and ensure that the outputs do not differ.

V. 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 corresponding flip-flop group. The ordinate represents the relative percentage activity difference. To compute this we calculate the switching activity in the actual circuit (golden circuit) and the Trojan affected circuit using the random vectors. Then we compute the absolute difference of these switching activity values and normalize them with the activity in the original circuit. We represent the normalized value in percentage. We repeat the same procedure for the vector set generated by our proposed approach. We plot both the the curves on the same graph to figure out the regions that shows marked difference in the activity profile. The curve corresponding to the random vector set is shown by the blue curve and a square legend and the curve corresponding to our approach is shown by the brown curve and a diamond legend. The experimental circuits are from a subset of ISCAS'89 sequential benchmark circuits. There are three process variation magnitudes that has been considered in [1]. We have also assumed the same values for the process variation, viz. 2.5%, 5.0% and 7.5%. In order to make sure that the discrepancies are observable in the face of manufacturing variations, we consider those regions in which the *Relative Toggle Count Magnification* is greater than 5%. The abbreviation *TCM* in the Figures stand for *Toggle Count Magnification*.

A. 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 Figures 3, 4 and 5. The Trojan size in this circuit is about 6% of the total gate count in this small circuit. 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 3, 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, the peaks of the regions defined by vector sets50-60 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 5 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.

B. s1196

Results for Toggle Count Magnification for s1196 are plotted in Figure 6 and Figure 7. The Trojan size in this case is about 2% of the total gate count in the circuit. Figure 6 is for a partition radius of 3 while Figure 7 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 6. These refer to the regions containing the flip-flops (16 and 17), (16 and 17), (20 and 28) and (23 and 24). In Figure 7, 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 truly enough our Trojan circuit derives its input from the set (16, 17, 23 and 24).

C. s1423

For circuit s1423, the target vector sets selected for observation pertain to regions 10, 15 and 20 for the peaks in the Figure 8. The Trojan size in this circuit is about 1% to 1.5% of the total gate count. 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.

D. s3271

Plots for s3271 shows a marked difference in the Toggle Count Difference Profile for the two (actual and Trojan infected) circuits under consideration. The Trojan size used in this circuit is less than 1% of the total gate count. 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 9) and after vector 230 (in Figure 10) 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 9 after vector 320 that is a little better than the random one. For Figure 10, 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.

VI. Conclusion

In this work, we have presented a simple yet effective approach for isolating and distinguishing circuit portions accountable for embedded Trojans. We have discussed the possible tracks of the solution approach with an indication of their potential tradeoffs. In the process, we have devised an algorithm for non-destructive testing of ICs that may be tampered by a 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 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. 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. Considering the high variation in the leakage current in the submerging process geometries is also a challenge for future research.

References

[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 IS Sensors for Enhanced IC Trust*,2006.

[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.

[4] S. Pilli and S. Sapatnekar, *Power estimation considering statistical IC parametric variations*, ISCAS 1997, pp. 1524 - 1527, vol.3.

[5] 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.

[6] 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.

[7] W. T. Cheng, M. Sharma, T. Rinderknecht and C. Hill; *Signature Based Diagnosis for Logic BIST*, ITC 2006, Oct. 2006, pp. 1 - 9.

[8] L. J. Kohout, A. Yasinsac and E. McDuffie; *Activity profiles for intrusion detection*, Fuzzy Information Processing Society, 2002. pp. 463 - 468.

[9] D. Agarwal. et al, *The EM side-channel(s)*, CHES 2002, Lecture Notes on Computer Science, Springer-Verlag, pp. 29-45, 2002.

[10] 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.

[11] M. A. Williams, *Anti-Trojan and Trojan Detection with In-Kernel Digital Signature testing of executables*, 2002.

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] D. P. Vallett, *An overview of CMOS VLSI failure analysis and the importance of test and diagnostics*, Proc. International Test Conference, 1996, pp. 930.

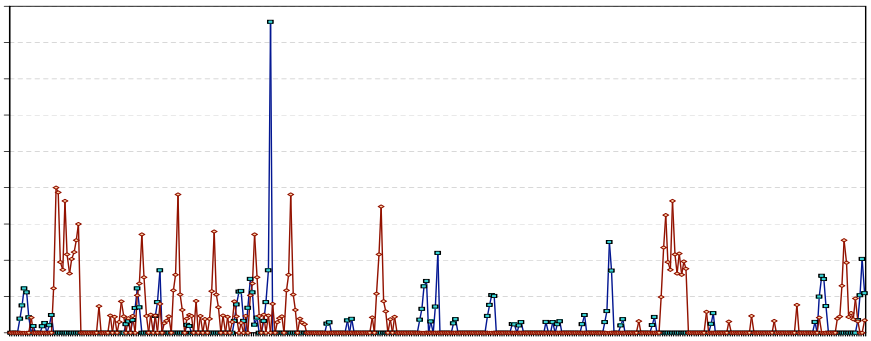


Fig. 3. TCM(Radius 2, flip-flop Count 3) for s444

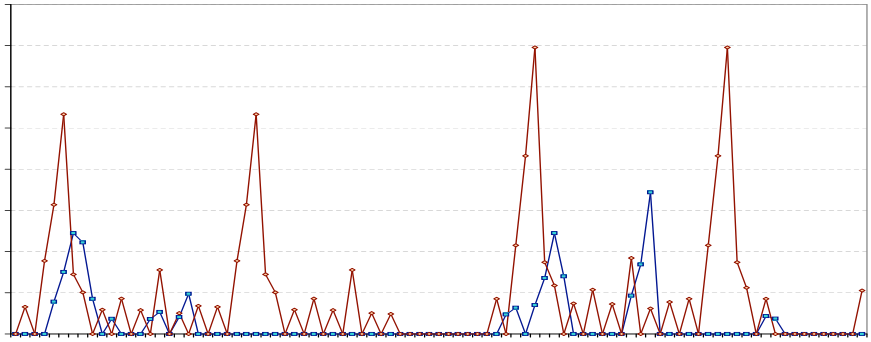


Fig. 4. TCM(Radius 2, flip-flop Count 4) for s444

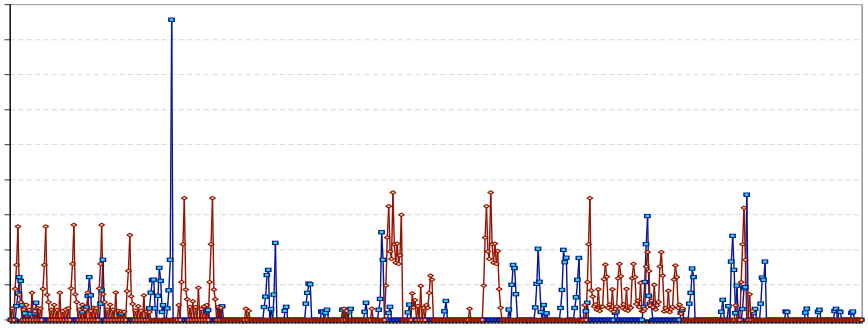


Fig. 5. TCM(Radius 3, flip-flop Count 3) for s444

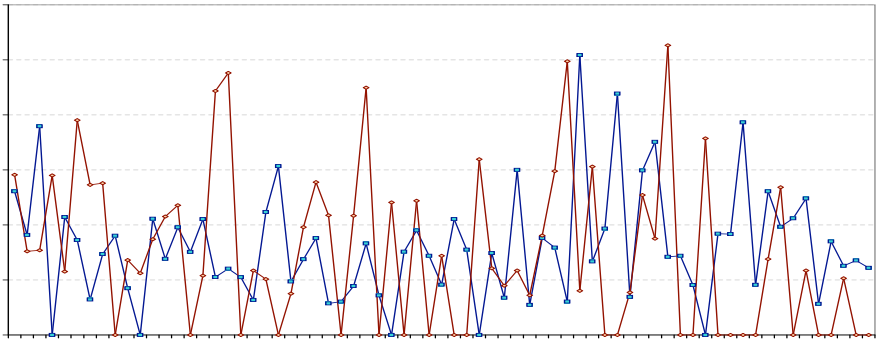


Fig. 6. TCM(Radius 3, flip-flop Count 2) for s1196

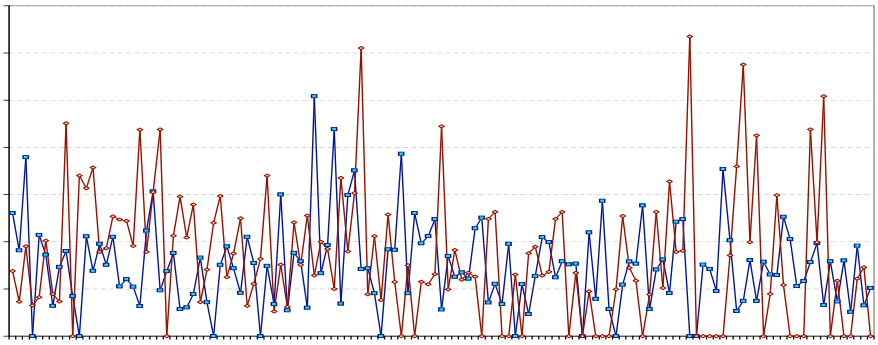


Fig. 7. TCM(Radius 4, flip-flop Count 2) for s1196

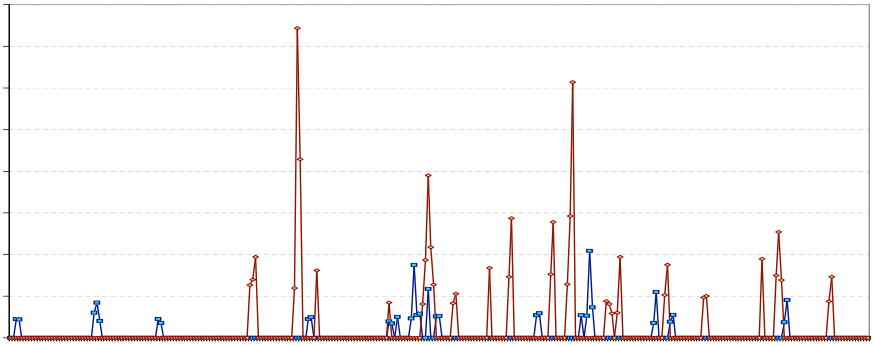


Fig. 8. TCM(Radius 4, flip-flop Count 5) for s1423

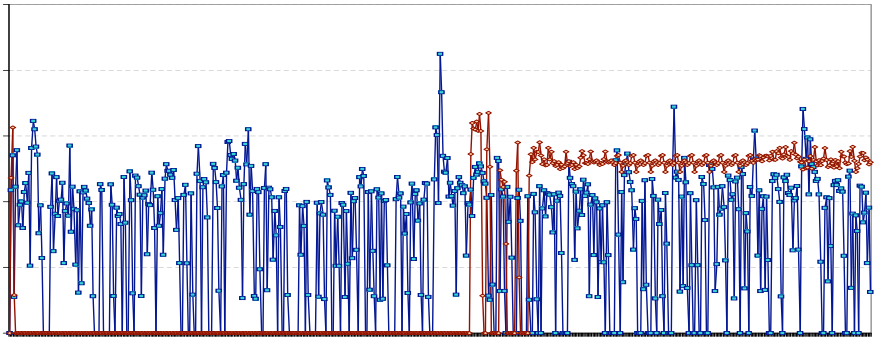


Fig. 9. TCM(Radius 3, flip-flop Count 3) for s3271

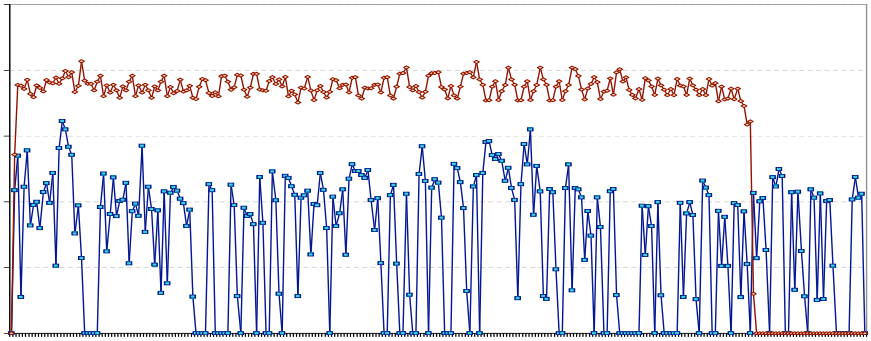


Fig. 10. TCM(Radius 4, flip-flop Count 5) for s3271