Malicious Circuitry Detection Using Fast Timing Characterization via Test Points

Sheng Wei
Computer Science Department
University of California, Los Angeles
Los Angeles, California 90095

Miodrag Potkonjak
Computer Science Department
University of California, Los Angeles
Los Angeles, California 90095

***Abstract*—We develop a region-based timing characterization approach to detect hardware Trojans (HTs) on integrated circuits (ICs). In order to ensure the scalability of the approach, we partition the target IC into well-formed and non-overlapping regions and detect hardware Trojans on all circuit locations by examining the timing properties of the transistor paths. Based on the circuit partition, we insert a minimal number of test points that provide additional observation interfaces for the delay measurements of all circuit locations. Our evaluations on ISCAS and ITC benchmarks show that the region-based Trojan detection via test points can detect hardware Trojans accurately with well controlled area overhead and test time.**

I. Introduction

Integrated circuit (IC) outsourcing has become a trend in the IC industry in order for the IC companies to increase their revenue. In the current IC design and manufacture model, the IC is exposed to a variety of threats such as hardware Trojans (HTs) [19][9] and unauthorized intellectual property (IP) usage [16], due to the fact that the IC designers and the manufacturing foundries are completely separate and different entities. Since ICs are the fundamental building blocks of all digital computing and communications platforms for the modern business, military, and government affairs, the potential vulnerabilities against security attacks have become a major concern in the IC industry.

A malicious circuitry, or hardware Trojan (HT) [19][9], is a malicious modification on an IC, which is intentionally conducted by untrusted designers or foundries in order to attack the target hardware. There is a wide consensus that HTs have huge impact on the security and integrity of the IC systems. For example, HTs may alter the functionality of the IC, change the original characteristics (propagation delay or leakage power), or even leak confidential information. Similar to the trend of software Trojans in the early days, the detection of HTs has triggered a great deal of attention in the IC community, due to the fact that the consequences of HTs can be extremely severe for security-sensitive systems.

As the potential risk arising from HTs increases, many research efforts have been put on how to detect and prevent the embedded HTs effectively. Among them, side channel-based monitoring schemes [8][2][11] have been considered as effective and economic compared to the otherwise expensive physical inspections. In the existing approaches, leakage power is the most often monitored property for the checking against HTs [15][23], because any unexpected malicious components on the circuit would result in a systematic bias in the total leakage consumption. Therefore, the leakage power-based approach can provide us with a full coverage of the circuit in terms of identifying HTs. However, as a side effect caused by the full coverage detection, most of the existing power tracing-based approaches require creating and solving huge sizes of power equations that cover essentially all gates on the target circuit, making it difficult to scale to modern IC designs with millions of gates on each chip. Furthermore, the existence of process variation (PV) during the IC manufacturing process may compromise the detection scheme, since it is difficult to distinguish the leakage power variation caused by PV from that caused by HTs.

Besides leakage power, circuit frequency (or delay) has also been considered as a side channel for HT detection [8][11]. Delay-based detection methods are in general scalable, since only a small subset of gates that are on the same path are under consideration. On the other hand, it is challenging for the delay-based methods to cover all the locations on the circuit for HT detection, due to the fact that there exist paths that are in parallel and reconverge to a single point, and it is difficult to map the measured path delay to a specific path for the consideration of HTs. As shown in Figure 1, there is a HT gate H embedded on the path A. Even though we can measure the delay between inputs $I1/I2$ and output O, we are unable to determine whether the measured delay is for path A or path B. In fact, since path B (gates 1-3-4-5-6-7) is much longer than path A (gates 1-2-H-7), it is difficult to detect the addition of HT gate H using delay measurements from inputs $I1/I2$ to output O.

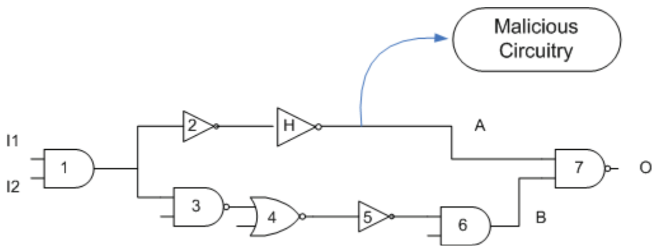

Fig. 1. Example of HT attack with no delay impact.

We develop a gate-level delay characterization approach that covers all locations on the circuit for HT detection. Although the lack of observability in delay caused by reconvergences, as shown in Figure 1, can be addressed by inserting additional test points (i.e., flip-flops) [22], we argue that the problem of identifying all the reconvergent points is

NP-complete, which cannot scale to large designs. In order to solve this issue, we employ a circuit partition method to scale down the problem space to a limited number of non-overlapping regions. In this way, we ensure that our timing characterization at all circuit locations is scalable to large IC designs. Also, based on the circuit partition, we introduce a test point insertion scheme that separates the parallel paths and enables their observability via delay measurements. To summarize, our innovations and contributions include the following:

- a region-based circuit partition approach to reduce the overhead and ensure the scalability of the timing characterization;
- a malicious circuitry detection approach using delay characterization that covers all circuit locations; and
- a test points insertion scheme to break the reconvergences and measure the required paths for their timing properties.

II. Related Work

Tehranipoor et al. [19][20] and Karri et al. [9] presented comprehensive surveys of hardware Trojan attacks and detection techniques. Agrawal et al. [1] proposed one of the first HT detection techniques, which employs side channels (e.g., power and temperature) as fingerprints to authenticate the ICs. Several other early HT detection approaches focused on functional test techniques, which monitor and check the outputs of the target ICs under certain input patterns via simulations [29][4]. More recently, the side channel-based approaches have become a trend in HT detection. For example, there are many HT detection approaches that have been proposed using leakage power [23][26], switching power [5][30], delay [8][11][18], or combinations of the above to detect or diagnose hardware Trojans on the target ICs [10][15].

III. Preliminaries

A. Hardware Trojan Threat Model

HT attack is a common IC threat that has drawn a great deal of attention in the hardware security community [21][28][18]. There are typically two types of HT attacks: (1) adding malicious components into the IC, and (2) altering the properties of the existing IC components. In this paper, we focus on the first category, since it is more commonly used by an attacker during the IC design and manufacture process and would induce more severe security attacks. In particular, we assume that attackers tend to hide and prevent the HT from exposing to delay measurements. For example, they may size the HT ultra small to minimize its observable delay. Furthermore, they can hide the HT under one of the parallel paths to bypass direct delay measurements.

B. Delay Model and Measurements

The delay of a single logic gate can be expressed as $d = gh + p$, where g and h are logical effort and electrical effort, respectively; and p is parasitic delay. In particular, We use the delay model in [14] that connects the gate delay to its sizing and operating voltages:

$$\begin{aligned} Delay = \frac{k_{tp} \cdot k_{fit} \cdot L^2}{2 \cdot n \cdot \mu \cdot \phi_t^2} \cdot \frac{V_{dd}}{(ln(e^{\frac{(1+\sigma)V_{dd}-V_{th}}{2 \cdot n \cdot \phi_t}} + 1))^2} \\ \cdot \frac{\gamma_i \cdot W_i + W_{i+1}}{W_i} \end{aligned} \quad (1)$$

where L is effective channel length, V_{th} is threshold voltage, W is gate width, V_{dd} is supply voltage, n is substreshold slope, μ is mobility, C_{ox} is oxide capacitance, D is clock period, ϕ_t is thermal voltage $\phi_t = kT/q$, σ is drain induced barrier lowering (DIBL) factor, subscripts i and $i+1$ represent the driver and load gates, respectively, γ is the ratio of gate parasitic to input capacitance, and k_{tp} and k_{fit} are fitting parameters.

In order to measure the delay of a transistor path, we leverage the delay fault testing technique introduced in [13]. In particular, we insert flip-flops into the target circuit that serve as additional observation and control interfaces to facilitate the delay measurements based on delay faults. In order to reduce the cost of the embedded test points, we ensure that the flip-flops are only activated during the test mode and deactivated during normal IC operations.

C. Gate-Level Characterization (GLC)

Gate-level Characterization (GLC) [24][25][27][22] is the process of identifying the process variation in the manufactured IC. The effect of process variation can be represented as a scaling factor towards the gate-level manifestational properties, such as delay. Then, a system of linear equations can be obtained by summing up the gate-level properties and measuring the total delay:

$$\tilde{d_j} = e_{sj} + e_{rj} + \sum_{\forall gate\ i=1,\ldots,n} K_{ij}\ s_i + H \quad (2)$$

where $\tilde{d_j}$ is the delay of the path when input vector j is applied; s_i is the process variation scaling factor of gate i, which represents the deviation of the delay from its nominal design values due to process variation; K_{ij} is the nominal delay for the gate at input state j; and e_{sj} and e_{rj} are systematic and random measurement errors, respectively. Furthermore, we add one additional variable H in each of the equations that represents the additional delay caused by malicious circuitries. Although it is unknown whether any HTs exist or not, we assume there is a HT on each delay path while formulating the equations. Our idea is that the H would capture the systematic bias caused by HTs, if there are any, or becomes an ultra-small value if the circuit is HT-free. Therefore, by solving the system of linear equations and evaluate the value of H, we are able to detect the existence of HTs.

IV. Overall Flow

Figure 2 shows the overall flow of our HT detection approach. We employ three systematic steps in order to determine whether any hardware Trojans have been embedded in the target circuit. First, we partition the circuit into small regions, which consist of limited number of gates under one or multiple delay paths, to ensure scalability of our approach toward large designs. We employ a maximum fanout free cone (MFFC)-based method [7] for the original circuit partition. Then, we merge the MFFCs to form the regions that are non-overlapping and that can cover all circuit locations. After obtaining the well-formed regions for delay measurements, we insert test points, i.e., scan chain flip-flops, to the source or destination gates of the paths that are not directly observable through inputs, outputs, or existing flip-flops. Secondly, we manipulate the input vector of the circuit and generate delay variations for each path. With the delay measurements using the test points, we are able to formulate a linear program for each path concerning the path delay and the delays of individual gates. Finally, for the detection of hardware Trojans, we insert an additional HT variable in the linear program in order to capture the systematic bias caused by the HT. By evaluating the solved HT variable values from the linear program, we determine whether any malicious circuitry exists in the target circuit.

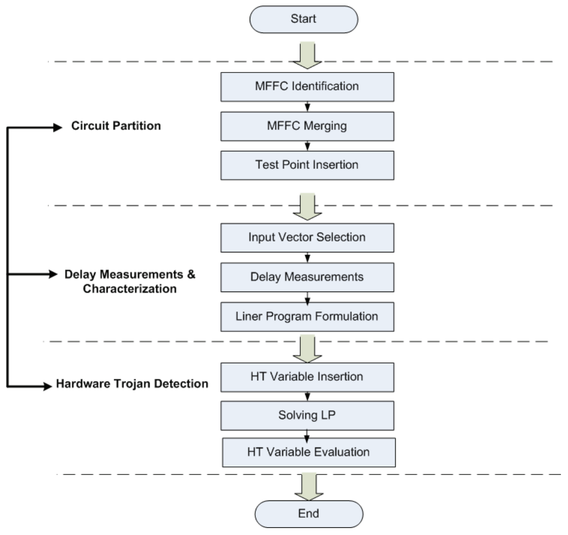

Fig. 2. Overall flow of the region-based Trojan detection using delay characterization.

V. Region-based Delay Characterization

A. Motivation

We note that we must address the following three issues regarding the overhead and scalability of our approach, in order to provide a reliable and scalable HT detection solution.

- *Area overhead*. The area overhead caused by the inserted test points (i.e., flip-flops) cannot be ignored, as the area of a flip-flop is often 4 to 6 times larger than a regular gate. It is essential to take into account of this critical overhead toward the design and manufacture of the ICs.
- *Test Time*. The test time required by the delay measurements is an important metric for the cost of the proposed approach. Since it requires input vector control to conduct gate-level timing characterization, the GLC of multiple paths cannot be fully parallelized. Therefore, we must minimize the total number of measured paths to reduce the test time while still covering all circuit locations.
- *Scalability*. the search of reconvergence points in all circuit locations is a NP-complete problem that raises scalability concerns. The complexity grows exponentially with the increase of the number of inputs, outputs, or gates. The efficiency and scalability of the identification procedure must be improved for the consideration of large designs, e.g., with millions of transistors, in the modern IC industry.

To address these issues, we further develop a circuit partition scheme that partitions the large IC into smaller regions, so that the scope of the timing characterization is reduced to the partitioned regions to address the scalability issue. Also, we aim to minimize the number of test points that we must add during the course of circuit partition, while the goal is still to cover all the circuit locations in terms of gate-level timing characterization.

B. Problem Formulation

We define the circuit partition problem as a specialized graph (netlist) partition problem with the goal of minimizing the required test points while controlling the number of gates in each region:

Circuit partition problem for timing characterization. Given a graph $G = (V, E, PI, PO)$ that represents the netlist of an IC, where V is the set of vertices (gates), E is the set of edges (wires), PI is the set of primary inputs, and PO is the set of primary outputs, find a graph partition that consists of k subgraphs (regions) so that (1) each region consists of $P_i(i = 1...k)$ paths, which ensures that the total number of source and destination nodes of the paths that do not belong to $PI \cup PO$ is minimized; and (2) the number of gates in each component $N_i(i = 1...k) < Th$, where Th is a configurable threshold determined by the gate-level timing characterization approach.

After partitioning the circuit, we can conduct the reconvergence identification and delay characterization in the scope of each single region. Since the size of the problem domain is reduced by factor of k, we argue that the complexity of the identification and characterization processes is reduced exponentially.

C. MFFC-based circuit partition

Our intuitions in addressing the circuit partition problem are three-fold: (1) We should find regions that contain paths traversing from primary outputs toward the direction of primary inputs. In other words, the addition of nodes in the region during the search process should go vertically (i.e., depth first). This is based on the consideration of leveraging as many primary inputs/outputs as possible, in order to reduce the additional test points. (2) The number of gates in each region should be maximized as long as it can be handled by the characterization approach (i.e., below threshold Th), which would reduce the number of regions and thus the number of delay measurements. (3) It is beneficial that there is no or little overlap between different regions, in order to reduce unnecessary measurements and characterizations.

Based on the above intuitions and thoughts, we develop a circuit partition method using maximum fanout free cones (MFFCs) [7]. a MFFC for a node g_i is a sub-netlist where each node g_j is a transitive fan-in of g_i, and all the transitive fan-in's of g_j is included in the MFFC. In other words, a MFFC is a self-contained region that grows maximally from PO toward PI of the circuit G, which satisfies our intuitions (1) and (2). Furthermore, the MFFCs have an important property that two different MFFCs are either non-overlapping or one contains the other, which satisfies our intuition (3) as long as we remove the smaller MFFC from consideration in terms of circuit partition. Our circuit partition algorithm using MFFCs is presented in Algorithm 1. We first find the MFFC for each gate in the circuit using known algorithms [7]. Then, we sort the found MFFC set in the ascending order in terms of the size (i.e., number of gates) in the MFFC. Finally, we remove the MFFCs that have been fully covered by at least one other MFFC to eliminate the redundancy.

Algorithm 1 MFFC-based circuit partition.

```
Input: Circuit Netlist ($Net$), Gate Set ($G$);
Output: A circuit partition with $k$ regions;
1:  for each gate $g_i$ in $G$ do
2:      Find the MFFC $C_i$ for $g_i$ [1]
3:  end for
4:  Sort the MFFC set $C = \{C_i | i = 1...N\}$ for each node in
    ascending order in terms of the number of gates involved.
5:  for each $C_i \in C$ do
6:      if $\exists C_j$ where $C_j \supset C_i$ then
7:          Remove $C_i$ from $C$
8:      end if
9:  end for
10: Output $C$ as the circuit partition.
```

Figure 3 shows a small example of the proposed circuit partition method using MFFC. We partition the 6-gate circuit into 3 regions. Each region is a maximum fanout free cone from the bottom node. With this partition, the delay of each path is measurable, and it only requires one test point at location 16, since all other observation points are primary inputs or outputs of the target circuit.

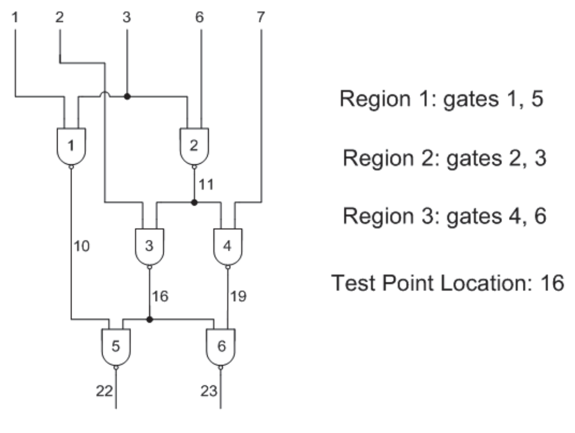

Fig. 3. Example of MFFC-based circuit partition for timing characterization.

D. Test Point Insertion

After partitioning the circuit, we insert test points in the target circuit to facilitate the delay measurements of all circuit locations. There are two types of locations where we must insert test points: (1) where there are no input, output, or flip-flops in the original design that can be used to measure the delay, such as the location 16 in Figure 3; and (2) where the corresponding path is not measurable due to reconvergence, such as the two paths in Figure 1.

The Type (1) test points are highly dependent on the partition of the circuit. We must make sure that the source and destination gates of all the paths in the selected regions are covered by either inserted test points or observation points (i.e., primary input/output or existing flip-flops) in the original design. For the Type (2) test points, we employ a depth-first search algorithm to identify all the reconvergences within the selected regions. Since the size of a region is significantly smaller than that of the entire circuit, the complexity of the reconvergence identification is greatly reduced even though it is NP-complete.

VI. Evaluation Results

A. System Implementation and Metrics

We evaluate our delay characterization-based Trojan detection method on a set of ISCAS'85, ISCAS'89, and ITC'99 benchmarks. We simulate the 45nm technology and generate the variation of threshold voltage following the Gaussian process variation model [3]. Also, we employ the spatial correlation of effective channel length following the quad-tree model [6]. For the prototype implementation, we use Synopsys Design Compiler Version E-2010.12-SP5 to synthesize. The target library is the FreePDK 45nm Standard cell library [12]. Finally, we use Cadence Soc Encounter Version 9.1 to place and route our design.

We consider the following two metrics for the evaluation of our approach: (1) the accuracy of HT detection, which is represented by the false positives and false negatives in the detection results; and (2) the overheads caused by the inserted test points, which include the area overhead (i.e., the additional area caused by the inserted flip-flops) and the cost of test (i.e., the test time required by measuring the delay of the selected paths).

B. Accuracy of Hardware Trojan Detection

We evaluated the accuracy of our HT detection approach on a set of ISCAS'85 benchmarks. For each benchmark, we repeat the detection process 50 times and solve the value of the HT variable H using a LP solver. Table I shows the statistics of the HT variable (H) in each tested benchmark. For the ease of comparison, we evaluated two cases where there is a HT gate inserted at a random location on the circuit, and where the circuit is HT-free. We observe from the results that there are large gaps and little overlaps in terms of the HT variables between the HT-free case and the HT-present case. This enables us to draw a decision line between the H in the two cases and use it to determine whether HTs exist or not.

TABLE I. THE DISTRIBUTION OF HARDWARE TROJAN VARIABLE (H) IN THE HT-PRESENT AND HT-FREE CASES.

Benchmark	HT Status	Max	75th Percentile(Q3)	Median (Q2)	25th Percentile (Q1)	Min
C432	no HT	0.006	0.001	0	0	0
	with HT	1.017	0.788	0.388	0.198	0.055
C499	no HT	0.052	0.011	0	0	0
	with HT	1.065	0.787	0.402	0.192	0.056
C880	no HT	0.004	0.002	0	0	0
	with HT	1.012	0.791	0.387	0.197	0.054
C1355	no HT	0.006	0.002	0	0	0
	with HT	1.015	0.802	0.387	0.197	0.055
C1908	no HT	0.004	0.001	0	0	0
	with HT	1.008	0.796	0.386	0.198	0.053
C2670	no HT	0.010	0.001	0	0	0
	with HT	1.014	0.790	0.386	0.198	0.056
C3540	no HT	0.002	0	0	0	0
	with HT	1.046	0.683	0.458	0.263	0.050
C5315	no HT	0.016	0.001	0	0	0
	with HT	1.043	0.691	0.466	0.257	0.040
C6288	no HT	0.009	0.001	0	0	0
	with HT	1.047	0.684	0.463	0.256	0.044
C7552	no HT	0.004	0.001	0	0	0
	with HT	1.044	0.685	0.459	0.264	0.049

C. Test Points Overhead

Our delay characterization process incurs the following sources of overhead: (1) the area cost for the inserted test points; and (2) the test time required by the delay measurements.

We evaluate overhead (1) by identifying the number of test points (i.e., flip-flops) that are required to measure the delay paths and cover all the gates in the target circuit. The column "# Test Points" in Table II shows the total number of test points required by our approach in each benchmark. We see that the number of test points is relatively small compared to the size of the circuit. In particular, we evaluate the area overhead by referring to the transistor counts, as reported in [17], which are needed to implement the test points as well as the existing gates in the target circuit. Figure 4 shows evaluation results of the area overhead, which is calculated as the ratio between the transistor count of the inserted test points and that of the original design. The results indicate that the area overheads in all tested benchmark circuits are below 25%.

In Table II, we show the number of characterized paths required by our approach as the indicator of overhead (2) (i.e., the cost of test). Our intuition is that the number of measured paths is an important metric to evaluate the test time of the hardware Trojan detection approach, since it is often the case that the measurements of multiple paths cannot be parallelized. Therefore, the overall test time is approximately the sum of test time on each path. The results show that our region-based approach requires a limited number of delay paths to be measured, compared to the huge number of paths in the circuit. The low cost of test is obtained from the non-overlapping circuit partition.

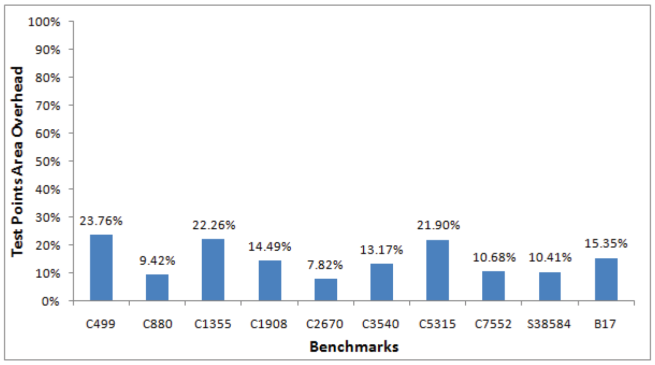

Fig. 4. Area overhead of the inserted test points.

VII. CONCLUSION

We have developed a region-based test point insertion method that can be used for gate-level delay characterization to detect the hardware Trojans. We partitioned the circuit into small non-overlapping regions using maximum fanout free cones. The test point insertion approach based on the circuit partition ensured that all the components on the target circuit are characterizable in terms of gate-level delay properties. Consequently, our HT detection method, which uses an additional HT variable during the course of delay characterization, is capable of covering all possible circuit locations. We evaluated the approach on a set of ISCAS and ITC benchmarks. The results indicated accurate HT detection with limited area overhead and test time.

TABLE II. OVERHEAD OF TEST POINT INSERTION IN TERMS OF THE NUMBER OF TEST POINTS (I.E., AREA OVERHEAD) AND THE NUMBER OF MEASURED PATHS (I.E., COST OF TEST)

Benchmark	# Gates	# Inputs	# Outputs	# Test Points	# Measured Paths	Avg # Measurements Per Path
C499	202	41	32	36	58	7.77
C880	383	60	26	25	41	11.61
C1355	546	41	32	68	58	9.41
C1908	880	33	25	82	49	19.13
C2670	1193	233	140	65	46	32.24
C3540	1669	50	22	155	125	20.10
C5315	2307	178	123	354	313	13.71
C7552	3512	207	108	296	155	26.14
S38584	19253	38	304	1328	1073	22.02
B17	32192	37	97	3047	1666	18.47

VIII. ACKNOWLEDGEMENTS

This work was supported in part by the NSF Award CNS-0958369, Award CNS-1059435, and Award CCF-0926127, and in part by the Air Force Award FA8750-12-2-0014.

REFERENCES

[1] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar. Trojan detection using IC fingerprinting. In *IEEE Symposium on Security and Privacy (S&P)*, pages 296–310, 2007.

[2] Y. Alkabani and F. Koushanfar. Consistency-based characterization for IC Trojan detection. In *International Conference on Computer-Aided Design (ICCAD)*, pages 123–127, 2009.

[3] A. Asenov. Random dopant induced threshold voltage lowering and fluctuations in sub-0.1 μm MOSFET's: A 3-D "atomistic" simulation study. *IEEE Transactions on Electron Devices*, 45(12):2505–2513, 1998.

[4] M. Banga and M. Hsiao. A region based approach for the identification of hardware Trojans. In *International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 40–47, 2008.

[5] M. Banga and M. Hsiao. VITAMIN: Voltage inversion technique to ascertain malicious insertions in ICs. In *International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 104–107, 2009.

[6] B. Cline, K. Chopra, D. Blaauw, and Y. Cao. Analysis and modeling of CD variation for statistical static timing. In *International Conference on Computer-Aided Design (ICCAD)*, pages 60–66, 2006.

[7] J. Cong, H. Li, S. Lim, T. Shibuya, and D. Xu. Large scale circuit partitioning with loose/stable net removal and signal flow based clustering. In *International Conference on Computer Aided Design (ICCAD)*, pages 441–446, 1997.

[8] Y. Jin and Y. Makris. Hardware Trojan detection using path delay fingerprint. In *International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 51–57, 2008.

[9] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor. Trustworthy hardware: Identifying and classifying hardware Trojans. *IEEE Computer Magazine*, 43(10):39–46, 2010.

[10] F. Koushanfar and A. Mirhoseini. A unified framework for multimodal submodular integrated circuits Trojan detection. *IEEE Transactions on Information Forensics and Security*, 6(1):162–174, 2011.

[11] J. Li and J. Lach. At-speed delay characterization for IC authentication and Trojan horse detection. In *International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 8–14, 2008.

[12] http://www.eda.ncsu.edu/wiki/freepdk.

[13] M. Majzoobi, E. Dyer, A. Elnably, and F. Koushanfar. Rapid FPGA characterization using clock synthesis and signal sparsity. In *International Test Conference (ITC)*, pages 1–10, 2010.

[14] D. Markovic, C. Wang, L. Alarcon, T. Liu, and J. Rabaey. Ultralow-power design in near-threshold region. *Proceedings of the IEEE*, 98(2):237–252, 2010.

[15] M. Potkonjak, A. Nahapetian, M. Nelson, and T. Massey. Hardware Trojan horse detection using gate-level characterization. In *Design Automation Conference (DAC)*, pages 688 –693, july 2009.

[16] G. Qu and M. Potkonjak. *Intellectual Property Protection in VLSI Design Theory and Practice*. Kluwer Publishing, 2003.

[17] J. Rabaey, A. Chandrakasan, and B. Nikolic. *Digital Integrated Circuits*. Prentice-Hall, 2003.

[18] J. Rajendran, V. Jyothi, and R. Karri. Blue team red team approach to hardware trust assessment. In *International Conference on Computer Design (ICCD)*, pages 285–288, 2011.

[19] M. Tehranipoor and F. Koushanfar. A survey of hardware Trojan taxonomy and detection. *IEEE Design Test of Computers*, 27(1):10–25, 2010.

[20] M. Tehranipoor, H. Salmani, X. Zhang, X. Wang, R. Karri, J. Rajendran, and K. Rosenfeld. Trustworthy hardware: Trojan detection and design-for-trust challenges. *IEEE Computer Magazine*, 44(7):66–74, 2011.

[21] S. Wei, J. Ahnn, and M. Potkonjak. Energy attacks and defense techniques for wireless systems. In *ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec)*, pages 185–194, 2013.

[22] S. Wei, K. Li, F. Koushanfar, and M. Potkonjak. Provably complete hardware Trojan detection using test point insertion. In *International Conference on Computer-Aided Design (ICCAD)*, pages 569–576, 2012.

[23] S. Wei, S. Meguerdichian, and M. Potkonjak. Gate-level characterization: Foundations and hardware security applications. In *Design Automation Conference (DAC)*, pages 222–227, 2010.

[24] S. Wei, S. Meguerdichian, and M. Potkonjak. Malicious circuitry detection using thermal conditioning. *IEEE Transactions on Information Forensics and Security*, 6(3):1136–1145, 2011.

[25] S. Wei, A. Nahapetian, M. Nelson, F Koushanfar, and M. Potkonjak. Gate characterization using singular value decomposition: Foundations and applications. *IEEE Transactions on Information Forensics and Security*, 7(2):765–773, 2012.

[26] S. Wei and M. Potkonjak. Scalable consistency-based hardware Trojan detection and diagnosis. In *International Conference on Network and System Security (NSS)*, pages 176–183, 2011.

[27] S. Wei and M. Potkonjak. Scalable hardware Trojan diagnosis. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 20(6):1049–1057, 2012.

[28] S. Wei and M. Potkonjak. The undetectable and unprovable hardware trojan horse. In *Design Automation Conference (DAC), to appear*, 2013.

[29] F. Wolff, C. Papachristou, S. Bhunia, and R. Chakraborty. Towards Trojan-free trusted ICs: Problem analysis and detection scheme. In *Design, Automation and Test in Europe (DATE)*, pages 1362–1365, 2008.

[30] J. Zhang, H. Yu, and Q. Xu. HTOutlier: Hardware Trojan detection with side-channel signature outlier identification. In *International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 55–58, 2012.