ON DESIGN VULNERABILITY ANALYSIS AND TRUST BENCHMARKS DEVELOPMENT

Hassan Salmani and Mohammad Tehranipoor[1]
The University of Connecticut
{salmani_h, tehrani}@engr.uconn.edu

Ramesh Karri
Polytechnic Institute of New York University
rkarri@duke.poly.edu

Abstract — **The areas of hardware security and trust have experienced major growth over the past several years. However, research in Trojan detection and prevention lacks standard benchmarks and measurements, resulting in inconsistent research outcomes, and ambiguity in analyzing strengths and weaknesses in the techniques developed by different research teams and their advancements to the state-of-the-art. We have developed innovative methodologies that, for the first time, more effectively address the problem. We have developed a vulnerability analysis flow. The flow determines hard-to-detect areas in a circuit that would most probably be used for Trojan implementation to ensure a Trojan goes undetected during production test and extensive functional test analysis. Furthermore, we introduce the Trojan detectability metric to quantify Trojan activation and effect. This metric offers a fair comparison for analyzing weaknesses and strengths of Trojan detection techniques. Using these methodologies, we have developed a large number of trust benchmarks that are available for use by the public, as well as researchers and practitioners in the field.**

1. INTRODUCTION

Adopted in the interest of economy, the horizontal integrated circuit design process has raised serious concerns for national security and critical infrastructures. An adversary is afforded plenty of opportunities to interfere with its design process, and circuit parametric or functional specifications may be jeopardized, allowing malicious activities [1][2][3][4]. Any intentional modification of design specifications and functionality to undermine its characteristics and correctness is called a hardware Trojan.

Hardware Trojans can be realized by including additional circuits at the register transfer or gate level or by changing circuit parameters like wire thickness or component size at the layout level, to name a few. Hardware Trojans can reduce circuit reliability, change or disable its functionality at a certain time, reveal its detailed implementation, or grant covert access to unauthorized entities [5][6].

A number of proposed approaches facilitate hardware Trojan detection by analyzing circuit side-channel signals or by increasing the probability of Trojan full activation. Incurred extra switching activity or induced additional wiring and gate capacitance affects circuit side-channel signals such as power and delay. Path delay fingerprint and delay measurement based on shadow registers are techniques intended to capture Trojan impact on circuit delay characteristics [7][8]. Transient current integration, circuit power fingerprint, and static current analysis are power-based Trojan detection techniques [9][10][11]. Efficient pattern generation is also necessary to discover a Trojan's impact upon circuit characteristics beyond process and environmental variations [12][13]. In addition to new pattern generation techniques, design-for-hardware-trust methodologies have been proposed to increase switching activity inside a Trojan circuit while reducing main circuit switching activity acting as background noise, to enhance Trojan detection resolution [1][2][4].

While a large number of research groups are actively working on the hardware Trojan detection problem, there is little effort in developing a standard approach or platform for evaluating and comparing contributions and results. There have been some efforts toward developing Trojans through contests like the Embedded Systems Challenge (ESC) during Cyber Security Awareness Week (CSAW) [14], but these are mainly implemented in an ad hoc fashion. The authors in [15] developed a technique to insert a Trojan in a circuit as small as a gate and demonstrated the difficulty of detection of such Trojans in several circuits. Furthermore, experiments are usually individualized to different circuits and measurements are presented without a baseline for comparison. As a result, it is nearly impossible to analyze the effectiveness of different techniques and to compare their strengths and weaknesses.

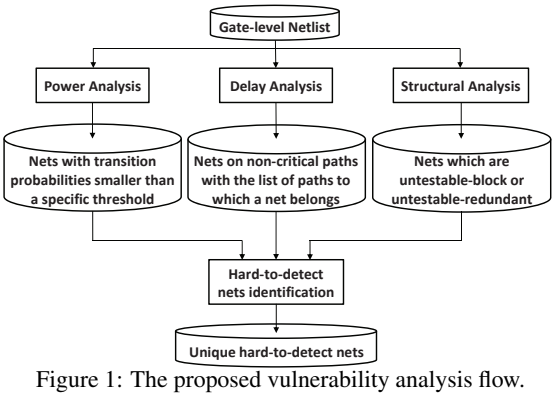

Figure 1: The proposed vulnerability analysis flow.

This paper presents a novel and systematic approach to address these issues. We propose a design vulnerability analysis flow that determines which parts of a circuit are more susceptible to Trojan insertion. This is based on the assumption that a Trojan must pass all production tests and extensive functional testing. To determine detectability of a Trojan, we also introduce the Trojan detectability metric to quantify resiliency of the Trojan against side-channel analyses. Using our proposed vulnerability analysis flow, we generate a large number of trust benchmarks that are circuits intentionally tampered with. The Trojan detectability metric and Trust benchmarks create standard measurements to compare different techniques fairly and to address the hardware Trojan problem more effectively.

Section 2 presents the design vulnerability analysis flow, and the Trojan detectability metric is defined in Section 3. In Section 4, we will present trust benchmarks, as well as results on vulnerability and detectability of several benchmarks. Section 5 concludes the paper.

2. DESIGN VULNERABILITY ANALYSIS

Functional hardware Trojans are realized by adding or removing gates; therefore, the inclusion of Trojan gates or the elimination of circuit gates affects circuit side-channel signals such as power consumption and delay characteristics, as well as the functionality. To minimize a Trojan's contribution to the circuit side-channel signals, an adversary can exploit hard-to-detect areas (e.g. nets) to implement the Trojan. Hard-to-detect areas are defined as areas in a circuit not testable by well-known fault-testing techniques (stuck-at, transition delay, path delay, and bridging faults) or not having noticeable impact on the circuit side-channel sig-

[1]THIS WORK WAS SUPPORTED IN PART BY THE NATIONAL SCIENCE FOUNDATION (NSF) UNDER GRANTS CNS-0844995 AND CNS-1059390.

nals. We propose a vulnerability analysis flow to identify such hard-to-detect areas in a circuit. These areas provide opportunities to insert hard-to-detect Trojans and invite researchers to develop techniques to make it difficult for an adversary to insert Trojans.

As Figure 1 shows, our proposed vulnerability analysis flow performs power, delay, and structural analyses on a circuit to extract the hard-to-detect areas. Any transition inside a Trojan circuit increases the overall transient power consumption; therefore, it is expected that Trojan inputs are supplied by nets with low transition probabilities to reduce activity inside the Trojan circuit.

The *Power Analysis* step in Figure 1 is based on analyzing switching activity; it determines the transition probability of every net in the circuit assuming the probability of 0.5 for '0' or '1' at primary inputs and at memory cells' outputs. Then, nets with transition probabilities below a certain threshold are considered as possible Trojan inputs. The *Delay Analysis* step performs path delay measurement based on gates' capacitance. This allows to measure the additional delay induced by Trojan by knowing the added capacitance to circuit paths. The Delay Analysis step identifies nets on non-critical paths as they are more susceptible to Trojan insertion and harder to detect their changed delay. To further reduce Trojan impact on circuit delay characteristics, it also reports the paths to which a net belongs to avoid selecting nets belonging to different sections of one path. The *Structural Analysis* step executes the structural transition delay fault testing to find untestable blocked and untestable redundant nets. Untestable redundant nets are not testable because they are masked by a redundant logic, and they are not observable through primary output or scan cells. Untestable blocked nets are not controllable or observable by untestable redundant nets. Tapping Trojan inputs to untestable nets hides Trojan impact on delay variations.

At its end, the vulnerability analysis flow reports unique hard-to-detect nets that are the list of untestable nets with low transition probabilities and nets with low transition probabilities on non-critical paths while not sharing any common path. Note that when a Trojan impacts more than one path, it provides greater opportunities for detection. Avoiding shared paths makes a Trojan's contribution to affected paths' delay minimal, which can be masked by process variations, making it difficult to detect and distinguish the added delay from variations. The reported nets are ensured to be untestable by structural test patterns used in production tests. They also have low transition probabilities so Trojans will negligibly affect circuit power consumption. As the nets are chosen from non-critical paths without any shared segments, it would be extremely difficult to detect Trojans by delay-based techniques.

The vulnerability analysis flow can be implemented using most electronic design automation (EDA) tools, and the complexity of the analysis is linear with respect to the number of nets in the circuit. We apply the flow to the Ethernet MAC 10GE circuit [16], which implements 10Gbps Ethernet Media Access Control functions. Synthesized at 90nm Synopsys technology node, the Ethernet MAC 10GE circuit consists of 102,047 components, including 21,830 flip-flops. The Power Analysis shows that out of 102,669 nets in the circuit, 23,783 of them have a transition probability smaller than 0.1, 7003 of them smaller than 0.01, 367 of them smaller than 0.001, and 99 of them smaller than 0.0001. The Delay Analysis indicates that the largest capacitance along a path, representing path delay, in the circuit is 0.065717825 pF, and there are 14,927 paths in the circuit whose path capacitance are smaller than 70% of the largest capacitance, assuming that paths longer than 70% in a circuit can be tested using testers. The Structural Analysis finds that there is no untestable fault in the circuit. By excluding nets sharing different segments of one path, there are 494 nets in the Ethernet MAC 10GE circuit considered to be areas where Trojan inputs could be used while ensuring the high difficulty of detection based on side-channel and functional test techniques.

3. TROJAN RANKING

A Trojan's impact on circuit characteristics depends on its implementation. Trojan inputs tapped from nets with higher transition probabilities will aggrandize switching activity inside the Trojan circuit and increase its contribution to circuit power consumption. Furthermore, the Trojan might affect circuit delay characteristics due to additional capacitance induced by extra routing and Trojan gates. To quantitatively determine the difficulty of detecting a gate-level Trojan, a procedure is developed to determine Trojan detectability based on its impact on delay and power across different circuits. Trojan detectability can establish a fair comparison among different hardware Trojan detection techniques since it is based on induced variations by a Trojan in side-channel signals.

The Trojan detectability metric is determined by (1) the number of transitions in the Trojan circuit and (2) extra capacitance induced by Trojan gates and their routing. This metric is designed to be forward-compatible with new approaches for Trojan detection by introducing a new variable, for example a quantity related to the electromagnetic field.

Transitions in a Trojan circuit reflect Trojan contribution to circuit power consumption, and Trojan impact on circuit delay characteristic is represented by measuring the added capacitance by the Trojan. Assuming A_{Trojan} represents the number of transitions in the Trojan circuit, S_{Trojan} the Trojan circuit size in terms of the number of cells, A_{TjFree} the number of transitions in the Trojan-free circuit, S_{TjFree} the Trojan-free circuit size in terms of the number of cells, TIC the added capacitance by Trojan as Trojan-induced capacitance, and C_{TjFree} the Trojan-affected path with the largest capacitance in the corresponding Trojan-free circuit, Trojan detectability ($T_{Detectability}$) at the gate-level is defined as

$$T_{Detectability} = |t| \tag{1}$$

where

$$t = \left(\frac{A_{Trojan}/S_{Trojan}}{A_{TjFree}/S_{TjFree}}, \frac{TIC}{C_{TjFree}} \right) \tag{2}$$

$T_{Detectability}$ at the gate-level is calculated as follows:

1. Apply random inputs to a Trojan-free circuit and obtain the number of transitions in the circuit (A_{TjFree}).
2. Apply the same random vectors to the circuit with a Trojan and obtain the number of transitions in the Trojan circuit (A_{Trojan}).
3. Perform the Delay analysis on the Trojan-free and Trojan-inserted circuits.
4. Obtain the list of paths whose capacitance are changed by the Trojan.
5. Determine the Trojan-affected path with the largest capacitance in the corresponding Trojan-free (C_{TjFree}) and the added capacitance (TIC).
6. Form the vector t in Eq. (2) and compute $T_{Detectability}$ as defined in Eq. (1). Note that Trojan detectability represents the difficulty of detecting a Trojan.

As an example, a 16-input comparator Trojan consisting of 12 gates is inserted at four different locations, namely TjG-Loc1, TjG-Loc2, TjG-Loc3, and TjG-Loc4, in the Ethernet MAC 10GE circuit, and Table 1 shows their detectability. The Ethernet MAC 10GE circuit consists of 102047 cells, Column 3 S_{TjFree}, while the Trojan size with 12 cells, Column 5 S_{Trojan}, is only about 0.011% of the entire circuit. TjG-Loc4, in Row 5, experiences the largest switching activity (13484 in Column 4) and relatively induces high TIC (0.004932996 pF in Column 6). It is expected that TjG-Loc4 will be the easiest Trojan to be detected due to more impact on circuit side-channel signals, and in turn the detectability of TjG-Loc4 ($T_{Detectability} = 1.079105$ in Column 8) is

Table 1: The detectability of the comparator Trojan placed at four different locations in Ethernet MAC 10GE circuit.

Trojan	A_{TjFree}	S_{TjFree}	A_{Trojan}	S_{Trojan}	$TIC(pF)$	$C_{TjFree}(pF)$	$T_{Detectability}$
TjG-Loc1	106,664,486	102,047	10,682	12	0.000286935	0.041358674	0.851659
TjG-Loc2	106,664,486	102,047	4,229	12	0.004969767	0.072111502	0.344132
TjG-Loc3	106,664,486	102,047	3,598	12	0.005005983	0.049687761	0.304031
TjG-Loc4	106,664,486	102,047	13,484	12	0.004932996	0.052602269	1.079105

higher than the others. Although the induced capacitance by TjG-Loc2 (0.004969767 pF), in Row 3, is more than the capacitance induced by TjG-Loc1 (0.000286935 pF), in Row 2, TjG-Loc1 has more significant contribution into circuit switching activity, 10682 versus 4229 in Column 4. Therefore, TjG-Loc1 has the second largest detectability (0.851659) after TjG-Loc4. Among TjG-Loc2 and TjG-Loc3, although TjG-Loc3, in Row 4, has slightly larger induced capacitance (0.005005983 pF), TjG-Loc2 experiences more switching activity (4229 versus 3598 in Column 4). The two Trojans have close detectability where TjG-Loc2 stands above and TjG-Loc3 remains the hardest Trojan to be detected with the lowest Trojan detectability.

4. TRUST BENCHMARKS

Using the above mentioned methodologies, we still develop a set of benchmarks at different levels of abstraction with different $T_{Detectability}$. For each original circuit, we first apply our circuit vulnerability analysis flow to distinguish hard-to-detect nets. We then implement different Trojans, and finally $T_{Detectability}$ of each Trojan is determined.

A trust benchmark is generated from a circuit to which a Trojan is deliberately added. We initiate the first efforts to develop static trust benchmarks, which we define as those that a Trojan is inserted into, and the location and size of the Trojan does not change. The goal is to provide standard benchmarks that ensure fair comparison between hardware Trojan detection techniques and to enable researchers to more effectively address this challenging problem. Our developed trust benchmarks are available at http://www.trust-hub.org/taxonomy. This is an ongoing effort, and we continue to generate various trust benchmarks. Further, Trojans are being designed with different activation probabilities to evaluate the limitations of Trojan detection techniques.

Table 2 presents a complete list of trust benchmarks that have been developed so far. They are categorized based on the Trojan taxonomy in [17], and the number of trust benchmarks for each type and main circuits are presented. For instance, the table shows 25 Trojans are inserted at the gate-level, 51 at the register-level, and 12 at the layout level, under Abstraction Level row. As another example, the Effect row shows that 35 Trojans change circuit functionality, 3 degrade circuit performance, 24 leak information to outside of chip, and 34 will deny the service when activated.

Tables 3 and 4 show detailed analysis of a selected number of gate-level benchmarks. In Table 4, Column 3 indicates that b19 circuit, in Row 2, is the largest circuit in size (62835) among the selected circuits. Table 3 also shows the number nets with transition probability less than 0.0001 in b19, 4530 in Row 2 and Column 6, is larger than the other circuits, and b19 has considerable number of paths whose capacitances are less than 70% of its critical path' capacitance, 474358 in Column 8. Further, there are eight untestable faults is b19, in Column 9. These provide significant opportunity for implanting Trojans resilient against power and delay side-channel analyses in b19. Table 4 confirms that b19-T100 with $T_{Detectability} = 0.024978531$, in Column 8, is the second most difficult Trojan to detect as no transition inside the Trojan is observed, 0 in Column 4, and it induces small capacitance, 0.000945429 pF in Column 6, on a non-critical path, 0.037849663 pF in Column 7. s38584-T200, in Row 7, has the lowest detectability, 0.0139008691 in Column 8; similar to b19-T100, there is no switching activity in s38584-T200, 0 in Column 4, and s38584-T200 induces less capacitance, 0.000414827 pF in Column 6, on a shorter path, 0.029841803 pF in Column 7, compared to b19-T100.

5. CONCLUSIONS

The hardware trust community needs to place in a common ground to more effectively address the Trojan detection problem. As no standard measurements, benchmarks, or tools have previously been developed, we put our effort into developing tools that, for the first time, analyze circuit vulnerabilities to hardware Trojans, determine Trojan detectability, and generate trust benchmarks. The vulnerability analysis flow determines areas in a circuit that are more probable to be used for Trojan implementation. Then, we defined the Trojan detectability metric to quantify Trojan impact on circuit power consumption and circuit performance. This metric helps determine the capability of a technique in Trojan detection. We also generated a large number of trust benchmarks available to researchers for evaluating their contributions.

6. REFERENCES

[1] S. Bhunia, M. Abramovici, D. Agarwal, P. Bradley, M. S. Hsiao, J. Plusquellic, and M. Tehranipoor, "Protection against Hardware Trojan Attacks: Towards a Comprehensive Solution," *IEEE Design & Test of Computers,* Issue: 99, pp. 1, 2012.

[2] M. Tehranipoor and F. Koushanfar, "A Survey of Hardware Trojan Taxonomy and Detection," *IEEE Design & Test of Computers,* Volume: 27, Issue: 1, pp. 10-25, 2010.

[3] R. Karri, J. Rajendran, K. Rosenfeld, M. Tehranipoor, "Trustworthy Hardware: Identifying and Classifying Hardware Trojans," *IEEE Computer,* Volume: 43, Issue: 10, pp. 39-46, 2010

[4] 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,* Volume: 44, Issue: 7, pp. 66-74, 2011.

[5] Y. Jin, D. Maliuk, and Y. Makris, "Post-deployment Trust Evaluation in Wireless Cryptographic ICs," in Proc. of the *IEEE Design, Automation & Test in Europe Conference & Exhibition (DATE12)*, PP. 965-970, 2012.

[6] X. Zhang and M. Tehranipoor, "Case study: Detecting Hardware Trojans in Third-party Digital IP Cores," in Proc. of the *IEEE International Workshop on Hardware-Oriented Security and Trust (HOST11)*, pp. 67-70, 2011.

[7] Y. Jin and Y. Makris, "Hardware Trojan Detection using Path Delay Fingerprint," in the Proc. of the *IEEE International Workshop on Hardware-Oriented Security and Trust (HOST08),* pp. 51-57, 2008.

[8] J. Li and J. Lach, "At-speed Delay Characterization for IC Authentication and Trojan Horse Detection," in the Proc. of the *IEEE International Workshop on Hardware-Oriented Security and Trust (HOST08),* pp. 8-14, 2008.

[9] X. Wang, H. Salmani, M. Tehranipoor, and J. Plusquellic, "Hardware Trojan Detection and Isolation Using Current Integration and Localized Current Analysis," in the Proc. of the *IEEE International Symposium on Fault and Defect Tolerance in VLSI Systems (DFT08),* pp. 87-95, 2008.

[10] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar, "Trojan Detection using IC Fingerprinting," in the Proc. of the *IEEE Symposium on Security and Privacy,* pp. 296-310, 2007.

[11] R. Rad, X. Wang, J. Plusquellic, and M. Tehranipoor, "Power Supply Signal Calibration Techniques for Improving Detection Resolution to Hardware Trojans," in the Proc. of the *International Conference on Computer-Aided Design(ICCAD08),* pp. 632-639, 2008.

[12] M. Banga and M. S. Hsiao, "A Novel Sustained Vector Technique for the Detection of Hardware Trojans," in the Proc. of the *International Conference on VLSI Design (VLSID09),* pp. 327-332, 2009.

[13] F. Wolff, C. Papachristou, S. Bhunia, and R.S. Chakraborty, "Towards Trojan-Free Trusted ICs: Problem Analysis and Detection Scheme," in the Proc. of the *Design, Automation and Test in Europe (DATE08),* pp. 1362-1365, 2008.

[14] Cyber Security Awareness Week (CSAW) *http://http://www.poly.edu/csaw2012*

[15] S. Wei, K. Li, F. Koushanfar, and M. Potkonjak, "Hardware Trojan horse benchmark via optimal creation and placement of malicious circuitry" in Proc. of the *ACM Design Automation Conference (DAC12)*, pp. 90-95, 2012.

[16] Ethernet 10GE MAC, *http://opencores.org/project,xge_mac*

[17] TrustHub, *http://www.trust-hub.org/*

Table 2: Trust benchmarks according to the taxonomy [17].

Category	Trojans		Main circuits
	Type	# of Trust benchmarks (Sample Benchmarks)	
Insertion phase	Specification	0	-
	Design	80 (AES-T100, EthernetMAC10GE-T700, MC8051-T800, ...)	AES, BasicRSA, EthernetMAC10GE, MC8051, PIC16F84, RS232, s15850, s35932, s38417, s38584, vga_lcd, wb_conmax
	Fabrication	8 (EthernetMAC10GE-T100, MultPyramid-T100, ...)	EthernetMAC10GE, MultPyramid
	Testing	0	-
	Assembly and Package	0	-
Abstraction Level	System	0	-
	Development Environment	0	-
	Register Transfer	51 (b19-T300, BasicRSA-T100, PIC16F84-T100, ...)	AES, b19, BasicRSA, MC8051, PIC16F84, RS232, wb_conmax
	Gate	25 (RS232-T1000, vga_lcd-T100, wb_conmax-T100, ...)	b19, EthernetMAC10GE, RS232, s15850, s35932, s38417, s38584, vga_lcd, wb_conmax
	Layout	12 (EthernetMAC10GE-T600, MultPyramid-T200, RS232-T2000, ...)	EthernetMAC10GE, MultPyramid, RS232
	Physical	0	-
Activation Mechanism	Always On	11 (AES-T300, EthernetMAC10GE-T600, MultPyramid-T200, ...)	AES, EthernetMAC10GE, MultPyramid
	Triggered	79 (PIC16F84-T300, s38584-T100, vga_lcd-T100, ...)	AES, b19,BasicRSA, EthernetMAC10GE, MC8051, MultPyramid, PIC16F84l,RS232, s15850, s35932, s38417, s38584, vga_lcd, wb_conmax
Effect	Change the Functionality	35 (b19-T500, s38417-T100, wb_conmax-T200, ...)	b19, EthernetMAC10GE, MC8051, RS232, s15850, s35932, s38417, s38584, vga_lcd, wb_conmax
	Degrade Performance	3 (EthernetMAC10GE-T100, MultPyramid-T100, s35932-T300)	EthernetMAC10GE, MultPyramid, s35932
	Leak Information	24 (BasicRSA-T100, s35932-T100, s38584-T200, ...)	AES, BasicRSA, PIC16F84, s35932, s38584
	Denial of Service	34 (s15850-T100, s38584-T100, wb_conmax-T300, ...)	AES, BasicRSA, EthernetMAC10GE, MC8051, MultPyramid, PIC16F84, RS232, s15850, s35932, s38417, s38584, vga_lcd, wb_conmax
Location	Processor	51 (BasicRSA-T300, s38584-T100, vga_lcd-T100, ...)	AES, b19, BasicRSA, MC8051, MultPyramid, PIC16F84, s15850, s35932, s38417, s38584, vga_lcd
	Memory	0	-
	I/O	4 (MC8051-T300, wb_conmax-T100, wb_conmax-T200, ...)	MC8051, wb_conmax
	Power Supply	2 (EthernetMAC10GE-T400, EthernetMAC10GE-T500)	MC8051-T300, wb_conmax
	Clock Grid	2 (EthernetMAC10GE-T200, EthernetMAC10GE-T300)	EthernetMAC10GE
Physical Characteristic	Distribution	2 (b19-T100, b19-T200)	b19
	Size	0	-
	Type	86 (b19-T500, BasicRSA-T100, MC8051-T700, ...)	AES, b19, BasicRSA, EthernetMAC10GE, MC8051, MultPyramid, PIC16F84, RS232, s15850, s35932, s38417, s38584, vga_lcd, wb_conmax
	Structure	8 (b19-T200, EthernetMAC10GE-T600, MultPyramid-T200, ...)	b19, EthernetMAC10GE, MultPyramid
Total	NA	88	NA

Table 3: Design vulnerability analysis of a selected number of Trojan-free circuits presented at [17].

Circuit	Power Analysis					Delay Analysis		Structural Analysis
	# Nets	< 0.1	< 0.01	< 0.001	< 0.0001	Critical path Capacitance(pF)	$< 70\%$ of Critical Path Capacitance	# Untestable faults
b19	70,259	14,482	8,389	5,533	4,530	0.37723719	474,358	8
s38417	5,669	589	291	219	69	0.050146392	41,901	0
s38584	7,203	817	197	85	30	0.044666893	27,689	0
s35932	6,269	0	0	0	0	0.00851317	3,156	0

Table 4: The detectability ($T_{Detectability}$) of a selected number of gate-level Trojans inserted in the circuits in Table 3

Trojan	A_{TjFree}	S_{TjFree}	A_{Trojan}	S_{Trojan}	$TIC(pF)$	$C_{TjFree}(pF)$	$T_{Detectability}$
b19-T100	4,037,383	62,835	0	83	0.000945429	0.037849663	0.024978531
s38417-T100	2,717,682	5,329	59	11	0.004167929	0.032341219	0.139390942
s38417-T200	2,717,682	5,329	1,328	11	0.005313744	0.030518107	0.4108472958
s38417-T300	2,717,682	5,329	257	15	0.000457899	0.03078216	0.0484715731
s38584-T100	423,986	6,473	705	9	0.000945429	0.015039353	1.2587797351
s38584-T200	423,986	6,473	0	83	0.000414827	0.029841803	0.0139008691
s38584-T300	423,986	6,473	16	731	0.012461155	0.004379333	2.8457800321
s35932-T100	353,304	5,426	354	15	0.000494403	0.006985635	0.8664403555
s35932-T200	353,304	5,426	733	12	0.003160179	0.009732743	1.2628060457
s35932-T300	353,304	5,426	738	36	0.000497869	0.008230419	0.3753278457