Preventing Hardware Trojan Insertion through Logic Masking

Sophie Dupuis, Marie-Lise-Flottes, Giorgio Di Natale, Bruno Rouzeyre
LIRMM (Université de Montpellier / CNRS UMR 5506)
Montpellier, France
firstname.lastname@lirmm.fr

Abstract —**Due to the evolution in the IC supply chain, IPs and dies come from numerous, and possibly untrusted, sources. This loss of control over the entire production flow may thus lead to several threats including mask theft, overproduction, as well as the insertion of malicious alterations to the circuits, referred to as Hardware Trojans (HTs). To protect circuits from overproduction, the target circuit's function can be masked so that only authorized customers can use them. In this paper, we propose a masking technique for prevention of overproduction that also helps thwarting HT insertion.**

***Index Terms* — Hardware Trojan; Logic Masking; Design for Hardware Trust; Logic testing**.

I. INTRODUCTION

With ever-shrinking transistor technologies, the cost of new fabrication facilities is becoming prohibitive and outsourcing the fabrication process to low-cost locations has become a major trend in Integrated Circuits (ICs) industry in the last decade. In addition, the complexity of today integrated system requires the use of third party Intellectual Property (IP) cores. This raises the question about untrusted foundries and IP suppliers and therefore hardware piracy: mask theft and overproduction [1] as well as the insertion of malicious circuitry or alterations, referred to as Hardware Trojans (HTs) [2, 3, 4]. Two challenges arise: protect the ICs from mask theft and overproduction as well as be able to verify the trustworthiness of the manufactured ICs.

In order to protect ICs from theft and overproduction, the functionality of the ICs can be masked so that only authorized users can properly use them. This procedure, often referred to as encryption in literature [5, 6, 7], is achieved by the addition of a key, which the correct value must be provided for the proper operation of the ICs.

In order to protect ICs from HTs, several methods have been proposed that aim at detecting their presence. HTs detection methods are divided into two categories: methods based on *side-channel analysis* [8, 9], or *logic testing* [10, 11, 12]. *Side channel analysis* methods focus on observing some physical parameters of the circuit, such as power consumption [8] or timing [9]. Relying on golden ICs (i.e. circuit that have been ensured to be HT-free by destructive methods), a comparison is made with the circuits under test. The assumption is that the introduction of additional logic gates should become visible because of an increase of the power consumption of the circuit or an increase of the delay in the logic path containing the HT. However, the main weakness of these methods is to manage environment variations. The second category relies on testing-based approaches. These methods focus on so-called *rare values* based HTs i.e. HTs that aim at modifying the functionality of a circuit under very rare conditions as introduced in [10]. The main concern is therefore to be able to activate potential HTs i.e. find test vectors that can maximize the chances of activating the HTs [11, 12].

In addition to detection methods, so-called *design for hardware trust* methods aim at improving HT detectability or preventing an attacker from inserting HT thanks to specific design rules [13]. These methods can e.g. modify the state machine [14, 15] or the combinational logic [16, 17].

The insight of using logical encryption in order to fight against HTs has recently been introduced in [18]. In this paper, we propose such a technique whose general idea is to minimize so-called *rare values* in a circuit (i.e. low controllable signals). Assuming that a HT requires a triggering condition and that an attacker is likely to attach this condition on signals with low controllability in order to make the HT stealthy, the minimization of these rare values is supposed to make it harder for an attacker to exploit them to incorporate a HT's trigger.

The first contribution of this paper is a novel structural modification using XOR/XNOR gates as well as complex logic functions (e.g. Z=A.B+C) referred as ANDORI gates. The second contribution of this paper is an iterative algorithm that allows resulting in less impact on the original circuit in terms on extra delays and area for the same improvement in terms on controllability. Compared to the method presented in [18], the proposed approach differs in signal selection and structural modification. As presented in Section IV, it results in better controllability improvements.

This paper is organized as follows. In Section II, we recall existing "encryption" methods, HTs detection methods as well as design-for-Hardware-Trust methods. In Section III, we present our logic masking technique. Experimental data are reported in Section IV. Finally, Section V concludes the paper.

II. PRIOR WORK

A. Fight against overproduction

As introduced in [6], logic encryption means hiding the hardware functionality (not encrypting the design by a cryptographic algorithm).

The terms obfuscation and masking are also used in literature. Whatever the term used, the goal is to protect ICs from mask theft and overproduction by preventing unauthorized IC use thanks to extra logic that changes the circuit functionality when a proper key is not provided. This way, only authorized users (who know the key) can use the ICs. We will use for each paper the term used by the authors concerning the technic. Logic masking will be used for the proposed approach.

1) Combinational encryption

The combinational encryption technique proposed in [5] consists in randomly inserting XOR/XNOR gates into the design. One input of each of these gates is connected to an original circuit signal while the other one is controlled by a bit of the key. For proper IC use, the correct key value must be given so that the newly added gates do not inverse the value of the signals they are connected to.

However, a random insertion of the encryption gates cannot ensure that wrong keys corrupt the outputs as wanted: if a wrong key produces a correct output, or affects only one or a few bits of the output, this renders the encryption procedure weak, as explained in [6]. In this paper, an improvement of the previous approach is presented that aims at ensuring that any wrong key affects half of the output bits. The goal is to make it difficult for an attacker to find the secret key. The encryption procedure inserts XOR/XNOR, and the places to insert these gates are therefore chosen in order to achieve 50% Hamming distance between the correct and the wrong outputs.

2) Sequential encryption

A so-called netlist level obfuscation is presented in [7]. The goal is to combine a combinational encryption and a sequential encryption i.e. a modification of the state transition function (FSM). The newly inserted FSM defines two modes of operations: the normal and the obfuscated mode. By default, the IC is in obfuscated mode (FSM output = '1') and the key (a specific input sequence) allows toggling from the obfuscated mode to the normal one (FSM output = '0'). The combinational encryption is made with the addition of XOR gates connecting the FSM output and several signals in the circuit. Original signals are thus inverted in obfuscated mode thanks to the FSM output. The choice is made to encrypt signals with large fan-in and fan-out cones, assuming that modification of such nodes will affect a larger number of internal nodes and primary outputs.

B. Fight against HTs insertion

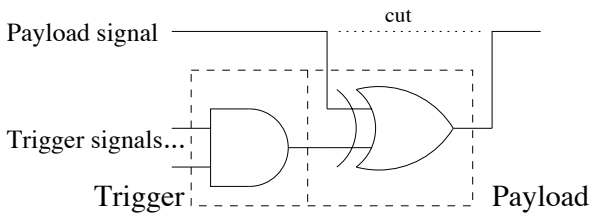


Figure 1. *Rare value* HT circuit model.

1) Logic testing and HT activation

In order to be able to detect a HT by logic testing, the main concern is to activate the HT in order to obtain an erroneous output. As introduced in [10], a HT is described in terms of its activation mechanism (referred as the *triggering condition* or *trigger*) and the introduced effect (referred as the *payload*) (cf. Fig. 1). Based on the assumption that the HT activation should occur under very rare conditions in order to minimize/avoid detection at test time, the goal is to create pattern generation techniques dedicated to detection of HTs, i.e. aiming at producing a reduced set of patterns that maximizes the chances of triggering potential HTs.

In [10], the main assumption is that triggers are supposed to depend on the least controllable signals, and payloads on the least observable signals. The goal is then to find the "most likely signals" on which an attacker could have attached a HT trigger and stitched a HT payload, and then, to generate patterns that aim at activating the potential triggers and propagating the potential payloads.

In [11], Chakraborty et al. propose a methodology called Multiple Excitation of Rare Occurrence (MERO). The idea is to improve the previous work by activating each potential trigger a given number of times.

In [12], the goal is the same, but several criteria are considered to better reflect the choices that may be made by an attacker for inserting HTs, assuming that she/he has access to the layout information. The attacker's goal in this case is to insert HT as stealthy as possible, from the functional, the performance, and the layout point of views. The selection of the sites is therefore based on the assumption that the HT is triggered (i) by signals with low controllability, (ii) in paths that are not critical in terms of delay, and (iii) combining multiple signals that are close from each other in the circuit's layout,.

2) Design-for-Harware-Trust

Inspired by Design-For-Testability solutions, so-called Design-For-Hardware-Trust solutions propose to modify the design methodology in order to incorporate features in the ICs' that can either help detections methods, or prevent an attacker from inserting a HT [13].

In [14] and [15], the state transition graph is modified in order to prevent HT insertion and facilitate detection during test. The idea is to help controlling the least controllable signals in a design, and observing the least observable signals. To do that, an input key allows activating a state machine into a *transparent mode* in which the key forces the least controllable signals to their rare value, in order to simulate the occurrence of a rare event that is likely to detect a HT. Besides, to improve the observation of the effect of the HT on the outputs, the values at the least observable signals are compacted into an observable signature.

Salmani et al. propose also in [16] a method that aims at increasing the probability of activating a HT during logic testing. The transition probability of a signal (i.e. the result of multiplying the probability of being '0' and the

probability of being ‘1’ of a signal) is used to estimate the number of clock cycles required to generate a. The method consists in inserting so-called *dummy scan flip-flops* to remove rare transitions. The goal is then that all signals have a transition probability above a given transition probability. This way, it makes it harder for an adversary to find which signals to use as a trigger.

Again with the aim of raising transition probabilities, it is proposed in [17] to insert a Probability Increase Circuit (PIC) in the ICs. This PIC consists in control-points based on OR or AND gates, depending on which value is rare.

C. Fight against overproduction and HTs insertion

It has been evoked in [6] that, if an IC is encrypted while it passes through the untrusted design phages, its functionality is not revealed. Encryption therefore prevents reverse engineering, cloning, as well as HT insertion. However, this HT protection is only a side effect if the encryption method does not focus on protecting against HT.

In [14, 15], the interest of a sequential obfuscation scheme to fight HT insertion is presented. It is assumed that such an approach is particularly oriented toward thwarting HT insertion since it prevents an attacker from exploiting the true state transition to insert a HT. Furthermore, it makes some inserted HTs benign: those only effective in obfuscated mode.

To the best of our knowledge, the first method combining combinational encryption and HT detection was proposed in [18]. The principle of this method is a combinational encryption that consists in the modification of the gate level netlist in order to encrypt the circuit with a newly added key, as in [5, 6]. However, the newly added gates are in this case inserted with the intention of helping thwarting the insertion of potential HTs. To do so, the goal is to improve the controllability of the low controllable signals, and then to "hide" their low controllable property from the attacker point of view.

III. RARE VALUE BASED LOGIC ENCRYPTION

In this paper, an encryption method dedicated to the detection of HTs is proposed, in the sense that it prevents an attacker from exploiting the “truly” low controllable signals of a circuit. In other words, the addition of new gates driven by a key aims at virtually changing the probabilities of every rare signal to be set to ‘0’ or ‘1’. As opposed to the work in [15, 16], in which the modification of the probabilities of the signals is effective only in a particular mode (scan test mode for detection at test time in [15] and a so-called testing mode in [16]), our encryption scheme modifies the probabilities of the signals in functional mode too. Based on the principle introduced in [18], the presented encryption scheme is more efficient in terms of removing low controllable signals, and is more secure regarding “decryption” algorithms [19].

A. Encryption

Encryption is done according to controllability values of signals. Signals with low controllability are encrypted, i.e. signals with a low probability of having value ‘0’ or ‘1’. The first parameter of our encryption technique is a chosen probability threshold (e.g. 0.1% chance of being ‘0’ or ‘1’). The aim of the procedure is then to obtain an encrypted circuit in which the number of signals with a probability below this threshold is as small as possible. It should be noted that the ideal would be to get close to 50% probability for all signals but the exact balance of 50% is not needed to control a signal to 1 or 0. That is why we seek to re balance the probabilities of nodes having the most unbalanced probabilities above a certain threshold. A second parameter can be given: the maximum number of gates used for the encryption. This value is then a trade-off between quality of encryption and area overhead.

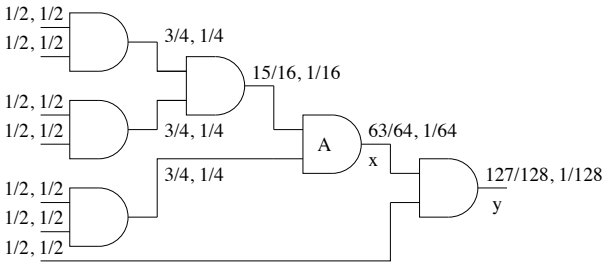
Our encryption is achieved by inserting XOR/XNOR as well as ANDORI gates in the design as presented in the example of Figure 3:

- First, the probabilities of all signals are computed (cf. Fig. 3.a). This is done thanks to the method described in [20]. Let us consider signal x with a 1-probability of 1/64. Let us assume that this probability is beyond the threshold.
- To increase this probability, an OR gate can be added (with a bit key equal to ‘0’) to “create” a signal x’ with more balanced probabilities (cf. Fig. 3.b-1) (a AND gate would be used in case of a rare 0). This new gate has also the effect of balancing the probabilities of downstream signals (cf. signal y). Nevertheless, signal x remains in the netlist. An encryption gate is therefore needed that aggregates the newly added gate and the gate driving the x signal (gate A in Fig. 3.b-1). In our example, a gate with the functionality “A.B+C” is needed. Gate A is therefore changed into this new gate, and the low controllable signal x is no longer in the netlist (cf. Fig 3.b-2). Note that a XOR/XNOR gate could also be used but gates aggregating the functionality “A.B xor C” usually do not exist, hence the interest of using AND/OR gate that permit the use of ANDORI gates.
- If the use of an ANDORI gate is not possible (typically if x is not the output of a AND or a OR gate), the idea is to encrypt one of the upstream signals i.e. the signals connected to the inputs of gate A. The idea is in that case to incorporate a XOR/XNOR gate. Since the encrypted signal is an upstream signal, there is indeed no need of aggregating the newly added gate. The upstream signal with the most unbalanced probabilities (as long as it does not belong to a critical path) is chosen since its probabilities modification will have the greatest impact on downstream signals (cf. Fig. 3.c).

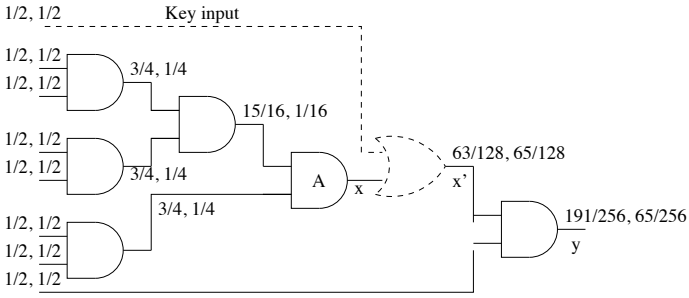
As one can see the encryption of x signal thanks to a ANDORI gate leads to far better results (1-probability=65/128) than the encryption on an upstream signal (1-probability=1/8=16/128). This is why this encryption is done in priority, and the encryption of an upstream signal, only if the first one was not possible.

The encryption scheme presented in [18] generates a 1-probability of 17/128 for signal x. The use of a OR gate

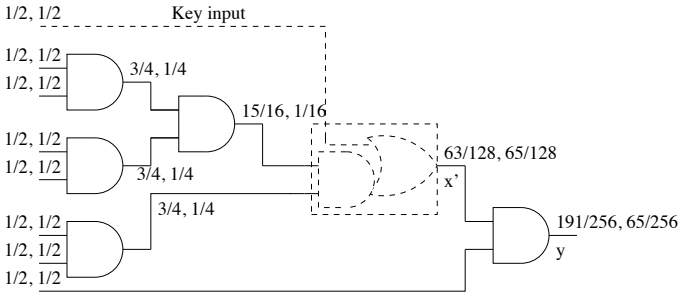
instead of a XOR gate leads indeed to lightly better improvements. However, the idea of using XOR/XNOR gate when only the encryption of an upstream signal is possible is preferred because it was presented in [19] that XOR/XNOR gates were more resilient than AND/OR gates to decryption attacks.



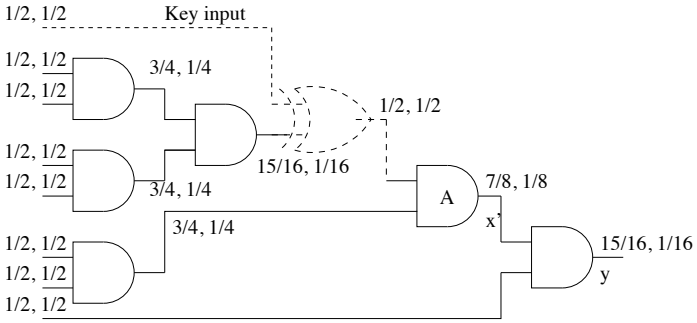
a) Probabilities (0-probability, 1-probability)



b-1) Change in probabilities



b-2) Encryption of the signal



c) Encryption of an upstream signal

Figure 3. Example of encryption (encryption in dotted lines)

An iterative process is used in order to change the probabilities of all the rare signals: among rare signals, the encryption is made primarily for the signals that are closer to the input pins. It is assumed that these are the signals for which controllability improvement will have the greatest impact on downstream signals: for example in Fig. 3, signal y is also a rare signal in the original netlist, but the encryption of this signal is not necessary anymore after signal x has been encrypted. Signal y would have unnecessarily been encrypted with a greedy algorithm as in [18].

Several phases of encryption are done and probabilities are computed between each phase of encryption. Furthermore, a check is done to remove an encryption gate if it does not change the probabilities as much as desired. This way, in case of a limitation in the possible number of encryption gates, this leaves room for a future encryption gate that could lead to better changes.

B. Encription key

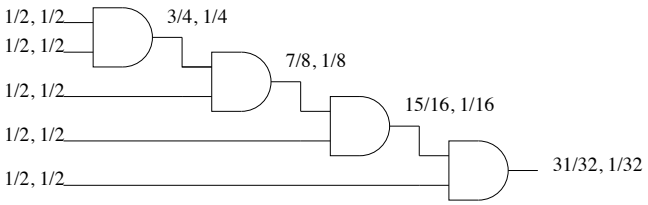
1) Size of the key

Depending on the size of the circuit to be encrypted, the number of necessary gates to encrypt the circuit may become very large. It is then not conceivable that the number of key bits represents a 1:1 ratio with respect to the number of encryption gates. In that case, so-called encryption signals (i.e. encryption gates inputs) have to be grouped to fit with the desired number of key bits.

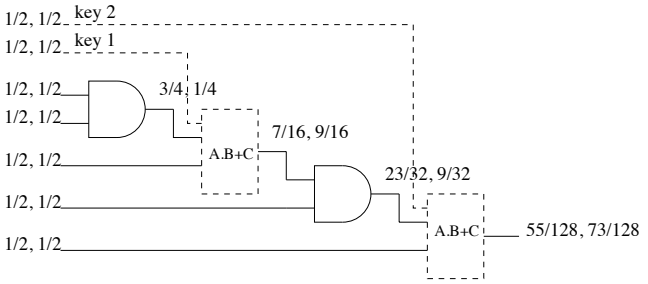
Nevertheless, merging two signals may change the signals probabilities. This can indeed lead to create divergences/reconvergences that would modify the probabilities of concerned signals. As presented in the example in Figure 4, the encryption of two rare-1 signals with distinct keys lead to a probability of 73/128 of being '1' for the output signal, whereas it is only 17/32 (=68/128) if the two key bits are merged.

One way to minimize the effect of this added procedure is to merge signals that do not belong to the same logic cone. This restriction prevents creating reconvergences. We have developed a heuristic that combines encryption signals together, after sorting them in descending order according to the number of logic cones that they belong to. This way, the most problematic signals are processed first. Encryption signals are then grouped according to two conditions: 1) they correspond to the same key value (note that this condition can be easily alleviated with the insertion of invertors, but we choose not to use invertors in order not to add further logic cells in the design); 2) they have no common logic cone.

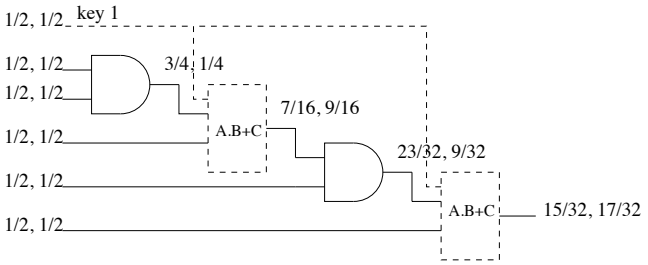
This may not be sufficient to reach a desired "small" number of key bits. In such a case, signals belonging to the fewest logical cones possible in common can be grouped. To do so, from the preceding sets, signals belonging to the smallest sets are reallocated into other existing sets, choosing each time the set for which there is the least logical cones in common. This is done until the number or remaining sets equals the user-chosen number of key bits.



a) Probabilities (0-probability, 1-probability)



b) Encryption with distinct key bits



c) Encryption with merged key bits

Figure 3. Example of key merging (encryption in dotted lines)

2) Uniqueness of the key

The presented encryption procedure produces an identical key for every IC. However, in order to thwart overproduction, the key must be unique to each IC. Solutions have already been proposed in [5, 6]. On of them consists in the insertion of a Physically Unclonable Function (PUF) into each IC, such as presented in Figure 4. In this example, the "encryption key" is the same for every IC (produced by the encryption algorithm), but the "user key" is IC dependent thanks to the PUF that, given a challenge, returns a single response per IC. The challenge and the user key, unique for each IC, are accessible to the user, and the encryption key (the same for every IC) is not accessible to the user.

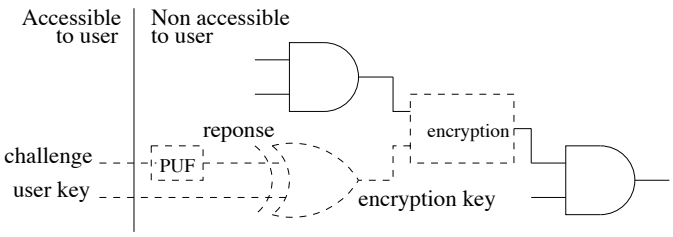


Figure 4. Use of PUF in [6]

IV. Experimental results

Firstly, we discuss the quality of our encryption with regards to an attack algorithm aiming at discovering the secret key [19]. Secondly, we present the experiments made in order to evaluate our algorithm in terms of controllability improvements. Besides, we compare these results with the ones obtained with the encryption algorithm presented in [18].

A. Protection against overproduction

In [19], a SAT-based algorithm is presented, that aims a determining the key values of an encrypted design. Several combinational encryption algorithms are tested including [5, 6, 18]. From the experiments made, the insertion of AND/OR and MUX gates is less secure than the insertion of XOR/XNOR gates. That is why we chose to use XOR/XNOR gates for the encryption of an upstream signal. However, even if the algorithm manages to decrypt most encrypted circuits, it is efficient only on combinational circuits.

Twelve encryptions were made: six with our algorithms, and 6 with the algorithm of [18]. The algorithm decryption managed to decrypt the key and the execution time was in the same order of magnitude for both algorithms. This is because very few (if any) XOR gates were used by our algorithm in the encryption of the benchmarks used.

B. Protection against HT

We evaluated our method on ISCAS benchmarks. Table I presents, for each benchmark:

- The parameters chosen: probability threshold (as well as the number of rare signals under that assumption) and maximum number of masking gates allowed,
- The results after encrypting/masking: in terms of area (number of gates added) and improvements in controllability (number of signals above, or still below the threshold).

For each experiment, the number of masking gates allowed was chosen below the number of low controllable signals, in order to test the limits of the methods. One can conclude from the data that our approach manages to remove all rare signals in many more cases than the approach in [18], with fewer gates.

V. Conclusion

The goal of our encryption technique is to thwart not only overproduction, but also the insertion of HTs, by minimizing the number of *rare values* in a circuit. That way, an attacker cannot exploit these rare values to insert a HT. The goal is to make the insertion more difficult, if not impossible.

From the logic masking of several benchmarks, results show that the proposed algorithm is more efficient than the first algorithm that introduced the idea, in terms of controllability improvements and area overhead.

REFERENCES

[1] U. Guin, K. Huang, D. DiMase, J. M. Carulli, M. Tehranipoor and Y. Makris. Counterfeit Integrated Circuits: A Rising Threat in the Global Semicondictor Supply Chain. In *Proceedings of the IEEE, Special Issue on Trustworthy Hardware*, 102(8):1207–1228, 2014.

[2] X. Wang, M. Tehranipoor and J. Plusquellic. Detecting Malicious Inclusions in Secure Hardware: Challenges and Solutions. *IEEE International Workshop on Hardware-Oriented Security and Trust (HOST'08*), pages 15–19, 2008.

[3] M. Tehranipoor and F. Koushanfar. A Survey of Hardware Trojan Taxonomy and Detection. *IEEE Design & Test of Computer,* 27:10–25, 2010.

[4] S. Bhunia, M. S. Hsiao, M. Banga, S. Narasimhan. Hardware Trojan Attacks: Threat Analysis and Countermeasures. In *Proceedings of the IEEE, Special Issue on Trustworthy Hardware*, 102(8):1229–1247, 2014.

[5] J. A. Roy, F. Koushanfar and I. L. Markov. EPIC: Ending Piracy of Integrated Circuits. In *Design, Automation and Test in Europe (DATE'08)*, pages 1069–1074, 2008.

[6] J. Rajendran, Y. Pino, O. Sinanoglu and R. Karri. Logic Encryption: A fault Analysis Perspective. In *Design, Automation and Test in Europe (DATE'12)*, pages 953–958, 2012.

[7] R. S. Chakraborty and S. Bhunia. Hardware Protection and Authentication Through Netlist Level Obfuscation. In *IEEE/ACM International Conference on Cmputer-Aided Design (ICCAD'08)*, pages 674–677, 2008.

[8] D.Agrawal, S.Baktir, D.Karakoyunlu, P.Rohatgi, and B.Sunar. Trojan Detection using IC Fingerprinting. In *IEEE Symposium on Security and Privacy (SP'07),* pages 296–310, 2007.

[9] Y. Jin and Y. Makris. Hardware Trojan Detection Using Path Delay Fingerprint. In *IEEE International Workshop on Hardware-Oriented Security and Trust (HOST'08),* pages 51–57, 2008.

[10] F. Wolf, C. Papachristou, S. Bhunia and R. S. Chakraborty. Towards Trojan-Free Trusted ICs: Problem Analysis and Detection Scheme. In *Design, Automation and Test in Europe (DATE'08)*, pages 1362–1365, 2008.

[11] R. S. Chakraborty, F. Wolff, S. Paul, C. Papachristou, and S. Bhunia. MERO: A Statistical Approach for Hardware Trojan Detection. In *International Conference on Cryptographic Hardware and Embedded Systems (CHES'09),* pages 396–410, 2009.

[12] S. Dupuis, P.-S. Ba, M.-L. Flottes, G. Di Natale and B. Rouzeyre. New Testing Procedure for Finding Insertion Sites of Stealthy Hardware Trojans. In *Design Automation & Test in Europe (DATE'15)*, pages 776–781, 2015.

[13] J. Rajendran, O. Sinanoglu, R. Karri. Regaining Trust in VLSI Design: Design-for-Trust Techniques. In *Proceedings of the IEEE, Special Issue on Trustworthy Hardware*, 102(8):1266–1282, 2014.

[14] R. S. Chakraborty, S. Paul, S. Bhunia. On-Demand Transparency for Improving Hardware Trojan Detectability. In IEEE *International Workshop on Hardware-Oriented Secutity and Trust (HOST'08)*, pp. 48–50, 2008.

[15] R. S. Chakraborty and S. Bhunia. Security Against Hardware Trojan Attacks Using Key-Based Design Obfuscation. In *Journal of Electronic Testing*, 27(6):767–785, 2011.

[16] H. Salmani, M. Tehranipoor, and J. Plusquellic. A Novel Technique for Improving Hardware Trojan Detection and Reducing Trojan Activation Time. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,* 20(1):112–125, 2012.

[17] H. Xue, T. Moody, S. Li, X. Zhang and S. Ren. Low Overhead Design for improving Hardware Trojan Detection Efficiency. In *Aerospace and Electronics Conference (NAECON'14)*, pages 379–383, 2014.

[18] S. Dupuis, P.-S. Ba, G. Di Natale, , M.-L. Flottes, and B. Rouzeyre. A Novel Hardware Logic Encryption Technique for thwarting Illegal Overproduction and Hardware Trojans. *IEEE International On-Line Testing Symposium (IOLTS'14*), 2014.

[19] P. Subramanyan, S. Ray and S. Malik. Evaluating the Security of Logic Encryption Algorithms. In *IEEE International Symposium on Hardware-Oriented Security and Trust (HOST'15)*, pages 137–143, 2015.

[20] G. Di Natale, S. Dupuis M.-L. Flottes, and B. Rouzeyre. Identification of Hardware Trojans triggering signals. In *Workshop on Trustworthy Manufacturing and Utilization of Secure Devices (TRUDEVICE'13)*, 2013.

TABLE I. LOGIC MASKING RESULTS REGARDING CONTROLLABILITY

Benchmark	Parameters		Initial	Encryption in [18]				Proposed logic masking			
	Threshold	Budget	Number of rare signals	Number of masking gates	Number of rare signals remaining	**% area overcost**	**% rare signals removed**	Number of masking gates	Number of rare signals remaining	**% area overcost**	**% rare signals removed**
apex2	0.01	32	21	21	0	**+3%**	**-100%**	8	0	**+1%**	**-100%**
	0.05	64	103	64	9	**+10%**	**-91%**	26	0	**+4%**	**-100%**
apex4	0.01	32	58	32	4	**+0.6%**	**-93%**	29	0	**+0.5%**	**-100%**
	0.05	512	683	512	77	**+10%**	**-89%**	229	0	**+4%**	**-100%**
c432	0.1	8	7	7	0	**+4%**	**-100%**	7	0	**+4%**	**-100%**
	0.2	32	53	32	33	**+20%**	**-38%**	31	0	**+19%**	**-100%**
c499	0.2	32	32	32	0	**+16%**	**-100%**	32	0	**+16%**	**-100%**
	0.3	32	58	32	50	**+16%**	**-16%**	26	0	**+13%**	**-100%**
c1355	0.1	32	64	32	32	**+6%**	**-50%**	32	32	**+6%**	**-50%**
	0.2	64	112	64	96	**+12%**	**-14%**	64	0	**+12%**	**-100%**
c1908	0.1	16	31	16	14	**+2%**	**-55%**	16	6	**+2%**	**-81%**
	0.2	64	110	64	65	**+7%**	**-41%**	64	9	**+7%**	**-92%**
c2670	0.001	16	28	16	8	**+1.4%**	**-71%**	11	0	**+1%**	**-100%**
	0.01	16	33	16	8	**+1.4%**	**-76%**	12	0	**+1%**	**-100%**