State of the Art on: IP Protection through Logic Locking at Register-Transfer Level

Luca Collini, luca.collini@mail.polimi.it

1. Introduction

The scaling of semiconductor technology with the associated rise of fabrication costs has limited the number of companies that can afford the billion-dollar manufacturing foundries. With this prohibitive cost, many companies are forced to outsource the integrated circuit (IC) fabrication to third-party foundries [7]. This trend is also known as the *globalization of the IC supply chain* [10]. However, the introduction of third parties in the IC supply chain introduces new security issues [1]. The two major concerns are IP theft and malicious modifications of the design [10]. Both issues require reverse engineering of the design under attack to extract and copy the functionality. As the market for domain-specific hardware components is becoming more and more lucrative, the high-value effort put into hardware design increases and makes it crucial to prevent IP theft [11]. In fact, the estimated loss due to IP violations alone was $4 billions in 2008 [9], while the total loss from IC counterfeiting was estimated to be about $169 billions in 2011 [5]. Hardware obfuscation aims at hiding and disabling the functionality of a chip to prevent a malicious foundry to thwart reverse engineering of the design. Many techniques have been proposed to perform hardware obfuscation and can be divided in two classes: *key-less obfuscation*, such as split manufacturing, and *key-based obfuscation*, such as logic locking. Split manufacturing is based on manufacturing parts of the circuit in different, potentially untrusted, foundries and then stitch them together in a trusted facility. Logic locking is based on adding some form of "programmability" to the design and ensuring that the circuit cannot function properly without a secret string of configuration data called “key” [10]. Researchers have recently put effort into unifying benchmarks and finding metrics in order to be able to compare different locking techniques.

Hardware obfuscation lays across hardware design and computer security therefore the main venues and papers of interest for hardware obfuscation are the ones that are focused on these two topics. Table 1 shows the most prestigious conferences and papers where these topics are usually discussed.

Conference name	**Acceptance rate % (lower is better)**
IEEE Security & Privacy (S&P)	12
ACM Conference on Computer and Communications Security (CCS)	16.6
IEEE/ACM International Conference on Computer-Aided Design (ICCAD)	23.9
IEEE/ACM Design Automation Conference (DAC)	29
IEEE Design, Automation and Test in Europe Conference (DATE)	32
Journal Name	**Impact Factor (higher is better)**
IEEE Transactions on Dependable and Secure Computing	8.948
IEEE Transactions on Information Forensics and Security	8.543
IEEE Transactions on Computer-Aided Design and Systems	2.98
IEEE Transactions on VLSI Design	2.933
ACM Journal on Emerging Technologies in Computing Systems	2.911
ACM Transactions on Design Automation of Electronic Systems	1.183

Table 1: List of most prestigious Conferences and Journals related to Hardware Obfuscation.

1.1. Preliminaries

ASIC flow
The ASIC flow is composed of a sequence of different steps. The first steps of the ASIC flow take place in the design house. Starting from a set of specifications, a high-level or a register-transfer level (RTL) description is elaborated by the digital designers. High-level synthesis can be used to automatically generate the RTL description from high-level languages. The description is given in input to a logic synthesis tool that produces a netlist. The netlist is mapped onto a physical layout via a process called place and routing. The resulting layout is sent for fabrication to an external foundry. After manufacturing, the chips are sent to a test facility and eventually they get back to the design house for distribution to the end users. In the ASIC supply chain the IP is exposed to different potential adversaries such as SoC integrators, foundries, test facilities and end-users. The major threat is reverse engineering that can lead to IP theft and overbuilding, or hardware Trojan insertion.

Threat models
The threat model for a security problem defines capabilities and intentions of the attacker. As in any other security field, it is crucial to define the threat model. Threat models for logic obfuscation primarily rely on the concept of *Oracle* and *Ambiguity*. An Oracle is a chip that performs the correct computation, whereas Ambiguity refers to the ability of the attacker to distinguish between key inputs and normal inputs.
In an *oracle-less* scenario we suppose that the attacker does not have access to a functional unit. This setting is plausible for all those application that are not mass produced for retail market or where the chip is produced for the first time.
In a *oracle-guided* scenario we suppose that the attacker has access to a functional chip that is treated as a black-box unit. The attacker can only query the chip with input patterns and observe their output values.
Distinct Ambiguity is used to describe a situation in which the attacker is able to distinguish between key inputs and normal inputs.
Ubiquitous Ambiguity is used to describe a situation in which the attacker is not able to distinguish key inputs and normal inputs [10].

Software obfuscation and opaque predicates
Software obfuscation is currently more mature than hardware obfuscation. The goal of software obfuscation is to transform an application in another that has the same behaviour as the original one but is way more difficult to understand by third parties. A key building block in software obfuscation is the *Opaque Predicate*. A predicate is said to be opaque if its outcome is known at compile time but is hard to deduce for a deobfuscator [3].

1.2. Research topic

When applying logic locking, one must accept a certain area overhead. Logic locking can be applied at different abstraction levels, leading to different results in terms of area overhead and robustness towards attacks. Some metrics have been proposed to compare different obfuscated designs but there is not a unified view on the matter yet. Having metrics to compare different designs is not only crucial to compare different techniques but most importantly to compare different applications of the same technique. In case the key is limited to a fixed amount of bits (to limit area overhead for example), it is important to understand where it is best to spend this limited amount of bits. The goal of our research is to develop a complete logic locking framework at register-transfer level that will allow us to say which parts of the design are better to obfuscate in order to obtain the most secure design given a constraint on the area or timing overheads. Our approach will operate at the register-transfer level, hoping to find a good trade-off between high-level synthesis and post-synthesis techniques. We will consider an oracle-less scenario with distinct ambiguity as our threat model. This is plausible scenario for low-volume markets where the attackers have strong methods to distinguish between functional and key signals.

2. Main related works

2.1. Classification of the main related works

The obfuscation techniques can be classified according to the level at which they are applied, by how they score after some proposed metrics, and the threat model they best fit into.

Logic obfuscation techniques can be applied at different steps of the design phase. In case of high level descriptions, a high level synthesis (HLS) tool may apply algorithmic-level obfuscation [7]. It has been shown that it is possible to apply obfuscation techniques at register-transfer level [6, 2], even though the two approaches are preliminary and require further investigation. Post-synthesis obfuscation is the most widespread category at the moment, post-synthesis techniques can be applied by modifying the netlist, for example adding extra logic ports.

Many metrics have been proposed to evaluate obfuscated designs. Some of them directly describe a physical property of the design, such as area or timing overhead, whereas others are values that have been shown to correlate with physical properties [1]:

- **Verification failure metric:** experimental metric that measures how many, and to which extent, outputs are affected by the obfuscation technique. To evaluate this metric an equivalence checking tool such as Synopsys Formality is needed to evaluate the functional difference between the original and the locked designs.
- **Entropy** (also known as Shannon Entropy): experimental metric that measures the amount of information in a data source. In the case of a combinational circuit it relates to the number of distinct outputs of the circuit. It tells us about two properties of the design:
  - *Power:* a design with high entropy must have many different possible outputs and therefore many transitions between logic-0 and logic-1, increasing the dynamic power of the circuit.
  - *Implications for obfuscation:* An obfuscated design with maximum entropy most resembles a random function.
- **Differential entropy:** experimental metric that is calculated with a miter circuit obtained by XORing each bit of the output of the locked circuit with the corresponding of the unlocked circuit. The entropy of the miter circuit is the evaluated to obtain the differential entropy. This metric represents the proportion of bits that differ between the obfuscated and the plain design. Experiments found a close relation between differential entropy and power overhead.
- **Reconvergence:** structural metric that represents the rate of internal signals converging in other nodes. Experiments showed that more resilient circuits to the key sensitization attacks have higher reconvergence.
- **Key structure metric:** normalized metric that indicates the structural interconnection between the key gates. A high value for this metric indicates a high resiliency to the key sensitization attacks.

2.2. Brief description of the main related works

It has been shown that obfuscation at higher levels of abstraction brings advantages as it allows designers to hide the semantic information of the chip design. On the contrary, post-synthesis techniques cannot protect information that is already embedded in the design by logic synthesis optimizations. On the other hand, obfuscation at the HLS level requires to adapt the design flow. For these reasons, an RTL approach is highly attractive as it is placed in between the existing techniques. A first approach at register-transfer level was proposed in [2] while a new promising preliminary approach has been presented in [6].

Table 2 summarizes the classification of the main logic techniques.

Table 2: Summary of the main logic locking techniques and scoring for the proposed metrics [1]

		Area overhead	Power overhead	Timing overhead	SAT attack resiliency	Key sens. attack resiliency	Verification failure metric	Entropy	Differential entropy	Reconvergence	Key structure metric
Pre-synthesis	TAO [7]	High	n.a.	Low	n.a.	n.a.	n.a.	n.a.	High [a]	n.a.	n.a.
	ASSURE [6]	Low	n.a.	Low	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.
	CDFG [2]	Low	Low	n.a	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.
	BDD Random [4]	High	High	Medium	Low	High	High	Low	Medium	Medium	High
	BDD AntiSAT [4, 12]	High	High	Medium	Medium	High	Low	Medium	Low	High	High
	BDD Entropy [1]	Medium	Medium	Low	Low	High	High	High	Medium	Low	High
Post-synthesis	RLL [8]	Low	Low	Low	Low	Low	High	Medium	High	Medium	Low
	SLL [13]	Low	Low	Low	Low	Medium	medium	Medium	High	Medium	Low
	Cone size [1]	Low	Low	Medium	Low	Medium	Low	Medium	High	Medium	Medium
	AntiSAT Random [8, 12]	Medium	Medium	Low	High	Low	High	Medium	High	Low	Medium
	AntiSAT SLL [13, 12]	Medium	Medium	Low	High	Medium	Medium	Medium	High	Low	Medium
	Anti SAT Cone size [1]	Medium	Medium	Medium	High	Medium	Low	Medium	High	Low	Medium

[a] Calculated as Hamming distance

For completeness, a brief overview of the main logic locking techniques (listed in Table 2) is provided:

- **TAO** [7]: is an extension for HLS tools to produce obfuscated RTL descriptions. TAO obfuscates the algorithm via obfuscation of constants, control branches and adding variants in the flows of data.
- **ASSURE** [6]: is a pre-synthesis tool that works at RT level. Assure applies obfuscation to branches, operations and constants.
- **CDFG** [2]: is a RT level technique that obfuscates the data flow graph of a design.
- **BDD** [4]: is a pre-synthesis technique that works on Binary Decision Diagrams. Key bits are added in the original BDD by randomly taking a node and adding two child nodes to it controlled by the key bit.
- **RLL** [8]: is the first proposed logic locking technique. It randomly inserts logic gates (tipically XOR or XNOR gates) controlled by a key bit.
- **SLL** [13]: Strong Logic Locking is a technique that strengthens the insertion of logic gates by inserting key-gates with complex interference among them.
- **Anti-SAT** [12]: is a technique that aims at making unfeasible SAT attacks by increasing the number of iterations to the exponential of number of primary inputs used to implement the AntiSAT block.
- **Cone size** [1]: is a heuristic technique that integrates the key gates with other gates that have the largest fanin or fanout cone or both.

2.3. Discussion

The proposed metrics are empiric and experimental and do not allow their use with optimization methods. An open challenge is to find security properties to compare two designs and tell which one is more secure. In contrast with most other areas of computer security, hardware obfuscation is missing security properties clearly defined by mathematical terms [10]. This would allow the use of optimization techniques to find the best obfuscated design with a fixed maximum area overhead.

ASSURE [6] and CDFG [2] show that logic locking at register-transfer level deserves further investigations. A register-transfer level approach would be similar to code obfuscation as we could start from the concept of opaque predicates to build predicates whose outcome depends on the input key. It may also be possible to inherit other useful knowledge currently used in the protection of software applications.

References

[1] Amir, S., Shakya, B., Xu, X., Jin, Y., Bhunia, S., Tehranipoor, M. M., and Forte, D. Development and evaluation of hardware obfuscation benchmarks. *Journal of Hardware and Systems Security 2* (2018), 142–161.

[2] Chakraborty, R., and Bhunia, S. Rtl hardware ip protection using key-based control and data flow obfuscation. pp. 405–410.

[3] Collberg, C., Thomborson, C., and Low, D. A taxonomy of obfuscating transformations. *http://www.cs.auckland.ac.nz/staff-cgi-bin/mjd/csTRcgi.pl?serial* (01 1997).

[4] Massad, M. E., Zhang, J., Garg, S., and Tripunitara, M. V. Logic locking for secure outsourced chip fabrication: A new attack and provably secure defense mechanism, 2017.

[5] Omdia. Top 5 most counterfeited parts represent a $169 billion potential challenge for global semiconductor market. Available at: https://www.electronicproducts.com/top-5-most-counterfeited-parts-represent-a-169-billion-potential-challenge-for-global-semiconductor-market/ (Last accessed: November 1, 2020), 2012.

[6] Pilato, C., Chowdhury, A. B., Sciuto, D., Garg, S., and Karri, R. Assure: Rtl locking against an untrusted foundry, 2020.

[7] Pilato, C., Regazzoni, F., Karri, R., and Garg, S. TAO: Techniques for algorithm-level obfuscation during high-level synthesis. In *Proceedings of the 55th Annual Design Automation Conference* (2018), DAC '18.

[8] Roy, J. A., Koushanfar, F., and Markov, I. L. Epic: Ending piracy of integrated circuits. In *2008 Design, Automation and Test in Europe* (2008), pp. 1069–1074.

[9] Semi. Innovation is at risk as semiconductor equipment and materials industry loses up to $4 billion annually due to ip infringement. Available at: http://dev7.semi.org/en/white-paper-ip-infringement-causes-4-billion-loss-industry-annually (Last accessed: November 1, 2020), 2008.

[10] Shamsi, K., Li, M., Plaks, K., Fazzari, S., Pan, D. Z., and Jin, Y. Ip protection and supply chain security through logic obfuscation: A systematic overview. *ACM Trans. Des. Autom. Electron. Syst. 24*, 6 (Sept. 2019).

[11] Tan, B., Karri, R., Limaye, N., Sengupta, A., Sinanoglu, O., Rahman, M. M., Bhunia, S., Duvalsaint, D., D., R., Blanton, Rezaei, A., Shen, Y., Zhou, H., Li, L., Orailoglu, A., Han, Z., Benedetti, A., Brignone, L., Yasin, M., Rajendran, J., Zuzak, M., Srivastava, A., Guin, U., Karfa, C., Basu, K., Menon, V. V., French, M., Song, P., Stellari, F., Nam, G.-J., Gadfort, P., Althoff, A., Tostenrude, J., Fazzari, S., Breckenfeld, E., and Plaks, K. Benchmarking at the frontier of hardware security: Lessons from logic locking, 2020.

[12] Xie, Y., and Srivastava, A. Anti-sat: Mitigating sat attack on logic locking. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 38*, 2 (2019), 199–207.

[13] Yasin, M., Rajendran, J. J., Sinanoglu, O., and Karri, R. On improving the security of logic locking. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 35*, 9 (2016), 1411–1424.