IEEE Copyright Notice

arXiv:2108.11335v1 [cs.CR] 19 Aug 2021

A Tutorial on Design Obfuscation: from Transistors to Systems

Samuel Pagliarini

Tallinn University of Technology (TalTech)

Department of Computer Systems

Centre for Hardware Security

E-mail: samuel.pagliarini@taltech.ee

Abstract—**The recent advances in the area of design obfuscation are encouraging, but may present themselves as hard to read for a non-specialist audience. This tutorial uncovers these advances in a clear language, contrasting the approaches that can be implemented at layout level, in the netlist of a circuit, or even at chip level. This tutorial also highlights the available support, both from the tooling side and the logistics of fabricating an obfuscated integrated circuit.**

Index Terms—**design obfuscation, logic locking, dummy vias, split fabrication, split manufacturing.**

I. INTRODUCTION

Historically, we have seen a constant offshoring trend for integrated circuit (IC) foundries that is supported by the fabless business model that most design companies subscribe to. Eventually, this trend that has been going on for decades, started to raise alarms that governments would lose access to state-of-the-art silicon [1] and instead rely on offshore foundries. That being said, foundries are typically considered a potential adversary in this scenario since the design house that creates the IC is not a stakeholder of the foundry. Compounded by geographical and political concerns, these foundries are then termed as *untrusted foundries*.

When it comes to attacks that can be mounted by untrusted foundries, the ICs and their many intellectual property (IP) modules are vulnerable to a diverse selection of threats. We direct the reader to [2] for a thorough discussion on the supply chain and related threats, while here we provide a short description of commonly studied threats:

- **Hardware trojans**. Malicious modification or insertion of logic into an IC.
- **IP piracy**. Unauthorized copy of an IP, by the foundry or any other user of the IP.
- **IC overbuilding** Production of parts beyond the contracted amount with the goal of reselling.
- **Reverse engineering** Process of deconstruction, either physical or logical, of an IP or IC with the intent of extracting knowledge from it.

Obfuscation has emerged as an interesting solution for directly combating IP piracy and IC overbuilding, while also

presenting some implications for the other threats. Let us now discuss some of the more relevant obfuscation techniques in the specialized literature.

II. CURRENT PRACTICES IN OBFUSCATION

A. Layout Obfuscation

In **camouflaged logic**, the central concept is to make one logic gate look like another, such that an attacker cannot discern the functionality of a particular gate based solely on its observable physical characteristics. One proposed camouflaging technique [3] uses a mix of dummy and real vias/contacts to obscure the gate's function. Dummy vias/contacts are not fully formed and do not make an electrical connection between layers, but they possibly may be mistaken for real vias/contacts during the delayering process of reverse engineering a fabricated circuit. The proposed design in [3] uses a camogate that can form a NAND, NOR, or XOR gate with identical layouts (except for the placement of real and dummy vias). The security of this technique relies on the limited ability of the attacker to distinguish between real and dummy vias/contacts.

However, the delayering and imaging capabilities of state-of-the-art reverse engineering labs are very advanced [4] and may not be fooled by such dummy vias/contacts. Further, dummy via/contact fabrication is caught in a dual-sided constraint between reliability and security as illustrated in Fig. 1. If the dummy has a large separation between layers, then reliability is not compromised, but it becomes more easily distinguished as a dummy. Conversely, a small separation is harder to distinguish, but is more likely to create a short. Additionally, the compatibility of building dummy 'broken' vias in a modern IC is questionable, as the fabrication process forms the vias in the same manufacturing step as the interconnect wires themselves. It is unclear how such a partially connected via/contact could be formed without additional special process steps and masks, increasing the cost and complexity of implementing this countermeasure. The contrast between conventional interconnect deposition and present deposition is shown in Fig. 1.

Another class of layout obfuscation techniques is worth mentioning: layout manipulation during place & route to avoid 'white spaces' [5]. But this is partially misleading: empty standard cell rows/slots eventually get filled by filler cells, while unused routing tracks may get filled with filler wires
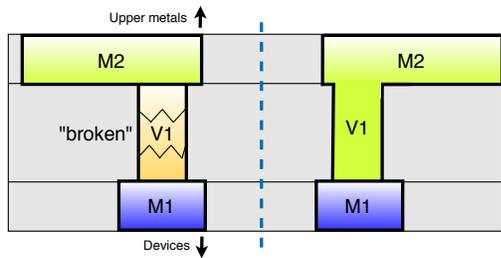
Fig. 1. Via and metal deposition processes. On the left, conventional deposition where vias and metals are formed one by one. On the right, the dual damascene process is highlighted where metals and vias are deposited at the same time.
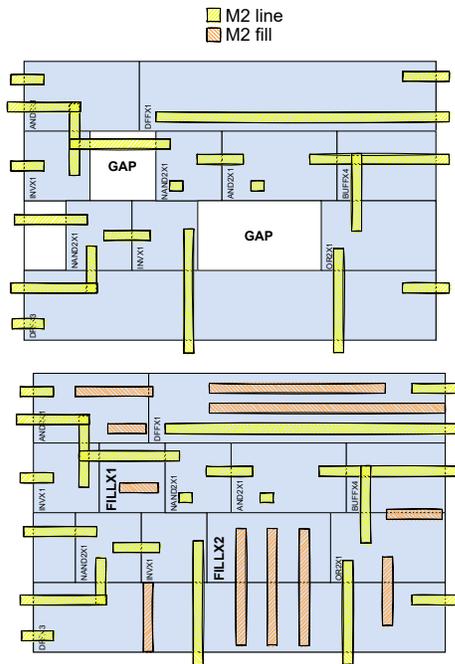

Fig. 2. Layout of a placed standard cell-based circuit, before and after fill procedures. The fill lines are overly simplified.

for DFM reasons. This effect is shown in Fig. 2 where we highlight only M2 for clarity. Notice how FILLX cells are placed in the gaps between standard cells and how specialized M2 fill lines are positioned in areas of low density.

The design rules for today's advanced nodes forcibly imply critical layers are gridded and regular, including poly, active, and fin layers. Effectively, circuits being fabricated today already resemble a sea of gates at their lowest layers, leaving even less margin for layout manipulation for obfuscation. Overall, layout modifications are *very limited* in combating threats coming from an untrusted foundry. Tooling support is largely non-existent and *ad hoc* solutions are employed.

B. Key-based Locking

Designs obfuscated with key-based locking require a key to operate correctly. The security strength of the technique comes from the fact that the key is not shared with an untrusted foundry. **Logic locking** is a class of techniques that has evolved over the last decade or so and gained
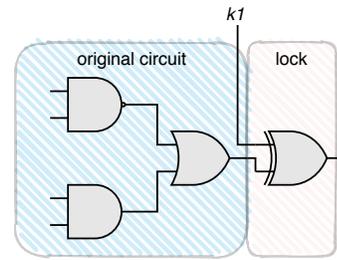

Fig. 3. Example of a locked circuit: the lock is the XOR gate controlled by key *k1*. The circuit behaves as expected when *k1=0*.
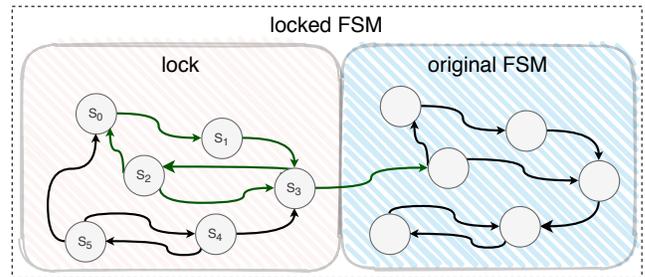

Fig. 4. FSM locking scheme where the transition S_0-S_1-S_3-S_2-S_3 has to be exercised to reach the original FSM.

significant traction. Some authors use other terms to refer to this class of techniques, such as logic encryption, logic obfuscation, or keyed logic. Conventionally, logic locking [6] has been considered for preventing reverse engineering, excess production, and, to a lesser extent, to prevent malicious circuit modifications. The original concept is relatively simple: additional logic is embedded into a circuit to provide it with a key-based operation. With the correct key, the circuit operates as intended. Without the correct key, some selected signals are corrupted. The key-based operation is usually implemented with XOR/XNOR gates, but MUXes and other logic cells can also be used for the same effect. A minimal example of an XOR-based lock is given in Fig. 3. For a large circuit, a number of XORs and keys would then be used, given that the difficulty to defeat logic locking increases with the number of locks.

FSM locking is the sequential counterpart of logic locking. Instead of additional combinational gates, new states and transitions are added to an existing FSM in order to create a locked FSM [7]. When the right key is applied, the locked FSM transitions from state to state until the original FSM becomes accessible and from there onwards, the circuit behaves as the original one would. This scheme is shown in Fig. 4. The cost of implementing FSM locking is relatively small: a few flops for new states and some combinational cells for new transitions, which can have close to no impact on the power, area, or performance of a system.

Overall, the effectiveness of key-based approaches is being constantly challenged [8]. However, tooling support is vast [9].

C. Macro approaches

In this obfuscation class, approaches make use of (some degree of) trusted fabrication, i.e., these macro approaches
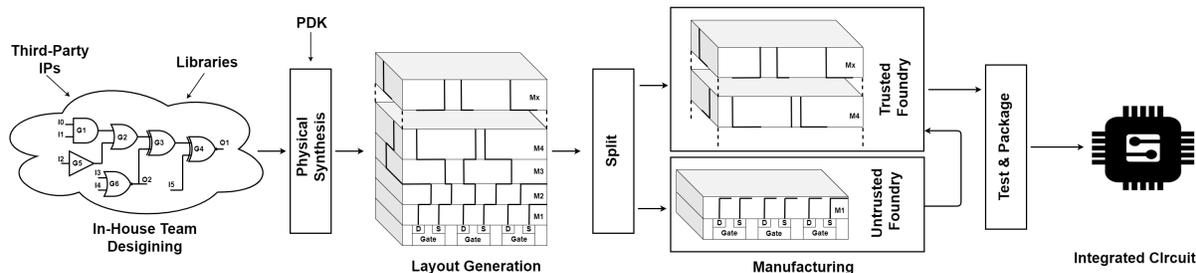
Fig. 5. Split Manufacturing Design Flow [10].

rely on foundry collaboration/trust that cannot be achieved by design means only. **Split fabrication** [10] is the most well-known example in this category. As illustrated in Fig. 5, a split-manufactured design is cut horizontally and different foundries are in charge of fabricating the top and bottom parts. Here, the typical assumption is that an untrusted foundry will mostly handle the fabrication of device layers while a trusted foundry will handle the interconnect. By doing so, the untrusted foundry does not fully understand how the transistors are connected, which by itself mitigates the piracy and overbuilding threats. The scheme is also reportedly efficient against hardware trojans.

Split fabrication suffers from one enormous drawback: the typical foundry business model does not contemplate the delivery of 'unfinished' wafers. On the contrary, foundries deliver wafers (or cut dice) that contain the entire stack, from devices to the uppermost layer metal (e.g., M10). In order to bypass this limitation and still benefit from the state-of-the-art transistors that untrusted foundries offer, a **Split-Chip** technique is defined in [11]. The concept is to create a vertical cut in the design, where modules are partitioned and assigned to either a trusted or untrusted side. If properly split, the two designs can be bound together with minimal performance and power penalties. Generally speaking, modules that are more control-oriented are good candidates for the trusted side of the system while computation-intensive modules are good candidates for the untrusted side.

Unfortunately, despite the obvious advantages, macro solutions require foundry cooperation and no clear framework exists for this activity today. Tools for implementing the aforementioned macro solutions are typically implemented as scripts within commercial CAD tools.

III. DISCUSSION AND CONCLUSIONS

Other active lines of research in obfuscation, although not covered in this tutorial, are the use of CMOS augmented by other materials (e.g., non-volatile storage of the circuit state with magnetic elements [12]) or paradigms (e.g., selective use of eFPGA [13]). However, these can still be considered emerging approaches.

While the literature offers a wide range of solutions for obfuscation, it is not possible to indicate a clear one-size-fits-all technique. From an academic perspective, logic locking is by far the most popular approach and an incredibly active research topic. While hardware security remains an afterthought in many instances, logic locking has the potential to break that barrier: It is conceivable that logic locking will be considered for protecting mass produced devices (i.e., consumer electronics) in the near future. For this reason, practitioners should familiarize themselves with the technique and its variants.

REFERENCES

[1] J. I. Lieberman, "White paper: National security aspects of the global migration of the us semiconductor industry," *Washington: Office of Senator Joseph I. Lieberman, Ranking Member, United States Senate Armed Services Committee*, 2003.

[2] M. Rostami, F. Koushanfar, and R. Karri, "A primer on hardware security: Models, methods, and metrics," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1283–1295, 2014.

[3] J. Rajendran, M. Sam, O. Sinanoglu, and R. Karri, "Security analysis of integrated circuit camouflaging," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, ser. CCS '13. Association for Computing Machinery, 2013, p. 709–720. [Online]. Available: https://doi.org/10.1145/2508859.2516656

[4] Chipworks, "Intel's 22-nm tri-gate transistors exposed," 2012, http://www.chipworks.com/blog/technologyblog/2012/04/23/intels-22-nm-trigate-transistors-exposed/.

[5] H. Hossein-Talaee and A. Jahanian, "Layout vulnerability reduction against trojan insertion using security-aware white space distribution," in *2017 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2017, pp. 551–555.

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

[7] R. S. Chakraborty and S. Bhunia, "Harpoon: An obfuscation-based soc design methodology for hardware protection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 10, pp. 1493–1502, 2009.

[8] P. Subramanyan, S. Ray, and S. Malik, "Evaluating the Security of Logic Encryption Algorithms," in *HOST*, 2015, pp. 137–143.

[9] CAD For Assurance. https://cadforassurance.org/.

[10] T. D. Perez and S. Pagliarini, "A survey on split manufacturing: Attacks, defenses, and challenges," *IEEE Access*, vol. 8, pp. 184 013–184 035, 2020.

[11] S. Pagliarini, J. Sweeney, K. Mai, S. Blanton, S. Mitra, and L. Pileggi, "Split-chip design to prevent ip reverse engineering," *IEEE Design & Test*, pp. 1–1, 2020.

[12] J. Yang, X. Wang, Q. Zhou, Z. Wang, H. Li, Y. Chen, and W. Zhao, "Exploiting spin-orbit torque devices as reconfigurable logic for circuit obfuscation," *Trans. Comp.-Aided Des. Integ. Cir. Sys.*, vol. 38, no. 1, p. 57–69, Jan. 2019. [Online]. Available: https://doi.org/10.1109/TCAD.2018.2802870

[13] J. Chen, M. Zaman, Y. Makris, R. D. S. Blanton, S. Mitra, and B. C. Schafer, "DECOY: DEflection-Driven HLS-Based Computation Partitioning for Obfuscating Intellectual PropertY," in *Proceedings of the 57th ACM/EDAC/IEEE Design Automation Conference*, 2020.