2019 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)

Hybrid Memristor-CMOS Obfuscation Against Untrusted Foundries

Amin Rezaei
Northwestern University
me@aminrezaei.com

Jie Gu
Northwestern University
jgu@northwestern.edu

Hai Zhou
Northwestern University
haizhou@northwestern.edu

***Abstract*—The high cost of IC design has made chip protection one of the first priorities of the semiconductor industry. In addition, with the growing number of untrusted foundries, the possibility of inside foundry attack is escalating. However, by taking advantage of polymorphic gates, the layouts of the circuits with different functionalities look exactly identical, making it impossible even for an inside foundry attacker to distinguish the defined functionality of an IC by looking at its layout. Moreover, since memristor is compatible with CMOS structure, it is possible to efficiently design hybrid memristor-CMOS circuits. In this paper, we propose a hardware obfuscation method based on polymorphic hybrid memristor-CMOS technology. Overhead of the polymorphic designs and the time complexity of possible attacks are discussed.**

***Keywords*-Hardware Obfuscation; Polymorphic Gates; Memristor; Inside Foundry Attack**

I. Introduction

With increasing the design costs of Integrated Circuits (ICs), chip protection has become one of the main concerns for the semiconductor industry. By using Reverse Engineering (RE) techniques, gate-level netlist can be extracted and duplicated without the authorization of the chip holder [1]. Moreover, many semiconductor companies contract out manufacturing of their designs to third party foundries. With the growing number of untrusted foundries, the possibility of Inside Foundry Attack (IFA) is also escalating [2]. To protect IC design from piracy, IC camouflaging [3], [4], [5] is proposed in which cells with different functions maintain an identical look in RE-based attacks. However, since state-of-the-art camouflaging schemes utilize dummy contact [6] and stealthy dopant [7] solutions, they are still vulnerable to IFA. Because in order to manufacture the chip, the third party foundry should have access to the connectivity information (i.e., dummy and true contacts as well as always-on/off and regular CMOS transistors.)

On the other hand, polymorphic designs [8], [9], [10] are multi-functional designs in which change of their behavior comes from modifications in the characteristics of their components after fabrication. In addition, memristor [11], [12], [13] is a nanoscale chip element where its resistance can be changed by the direction of the electric current. One of the interesting usage of memristor is to build logic gates. In some of the previous works like IMPLY [14] and MAGIC [15], only memristors are used for logic implementation. On the other hand, in some other works like MRL [16] and MeMOS [17] hybrid memristor-CMOS designs are introduced.

In security perspective, we can deem memristor as a switch that is *off* when it has the maximum resistance and *on* vice versa. By taking advantage of the memristors, it is possible to create polymorphic designs whose functionally can be controlled by changing memristor's resistance. In this paper, we propose our obfuscation scheme based on polymorphic hybrid memristor-CMOS [16], [18], [19] designs since they are scalable and fully compatible with existing CMOS technologies.

Although there are quite a few works in acyclic combinational obfuscation, cyclic combinational and sequential obfuscations have not been considered carefully. The proposed cyclic obfuscation scheme in [20] did not address the fact that SAT-based attack on cyclic combinational obfuscation can be as easy as the attack on acyclic combinational obfuscation if there exists only a correct key under which the circuit is acyclic [21]. Also, most of the solutions in sequential obfuscation [22], [23] depend on designing a strong Physical Unclonable Function (PUF) unit that is both costly and controversial. Thus, the contributions of this paper are twofold:

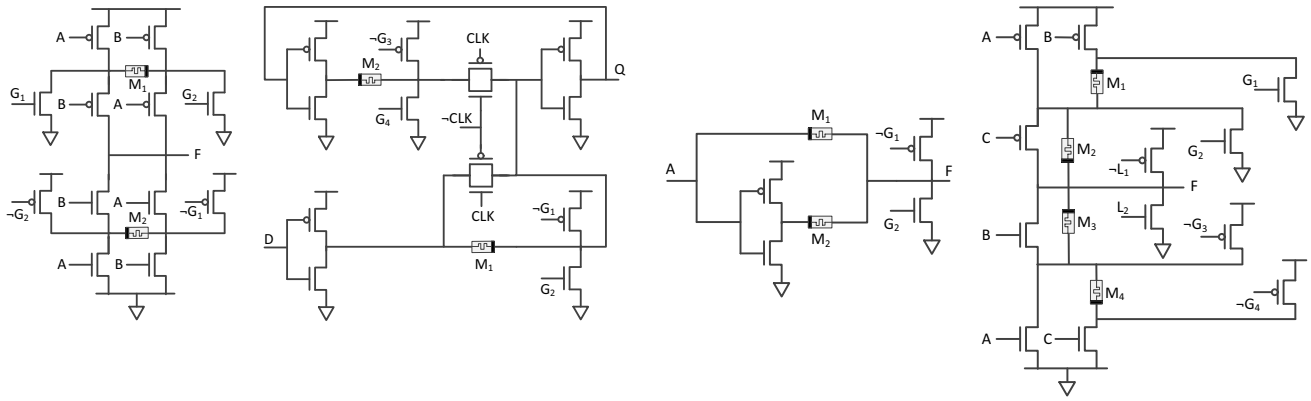
- Overcoming the problem of IFA by proposing polymorphic hybrid memristor-CMOS designs instead of using dummy contact and doping-based solutions.
- Defeating the SAT-based attack in particular and query-based learning attack in general by suggesting a hierarchical hardware obfuscation scheme that does not depend on PUF design.

The rest of the paper is arranged as follows. Several polymorphic hybrid memristor-CMOS designs are proposed in Section II. Section III introduces a hierarchical hardware obfuscation scheme based on the proposed designs and discusses the time complexity of the possible attacks. The experimental results are shown in Section IV. Finally, Section V concludes the paper.

II. Polymorphic Memristor-CMOS Designs

Two polymorphic memristor-CMOS gates called NOR-NAND (NONA) and Buffer-INV (BINV) are introduced in [24]. We review those designs and propose three additional polymorphic designs in the following subsections.

978-1-7281-3391-1/19/$31.00 ©2019 IEEE
DOI 10.1109/ISVLSI.2019.00102



(a) (b) (c) (d)

Figure 1: Hybrid memristor-CMOS designs (a) NONA [24] (b) BLat (c) BINV [24] (d) QFun

Table I: Configuration phase (a) NONA [24] (b) BLat (c) BINV [24] (d) QFun

(a)

	G_1	G_2	A	B
NAND	1	0	1	0
NOR	0	1	0	1

(b)

		G_1	G_2	G_3	G_4	D	CLK
Latch	Step1	0	1	0	0	0	0
	Step2	0	0	0	1	0	1
Buffer	Step1	0	1	0	0	0	0
	Step2	0	0	1	0	1	1
	Step3	1	0	0	0	1	0

(c)

		G_1	G_2	A
Buffer	Step1	0	1	1
	Step2	0	1	0
INV	Step1	1	0	0
	Step2	1	0	1

(d)

	M_1	M_2	M_3	M_4	Moves
NAND	On	On	Off	Off	2, 4
NOR	Off	Off	On	On	1, 5
INV	Off	On	On	Off	3, 6
E-INV	Off	Off	Off	Off	1, 2

All of the designs can be configured in parallel if they have independent logic inputs. After the configuration phase, the control signals are set to "0". To charge a memristor, its resistance is increased to the maximum resistance. In this case, the memrisor is considered as *off*. On the other hand, to discharge a memristor, its resistance is decreased to the minimum resistance. In this case, the memristor is considered as *on*.

A. NONA Design

The NONA polymorphic gate [24] is shown in Fig. 1(a). Based on different configurations of the memristors, the functionality of NONA is changed. If M_1 is on while M_2 is off, NONA functions as NAND gate. On the contrary, if M_1 is off while M_2 is on, NONA operates as NOR gate. The configuration procedure for NONA is shown in Table I(a). In configuration phase, the combination of the global control signals (i.e., G_1 and G_2) and the logic inputs (i.e., A and B) is used.

B. BLat Design

The proposed Buffer-Latch (BLat) polymorphic design is depicted in Fig. 1(b). If M_1 is on while M_2 is off, BLat functions as D-type latch. On the contrary, if M_1 is off while M_2 is on, BLat functions as a buffer. As shown in Table I(b) a two-step (for latch) or three-step (for buffer) configuration phase is required based on the targeted function.

C. BINV Design

The Buffer-INV (BINV) gate [24] is depicted in Fig. 1(c). If M_1 is on while M_2 is off, BINV works as a buffer. Contrariwise, BINV works as an INV. Table I(c) shows a two-step configuration procedure for BINV.

D. QFun Design

A polymorphic Quadruple Function (QFun) gate corresponding to basic NAND, NOR, INV, and E-INV gates is proposed in Fig. 1(d). The full functionality of QFun is shown in Table I(d). For example, if M_1 and M_2 are on while M_3 and M_4 are off, QFun acts like NAND gate. The combination of the global control signals (i.e., G_1 to G_4), the local control signals (i.e., L_1 and L_2), and the logic inputs (i.e., A, B, and C) is used to charge/discharge the targeted memristors. As shown in Table I(d), two of the six moves described below are required for shaping the QFun functionality.

Move 1: For charging M_1 and M_2, logic input A is set to "0". In addition, control signals G_1 and L_2 and logic input B are set to "1".
Move 2: For charging M_3 and M_4, logic input C and control signals L_2 and G_3 are set to "1". Also, logic input A is set to "0".
Move 3: For charging M_1 and M_4, logic input A is set to "0". In addition, logic inputs C and B and control signals G_1 and G_3 are set to "1".

Move 4: For discharging M_1 and M_2, logic input B is set to "0". Moreover, control signals L_1 and G_2 and logic input A are set to "1".
Move 5: For discharging M_3 and M_4, logic input A and control signals L_1 and G_4 are set to "1". Also, logic input C is set to "0".
Move 6: For discharging M_2 and M_3, logic input A and control signals L_1 and G_2 are set to "1". Also, logic input B is set to "0" while logic input C is set to "1".

In cascaded gates the charging/discharging procedure should be done sequentially. In this case, local control signals L_1 and L_2 can be used to set the desired output for QFun that drives another gate. This can be done regardless of the defined functionality of the already programmed QFun gates.

E. QLat Design

If all the memristors are off in Fig. 1(d), QFun operates as E-INV gate. In this case, if we connect CLK and $\neg CLK$ to logic inputs B and C respectively, the gate will work as an INV with level sensitive clock. Fig. 2 depicts the proposed D-type latch design (i.e., QLat) corresponding to QFun gates. The input E-INV activates with CLK while the latch feedback loop E-INV activates with $\neg CLK$. Input D is accepted when CLK is high. However, when CLK goes low, the input is open-circuited and the latch is set with the prior data D.

III. Hierarchical Hardware Obfuscation

We propose a hierarchical hardware obfuscation scheme in the following subsections based on the proposed polymorphic designs in Section II. We also analyze the time complexity of possible attacks.

A. Combinational Obfuscation

In order to obfuscate the original combinational circuit, first the acyclic combinational circuit will be converted to a cyclic one [24]. As shown in Fig. 3(a), it is enough to identify two AND and OR (or NAND and NOR) gates sharing a common input in a feed-forward path. Then, a passing gate is added to the output of the rearward gate and a feedback is introduced from output of the forefront gate to the input of the passing gate. Passing gates for logic "1" and "0" are OR and AND respectively. Based on the value of the common input, either the forefront gate or the passing gate becomes nullified; thus, the feedback is broken.

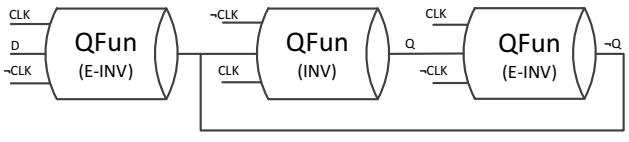
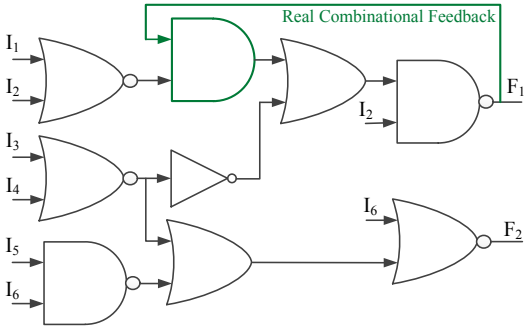
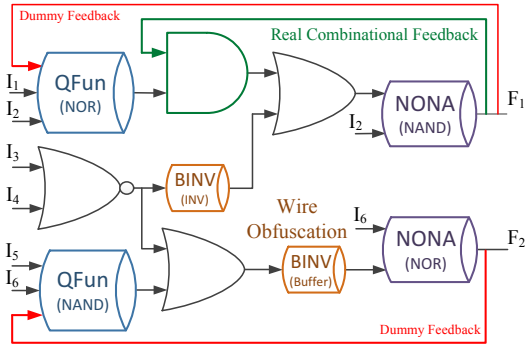


Figure 2: Hybrid memristor-CMOS QLat



(a)



(b)

Figure 3: (a) Cyclic combinational circuit (b) Obfuscated circuit

Next, some of the gates are chosen to be exchanged with the proposed polymorphic gates (i.e., NONA, BINV, and QFun) as shown in Fig. 3(b). NONA can be used to obfuscate NAND and NOR gates; BINV is suitable for obfuscating INV gates as well as the wires; QFun can be employed not only to obfuscate NAND, NOR, and INV gates, but also to produce dummy feedbacks [20]. Since QFun has three logic inputs, some dummy inputs can be available based on the usage of the gate. For example, if QFun is configured as NAND gate, the input C does not influence the output of the gate. Thus, with utilizing dummy inputs of QFun, dummy feedbacks can be introduced.

As mentioned in [24], both the original and the cyclic SAT-based attacks [25], [21] cannot efficiently deobfuscate the combinational circuits with real and dummy feedbacks.

B. Scan Chain Obfuscation

Almost all ICs are sequential circuits in which scan chain is widely used for testing purposes. In scan chain, the sequential circuit works in two different modes named as regular and testing. A 2-1 MUX is placed at the input of each Flip-Flop (FF) in order to connect all FFs in a shift register for one MUX selection while the FFs work in the regular mode for the other MUX selection. In the testing mode, the circuit acts like a combinational one.

For scan chain obfuscation, first two polymorphic FF-Buffers (FFBs) are randomly chosen for start and end

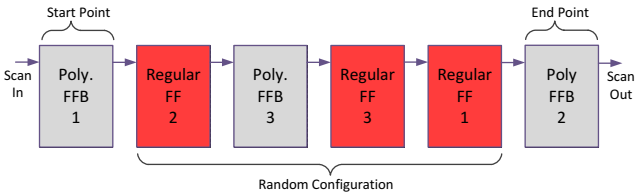


Figure 4: Scan chain obfuscation

points, and then the remaining regular FFs and polymorphic FFBs are randomly allocated in the chain. Fig. 4 shows an example. In order to test the IC, first the polymorphic FFBs are programmed based on their defined function, and then the scan chain can be used for the predefined test patterns.

Suppose there are n polymorphic FFBs in the scan chain, that only m of them are real FFs. In order to misuse the scan chain, although the attacker can figure out the number of real FFs, he cannot find out the position of them by looking at the layout. Thus, he requires to check $C(m, n)$ cases. By choosing an efficient distribution between m and n (e.g., $m = 0.5n$) the required number of tests will be a^n $(a > 1)$ that is practically impossible to achieve for large circuits.

C. Sequential Obfuscation

In order to obfuscate the original sequential circuit, a block with memory elements is replaced with the proposed polymorphic latch designs (i.e., BLat and QLat.) Note that cascading two polymorphic latch designs with reverse CLK signals, results in a buffer or a master-slave D-type FF. The complexity of the chosen block can be increased with wire obfuscation (i.e., adding BLat and QLat designs in the way of wires and programming them as buffers.)

Without the availability of scan chain, the SAT-based attacks [25], [21] are not suitable for attacking sequential circuit. In this case, a general learning algorithm is required to attack the whole circuit. By simulating the circuit as a Deterministic Finite Automata (DFA), the attacker may be able to learn the functionality of the circuit. It is supposed that the attacker has access to the physical layout. Thus, he can distinguish between polymorphic and regular designs. Moreover, he can acquire a functional circuit as a black-box and get the correct outputs for given input vectors. Also, the attacker knows the list of functions that each polymorphic design can implement. However, since the obfuscated block consists of exactly similar layouts for different polymorphic designs, he cannot figure out the actual functionality of each polymorphic design by looking at the layout even if he is an inside foundry attacker.

The attacker can learn an initially unknown DFA U from two types of the oracles based on the membership and equivalence queries. The input of the membership query is a regular expression t and the output is "Yes" if t is a member of U. Otherwise, the output is "No". The membership oracle is the functional circuit that the attacker has acquired from the market. The input of the equivalence query is a DFA E and the output is "Yes" if E is equivalent to U. Otherwise, the answer is "No". Also, the oracle provides the attacker with a counterexample t that is a regular expression with different acceptance status in E and U. The equivalence oracle can be simulated by random sampling. Based on the proof in [26], if s is the number of states in U and the equivalence oracle presents counterexamples of minimal length, running time of the learning algorithm is bounded by a polynomial in s for the best-case scenario. On the other hand, based on the state explosion problem [27], the size of the state space of a system grows exponentially in the number of its processes and variables. Thus, the above learning algorithm that is polynomial in s is exponentially large in the size of the circuit. This makes the exact query-based attack empirically impossible on sequential obfuscation.

IV. Experimental Results

In this section, first we show the layouts of the proposed polymorphic designs and compare the overhead of the polymorphic gates with the regular ones. Then, we study the possibility of brute force and query-based attacks on scan chain and sequential obfuscations.

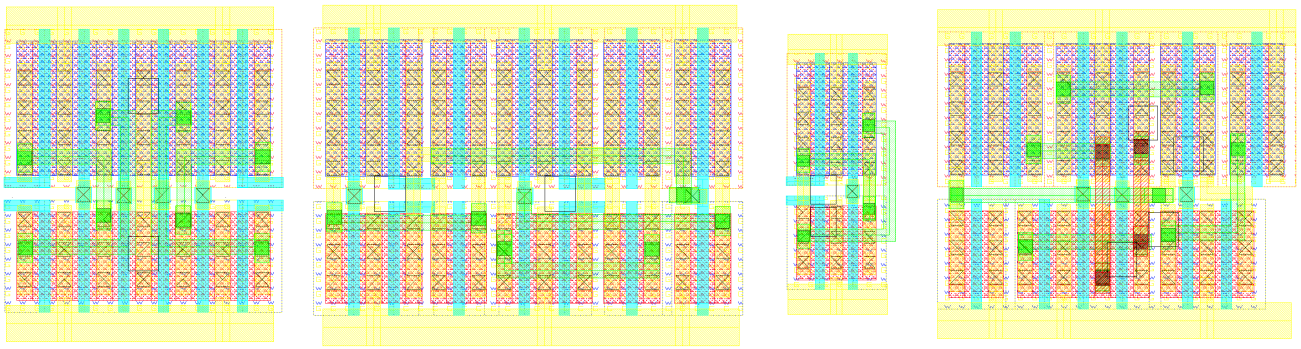
A. Polymorphic Designs Evaluation

Fig. 5 depicts the layouts of the polymorphic hybrid memrisor-CMOS designs. in 45nm CMOS technology. Also, triplicating the layout of QFun provides the layout of QLat. If the obfuscation scheme is reliable, the layout of any IC that is partially implemented using the proposed polymorphic designs can be even publicly available.

In addition, Table II shows the overhead of the polymorphic gates (i.e., NONA, BINV, QFun) compared with the regular gates in 45nm CMOS technology. The experiments are done in Cadence Virtuoso platform. On average, propagation delay of QFun, NONA, and BINV are 25%, 7%, and 6% more than regular gates respectively. Also, energy consumption of QFun, NONA, and BINV are on average 50%, 40%, and 20% more than regular gates respectively. Moreover, based on the fact that memristor can be implemented on top of the silicon layer, the size of QFun, NONA, and BINV are 3.5x, 3x, and 2x in comparison with the regular gates. As an example, if an obfuscation scheme exchanges 10% percent of the regular gates with polymorphic gates using an equal distribution of QFun, NONA, and BINV, the total area overhead is less than 20% that seems reasonable.

B. Attack Complexity Measurement

Fig. 6 shows the average number of cases that are required for a brute force attack to deobfuscate the circuit with scan chain scheme under different sizes. The area overhead of the obfuscation scheme with 5%, 10%, and 20% polymorphic elements with an equal distribution of BLat and QLat is less



(a) (b) (c) (d)

Figure 5: Hybrid memristor-CMOS layout (a) NONA [24] (b) BLat (c) BINV [24] (d) QFun

Table II: Polymorphic gates comparison

	NAND			NOR			INV		
	Reg.	NONA	QFun	Reg.	NONA	QFun	Reg.	BINV	QFun
Ave. Propagation Delay (ps)	9.25	10.03	10.24	9.97	10.42	10.24	6.03	6.12	9.87
Ave. Energy Consumption (nJ)	243.2	376.2	412.5	267.2	378.4	419.9	495.1	591.1	624.5
Num. of Transistors / Memristors	4/0	12/2	12/4	4/0	12/2	12/4	2/0	4/2	12/4

Table III: State explosion problem

Gates	100	500	1000	1500	2000	2500
States	1.13E+15	1.81E+75	3.27E+150	5.92E+225	1.07E+301	1.93E+376

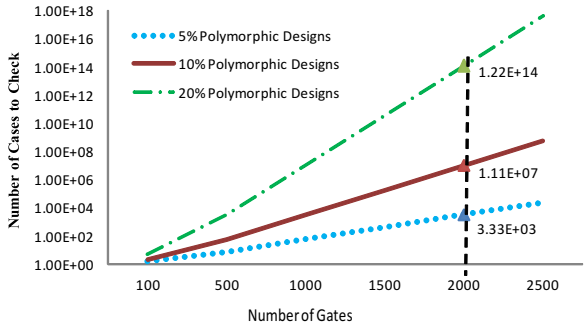


Figure 6: Brute force attack on scan chain obfuscation

than 10%, 20%, and 40% respectively. It is clear that the number of required tests exponentially grows to the linear expansion of the circuit.

Attacking the whole chip using a query-based learning algorithm is a more general option left for the attacker. Table III shows the effect of the state explosion problem in such attack. It is supposed that 50% of a circuit is memory element. As can be seen, even if the random sampling provides 100% accuracy as an equivalence oracle -that is highly optimistic-, the time complexity of the query-based learning algorithm that is polynomial to the number of states, grows exponentially with increasing the size of the circuit.

V. Conclusion

Recently, chip protection has attracted much attention because of the high IC design cost. In addition, with the growing number of untrusted third party foundries, IFA becomes an escalating problem. On the other hand, memristor is the forth basic nanoscale chip element that its resistance serves as a stored variable.

In this paper, first we proposed several polymorphic hybrid memrisor-CMOS designs in order to obfuscate the IC layout. Polymorphic designs look exactly identical for anyone that has access to the layout, including third party engineers who manufacture the chip. In this case, the design goal is to have only one functional configuration while the other configurations are meaningless. Experimental results showed that the overhead of the proposed designs are reasonable.

Next, utilizing the proposed polymorphic designs, we introduced a hierarchical hardware obfuscation consists of combinational, scan chain, and sequential obfuscations. The hierarchical scheme is secure against acyclic and cyclic SAT-based attacks, brute-force attack on scan chain, and query-based learning attack on the whole sequential circuit.

Acknowledgment

This work is partially supported by NSF under CCF-1533656, CNS-1441695, and CNS-1651695.

REFERENCES

[1] R. Torrance and D. James. The state-of-the-art in semiconductor reverse engineering. In *ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 333–338, 2011.

[2] Y. Shen, A. Rezaei, and H. Zhou. A comparative investigation of approximate attacks on logic encryptions. In *Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 271–276, 2018.

[3] J. Rajendran, M. Sam, O. Sinanoglu, and R. Karri. Security analysis of integrated circuit camouflaging. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 709–720, 2013.

[4] M. Li, K. Shamsi, T. Meade, Z. Zhao, B. Yu, Y. Jin, and D. Z. Pan. Provably secure camouflaging strategy for ic protection. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, article 28, 2016.

[5] B. Mazumdar M. Yasin, O. Sinanoglu, and J. Rajendran. Camoperturb: Secure ic camouflaging for minterm protection. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, article 29, 2016.

[6] R. P. Cocchi, J. P. Baukus L. W. Chow, and B. J. Wang. Circuit camouflage integration for hardware ip protection. In *ACM/EDAC/IEEE Design Automation Conference (DAC)*, article 153, 2014.

[7] G. T. Becker, F. Regazzoni, C. Paar, and W. P. Burleson. Stealthy dopant-level hardware trojans. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, pages 197–214, 2013.

[8] R. Ruzicka and V. Simek. Nand/nor gate polymorphism in low temperature environment. In *International Symposium on Design and Diagnostics of Electronic Circuits Systems (DDECS)*, pages 34–37, 2012.

[9] Z. Gajda and L. Sekanina. Gate-level optimization of polymorphic circuits using cartesian genetic programming. In *IEEE Congress on Evolutionary Computation (CEC)*, pages 1599–1604, 2009.

[10] F. Parveen, Z. He, S. Angizi, and D. Fan. Hybrid polymorphic logic gate with 5-terminal magnetic domain wall motion device. In *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pages 152–157, 2017.

[11] L. Chua. Memristor-the missing circuit element. In *IEEE Transactions on Circuit Theory*, volume 18, issue 5, pages 507–519, 1971.

[12] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams. The missing memristor found. In *Nature*, volume 453, issue 7191, pages 80–83, 2008.

[13] S. Kong, J. Gu, and H. Zhou. Memristor-based clock design and optimization with in-situ tunability. In *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pages 427–432, 2017.

[14] S. Kvatinsky, G. Satat, N. Wald, E. G. Friedman, A. Kolodny, and U. C. Weiser. Memristor-based material implication (imply) logic: Design principles and methodologies. In *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, volume 22, pages 2054–2066, 2014.

[15] S. Kvatinsky, D. Belousov, S. Liman, G. Satat, N. Wald, E. G. Friedman, A. Kolodny, and U. C. Weiser. Magic: Memristor-aided logic. In *IEEE Transactions on Circuits and Systems II: Express Briefs*, volume 61, pages 895–899, 2014.

[16] S. Kvatinsky, N. Wald, G. Satat, A. Kolodny, U. C. Weiser, and E. G. Friedman. Mrl: Memristor ratioed logic. In *International Workshop on Cellular Nanoscale Networks and their Applications (CNNA)*, pages 1–6, 2012.

[17] T. Singh. Hybrid memristor-cmos (memos) based logic gates and adder circuits. In *CoRR*, volume abs/1506.06735, 2015.

[18] Q. Xia et al. Memristor-cmos hybrid integrated circuits for reconfigurable logic. In *Nano Letters*, volume 9 10, pages 3640–5, 2009.

[19] D. B. Strukov et al. Hybrid cmos/memristor circuits. In *IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1967–1970, 2010.

[20] K. Shamsi, M. Li, T. Meade, Z. Zhao, D. Z Pan, and Y. Jin. Cyclic obfuscation for creating sat-unresolvable circuits. In *International Conference on Great Lakes Symposium on VLSI (GLSVLSI)*, pages 173–178, 2017.

[21] H. Zhou, R. Jiang, and S. Kong. Cycsat: Sat-based attack on cyclic logic encryptions. In *International Conference on Computer-Aided Design (ICCAD)*, pages 49–56, 2017.

[22] Y. Alkabani, F. Koushanfar, and M. Potkonjak. Remote activation of ics for piracy prevention and digital right management. In *International Conference on Computer-Aided Design (ICCAD)*, pages 674–677, 2007.

[23] J. Zhang, Y. Lin, Y. Lyu, and G. Qu. A puf-fsm binding scheme for fpga ip protection and pay-per-device licensing. In *IEEE Transactions on Information Forensics and Security*, volume 10, pages 1137–1150, 2015.

[24] A. Rezaei, Y. Shen, S. Kong, J. Gu, and H. Zhou. Cyclic locking and memristor-based obfuscation against cycsat and inside foundry attacks. In *Design, Automation and Test in Europe Conference and Exhibition (DATE)*, pages 85–90, 2018.

[25] 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)*, pages 137–143, 2015.

[26] D. Angluin. Learning regular sets from queries and counterexamples. In *Journal of Information and Computation*, volume 75, pages 87–106, 1987.

[27] A. Valmari. The state explosion problem. In *Lectures on Petri Nets I: Basic Models, Advances in Petri Nets, the Volumes Are Based on the Advanced Course on Petri Nets*, pages 429–528. Springer-Verlag, 1998.