Optimized and Automated Secure IC Design Flow: A Defense-in-Depth Approach

Kevin Immanuel Gubbi, *Student Member, IEEE*, Banafsheh Saber Latibari, *Student Member, IEEE*, Muhtasim Alam Chowdhury, Afrooz Jalilzadeh, Erfan Yazdandoost Hamedani, Setareh Rafatirad, Avesta Sasan, Houman Homayoun, and Soheil Salehi, *Member, IEEE*

***Abstract*— The globalization of the manufacturing process and the supply chain for electronic hardware has been driven by the need to maximize profitability while lowering risk in a technologically advanced silicon sector. However, many hardware IPs' security features have been broken because of the rise in successful hardware attacks. Existing security efforts frequently ignore numerous dangers in favor of fixing a particular vulnerability. This inspired the development of a unique method that uses emerging spin-based devices to obfuscate circuitry to secure hardware intellectual property (IP) during fabrication and the supply chain. We propose an Optimized and Automated Secure IC (OASIC) Design Flow, a defense-in-depth approach that can minimize overhead while maximizing security. Our EDA tool flow uses a dynamic obfuscation method that employs dynamic lockboxes, which include switch boxes and magnetic random access memory (MRAM)-based look-up tables (LUT) while offering minimal overhead and being flexible and resilient against modern SAT-based attacks and power side-channel attacks. An EDA tool flow for optimized lockbox insertion is also developed to generate SAT-resilient design netlists with the least power and area overhead. PPA metrics and security (SAT attack time) are provided to the designer for each lockbox insertion run. A verification methodology is provided to verify locked and unlocked designs for functional correctness. Finally, we use ISCAS'85 benchmarks to show that the EDA tool flow provides a secure hardware netlist with maximum security while considering power and area constraints. Our results indicate that the proposed OASIC design flow can maximize security while incurring less than 15% area overhead and maintaining a similar power footprint compared to the original design. OASIC design flow demonstrates improved performance as design size increases, which demonstrates the scalability of the proposed approach.**

***Index Terms*— EDA, hardware security, power side-channel, reverse engineering, defense-in-depth, STT-MRAM.**

Manuscript received 2 October 2023; revised 1 January 2024; accepted 30 January 2024. Date of publication 22 February 2024; date of current version 30 April 2024. This article was recommended by Associate Editor Y. Tang. *(Corresponding author: Soheil Salehi.)*

Kevin Immanuel Gubbi, Banafsheh Saber Latibari, Avesta Sasan, and Houman Homayoun are with the Department of Electrical and Computer Engineering, University of California at Davis, Davis, CA 95616 USA (e-mail: kgubbi@ucdavis.edu; bsaberlatibari@ucdavis.edu; asasan@ucdavis.edu; hhomayoun@ucdavis.edu).

Muhtasim Alam Chowdhury and Soheil Salehi are with the Department of Electrical and Computer Engineering, The University of Arizona, Tucson, AZ 85721 USA (e-mail: mmc7@arizona.edu; ssalehi@arizona.edu).

Afrooz Jalilzadeh and Erfan Yazdandoost Hamedani are with the Department of Systems and Industrial Engineering, The University of Arizona, Tucson, AZ 85721 USA (e-mail: afrooz@arizona.edu; erfany@arizona.edu).

Setareh Rafatirad is with the Department of Computer Science, University of California at Davis, Davis, CA 95616 USA (e-mail: srafatirad@ucdavis.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSI.2024.3364160.

Digital Object Identifier 10.1109/TCSI.2024.3364160

I. Introduction

THE increasing complexity of semiconductor Intellectual Property (IP), combined with the shrinking transistor dimensions in advanced process nodes, has led to an intricate and distributed Integrated Circuit (IC) supply chain. A complex network of international suppliers supports the design, production, and assembly of contemporary ICs, which include millions and billions of transistors and require complicated fabrication techniques. In addition to the more than 8,500 suppliers physically situated outside of the United States, U.S.-based semiconductor businesses have 7,300 suppliers scattered throughout 46 states. The worldwide expansion of the manufacturing process and the hardware supply chain is designed to primarily profit from the geographical advantages of suppliers and the wide range of talents of human resources. Even though the advantage of the global ecosystem has continued for all participants and their respective global economies, the security of the underlying hardware is currently under threat from various emerging hardware security risks, such as overproduction, Trojan insertion, reverse engineering, IP theft, and counterfeiting. With hardware security breaches becoming increasingly successful, the phrase "trust starts in silicon," which regards hardware as the foundation of security, is under doubt. The complexity involved in improving hardware security has led the hardware security community to devise various approaches for reevaluating hardware security. The main problem of hardware security concerns, such as IP theft, overproduction, and reverse engineering, to mention a few, has shown promise in being alleviated by hardware design-for-trust approaches such as split manufacturing, IC camouflaging, and logic locking. But among the numerous solutions discussed above, logic locking has the power to prevent the bulk of hardware security threats throughout different stages of the worldwide supply chain [1]. The right keys must be provided to unlock the design's full capabilities, and this is done in a trusted environment as part of the post-manufacturing process. The functioning of IP is unknown at this point because the IC is obfuscated. The SAT-attack, also known as the boolean attack, has the capacity to restore the functionality of the obfuscated IP thanks to continuing extensive research in hardware security. An oracle circuit (activated IC) and the

netlist must be obtained for the SAT-attack utilizing invasive reverse engineering techniques [2]. A differentiating input pattern (DIP) that can discriminate between two keys is sought by the SAT-attack. By using this DIP, you may get rid of the wrong keys from the oracle. This process is repeated by the SAT-attack until it cannot locate the DIP.

Many efforts attempt to increase the number of iterations needed to obtain the correct key for unlocking the circuit in order to prevent the SAT-attack. The most advanced technique is called Stripped Functionality Logic Locking (SFLL), which needs exponential SAT iterations to locate the correct key. It thwarts bypass and removal attacks in addition to the SAT-attack [3]. However, the studies in [4] and [5] reveal the flaw in the SFLL method implementation and demonstrate that it is possible to get the obfuscation key in a matter of minutes. Additionally, the obfuscation primitives SFLL, SAR-Lock, and Anti-SAT belong to the class of one-point functions that evaluate the correct output when used with a particular input pattern. The output corruptibility provided by such approaches is extremely low. Thus, even when the incorrect key is used, the circuit functions essentially identically to the oracle circuit. This is true even if SAT-attack must apply multiple inputs to locate the correct key [6]. The community also developed obfuscation techniques that prevent the obfuscated netlist from being transformed into SAT problems in an effort to fend off the SAT-attack. Delay-based locking and cyclic obfuscation are some techniques that fend off SAT-attack.

Attacks, such as cycSAT and satisfiability modulo theories (SMT), were able to model the cyclic or behavioral locking into an SAT or SAT with theory-solvable logic problems, albeit not long after the introduction of these obfuscation techniques. [7]. However, research in the reconfigurable domain prevents SAT-attacks by expanding the key search space. The attacker has a key search space of 2^{2^M} thanks to the M-input Look Up Table (LUT)-based obfuscation [8]. However, the LUT's higher overhead prevents their practical adoption for security reasons. A magneto-electric spin-orbit (MESO) device is used in another reconfigurable effort to resist most attacks while imposing lower overheads. MESO devices allow both reconfigurability and dynamic morphing. Even though this is a positive step, their obfuscation method can only be used in applications that can accept a certain degree of accuracy loss [9]. By creating an SAT-hard instance employing a key-configurable logarithmic-based network for route obfuscation, the work in [10] attempts to withstand the SAT-attack. Even though this appears to stop SAT-attacks, an alternative formulation of the SAT can aid the attacker in swiftly obtaining the keys [11].

As a result of this never-ending back and forth between SAT-attacks and obfuscation approaches, novel attacks have consistently been able to identify vulnerabilities. In this work, we provide a unique Security-Aware EDA tool flow and technique to resist and thwart multiple attacks to form a defense-in-depth approach, in contrast to earlier studies like [8], [11], and [12] that only concentrate on mitigating a single attack vector. This work presents the following contributions:

- Automated creation of SAT-hard instances for secure circuit design using MRAM-based LUTs and routing-based obfuscation. Ensures reliability, low power, and resilience to removal and scan-based attacks.
- Development of an EDA tool flow for optimized and automated lockbox placement and security evaluation. Determines optimal lockbox size and number configuration for a SAT-resilient design with minimal power and area overhead.
- Scalable EDA solution for empirical evaluation of lockboxes using the OASIC design flow. Demonstrates security against machine learning-assisted power side-channel attacks with minimal power and area overhead.

II. Background and Motivation

A. SAT Attack and Logic Locking

1) Logic Locking: Logic Locking is a hardware security approach that is based on inserting additional logic gates into the design to protect the IPs from reverse engineering, IP piracy, overproduction, and unauthorized activation. Logic locking is implemented at different abstraction levels: Layout level, Transistor level, gate level, Register Transfer Level (RTL) level, and high level. Logic locking methods can be classified into the subsequent categories: primitive, point function, cyclic, LUT/routing, scan-based, sequential/finite state machine (FSM), timing, FPGA-based, and high-level synthesis (HLS)-based [13]. In this approach, key-programmable XOR/XNOR gates, MUXes, or LUTs are inserted into the original netlist, and the key is in a tamper-proof memory. The location and the number of inserted key gates are key factors in the strength of this method [14], [15], [16]. If the inserted gates are isolated or mutable, the attacker can easily discover the keys. In Strong Logic Locking [17], the insertion location is such that the effect of input can be easily sensitized to the output. In other work, researchers used camouflaging to block the attacker's reverse engineering attempt [18]. Several works focused on presenting hardware security attack mechanisms, such as justification and sensitization attacks [19] to reverse engineer the design. Later, several powerful attack mechanisms using Automatic Test Generation Pattern (ATPG) and SAT-attack [2] were proposed. The next section will delve into the concept of the SAT-Attack and related works.

2) SAT-Attack and Defense: SAT-Attack [2] is based on the Boolean Satisfiability problem, and it is an oracle-guided attack. The SAT solver processes the input in conjunctive normal form (CNF), so the CNF format of the original and obfuscated circuits are obtained from the netlist. SAT-Attack uses a miter circuit to find Discriminating Input Patterns (DIPs), to eliminate the incorrect key based on the input patterns iteratively. The attack continues until no new DIP exits. Figure 1 shows an overview of the SAT-Attack. Every key input combination from the set of candidate keys (SCK) is considered a candidate key. If the attacker can find an input x_d and two different key values k_1 and k_2 in SCK such that $C(x_d, K_1, Y_1) \neq C(x_d, K_2, Y_2)$, the input x_d would be a DIP. So, the incorrect key is identified and removed from SCK in each iteration by finding a new DIP. Moreover, the

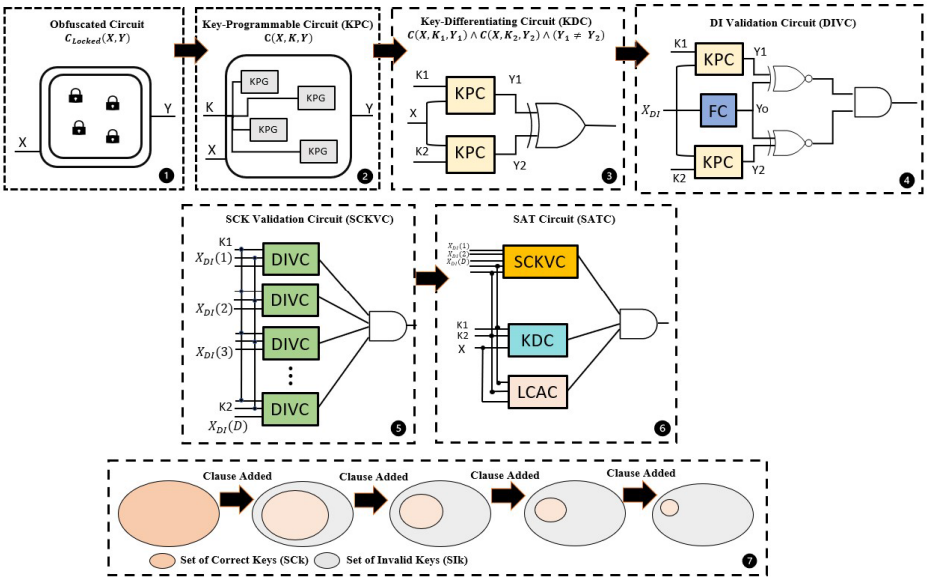


Fig. 1. Overview of the SAT-Attack: (1) An obfuscated circuit (2) Key programmable circuit (3) Key differentiating circuit (4) Discriminating Input (DI) validation circuit (5) SCK validation circuit (6) SAT circuit (7) Reduction of the Set of Candidate Keys (SCK) after each iteration.

new DIP should be different from the previously founded DIPs in previous iterations. Figure 1 (1) shows an obfuscated circuit, (2) shows the reverse-engineered netlist of the design with Key Programmable Gate (KPG) cells, (3) shows the miter circuit or Key Programmable Circuits (KPCs), that the output of the circuits is one when the output of the circuit for the same input X is different for K_1 and K_2, (4) shows the Discriminating Input (DI) Validation Circuit (DIVC) that checks two input keys to produce the correct output with respect to a previously discovered DIP, (5) shows the Set of Candidate Keys (SCK) validation circuit and is used to check a new DIP is compatible with all DIPs that have previously been discovered, and (6) shows the final SAT circuit. Many researchers in the community of hardware security introduced different tricks to make their defense mechanism SAT-hard by increasing the number of required iterations or the time of each iteration to find the correct key using SAT-Attack. Stripped Functionality Logic Locking (SFLL) [5] increases the number of required iterations exponentially. However, later future work showed that SFLL is easily breakable, and the key could be found within several minutes. SAR-Lock [20], and Anti-SAT [21] are classified as one-point obfuscation-based techniques meaning that the circuit works almost identically to the oracle circuit even with the wrong key. In other words, the output corruptibility is very low in these techniques. Authors in Delay-based locking [22] or cyclic obfuscation [23] avoid SAT-Attack by formulating the obfuscation techniques such that it makes transmission of the obfuscated netlist to SAT problems difficult. In CycSAT [24] and SMT [7], the proposed methods were successful in modeling the cycles and delays and translating these concepts to SAT solvable problems. Some works introduced defense mechanisms based on using reconfigurable blocks such as LUT. In LUT-based obfuscation group of gates is replaced with LUT. The LUT configuration bit is set based on the truth table of the gate. In this obfuscation method, the key that determines the logical functionality of LUT is stored in a tamper-proof memory. However, this obfuscation is still breakable by SAT-Attack. To apply the SAT-Attack on the LUT-based obfuscated circuit, LUTs are replaced with MUXes. To avoid SAT-attack, researchers proposed using a larger size of LUT in large quantities. However, it leads to design overhead.

B. Polymorphic Logic Using Emerging Devices for Security

LUTs have garnered significant attention as one of the most commonly used reconfigurable fabric options for logic obfuscation. This is primarily due to their adaptability, which allows for the creation of logic elements at moderate and fine levels of detail with minimal initial engineering costs. Moreover, researchers have explored LUTs as a potential platform to enhance reliability in the context of process-voltage-temperature variations [12], [25]. However, there are inherent challenges when it comes to achieving flexible designs using SRAM-based LUTs, primarily stemming from their increased space and power consumption SRAM-based LUTs also exhibit drawbacks such as elevated static power consumption, instability, and limited logic density [26]. Furthermore, from a security perspective, conventional SRAM-based LUTs are susceptible to Power Side-Channel Attacks (P-SCA). Efforts have been made to introduce innovations in LUT design by leveraging emerging devices, aiming to address the limitations associated with SRAM-based LUTs [25], [27]. One promising alternative explored in the literature involves high-endurance non-volatile spin-based LUTs, which are positioned as potential replacements for SRAM-based LUTs, Flash-based LUTs, and other state-of-the-art emerging LUT technologies like RRAM-based LUTs and PCM-based LUTs [25], [27], [28], [29]. Spin-based devices offer several advantages, including non-volatility, minimal static power consumption, robust endurance, high integration density, and compatibility with the CMOS manufacturing process. However, many of the spin-based LUTs proposed in existing research struggle to maintain a wide sense margin and are prone to high error

rates, especially when faced with Process Variation (PV) challenges [25]. In our study, we employ commercially available Spin Transfer Torque (STT) Magnetic Tunnel Junctions (MTJ) to create a secure and P-SCA-resilient non-volatile Magnetic RAM (MRAM)-based LUT with a generous read margin, addressing the aforementioned issues. Furthermore, we compare this MRAM-based LUT to a conventional SRAM-based LUT. Additionally, we provide an assessment of reliability under PV and evaluate the security and resistance to attacks of the LUT design. While previous research has showcased the potential for power dissipation reduction through polymorphic logic implementation using Magneto-Electric Spin-Orbit (MESO) devices [9], our study utilizes STT-MTJ devices due to their commercial availability, as opposed to MESO devices.

C. Spin Transfer Torque Magnetic RAM (STT-MRAM)

The fundamental idea behind spin-based Non-Volatile Memory (NVM) devices involves the manipulation of the inherent spin of electrons within a solid-state nano-device composed of a ferromagnetic thin film. MTJ devices are created by stacking pillars made of layers of ferromagnetic and insulating materials. This design enables the manipulation and detection of magnetic orientations using electrical signals. The non-volatile MTJ device comprises two ferromagnetic layers referred to as the fixed layer and the free layer, with a tunneling oxide layer in between [30]. These ferromagnetic layers can be aligned in two different magnetization configurations: Parallel (P) and Anti-Parallel (AP). Consequently, the MTJ exhibits either low resistance (R_P) or high resistance (R_{AP}), respectively [30]. Researchers have employed few different methods for writing data to MTJ cells [31]. From these methods, we focus on Current-Induced Magnetic Switching (CIMS), also known as Spin Torque Transfer (STT). The transition of MTJ states occurs when I_{MTJ} surpasses a critical current threshold, IC. The resistance of the MTJ in the P ($\theta = 0°$), and AP ($\theta = 180°$) states can be described by the following equations:

$$R(\theta) = 2R_{MTJ} \times \frac{1 + TMR(V_b)}{2 + TMR(V_b) + TMR(V_b).\cos(\theta)} = \begin{cases} R_P = R_{MTJ}, & \theta = 0° \\ R_{AP} = R_{MTJ}(1 + TMR), & \theta = 180°, \end{cases} \tag{1}$$

$$R_{MTJ} = \frac{t_{ox}}{Factor \times Area \cdot \sqrt{\phi}} \exp(1.025 \times t_{ox} \cdot \sqrt{\phi}), \tag{2}$$

$$TMR(V_b) = \frac{TMR(0)}{(1 + (\frac{V_b}{V_h})^2)}, \tag{3}$$

where V_b represents the bias voltage, and $V_h = 0.5V$ signifies the bias voltage at which the Tunnel Magneto-Resistance (TMR) ratio reaches half of the $TMR0$ value. Additionally, t_{ox} denotes the thickness of the oxide layer in the Magnetic Tunnel Junction (MTJ), and the Factor is determined from the resistance-area product of the MTJ, which depends on the material composition of its layers. The term "Area" pertains to the surface area of the MTJ, while ϕ corresponds to the energy barrier height of the oxide layer [32]. The method employed

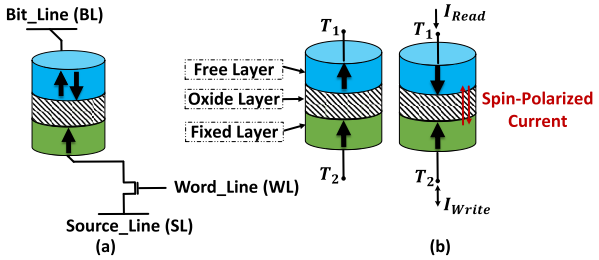


Fig. 2. (a) STT-MRAM cell structure with one transistor and one Magnetic Tunnel Junction (MTJ). (b) On the right: Anti-parallel state (characterized by high resistance), on the left: Parallel state (characterized by low resistance).

TABLE I
MTJ PARAMETERS DESCRIPTION AND VALUES USED

Parameters	Description	Value
MTJ_{Area}	$l_{MTJ} \times w_{MTJ} \times \pi/4$	$15nm \times 15nm \times \pi/4$
t_f	Free Layer thickness	$1.3nm$
RA	MTJ resistance-area product	$9\Omega.\mu m^2$
T	Temperature	$358K$
α	Damping coefficient	0.007
P	Polarization	0.52
V_0	Fitting parameter	0.65
α_{sp}	Material-dependent constant	$2e^-5$

for Spin-Transfer Torque (STT) switching relies on the application of a spin-polarized current through the MTJ junction. This current induces a change in the magnetization of the free layer when its magnitude surpasses a specific threshold known as the critical current. As shown in Figure 2a, STT-MRAM makes use of an MTJ device as its primary storage component. While STT-MRAM offers clear advantages such as enhanced write endurance, it comes with trade-offs in the form of increased write latency and higher energy consumption, intensifying the energy and reliability concerns associated with this technology. STT switching is particularly noteworthy as a data storage alternative due to its independence from external wires and magnetic fields, reduced current density requirements for switching, and lower power consumption compared to other methods mentioned in this context [31]. Figure 2b provides a visual representation of an STT-MRAM cell featuring an access transistor, a configuration commonly referred to as the "one-transistor-one-MTJ (1T-1R)" structure [30]. Furthermore, Table I lists the MTJ parameters and their value that we have considered in our simulations.

D. Electronic Design Automation for Hardware Security

This section provides an overview of the current state of the art in EDA for hardware security. Previously, the EDA tool focused mostly on power, performance, and area. However, security is another important aspect that matters a lot [33]. Some hardware security threats include Side-Channel Attacks (SCAs), Fault-Injection Attacks (FIAs), Piracy of Design IP/Counterfeiting of ICs, and Hardware Trojans (HTs). These threats could exist in various design stages, such as High-level synthesis [34], [35], [36], [37], Logic synthesis [15], [38], [39], [40], Physical synthesis [41], [42], [43], [44], Functional validation [7], [45], [46], [47], Timing and power validation [48], [49], [50], and Testing [51], [52]. In [53], they have proposed HW2VEC, which is a graph learning-based tool to resolve hardware security issues in EDA. HW2VEC is capable of

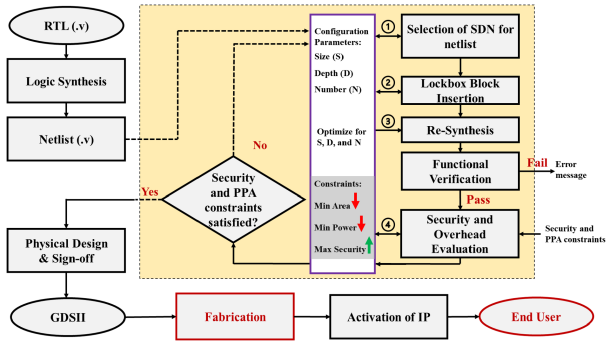


Fig. 3. Proposed OASIC design flow for the Defense-in-Depth synthesis of hardware using emerging MRAM-based lockboxes.

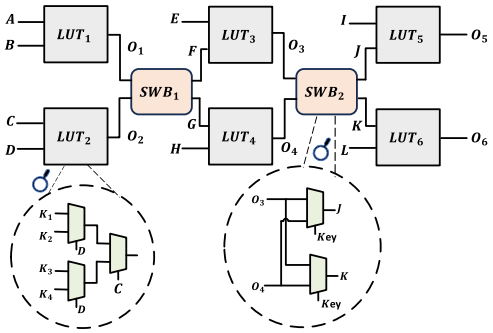


Fig. 4. The circuit-level diagram of the proposed $2 \times 2 \times 2$ lockbox.

extracting a graph representation from a hardware design in various abstraction levels. In this work, they used HW2VEC for Hardware Trojan Detection and IP piracy detection. In [54], they addressed the challenge of rising counterfeiting in EDA companies and the need for anti-counterfeiting solutions. In [55], they focused on approaches for the Independent Validation and Verification (IV&V) of security-aware EDA tools. They presented techniques to prevent reverse engineering, HT insertion, SCA, and attacks on the asset management infrastructure (AMI). EDA tools designed for security have been increasing in popularity to prevent P-SCAs. EDA tools include security features that help designers build secure electronic systems by detecting and mitigating vulnerabilities in their design in a more streamlined process. For instance, tools that incorporate Test Vector Leakage Assessment (TVLA) can help identify and eliminate security vulnerabilities through analysis of power traces during test vector execution. Moreover, EDA security tools enable designers to detect and prevent P-SCAs early in the design phase, saving a significant amount of time and effort in design verification. Previous work has leveraged EDA flows to minimize area and delay overheads. Johann et al., have demonstrated that side-channel resilience of optimized designs using differential power analysis is possible without incurring any drawbacks in terms of delay, power, and gate-count [56]. Prior research into the field has been heavily focused on replacing static circuit elements with elements that are better suited for hardware security such as the timing of circuit signals which randomizes side-channel information that is leaked [57] or wave dynamic differential logic (WDDL) implemented using static CMOS logic to standardize power consumption across different gate types [58].

E. Motivation

While dynamic camouflaging of MESO devices remains a potent defense against SAT attacks, its applicability is limited to approximate applications such as image processing, that can withstand a certain level of errors. The study presented in [9] illustrates that even without dynamic camouflaging, MESO devices can still generate an SAT-hard instance, leading to timeout states. Moreover, their SAT representation of MESO devices in benchmark files resembles a Multiplexer (MUX), featuring an additional 8 gates and 7 MUXes. However, the same functionality can be achieved by replacing the MESO device with a LUT of size 2, utilizing just 3 MUXes, as shown in Figure 4. This LUT of size 2 can effectively emulate all 8 functions offered by a MESO device, significantly reducing the SAT-attack runtime associated with MESO-based obfuscation. On the flip side, LUT-based obfuscation provides a substantial key search space for SAT solvers, making it challenging to find the key. Nevertheless, it imposes a notable overhead. Recent research in [12] aims to mitigate this overhead while preserving SAT resiliency, but the overhead remains substantial.

In contrast, work in [10] strives to enhance SAT hardness by employing key-configurable logarithmic-based networks, aiming for a balanced clause-to-variable ratio between 3 to 6 to create SAT-hard problems. However, despite their efforts to thwart SAT attacks, research in [11] demonstrates that the key can still be retrieved within polynomial time. Furthermore, they suggest an incremental variant of routing-based obfuscation, but the establishment of an SAT-hard problem remains subjective in terms of security. SAT-hard problems today can be solved more swiftly with emerging SAT solvers and hardware. While the majority of strategies concentrate on alleviating SAT attacks, it's crucial not to underestimate the potential for P-SCAs. To bolster hardware security, drawing inspiration from the aforementioned concepts, we employ lockboxes. These lockboxes can effectively counter SAT attacks, removal attacks, and probabilistic attacks while diminishing non-invasive P-SCAs by utilizing MRAM-based LUTs, as elaborated in [2]. In the following section, we delve into the EDA tool flow for the Secure Defense-in-Depth Synthesis of hardware and the construction of these lockboxes.

III. Proposed OASIC Design Flow

A. Defense-in-Depth EDA Tool Flow for Secure Hardware

The seamless integration of lockboxes into a circuit design, without any predetermined insertion policy, mitigates the need for manual intervention by the IP designer. In contrast to traditional hardware security practices that emphasize strategic obfuscation placement within logic cones featuring numerous gates, our approach involves stochastic lockbox insertion. This process entails the systematic replacement of random gates with LUTs. Despite the apparent randomness, this insertion strategy consistently yields outstanding SAT-resiliency, resulting in a notable elevation of output corruption and a commensurate enhancement of overall security. To systematize this intricate task, we propose an advanced security-aware

EDA tool flow. The initiation of the tool flow begins with a Verilog high-level RTL design description. This description undergoes a meticulous synthesis process to yield a gate-level netlist. Subsequently, this netlist serves as the input for our advanced Optimized and Automated Secure IC (OASIC) design tool flow. The determination of the optimal number and size of lockboxes, which is pivotal to our methodology, relies on a configuration file encapsulating desired security parameters. In cases where a valid configuration is not readily available, our tool initiates an iterative process. It commences with a single lockbox of size 2×2 and systematically explores a range of lockbox sizes, including $2 \times 2 \times 2$, 4×4, $4 \times 4 \times 4$, 8×8, and $8 \times 8 \times 8$, at various insertion frequencies (1x, 2x, 3x, 4x, 5x, 10x, and 25x). The EDA tool flow is constructed using a set of Python and Tcl scripts. A Python script processes the target netlist, inserting automatically generated lockboxes based on user inputs. Subsequently, a set of wrapper scripts facilitates seamless integration between various EDA tools and our security-aware optimization tool flow, enabling automated security-aware design. This systematic exploration ensures a thorough examination of the design space, optimizing SAT-resiliency while considering critical performance metrics such as SAT-attack time, power consumption, and area utilization. The integration of automation into our design flow is pivotal for enhancing overall efficiency and reducing the manual workload on designers. This expedites the lockbox insertion process and contributes to a more robust and secure design. Through systematic exploration of diverse configurations, our automated methodology ensures that the final design achieves optimal SAT-resiliency with minimal impact on critical performance metrics. This approach not only results in a more secure outcome but also emphasizes the crucial role of automation in navigating the intricate landscape of modern hardware security design.

B. Achieving SAT Resiliency

SAT-solvers are widely employed due to their effectiveness in solving practical problems using the Conflict-Driven Clause Learning (CDCL) algorithm. CDCL is an extension of the Davis-Putnam-Logemann-Loveland (DPLL) algorithm, incorporating the capacity to acquire new clauses and backtrack non-chronologically [59]. Notably, the CDCL algorithm emerged as the fastest SAT solver at the 2019 CaDiCaL SAT-solving competition, as noted in [59], [60], and [61]. As previously discussed, numerous obfuscation techniques employ the one-point function to increase the number of DIPs for uncovering the correct key. In the realm of SAT solvers, this phenomenon translates into a higher number of DPLL iterations. While this presents a promising approach, it necessitates a trade-off between reduced output susceptibility provided by the obfuscation and increased SAT resilience. To circumvent this trade-off, in LUT-based obfuscation, LUTs can be strategically placed within different output logic cones. This not only enhances output susceptibility but also capitalizes on the use of larger LUT sizes to bolster SAT resilience.

The enhanced LUT design introduced in [12] combines both small and large LUTs, resulting in the generation of SAT-hard instances. The symmetric issue arising from the MUX-tree structure of the LUT enforces a relationship between the number of clauses and the number of variables. This relationship maximizes the penalty associated with incorrect variable assignments uniformly across the search tree. Furthermore, a similar symmetric problem can be formulated using the logarithmic routing network, as explained in [10]. Taking inspiration from this concept and the structure of FPGA, the authors in [62] introduced a hybrid approach that combines LUTs with a logarithmic routing-based obfuscation. This hybrid technique not only expands the search key space for SAT solvers but also contributes to the creation of SAT-hard problems. Furthermore, in order to minimize the overhead associated with LUTs, the authors imposed a size constraint on the LUTs, opting for LUTs of size 2, which can be replaced by MESO or other emerging devices if necessary. While this approach effectively resists SAT attacks, augmenting the size of the LUTs can enhance security even further.

Figure 4 illustrates the implementation of these security mechanisms, featuring LUTs of size 2 and two Logarithmic banyan networks. It's worth noting that the banyan network utilized here is nearly non-blocking and incorporates $(N/2)log_2^N$ switching blocks. During the development of the lockbox, we explored various configurations. Figure 4 illustrates a $2 \times 2 \times 2$ lockbox featuring a single switch box positioned between each component. The inputs of this switch box are connected to the fanout of the LUTs (replacing 2-input gates in the circuit), with the output of the switch box serving as the input to one of the 2-input LUTs (denoted as F and G). Given that the $2 \times 2 \times 2$ lockbox can be relatively easily de-obfuscated using SAT attacks, we opted to increase the size of the switching network, which resembles a banyan network. For obfuscation purposes, we tested both an 8×8 lockbox with 16 switching elements and an $8 \times 8 \times 8$ lockbox with 32 switching elements. The implementation details of the 8×8 and $4 \times 4 \times 4$ lockboxes can be found in Figure 5 and Figure 6, respectively. Additionally, Figure 4 provides a representation of each switching block for use in SAT-solver simulations. Moreover, Figure 5 offers a basic example illustrating how the gates of the ISCAS s838 benchmark can be mapped to the 8×8 lockbox, aiding in a clearer comprehension of the implementation process.

In the simulation of SAT attacks, each LUT and routing block undergoes conversion into MUXes, contributing to the expansion of the recursive DPLL tree search. The introduction of these MUXes serves to augment the clause-to-variable ratio, resulting in symmetric switching networks within the circuit that are interconnected with LUTs. This, in turn, heightens SAT hardness, even for advanced solvers like FunSAT SAT-solver as outlined in [63]. The heightened challenge in SAT problem-solving arises from the repercussions of incorrect variable assignments within the DPLL method. When an improper assignment occurs, DPLL must backtrack from the farthest point in the tree to the branch where the erroneous assignment was made. The key strength of constructing SAT-hard solutions through the deployment of a lockbox lies in several factors: (1) Each iteration of the SAT attack poses SAT-hard problems. (2) The output corruption rate in this approach significantly surpasses that of obfuscation relying on a one-point function. (3) It remains impervious to bypass, removal,

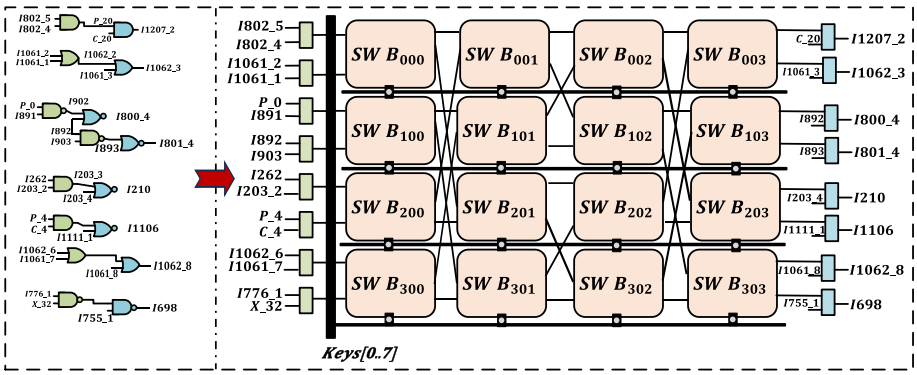


Fig. 5. The gate-level schematic of S838 (ISCAS benchmark) showing selected gates for lock insertion using an 8 × 8 lockbox.

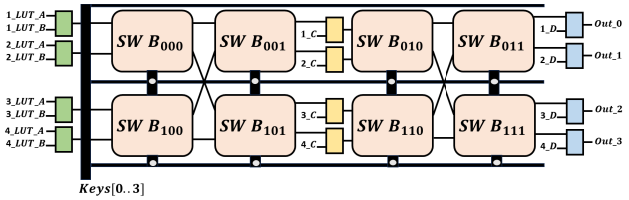


Fig. 6. Conceptual diagram of the proposed 4 × 4 × 4 lockbox.

or approximate attacks. (4) It also actively endeavors to thwart SAT attacks and mitigate P-SCAs. To assess the resilience and SAT hardness of the proposed lockbox, we applied obfuscation to the 'C7552' benchmark from ISCAS-85, employing lockboxes of different sizes and varying the number of lockboxes from 1 to 50 whenever feasible. The outcomes illustrated in Table III demonstrate that increasing the number of lockboxes enhances SAT hardness while enlarging the lockbox size enhances SAT resilience. Instead of obfuscating 75 gates with 2 × 2 lockboxes, utilizing three 8 × 8 × 8 blocks resulted in SAT timeouts. Moreover, the overhead incurred by employing the 8 × 8 × 8 blocks is approximately three times lower compared to the utilization of 75 lockboxes of size 2 × 2. This illustrates how the introduction of an increasing number of SAT-hard lockbox instances into the circuit leads to the repeated occurrence of extremely deep and notably extensive DPLL recursion trees. Consequently, this substantially and exponentially elongates the time required for the execution of DPLL recursive calls. In contrast to the approach outlined in [64], the switch box in this research employs just 2 MUXes instead of 4. Additionally, the inclusion of an extra inverter in the switch box, as seen in FullLock [10], introduces additional overhead and augments the count of correct keys within the circuit. To illustrate, a single incorrect inversion in the first switch box element can be rectified by applying another inversion element in the subsequent switch box. In the subsequent sections, we will provide further elaboration on the construction of the LUT utilized in these lockboxes.

C. Power-Balanced MRAM-Based LUT

Combinational logic implementation is the main aim of using LUT. In order to implement M-input Boolean functions, M-input LUTs typically contain 2M memory cells. To evaluate the correct LUT functionality, a select tree MUX circuitry is utilized to access the contents of the memory cell. Transmission gates (TGs) and pass transistors are used in the construction of the selected tree MUX. The 2-input example of the MRAM-based LUT design is shown in Figure 7. The WE and $\overline{WE}$ signals are used to control the Write operation by linking each memory cell to the Bit Line, BL, and its complementary signal, $\overline{BL}$, as shown in Figure 7. During the Write procedure, we may access each memory cell separately using inputs A and B, and we can modify the memory's content by adjusting BL and $\overline{BL}$. To ensure that MTJ_i and $\overline{MTJ_i}$ always have the opposite values, we also update the contents of MTJs in each memory cell in a complimentary manner during each write operation. In particular, MTJ_i will be in the AP or high-resistance state if the data stored in the $\overline{MTJ_i}$ is in the P or low-resistance state, and vice versa. As a result, we can read the data stored in the main memory cell accurately by using a sense amplifier where the complementary value of each cell is the reference ($R_P \ll R_{AP}$) rather than a reference cell with a resistance value of $R_{ref} = \frac{R_P + R_{AP}}{2}$ ($R_P < R_{ref} < R_{AP}$). The large sense margin during the read operation results in a more reliable sensing operation. To read the data stored in the MTJ devices, we will first enable the Pre-Charge signal, PC, to charge both Out and $\overline{Out}$ to VDD. Then we disable the PC signal and we can access the data in the MTJ devices by enabling the RE and $\overline{RE}$ signals. Read enable signals RE and $\overline{RE}$ enable the read path from $X1$ and $X2$ to the ground, resulting in a race condition due to the difference in the resistance of complementary MTJs. The sense amplifier observes the resistance difference between the MTJ_i and $\overline{MTJ_i}$ and depending on the data produces both the output Out and its complementary value $\overline{Out}$. The MTJ terminals are connected to the select tree MUXes and input signals A and B are used to select the output accordingly. This value is observed at $Q1$ and $Q2$ nodes. The MRAM-based LUT enhances design security through dynamic obfuscation and P-SCA mitigation, maintaining consistent power usage and minimal power variation in the output. MTJ device contents can be reconfigured within each LUT using key input via the BL signal to alter functionality. For example, to implement the AND function, A and B inputs select memory cells as 11, 10, 01, and 00, with corresponding keys via the BL signal as 1, 0, 0, and 0. Thus the proposed design can implement any of the 16 gates listed in Table II. In the next section, we'll explore Scan Enable obfuscation of the LUT output.

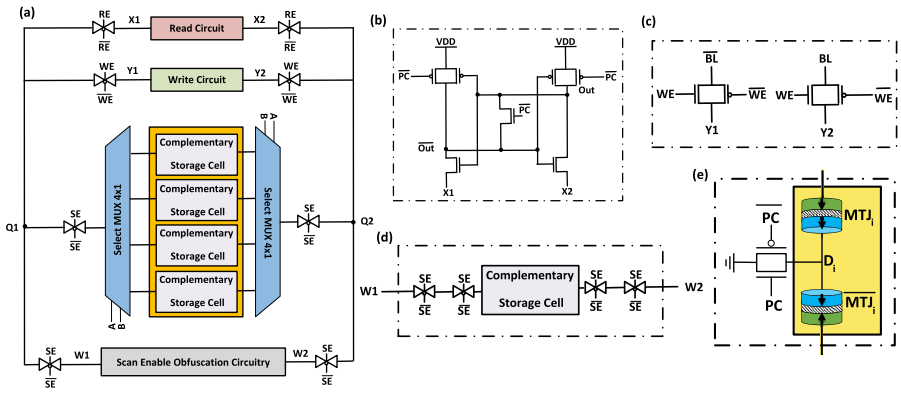


Fig. 7. Circuit-level diagram of the 2-input P-SCA-resilient LUT with scan-enable obfuscation mechanism using STT-MTJ devices. (a) Overall view of the design. (b) Read circuit. (c) Write circuit. (d) Scan-enable obfuscation circuit. (e) Complementary MRAM storage cell.

TABLE II
KEY BITS (K[1:4]) FOR IMPLEMENTING 16 DIFFERENT BOOLEAN FUNCTIONS UTILIZING THE STT-MTJ-BASED 2-INPUT LUT

Function	K[1:4]	Function	K[1:4]
0	0000	1	1111
A AND B	0001	Abar AND B	0100
A OR B	0111	A OR Bbar	1011
A NAND B	1110	A NAND Bbar	1101
A NOR Bbar	0100	A NOR B	1000
A	0011	Abar	1100
B	0101	Bbar	1010
A XOR B	0110	A XNOR B	1001

D. Scan Enable Obfuscation Mechanism

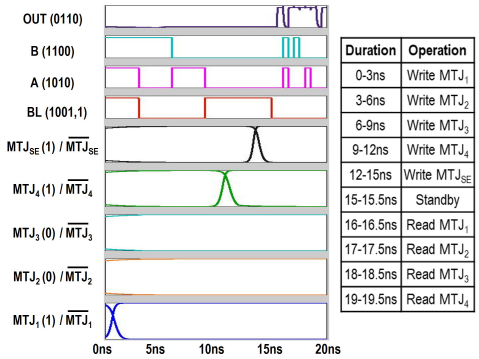
The proposed lockbox employs a combination of LUTs and interconnect obfuscation to enhance SAT hardness, with MRAM-based LUTs facilitating dynamic morphing and mitigation of P-SCA. To further fortify this primitive against SAT attacks and unforeseen threats, we suggest the incorporation of an additional complementary MTJ memory cell within the LUT. This memory cell stores a key value that can be dynamically configured for each LUT. If the Scan Enable signal (SE) is enabled during the read operation, the data stored in MTJ_{SE} and $\overline{MTJ}_{SE}$ will determine whether '0' is transmitted to the output or if '1' is randomly selected. The contents of these extra memory cells are randomized for each MRAM-based LUT incorporated into the design, using the same Write circuit. When an attacker attempts a SAT-attack, they would activate the SE signal in the Oracle IC to apply the DIP and collect the responses. In this scenario, the contents of MTJ_{SE} would be reflected in the output, yielding an obfuscated response, thus providing an added layer of security. It's worth noting that since the IP designer possesses knowledge of the values stored in the MTJ_{SE} memory cells, this circuitry doesn't cause any errors during the testing phase. The IP designer knows about this design aspect, while the attacker is assumed to lack access to MTJ_{SE} contents, preventing them from determining functionality.

IV. EXPERIMENTAL EVALUATION

In the Experimental Evaluation section, we assess the effectiveness of our suggested lockboxes against state-of-the-art SAT-attacks, utilizing the new FunSAT solver and the original SAT-solver. The benchmarking process involves the widely adopted ISCAS-89 suite, providing insights into real-world design scenarios and complexities.

A. EDA Tool Flow

To empirically demonstrate the efficacy of the lockboxes, we evaluate them against state-of-the-art SAT-attack with new FunSAT solver [63] and the original SAT-solver by [2]. FunSAT takes in a Verilog netlist and implements an SAT-solver on the locked Verilog netlist with the Oracle Verilog netlist. For BENCH files, we use the original SAT-attack [2]. The lockbox testing benchmark comprises the ISCAS-89 benchmark suite, which is widely adopted in the community and includes a few IPs serving as representative examples of real-world design size and complexity. All experiments were carried out on a 64-bit computer system featuring a quad-core Intel(R) Xeon(R) CPU E3-1271 v3 @3.60GHz processor and 64GB of memory. Our threat model assumes the key is stored in tamper-proof memory, and the attacker has access to a fully reverse-engineered IC and the activated Oracle IC. Our system sets a timeout of 250 seconds for SAT attack time. Additionally, we utilize the HSPICE circuit simulator to validate the MRAM-based LUT's functionality, employing 45nm CMOS technology and the STT-MRAM model described in [65]. You can find the MRAM-LUT HSPICE circuit simulation waveform in Figure 8. The EDA tool flow is technology node agnostic and can be applied to any process library defined in the config file for an automated and optimized secure design flow. We utilized the 45nm PDK library for HSpice simulations, demonstrating the resilience of MRAM-based LUTs to machine learning-based Power Side-Channel Analysis. However, we employed an STT-MRAM macro defined in Verilog that also uses the 32nm library for synthesis and PPA-security metric evaluation. The proposed EDA tool flow for Secure Hardware Design presents a comprehensive strategy to automate the security of Intellectual Properties (IPs) against reverse engineering and P-SCAs. This EDA flow can accommodate design netlists in either Verilog (.v) or BENCH (.bench) format, provided they are flattened netlists containing only one module. While Synopsys' Design Compiler is utilized for synthesis, alternative logic synthesis tools can be employed instead of Design Compiler. To conduct



Duration	Operation
0-3ns	Write MTJ_1
3-6ns	Write MTJ_2
6-9ns	Write MTJ_3
9-12ns	Write MTJ_4
12-15ns	Write MTJ_{SE}
15-15.5ns	Standby
16-16.5ns	Read MTJ_1
17-17.5ns	Read MTJ_2
18-18.5ns	Read MTJ_3
19-19.5ns	Read MTJ_4

Fig. 8. Simulation waveform for implementation of a 2-input XOR gate using MRAM-LUT with MTJ_{SE} being set to value of '0'.

experiments, we utilized Synopsys's SAED32nm RVT library in the typical-typical corner to assess the effectiveness of our EDA tool flow. A script is employed to insert the requisite number of lockboxes, each of a specific size, into the input netlist, adhering to predefined size and number constraints. The insertion script receives these constraints through a wrapper script, which systematically explores various combinations of number and size constraints to optimize SAT attack time, power, and area results. Meeting or surpassing the minimum SAT attack time is a prerequisite before proceeding to power and area analysis. Consequently, the tool ensures that SAT time requirements are met before delving into power and area optimizations. Furthermore, the EDA tool flow incorporates an additional verification step to affirm that the modified netlists maintain functional equivalence to the original netlists. The primary objective of RTL Design Verification is to ensure compliance with the requirement specification, especially in high-reliability or mission-critical applications. To attain the necessary standards, various techniques are employed, including Simulation, Structural Analysis, Formal Verification Methods, and Timing Analysis. In this project, we describe the verification methodology applied to the RTL netlist of modules to check functionality before and after lockbox logic locking. Verification was accomplished through Directed Stimulus verification and Universal Verification Methodology (UVM). In the case of Directed Stimulus verification, the testbench exhaustively checked all input combinations, verifying the output covering all cases. Additionally, complete functional coverage and code coverage were achieved due to testing of all input patterns using Synopsys VCS (note that similar EDA tools can be used as an alternative). Furthermore, to make the testbench more modular and reusable, a UVM testbench was created to generate tests verifying the functional equivalence before and after lockbox logic obfuscation. Power data in Figure 9 is obtained through power analysis on the synthesized netlist using Synopsys' Design Compiler (as illustrated in Figure 3). Design Compiler synthesizes the netlists, and power reports are collected from it. Notably, any Verilog netlist-compatible synthesis and power analysis tool can replace the Design Compiler in the EDA tool flow. In our methodology, we used the 45nm PTM library for HSpice simulations, validating MRAM-based LUTs' functionality and resilience against ML-based Power Side-Channel Analysis. For synthesis, Power, Performance, and Area (PPA) evaluation with respect to security metrics, we utilized an STT-MRAM macro defined in Verilog, leveraging the 32nm library, however, the tool is process node agnostic. In projects requiring a tape-out, substituting the non-volatile cell macro with the foundry-specific counterpart ensures compatibility with fabrication requirements. In Figure 9, with 25 lockboxes vs. 10, we hypothesize that a higher count enhances the optimization tool's flexibility, enabling concurrent improvements in area and power efficiency. The larger pool of lockboxes facilitates the exploration and implementation of resource-efficient configurations, potentially enhancing spatial and energy utilization in the circuit.

B. Resiliency Against SAT-Attack

The SAT attack on the obfuscated circuit employed a one-layer linear encoding process. This process replaced the original sub-CNF of the routing block with a CNF that featured a single layer of MUX controlled by a one-hot key. Further, CNF reduction was achieved using the BVA algorithm explained in [11]. This preprocessing step significantly improved the SAT solver's effectiveness in handling routing obfuscation challenges. Table III provides the time in seconds required by the SAT solver when $8 \times 8 \times 8$ lockboxes were utilized for obfuscation. The table demonstrates the lockbox's effectiveness in thwarting SAT attacks during static operational mode. With three lockboxes integrated into the circuit, the SAT solver failed to decipher the keys for all benchmark tests. Table III also indicates that AppSAT failed and terminated erroneously for all circuits when scan-enabled circuitry was activated. As demonstrated in [9] and [66], harnessing the dynamic morphing capabilities of emerging devices effectively counters SAT attacks. However, controlling device functionality using TRNG, as demonstrated in [9], limits the scope of obfuscation. Such obfuscation is only suitable for IPs capable of tolerating a certain degree of error. While we can design circuitry capable of controlling the dynamic morphing of emerging devices in real-time, attackers may shift their focus to reverse engineering this circuitry, or it could still be susceptible to removal attacks. Hence, in this approach, we employ statically programmed emerging devices that do not undergo dynamic morphing during runtime. Nonetheless, they offer superior SAT-attack resilience and do not limit the scope of obfuscation.

In cases where the emerging devices presented in [9] and [66] must be used for IPs intolerant to errors, we program them in a static version. They can then be replaced with LUTs of size two, and the key can be recovered within a few minutes. Lastly, while primitives proposed in [9] and [66] provide at most 16 logical functions, the LUT used in the lockbox can be expanded to increase the SAT-hardness of the resulting lockbox. The lockbox replaces the gates and their interconnects, and its removal does not benefit the attacker in any way. Moreover, as previously discussed, relying on a single countermeasure against a given attack is insufficient. Thus, as part of a defense-in-depth approach, the scan-enabled obfuscation method helps completely defeat the SAT attack. Since the SAT attack necessitates access to the oracle circuit, the attacker must enable the scan enable signal in

TABLE III
SAT-SOLVER TIME TAKEN (IN SECONDS) TO FIND KEY FOR C7552 BENCHMARK OBFUSCATED WITH DIFFERENT SIZES AND NUMBER OF LOCKBOX. ∼ SHOWS THAT THE INSTANCES ARE SAT-HARD

Lockboxes	Size of Lockboxes					
	2×2	$2\times2\times2$	4×4	$4\times4\times4$	8×8	$8\times8\times8$
1	0.31	1.62	0.43	4.2	5.74	158.31
2	0.35	6.833	3.83	25.19	6.33	1311.97
3	0.405	14.01	9.23	55.26	20.422	∼
4	0.55	48	28.38	161.83	180.938	∼
5	0.67	71.91	51.181	∼	316.231	∼
10	1.16	152.59	84.14	∼	∼	∼
25	34.5	229.6	149.3	∼	∼	∼
50	102.319	∼	∼	∼	∼	∼

the oracle circuit to apply the test vector and obtain the corresponding responses. Fortunately, the responses received are obfuscated by our proposed MRAM-based LUT. Consequently, this approach helps thwart all Oracle-based attacks, including the original SAT-attack, SMT-attack, CycSAT, and AppSAT.

C. Resiliency Against ScanSAT and Scan and Shift Attack

As explained in the ScanSAT technique, which specifically addresses obfuscated scan chain designs, its objective is to transform the circuit responsible for inverting the scan flip-flop responses into an SAT problem. This enables the SAT solver to identify the key responsible for these output inversions. However, it's important to note that in the lockbox, the circuitry handling the inversion process is not part of the scan chain. To reinforce this point, let's examine a single reconfigurable block that replaces the OR gate in the circuit with the 2-input MRAM-LUT and introduces the scan-enable obfuscation circuitry on top of the MRAM-LUT, as depicted in Figure 7. For Scan and Shift attacks, the key values are stored in the MTJ_{SE} and $\overline{MTJ}_{SE}$ cells, and a separate scan chain can be implemented for these cells. This dedicated scan chain allows for key shifting to configure the emerging devices, and the scan-out from this circuitry can be effectively blocked.

D. Resiliency Against Power Side-Channel Attack (P-SCA)

In the evaluation of resiliency against P-SCA, we leverage Monte Carlo (MC) simulations to rigorously assess SyM-LUT's robustness in reading and writing operations amid Process Variation (PV). The extensive simulation, encompassing 10,000 instances, considers PV effects on both the CMOS peripheral circuit and the MTJs. This analysis involves a 1% variation in MTJ dimensions, a 10% variation in threshold voltage, and a 1% variation in transistor dimensions [25]. The MC simulation confirms SyM-LUT's reliable write performance, with less than 0.0001% write errors observed across 10,000 error-free instances. Additionally, the inherent complementary states of the MTJs provide a substantial read margin, resulting in less than 0.0001% read errors induced by PV across all gates. These findings are reinforced by the comprehensive results of 10,000 error-free MC simulations. For a detailed analysis of MRAM-LUT's resiliency against ML-assisted P-SCA, refer to Table IV, which provides an explicit representation of its robustness in ML-assisted P-SCA scenarios. According to the MC simulation results, the MRAM-based LUT demonstrates reliable write performance, with less than 0.0001% write errors in 10,000 error-free MC instances. The complementary nature of the MTJs' states ensures a substantial read margin, resulting in fewer than 0.0001% read errors attributable to Process Variation, as confirmed by 10,000 error-free MC simulation outcomes. Notably, the power consumption for reading both '0' and '1' is nearly identical, a strategic feature employed to mitigate P-SCA. Table IV depicts the resiliency of the MRAM-LUT against ML-assisted P-SCA. We classify the LUT's functionality using power traces using a Deep Neural Network (DNN). Researchers have demonstrated that DNN can get over the ML-assisted P-SCAs' misalignment countermeasures. The scaled power trace vector, with values ranging from 0 to 1 for better convergence, is the input to the DNN. The architecture's output layer employs a softmax activation and a categorical cross-entropy loss function. The softmax activation provides a probability distribution for every possible function that the LUT could implement. For model training, we employ fully connected layers with the ReLu activation function and Adam optimizer. The 640,000 data traces are used for training, and 10-fold cross-validation is performed to assess the model. The DNN model accuracy was observed to be low, nearly 35% as shown in Table IV. Random Forest, as well as Logistic Regression, is used to evaluate the proposed MRAM-LUT against P-SCAs. As shown in Table IV, we can see that all three ML-assisted algorithms perform poorly on the STT-MRAM-based power-balanced LUTs. The F1 score is a metric commonly used in machine learning to assess the performance of a classification model. It combines precision and recall into a single value, providing a balanced measure of a model's accuracy. The F1 score ranges from 0 to 1, where a higher value indicates better model performance. The F1 score is then calculated as the harmonic mean of precision and recall:

TABLE IV
RESILIENCY OF THE PROPOSED MRAM-BASED LUT AGAINST ML-ASSISTED P-SCAS

Algorithm used	Accuracy	F1-Score
Log. Regression	29.48%	0.301
Random Forest	30.67%	0.312
DNN	35.84%	0.358

$$F1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

E. Optimization of Lockbox Insertion

In this section, we aim to develop an optimization model to find the best lockbox configuration that maximizes SAT-resiliency given the total power constraint ($p_{\max}$), which is set to be the same as the total power of the original circuit, and area constraint ($a_{\max}$), which is set to only allow less than 15% overhead. We define binary vector $x = \left[x[ij]\right]_{i,j}$ where $x[ij]$ denote whether lockbox with Size $S = 2^i$ and depth $D = j + 1$ has been used for any $i \in \{1, \ldots, n_S\}$ and

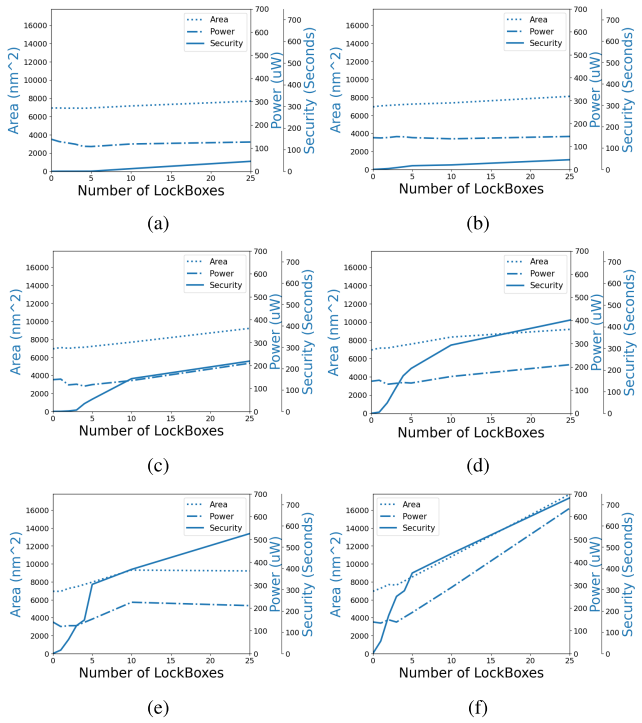


Fig. 9. AES core locked with (a) 2×2 (b) $2 \times 2 \times 2$ (c) 4×4 (d) $4 \times 4 \times 4$ (e) 8×8 (f) $8 \times 8 \times 8$ lockboxes showing power-area-security trade-off with varying lockbox size, depth, and number.

$j \in \{1, \ldots, n_D\}$. Moreover, we defined $n = n_S n_D$ and an integer variable y to denote the number of lockboxes. The decision variables are formally introduced as follows:

$$x[ij] = \begin{cases} 1 & \text{if } S = 2^i,\ D = j+1 \\ 0 & \text{o.w.} \end{cases} \tag{4}$$

$$y = \text{number of lockbox} \in \mathbb{Z}_+, \tag{5}$$

where $\mathbb{Z}_+$ denotes the set of non-negative integer numbers. Next, we define functions $f_{area}, f_{power}, f_{SAT} : \{0,1\}^n \times \mathbb{Z}_+ \to \mathbb{R}$ representing the area, power, and SAT-resiliency w.r.t. depth (D) and size (S) combination ($x[ij]$), and the number of lockboxes (y). We propose a nonlinear function to model the relation between the decision variables and the attributes (area, power, and SAT-resiliency). In particular,

$$f_{area}(x, y) = a_0^\top x + (a_1^\top x)y + (a_2^\top x)y^2 + x^\top A x, \tag{6}$$

$$f_{power}(x, y) = b_0^\top x + (b_1^\top x)y + (b_2^\top x)y^2 + x^\top B x, \tag{7}$$

$$f_{SAT}(x, y) = c_0^\top x + (c_1^\top x)y - (c_2^\top x)y^2 - x^\top C x, \tag{8}$$

where $a_\ell, b_\ell, c_\ell \in \mathbb{R}^n$ for any $\ell \in \{0, 1, 2\}$, $A, B, C \in \mathbb{S}_+^{n \times n}$ where $\mathbb{S}_+^n$ denotes the set of $n \times n$ positive semi-definite matrices. To identify the optimal lockbox combination, we formulate the optimization problem to maximize SAT-resiliency while adhering to specific power and area constraints. In particular, we have

$$\max_{\substack{x \in \{0,1\}^n \\ y \in \mathbb{Z}_+}} \quad f_{SAT}(x, y) \tag{9}$$

$$\text{s.t.} \quad f_{power}(x, y) \le p_{\max} \tag{10}$$

$$f_{area}(x, y) \le a_{\max} \tag{11}$$

$$\sum_{i,j} x_{ij} = 1. \tag{12}$$

The constraints in (10) and (11) restrict the lockbox combination to not exceed the maximum available power and area, respectively. Additionally, (12) ensures the selection of only one combination. Now, we formally introduce the optimization process to find the lockbox configuration.

Step 1: Find coefficients a_ℓ, b_ℓ, c_ℓ for any $\ell \in \{0, 1, 2\}$ and positive semi-definite matrices A, B, C in (6)-(8) using training dataset such that $a_2, b_2, c_2 \in \mathbb{R}_+^n$.

Step 2: Solve optimization problem (9).

To fulfill Step 1, exploring the structure of the function defined in (6) we observe that the coefficients can be found by fitting a piecewise-linear function. In particular, for any lockbox combination $k \in \{1, 2, \ldots, n\}$ corresponding to size $S = 2^i$ and depth $D = j+1$ we can find the coefficients $a_\ell[k]$ for $\ell \in \{0, 1, 2\}$ and the row $A_{k,:}$ by minimizing the following least-square problem

$$\min_{\beta} \|X\beta - z\|^2$$
$$\text{s.t.} \quad \beta_l \ge 0, \quad \forall l \in \{3, \ldots, n+3\},$$

where $\beta = [a_0[k], a_1[k], a_2[k], A_{k,:}]^\top \in \mathbb{R}^{n+3}$; moreover, $X \in \mathbb{R}^{m \times (n+3)}$ and $z \in \mathbb{R}^m$ denote the design matrix with m observations from the training dataset and observed values, respectively. A similar procedure can be applied to find the corresponding coefficients for functions (7) and (8). Next, given the coefficients obtained from Step 1, we can construct the optimization problem in (9). It is important to note that the constraint functions defined in (10) and (11) are convex in x and in y but not jointly convex, moreover, the objective function f_{SAT} defined in (9) is concave in x and in y but not jointly concave. Therefore, to solve (9) we use an alternating optimization method in which at each iteration one variable is fixed, and the optimization problem is solved with respect to the other variable. We manually cross-checked the output of the optimization approach against the collected data, and we could verify the correctness of the outcome. For instance, for the ISCAS benchmark circuit 'c7552', which has 3512 gates, with ($N = 2$) lockboxes of size ($S = 8$) and depth ($D = 3$) we can achieve maximum security while not violating the area and power constraints. Thus, the results of the optimization are promising, particularly for larger circuits, which demonstrates the scalability of the proposed approach.

F. Overhead Analysis

According to the results of our simulation, standby energy has been significantly reduced to 20aJ, with an average write energy consumption of 33fJ and read energy consumption of 4.6fJ. It is important to note that in LUTs, Write operations occur far less frequently than read operations. MRAM-LUT minimizes the overhead by using MTJ devices as storage components and needs 25 fewer MOS transistors, assuming a 6T-SRAM cell design is employed in an SRAM-LUT. It costs an additional 18 MOS transistors to implement the scan-enable obfuscation mechanism to the MRAM-LUT. It is important to note that reduced area overhead can be achieved by fabricating MTJs on top of baseline MOS transistors.

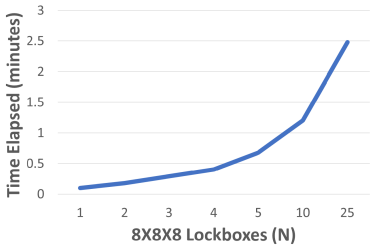


Fig. 10. Evaluation of OASIC: Time Taken for Insertion and Optimization of 8 × 8x8 lockboxes for the c7552 benchmark.

TABLE V
COMPARISON OF THE PROPOSED OASIC DESIGN FLOW WITH EXISTING HARDWARE SECURITY TECHNIQUES

Threats and Attacks		SFLL [3]	GHSE/MESO [66], [9]	InterLock [11]	CAS-Lock [6]	LUT [12]	OASIC Proposed
SAT-attack		✓	-	✓	✓	✓	✓
AppSAT-attack		✓	✓	✓	✓	✓	✓
ScanSAT-attack		-	-	-	-	✓	✓
Shift and Scan attack		-	-	-	-	✓	✓
Removal attack		✓	-	✓	✓	✓	✓
Power Side Channel attack		-	-	-	-	-	✓
Features		SFLL	GHSE/MESO	InterLock	CAS-Lock	LUT	OASIC
Dynamic morphing		✘	✓(Limited App)	✘	✘	✘	✓
EDA Tool Flow	Integration	✘	✘	✘	✘	✘	✓
	Optimization	✘	✘	✘	✘	✘	✓

G. Comparison With the State-of-the-Art

The uniqueness of this research lies not only in the incorporation of secure lockboxes or polymorphic designs resistant to side-channel attacks but also in the development of an EDA toolchain for creating robust hardware designs with multi-layered defense capabilities. This toolchain seamlessly integrates lockboxes, utilizing cutting-edge emerging devices, into a design netlist, effectively thwarting various attack strategies. The subsequent discussion offers a qualitative assessment in comparison to state-of-the-art obfuscation techniques. It is significant to note that a comprehensive and equitable comparison is challenging due to the absence of a standardized metric and model. Table V shows how our method fares against other state-of-the-art hardware security methods. When contrasted with existing secure design flows, our proposed solution demonstrates superior SAT-resiliency, particularly when compared to SFLL [3] and CASLock [6]. This enhanced resilience is attributed to our utilization of a power-balanced MRAM-LUT structure for SAT-hardness, as well as the obfuscation of oracle responses using scan-enable obfuscation. Furthermore, our approach does not suffer from the limited output corruption issues associated with one-point function-based techniques. While it's worth noting that SAT-resiliency can also be achieved through reconfigurable-based obfuscations like FullLock and InterLock [11], it's essential to acknowledge that these methods necessitate substantial effort to map gates onto the intricate structure we propose. The majority of cutting-edge obfuscation techniques mentioned in this discussion have been overcome and can no longer be deemed secure. In contrast, emerging technologies like those discussed in [9] and [66] successfully counter SAT-attacks and removal attacks, but their practicality is limited, and they do not offer resistance against P-SCAs. The application of the EDA tool flow to incorporate MRAM-LUTs introduces a multi-layer defense strategy by merging enhanced LUT-based obfuscation with SOM. Although both of these analyzed primitives lead to SAT timeouts, the proposed EDA approach incurs minimal overhead and can be employed to develop hardware that effectively thwarts SAT attacks, thanks to the integration of SOM. For the evaluation of OASIC, we have provided the time taken by OASIC to secure a sample benchmark (c7552) using 8 × 8x8 lockbox instances. As shown in Figure 10, the time taken to secure the c7552 benchmark and provide a physical design-ready netlist with 25 instances is less than 2.5 minutes. In comparison to the methodology outlined in [62], which can defend against both SAT attacks and P-SCAs, our approach provides an optimized and automated EDA tool framework for the insertion of lockboxes to maximize security while considering power and area constraints.

V. CONCLUSION

Herein, we developed an Optimized and Automated Secure IC (OASIC) design flow, which is an EDA tool flow to insert reconfigurable interconnect and logic lockboxes. The proposed OASIC combines the LUT-based logic & interconnect obfuscation with an optimized EDA tool framework to help create secure designs. The mix of logic obfuscation and interconnect obfuscation helps increase the number of DPLL calls, which the SAT-solver must trim down to find a key. By increasing the size of the lockbox, we can convincingly illustrate this effect, and a few 8 × 8 × 8 lockboxes are enough to derive SAT-hard instances. Additionally, by utilizing MRAM-based LUT, which enables the lockboxes to be altered while keeping minimal overheads compared to SRAM-based LUTs, the suggested primitive's security is strengthened. In addition to mitigating the threat of the P-SCAs, the complementing approach of MTJ reconfiguration aids in improving reliability when PV is present. The scan-enabled obfuscation circuitry is proposed to obscure the responses of the oracle circuit further. As a result, the proposed lightweight lockbox method can resist both approximate and removal attacks and has a more significant output error and higher output corruptibility. Finally, an iterative EDA tool flow is proposed to insert an appropriate number of lockboxes and secure hardware netlists from reverse engineering and P-SCAs. The proposed OASIC design flow sweeps through different configurations of STT-MRAM-based lockboxes and provides an end-design netlist that satisfies the user's SAT timeout while incurring minimum power and area overhead.

REFERENCES

[1] M. Yasin, J. J. Rajendran, O. Sinanoglu, and R. Karri, "On improving the security of logic locking," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 35, no. 9, pp. 1411–1424, Sep. 2016.

[2] P. Subramanyan, S. Ray, and S. Malik, "Evaluating the security of logic encryption algorithms," in *Proc. Int. Symp. Hardw. Orient. Secur. Trust (HOST)*, May 2015, pp. 137–143.

[3] M. Yasin, C. Zhao, and J. J. Rajendran, "SFLL-HLS: Stripped-functionality logic locking meets high-level synthesis," in *Proc. Int. Conf. Comput.-Aided Des.*, Nov. 2019, pp. 1–4.

[4] D. Sironee and P. Subramanyan, "Functional analysis attacks on logic locking," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 2514–2527, 2020.

[5] F. Yang, M. Tang, and O. Sinanoglu, "Stripped functionality logic locking with Hamming distance-based restore unit (SFLL-hd)—Unlocked," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 10, pp. 2778–2786, Oct. 2019.

[6] B. Shakya, X. Xu, M. Tehranipoor, and D. Forte, "CAS-lock: A security-corruptibility trade-off resilient logic locking scheme," *IACR Trans. Cryptograph. Hardw. Embedded Syst.*, vol. 2020, no. 1, pp. 175–202, 2020.

[7] K. Z. Azar, H. M. Kamali, H. Homayoun, and A. Sasan, "SMT attack: Next generation attack on obfuscated circuits with capabilities and performance beyond the SAT attacks," *Trans. Cryptograph. Hardw. Embedded Syst.*, vol. 2019, no. 1, pp. 97–122, 2019.

[8] G. Kolhe, P. D. S. Manoj, S. Rafatirad, H. Mahmoodi, A. Sasan, and H. Homayoun, "On custom LUT-based obfuscation," in *Proc. Great Lakes Symp. (VLSI)*, 2019, pp. 477–482.

[9] N. Rangarajan, S. Patnaik, J. Knechtel, R. Karri, O. Sinanoglu, and S. Rakheja, "Opening the doors to dynamic camouflaging: Harnessing the power of polymorphic devices," *IEEE Trans. Emerg. Topics Comput.*, vol. 10, no. 1, pp. 137–156, Jan. 2022.

[10] H. M. Kamali, K. Z. Azar, H. Homayoun, and A. Sasan, "Full-lock: Hard distributions of SAT instances for obfuscating circuits using fully configurable logic and routing blocks," in *Proc. Des. Autom. Conf.*, 2019, pp. 1–6.

[11] H. M. Kamali, K. Z. Azar, H. Homayoun, and A. Sasan, "InterLock: An intercorrelated logic and routing locking," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des. (ICCAD)*, Nov. 2020, pp. 1–9.

[12] G. Kolhe et al., "Security and complexity analysis of LUT-based obfuscation: From blueprint to reality," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Westminster, CO, USA, Nov. 2019, pp. 1–8.

[13] H. M. Kamali, K. Z. Azar, F. Farahmandi, and M. Tehranipoor, "Advances in logic locking: Past, present, and prospects," Cryptology ePrint Archive, 2022. [Online]. Available: https://eprint.iacr.org/2022/260

[14] J. A. Roy, F. Koushanfar, and I. L. Markov, "EPIC: Ending piracy of integrated circuits," in *Proc. Conf. Des. Automat. Test Europe*, 2008, pp. 1069–1074.

[15] J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri, "Security analysis of logic obfuscation," in *Proc. ACM/IEEE Des. Automat. Conf.*, Jun. 2012, pp. 83–89.

[16] J. Rajendran et al., "Fault analysis-based logic encryption," *IEEE Trans. Comput.*, vol. 64, no. 2, pp. 410–424, Feb. 2015.

[17] M. Yasin, J. Rajendran, and O. Sinanoglu, "Pre-SAT logic locking," in *Trustworthy Hardware Design: Combinational Logic Locking Techniques*. Cham, Switzerland: Springer, 2020. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-15334-2_3

[18] S. Dupuis and M.-L. Flottes, "Logic locking: A survey of proposed methods and evaluation metrics," *J. Electron. Test.*, vol. 35, no. 3, pp. 273–291, Jun. 2019.

[19] J. Rajendran, M. Sam, O. Sinanoglu, and R. Karri, "Security analysis of integrated circuit camouflaging," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2013, pp. 709–720.

[20] M. Yasin, B. Mazumdar, J. J. V. Rajendran, and O. Sinanoglu, "SARLock: SAT attack resistant logic locking," in *Proc. IEEE Int. Symp. Hardw. Orient. Security Trust*, May 2016, pp. 236–241.

[21] Y. Xie and A. Srivastava, "Anti-SAT: Mitigating SAT attack on logic locking," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 38, no. 2, pp. 199–207, Feb. 2019.

[22] Y. Xie and A. Srivastava, "Delay locking: Security enhancement of logic locking against IC counterfeiting and overproduction," in *Proc. 54th ACM/EDAC/IEEE Des. Autom. Conf. (DAC)*, Jun. 2017, pp. 1–6.

[23] K. Shamsi, M. Li, T. Meade, Z. Zhao, D. Z. Pan, and Y. Jin, "Cyclic obfuscation for creating SAT-unresolvable circuits," in *Proc. Great Lakes Symp. (VLSI)*, May 2017, pp. 173–178.

[24] A. Rezaei, Y. Li, Y. Shen, S. Kong, and H. Zhou, "CycSAT-unresolvable cyclic logic encryption using unreachable states," in *Proc. 24th Asia South Pacific Design Autom. Conf.*, Jan. 2019, pp. 358–363.

[25] S. Salehi, R. Zand, and R. F. DeMara, "Clockless spin-based look-up tables with wide read margin," in *Proc. Great Lakes Symp. VLSI*, May 2019, pp. 363–366.

[26] I. Kuon, R. Tessier, and J. Rose, "FPGA architecture: Survey and challenges," *Found. Trends Electron. Design Autom.*, vol. 2, no. 2, pp. 135–253, 2007.

[27] R. Zand, A. Roohi, S. Salehi, and R. F. DeMara, "Scalable adaptive spintronic reconfigurable logic using area-matched MTJ design," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 63, no. 7, pp. 678–682, Jul. 2016.

[28] J. Yang et al., "Exploiting spin-orbit torque devices as reconfigurable logic for circuit obfuscation," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 38, no. 1, pp. 57–69, Jan. 2019.

[29] X. Cui, X. Li, M. Zhang, and X. Cui, "Design of the RRAM-based polymorphic look-up table scheme," *IEEE J. Electron Devices Soc.*, vol. 7, pp. 949–953, 2019.

[30] S. Salehi, D. Fan, and R. F. Demara, "Survey of STT-MRAM cell design strategies: Taxonomy and sense amplifier tradeoffs for resiliency," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 13, no. 3, pp. 1–16, Jul. 2017.

[31] K. Chen, J. Han, and F. Lombardi, "On the restore operation in MTJ-based nonvolatile SRAM cells," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 23, no. 11, pp. 2695–2699, Nov. 2015.

[32] R. Zand, A. Roohi, D. Fan, and R. F. DeMara, "Energy-efficient nonvolatile reconfigurable logic using spin Hall effect-based lookup tables," *IEEE Trans. Nanotechnol.*, vol. 16, no. 1, pp. 32–43, Jan. 2017.

[33] J. Knechtel et al., "Towards secure composition of integrated circuits and electronic systems: On the role of EDA," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2020, pp. 508–513.

[34] C. Pilato, K. Wu, S. Garg, R. Karri, and F. Regazzoni, "TaintHLS: High-level synthesis for dynamic information flow tracking," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 38, no. 5, pp. 798–808, May 2019.

[35] B. Karp, M. Gay, O. Keren, and I. Polian, "Security-oriented code-based architectures for mitigating fault attacks," in *Proc. Conf. Design Circuits Integr. Syst. (DCIS)*, Nov. 2018, pp. 1–6.

[36] Y. Alkabani and F. Koushanfar, "Active hardware metering for intellectual property protection and security," in *Proc. USENIX Secur. Symp.*, vol. 20, 2007, pp. 1–20.

[37] K. Xiao and M. Tehranipoor, "BISA: Built-in self-authentication for preventing hardware trojan insertion," in *Proc. IEEE Int. Symp. Hardware-Oriented Secur. Trust (HOST)*, Jun. 2013, pp. 45–50.

[38] K. Tiri and I. Verbauwhede, "A digital design flow for secure integrated circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 25, no. 7, pp. 1197–1208, Jul. 2006.

[39] J. Breier, X. Hou, and S. Bhasin, *Automated Methods in Cryptographic Fault Analysis*. Cham, Switzerland: Springer, 2019.

[40] T. F. Wu, K. Ganesan, Y. A. Hu, H.-S. P. Wong, S. Wong, and S. Mitra, "TPAD: Hardware trojan prevention and detection for trusted integrated circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 35, no. 4, pp. 521–534, Apr. 2016.

[41] G. T. Becker et al., "Test vector leakage assessment (TVLA) methodology in practice," in *Proc. Int. Cryptograph. Module Conf.*, 2013.

[42] G. D. Natale, E. Ioana Vatajelu, K. S. Kannan, and L. Anghel, "Hidden-delay-fault sensor for test, reliability and security," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2019, pp. 316–319.

[43] C. McCants, "Trusted integrated chips (TIC) program," *Intell. Adv. Res. Projects Activity*, 2016.

[44] X. Zhang, A. Ferraiuolo, and M. Tehranipoor, "Detection of trojans using a combined ring oscillator network and off-chip transient power analysis," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 9, no. 3, pp. 1–20, Oct. 2013.

[45] M. R. Fadiheh, D. Stoffel, C. Barrett, S. Mitra, and W. Kunz, "Processor hardware security vulnerabilities and their detection by unique program execution checking," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2019, pp. 994–999.

[46] G. Fey, A. Sulflow, S. Frehse, and R. Drechsler, "Effective robustness analysis using bounded model checking techniques," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 30, no. 8, pp. 1239–1252, Aug. 2011.

[47] E. Love, Y. Jin, and Y. Makris, "Proof-carrying hardware intellectual property: A pathway to trusted module acquisition," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 1, pp. 25–40, Feb. 2012.

[48] J. Jiang, M. Aparicio, M. Comte, F. Azaïs, M. Renovell, and I. Polian, "MIRID: Mixed-mode IR-drop induced delay simulator," in *Proc. 22nd Asian Test Symp.*, Nov. 2013, pp. 177–182.

[49] J. Dutertre et al., "Laser fault injection at the CMOS 28 nm technology node: An analysis of the fault model," in *Proc. Workshop Fault Diagnosis Tolerance Cryptogr. (FDTC)*, Sep. 2018, pp. 1–6.

[50] Y. Jin and Y. Makris, "Hardware trojan detection using path delay fingerprint," in *Proc. IEEE Int. Workshop Hardware-Oriented Secur. Trust*, Jun. 2008, pp. 51–57.

[51] B. Yang, K. Wu, and R. Karri, "Secure scan: A design-for-test architecture for crypto chips," in *Proc. 42nd Design Autom. Conf.*, Jun. 2005, pp. 135–140.

[52] R. S. Chakraborty, F. Wolff, S. Paul, C. Papachristou, and S. Bhunia, "MERO: A statistical approach for hardware Trojan detection," in *Proc. Int. Workshop Cryptograph. Hardw. Embedded Syst.* Cham, Switzerland: Springer, 2009, pp. 396–410.

[53] S. Yu, R. Yasaei, Q. Zhou, T. Nguyen, and M. A. Al Faruque, "HW2VEC: A graph learning tool for automating hardware security," in *Proc. IEEE Int. Symp. Hardw. Oriented Secur. Trust (HOST)*, Dec. 2021, pp. 13–23.

[54] F. Koushanfar et al., "Can EDA combat the rise of electronic counterfeiting?" in *Proc. DAC Design Autom. Conf.*, Jun. 2012, pp. 133–138.

[55] B. Tan et al., "Invited: Independent verification and validation of security-aware EDA tools and IP," in *Proc. 58th ACM/IEEE Design Autom. Conf. (DAC)*, Dec. 2021, pp. 1299–1302.

[56] P. Slpsk, P. K. Vairam, C. Rebeiro, and V. Kamakoti, "Karna: A gate-sizing based security aware EDA flow for improved power side-channel attack protection," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Nov. 2019, pp. 1–8.

[57] A. G. Bayrak, N. Velickovic, F. Regazzoni, D. Novo, P. Brisk, and P. Ienne, "An EDA-friendly protection scheme against side-channel attacks," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2013, pp. 410–415.

[58] K. Tiri and I. Verbauwhede, "A VLSI design flow for secure side-channel attack resistant ICs," in *Proc. Design, Autom. Test Eur.*, Mar. 2005, pp. 58–63.

[59] K. Fazekas, "On SAT-based solution methods for computational problems," Ph. D. dissertation, Inst. Formal Models Verification, Johannes Kepler Univ. Linz, 2020.

[60] A. Biere, "Cadical, lingeling, plingeling, treengeling and yalsat entering the sat competition 2018," in *Proc. SAT Competition*, vol. 14, 2017, pp. 316–336.

[61] A. Fleury and M. Heisinger, "Cadical, kissat, paracooba, plingeling and treengeling entering the sat competition 2020," in *Proc. SAT Competition*, 2020, p. 50.

[62] G. Kolhe et al., "LOCK&ROLL: Deep-learning power side-channel attack mitigation using emerging reconfigurable devices and logic locking," in *Proc. 59th ACM/IEEE Design Autom. Conf.*, Jul. 2022, pp. 85–90.

[63] Y. Hu, Y. Zhang, K. Yang, D. Chen, P. A. Beerel, and P. Nuzzo, "FunSAT: Functional corruptibility-guided SAT-based attack on sequential logic encryption," in *Proc. IEEE Int. Symp. Hardw. Oriented Secur. Trust (HOST)*, Dec. 2021, pp. 281–291.

[64] G. Kolhe et al., "Securing hardware via dynamic obfuscation utilizing reconfigurable interconnect and logic blocks," in *Proc. 58th ACM/IEEE Design Autom. Conf. (DAC)*, Dec. 2021, pp. 229–234.

[65] J. Kim, A. Chen, B. Behin-Aein, S. Kumar, J.-P. Wang, and C. H. Kim, "A technology-agnostic MTJ SPICE model with user-defined dimensions for STT-MRAM scalability studies," in *Proc. IEEE Custom Integr. Circuits Conf. (CICC)*, Sep. 2015, pp. 1–4.

[66] S. Patnaik, N. Rangarajan, J. Knechtel, O. Sinanoglu, and S. Rakheja, "Advancing hardware security using polymorphic and stochastic spin-Hall effect devices," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2018, pp. 97–102.

Kevin Immanuel Gubbi (Student Member, IEEE) received the M.S. degree in embedded electrical and computer systems from San Francisco State University in 2021. He is currently pursuing the Ph.D. degree with the ASEEC Laboratory, UC Davis. He is also a Graduate Student Researcher with CHEST. He received the ACM DAC 2023 Young Fellow Grant. His research spans computer system security, spin-based devices, reconfigurable architectures, low-power VLSI circuits, neuromorphic AI hardware, and IoT security.

Banafsheh Saber Latibari (Student Member, IEEE) received the B.Sc. degree in computer engineering from the K. N. Toosi University of Technology in 2014 and the M.Sc. degree in computer architecture from the Sharif University of Technology in 2017. She is currently pursuing the Ph.D. degree with the Electrical and Computer Engineering Department, University of California at Davis. From 2019 to 2021, she was a Graduate Research Assistant with the GATE Laboratory, George Mason University. Her research focuses on machine learning, embedded system security, and computer architecture.

Muhtasim Alam Chowdhury received the B.S. degree in electrical and electronics engineering from North South University in 2019. He is currently pursuing the Ph.D. degree with the Electrical and Computer Engineering Department, The University of Arizona. His research interests are IoT hardware supply chain security, hardware security, in-memory computing, spin-based devices, AI applications for security, and robotics.

Afrooz Jalilzadeh received the bachelor's degree in mathematics from the University of Tehran and the Ph.D. degree in industrial engineering and operations research from The Pennsylvania State University. She is an Assistant Professor with the Department of Systems and Industrial Engineering, The University of Arizona. Her research interests include designing, analyzing, and implementing stochastic approximation methods for solving stochastic optimization and variational inequality problems.

Erfan Yazdandoost Hamedani received the B.S. degree in mathematics and applications from the University of Tehran, Tehran, Iran, in 2015, and the Ph.D. degree in industrial engineering and operation research with a minor in statistics from The Pennsylvania State University in August 2020. He is an Assistant Professor with the Department of Systems and Industrial Engineering, The University of Arizona. His research interests include distributed optimization, large-scale saddle point problems, and bi-level optimization in machine learning.

Setareh Rafatirad received the M.Sc. and Ph.D. degrees in computer science from the University of California at Irvine, in 2009 and 2012, respectively. She is an Associate Professor with the Department of Computer Science, University of California at Davis. Prior to that, she was an Associate Term Professor with the Department of Information Sciences and Technology, George Mason University. Her research interests include applied machine learning, IoT security, and natural language processing.

Avesta Sasan received the B.Sc. degree in computer engineering and the M.Sc. and Ph.D. degrees in electrical and computer engineering (ECE) from the University of California at Irvine (UCI) in 2005, 2006, and 2010, respectively. In 2010, he joined the Office of CTO in Broadcom Company, working on the physical design and implementation of ARM processors, as a Physical Designer, the Timing Signoff Specialist, and the lead of signal and power integrity signoff in this team. He joined George Mason University in 2016 as an Associate Professor with the Department of ECE, where he is also the Associate Chair for Research. In 2021, he joined the Faculty at the ECE Department, University of California at Davis. His research spans hardware security, machine learning, neuromorphic computing, low-power design, approximate computing, and the IoT.

Houman Homayoun received the B.S. degree in electrical engineering from the Sharif University of Technology in 2003, the M.S. degree in computer engineering from the University of Victoria in 2005, and the Ph.D. degree in computer science from the University of California at Irvine in 2010. He is currently a Professor with the Department of Electrical and Computer Engineering (ECE), University of California at Davis (UC Davis). Prior to that, he was an Associate Professor with the Department of ECE, George Mason University. From 2010 to 2012, he spent two years with the University of California at San Diego, as an NSF Computing Innovation Fellow awarded by CRA-CCC. He carried out research in hardware security and trust, data-intensive computing, and heterogeneous computing.

Soheil Salehi (Member, IEEE) received the Ph.D. and M.S. degrees in ECE from the University of Central Florida (UCF) in 2016 and 2020, respectively. He is an Assistant Professor with the Electrical and Computer Engineering (ECE) Department, The University of Arizona (UofA). Prior to joining the UofA, he was an NSF-Sponsored Computing Innovation Fellow with the Accelerated, Secure, and Energy-Efficient Computing Laboratory and the Center for Hardware and Embedded Systems Security and Trust, University of California at Davis (UC Davis). He has expertise in the areas of hardware security, IoT supply-chain security, applied ML for secure hardware design.