Rethinking Watermark: Providing Proof of IP Ownership in Modern SoCs

N. Nalla Anandakumar, M. Sazadur Rahman, Mridha Md Mashahedur Rahman, Rasheed Kibria, Upoma Das, Farimah Farahmandi, Fahim Rahman, and Mark M. Tehranipoor
Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL, USA
{nnachimuthu, mohammad.rahman, mrahman1, rasheed.kibria, upoma.das}@ufl.edu
{farimah, fahimrahman, tehranipoor}@ece.ufl.edu

***Abstract*—Intellectual property (IP) cores are essential to creating modern system-on-chips (SoCs). Protecting the IPs deployed in modern SoCs has become more difficult as the IP houses have been established across the globe over the past three decades. The threat posed by IP piracy and overuse has been a topic of research for the past decade or so and has led to creation of a field called watermarking. IP watermarking aims of detecting unauthorized IP usage by embedding excess, non-functional circuitry into the SoC. Unfortunately, prior work has been built upon assumptions that cannot be met within the modern SoC design and verification processes. In this paper, we first provide an extensive overview of the current state-of-the-art IP watermarking. Then, we challenge these dated assumptions and propose a new path for future effective IP watermarking approaches suitable for today's complex SoCs in which IPs are deeply embedded.**

***Keywords*-IP Watermarking, System-on-Chip (SoC), Intellectual Property (IP) cores.**

I. Introduction

The advancement of processing technology has led to rapid growth in integrated circuit (IC) design complexity. There are now more than tens of billions of transistors integrated on a single chip, and the upward trend is expected to increase in the coming years. The continuing device scaling creates a design productivity gap between IC design (which typically increased at about 20% per year) and IC fabrication (which increased at about 40% per year) [1], and this gap is becoming wider. IP reuse emerged as the most important design technology innovation in the past two decades to close this productivity gap and expedite the development of modern SoC products.

The concept of reuse takes advantage of the pre-designed and pre-verified functional blocks, called IP cores. IP reuse and exchange usually take the form of soft, firm, or hard [2, 3] IPs. Soft IPs, delivered in the form of synthesizable hardware description language (HDL) codes. Hard IPs are commonly provided in the form of GDSII (Graphical Database System II) files representation of a fully placed and routed design. Firm IPs typically come in the form of entirely placed netlists. These IP cores can be generic, proprietary, or open-source. Generally, open-source functions include processor units, ADCs/DACs, RAM, UARTs, ethernet controllers, power modules, media access controllers (MACs), etc. According to a recent report published by the ESD alliance, the semiconductor IP sector has become the largest segment in the electronic design automation (EDA) industry with total revenue of $1.0529 billion in the 4th quarter of 2020, which is more than 34% of worldwide EDA industry revenue [4]. The flexibility of reusable IPs expedites the creation of SoC products and brings a lot of potential profits to the IP providers. Systems-on-chips (SoCs) combine dozens of intellectual property (IP) cores licensed from different vendors [5, 6]. The reuse of IP cores comes with a risk of IP piracy and overuse. Problems could be in various forms, such as claiming someone else's IP as your own or reselling it, not giving an IP designer credit where it is due and open-source IPs being used for commercial purposes. Therefore, a need exists for provable identification of an IP core within a suspect design. Each IP must have a unique identification that represents the version, ownership rights, design information, and provider. Moreover, the identification can also provide ownership proof, designer information, and IP tracing. The ability to prove the identity of IP is increasing in importance [2, 3, 7–10]. After the IP has been incorporated into a SoC and packaged, designers can still check the identity of the IP.

Researchers have proposed several possible protection methods against illegal IP usage [2, 7, 8, 10–18]. One potential solution for claiming ownership of an IP core is to use watermarks. Watermarking, the process of marking an asset with a known structure, has been proposed to detect IP theft and overuse. Watermarking in hardware IPs is the mechanism of embedding a signature (or a unique code) into an IP core without altering the original functionality of the design. The ownership of the IP can be later verified when the watermark is extracted. The IP watermarking steps are illustrated in Fig. 1. Typically, an IP core developer embeds a watermark inside the core and sells the protected IP core in the market. To reduce time-to-market and design complexity, SoC designers use IPs from various IP owners in their designs. The SoCs are fabricated in foundries located around the globe. So, the IP owners have very little control over any illegal usage of their IPs in an SoC [13, 19]. If the IP owner encounters such an issue, they can retain a suspected chip from the market and extract the watermark using their known parameters. If the extracted watermark is matched with the originally embedded watermark into the IP, then the IP owner can easily prove that his/her IP core is illegally used in the SoC. An ideal watermark mechanism is embedded and verified easily and yet does not suffer from high overhead and is resistant to attacks [2].

It is important to note that watermarking is a passive

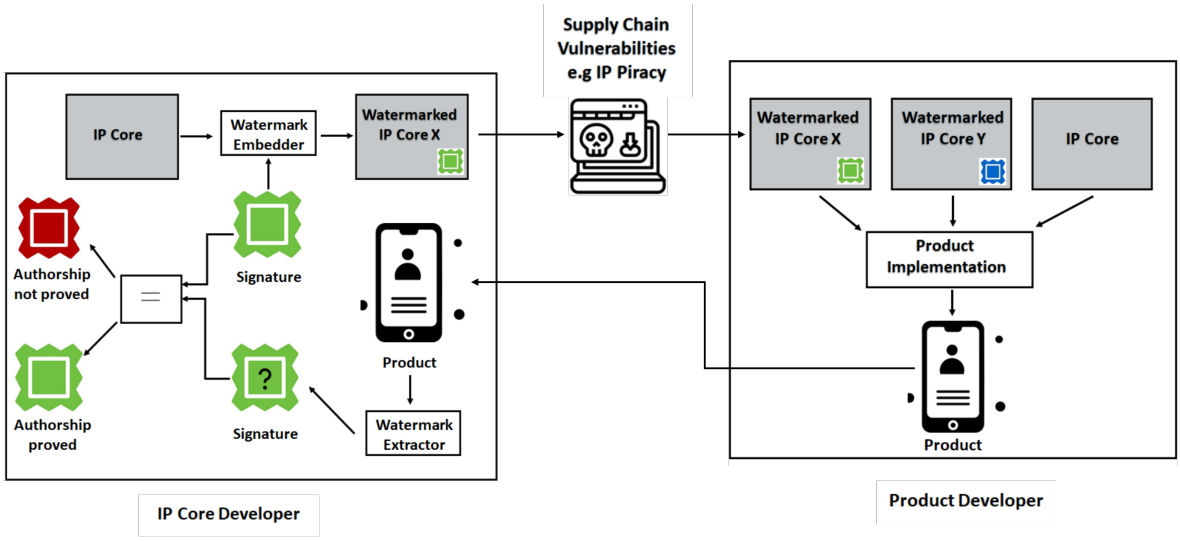

Fig. 1: IP watermarking embedding and verification.

technique that cannot prevent IP infringement, piracy, or overproduction of ICs. It is only suitable for proving intellectual property use. In the past two decades, the semiconductor industry has witnessed legal battles among technological giants in response to intellectual property theft, piracy, and digital rights management. In 2003, Cisco Systems Inc. filed a lawsuit against network equipment manufacturer Huawei Technologies and its subsidiaries, claiming illegitimate copying of its IPs, including source codes, software, documentation, and copyrighted materials [20]. A tough legal battle took place between Intel Corporation and Micron Technology over 3D memory technology development [21]. Semiconductor startup CNEX Labs sued Huawei Technologies for stealing their IP [22]. These copyright infringement activities are not uncommon, and the worst part is that these incidents stay undiscovered in most cases.

Globalization in the semiconductor supply chain granted third parties access to advanced technologies manufactured for critical applications. Outsourcing of semiconductor design tasks, therefore, introduces vulnerabilities into the supply chain that adversaries can exploit. From a global perspective, where IP protection laws vary vastly from one country to another, IP protection and authorship can no longer be limited to passive methods such as patents, copyrights and watermarks that merely deter these threats. Furthermore, the existing watermarking techniques [2, 7, 8, 14, 15] are only limited to the specific IP blocks themselves, which are not capable of providing SoC level detection of adversarial footprints [23]. Therefore, there is a critical need to develop a new and innovative watermarking techniques to prevent IP piracy/theft in modern complex SoC environment.

A number of watermarking methods have been reported in the literature for providing proof of IP ownership [24–39]. Researchers have investigated the IP identification/detection methods at various design levels such as system design, behaviour design, logic design and physical design. However, the existing techniques do not explicitly discuss the watermark detection or extraction process in many practical scenarios to prove the ownership of the IP core in complex SoCs. For example, how do the IP designer detect and prove IP overuse in a suspect SoC after the chip has been packaged when the IP designer has access to the chip but not the IP?. Therefore, the assumptions to leverage existing approaches are incompatible with the modern threat landscape and must be revisited. In this article, we first provide a high-level overview of the state-of-the-art IP watermarking techniques. Then, we provide guidance to shape future research directions to maximize the applicability and impact of watermarking techniques in modern SoCs.

The rest of this paper is structured as follows. Section II discusses the existing watermarking techniques and their pros and cons. General requirements for IP watermarking and attack analysis are briefly discussed in Section III. The threat models for IP piracy and overuse in a modern SoC design is discussed in Section IV. Section V discusses research roadmap on hardware IP watermarking. Finally, the paper is concluded in Section VI.

II. Existing IP Watermarking Techniques

IP watermarking approaches can be roughly classified into five groups: i) Constraint-based watermarking, ii) Digital signal processing (DSP)-based watermarking, iii) Finite state machine (FSM)-based watermarking, iv) Test structure-based watermarking and v) Side channel-based watermarking, as shown in Fig. 2

A. Constraint-based Watermarking

Several NP-hard optimization problems are used in every phase of the IC design process (i.e., system synthesis, behavioral synthesis, logic synthesis, and physical synthesis). A detailed enumeration of all possible solutions would be impossible due to the complexity of the problems. Heuristic algorithms with some design constraints are used to search for near-optimal or quasi-optimal solutions. This is where constraint-based IP watermarking techniques come into play. Kahng et al. [40] proposed one such approach that is applicable at different stages of the design process. In this approach,

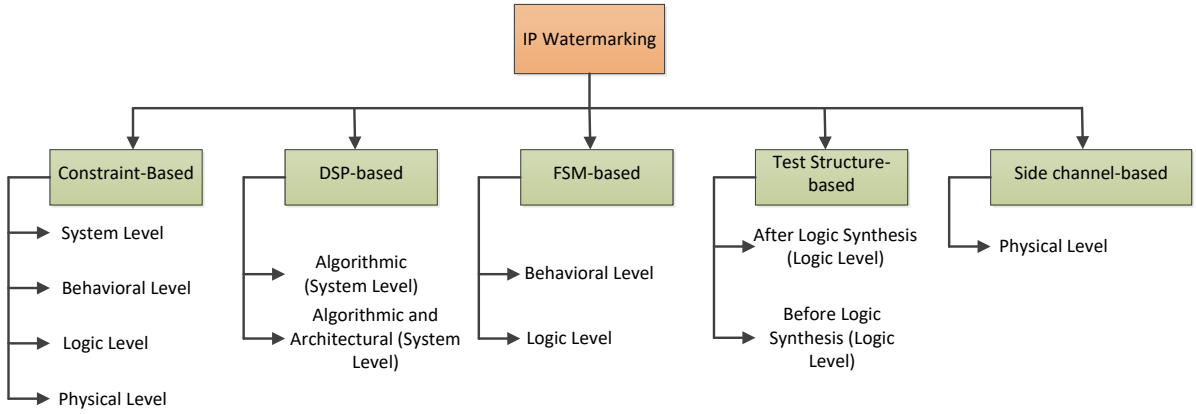

Fig. 2: IP watermarking strategies.

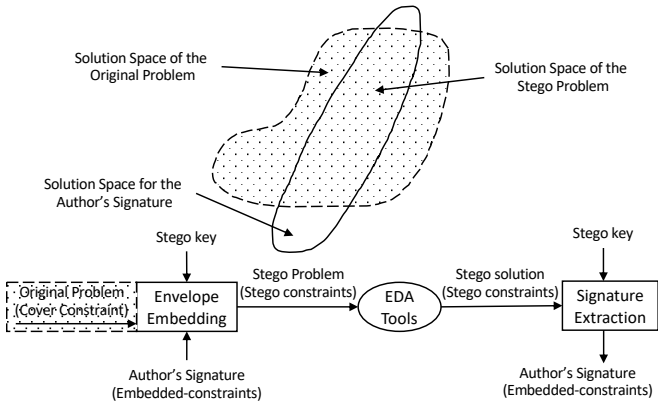

Fig. 3: Basic concept of the constraint-based watermarking approach [41].

NP-hard problems are solved using EDA tools available at that stage. A generic optimizer is used to solve constraint-satisfaction problems (CSP) [40]. As a result, the watermarked design can be derived from the algorithmic constraints added to such a solution.

Fig. 3 shows the basic conceptual overview of how constraint-based watermarking works. The heuristic algorithm takes the original design specifications. It covers constraints as inputs for design space exploration to select a good solution as the original IP core from an ample solution space. The encrypted authorship message is first converted into a set of embedded constraints. These constraints are then used as additional inputs to the envelope embedding unit. The embedded constraints derived from the author's signature are blended with the cover constraints to generate the stego constraints. At this point, the original problem becomes the stego problem which is fed to the EDA tools to find a near-optimal solution. The final result will be a watermarked IP core that satisfies both the original and the embedded constraints. Therefore, this provides a probabilistic proof of authorship, which is expressed in Equation 1 [40].

$$P_c = P(x \leq b) = \sum_{i=0}^{b} [(C!/(C-i)!] \cdot i!) \cdot p^{C-i} \cdot (1-p)^i] \quad (1)$$

Here, p = probability of satisfying one random constraint by coincidence, P_c = proof of authorship, b = number of constraints unsatisfied and C = number of imposed constraints. P_c should be kept as low as possible when designing constraint-based watermarking strategies so that only the author can satisfy the final solution to the IP design C watermark problem. In addition, the average P_c could be in the range of 10^{-30} [42], making it impossible for anyone (in terms of time and effort) to copy the watermark. Constraint-based watermarking, therefore, allows IP digital rights protection, where it is computationally infeasible to guess the correct solution without a signature, key, etc. Following are the components of a generic constraint-based watermarking procedure as outlined by Kahng et al. [40].

- Constraint-based watermarking should be a complex optimization problem where achieving an acceptable solution or enumerating enough acceptable solutions grows exponentially with the input size.
- Intellectual property can be defined as the solution to the well-defined optimization problem.
- An optimization problem can be solved by using existing algorithms and/or off-the-shelf software.
- Security requirements are considered as that are similar to those known from currency watermarking.

To solve the stego problem, the heuristic algorithm of the EDA tools uses the stego constraints rather than the design constraints. The constraints can either be incorporated into the optimizer's inputs (i.e., pre-processing) or applied to the optimizer's output (i.e., post-processing). Pre-processing appears to be the most popular approach. Since its first introduction in [40, 43], many techniques have been proposed for the constraint-based IP watermarking at different abstraction levels, which range from the system synthesis level [24, 43] to the behavioral synthesis level [25, 26], logic synthesis level [27–30] and physical synthesis level [31, 32, 40]. In the following we will review some of the most influential proposals at each level.

1) System Synthesis Level: The nodes of the graph to be partitioned are randomly numbered using integers as illustrated

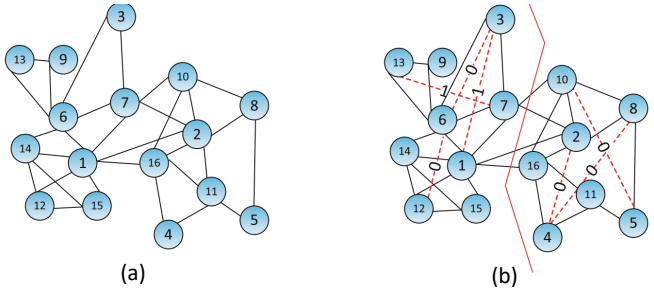

Fig. 4: (a) Graph to be partitioned with nodes indexed by integers (b) The partitioned graph with embedded watermark [43].

in Fig. 4(a). The stego constraints associated with watermarks mandate that nodes within a partition remain together. A pair of origin and terminal nodes are selected for each watermark bit. Nodes that have not been selected as the origin node are used to determine the original node. Nodes with the smallest index are selected. The terminal node is determined by the watermark bit. The terminal node is selected when the watermark bit is '1'. Alternatively, when the watermark bit is '0', the terminal node is the one with the smallest even index that has not been paired. Consider the letter 'A' with the ASCII code "1000001" as a watermark (or a part of the watermark). According to the embedding criteria described above, the pairs of nodes (1, 3), (2, 4), (3, 6), (4, 8), (5, 10), (6, 12), and (7, 13) in Fig. 4(b) should all be in the same partition. According to Fig. 4(b), one possible watermarked solution is to balance partitioning, so that node differences between two partitions are less than 20%. The graph partitioning-based watermarking described above can also be extended to a graph coloring problem. In graph coloring, the aim is to find a way to color the vertices of a graph such that no two adjacent vertices are the same color. The graph coloring problem resembles many optimization tasks in the VLSI (Very Large-Scale Integration) design flow. As an example, a cache-line code optimization problem [24] can be solved by finding a solution with a fixed number of colors, where each color represents one cache line. An additional edge can be added to the graph to represent a watermark.

2) Behavioral Synthesis Level: Task scheduling, resource assignment, transformations, resource allocation, and template mapping are all NP-hard optimization problems that are excellent for embedding the watermark. Koushanfar et al. [26] proposed watermark insertion during register allocation. Fig. 5

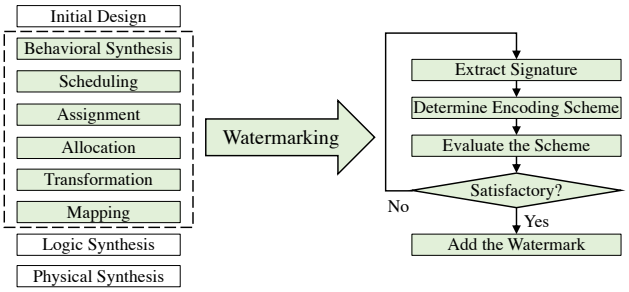

Fig. 5: A generic approach for watermark insertion at the behavioral synthesis level [26].

shows a generic approach for watermark insertion at the behavioral synthesis level. After scheduling, the intermediate output generated in one stage needs to be stored in registers and applied to the next stage to maintain the pipelining. The period between the first time a variable is generated and the last time it is used is its lifetime. If two variables exist in two different time frames, then they can share the same register. A graph coloring problem can be solved for an interval graph by allocating registers. The embedded constraints generated by the author's signature can be inserted as an additional edge in the interval graph and solved for the overall solution. Researchers also proposed a don't care condition-based solution for constraint-based watermarking insertion [25]. Occasionally, a designer may not care about the output of a truth table for specific inputs, which are termed as "don't care" conditions. IP watermarking can be accomplished with don't care conditions that force the output of IPs.

3) Logic Synthesis Level: The logic synthesis process transforms abstract design behavior into a specific implementation comprising logic gates based on the RTL HDL. Multilevel logic minimization and technology mapping are two optimization tasks performed in combinational logic synthesis. According to [28], both tasks are a suitable candidate for constraint-based watermarking. Technology mapping maps the logic network of a design to as few pre-defined library cells as possible, the complexity of which is NP-hard. Cui et al. [29] proposed an adaptive watermarking technique by modulating some closed cones as part of an original, optimized logic network (master design) for technology mapping. Various disjoint closed cones are analyzed based on their slack and sustainability. If closed cones in the critical path can be more effectively preserved upon remapping using the notion of slack sustainability, then they are qualified to host watermarks. Only qualified disjoint closed cones are selected at random, and templates constrained by the signature are remapped and embedded as the watermark. By using this parametric formulation, designers can take advantage of a design's headroom to increase the signature length or strengthen the watermarks. A similar approach has been proposed where watermarking constraints have been imposed only to a careful selection of non-critical paths based on the design specification [30]. A watermarking technique based on the Schmitt Trigger insertion at logic synthesis level has been proposed [27]. The signature is presented as a BASE64 hash-code, an MD5 hash-code and an ASCII string to create a unique sequence of watermarked bits.

4) Physical Synthesis Level: The logic synthesis process produces a gate-level netlist and circuit layout derived from the mapped gate-level netlist. Based on the mapped gate-level netlist, physical synthesis produces an optimized netlist and circuit layout. Floorplanning, clock tree synthesis, placement, scan chain reordering, routing, physical signoff are common steps of physical synthesis. In [40], several watermarking techniques based on these tasks are presented. Authorship signatures are used to determine path timing constraints. These timing constraints are replaced with "sub-path" constraints to insert watermarks. The synthesis solution may not satisfy both sub-path constraints under the original timing constraint. As

indicated by [40], there is a slight chance of satisfying both sub-path constraints when the original constraint is satisfied. A strong proof of authorship can be achieved by constraining hundreds of timing paths. Standard-cell routing-based watermarking approach at physical synthesis level has been proposed [40]. According to this approach, the watermarking constraints are based on the (per-net) costs associated with the routing resources. In other words, if the watermark bit is '1', the network will incur unusual costs if traffic is redirected in the wrong direction and/or vice versa. Thus, a watermark can be verified by examining the distribution of resources in specific nets. By constraining the placement of selected logic cells in rows with the specified row parity, the watermark can also be inserted [40]. In other words, some cells are constrained to be placed in even-index rows, while others are constrained to be placed in odd-index rows. The typical placement instance has tens of thousands of standard cells, and this method provides an authorship proof with a long watermark. Still, some cells may incur a higher routing cost due to watermarking constraints. Therefore, watermarked logic cells should be carefully chosen. Routing grid-based watermark insertion has been proposed [31] where each grid cell in the layout picture is assigned a judging parameter value, and the picture is partitioned with small grids of appropriate size. A watermarking signature is then drawn on the grided layout. Text messages or meaningful symbols may be used as signatures. The cells are scattered using some algorithm, such as the pseudo random coordinate transformation (pReT) algorithm, to generate a new distribution of the original watermarking cells. As a result, the watermarking can be spread out over the entire layout, making it difficult to tamper with the watermark by changing just a tiny part of the layout. By combining directed NBTI (Negative bias temperature instability) aging [44] and delay logic, Zheng et al. [32] presented a new approach to IC digital watermarking. Their work demonstrates that delay logic measures relative speed differences between competing logic paths due to variation in the process.

B. Digital Signal Processing Watermarking

Watermarking using digital signal processing (DSP) is introduced at the algorithmic level of the design flow in [33] and [34]. The primary goal of both techniques is to enable designers to make slight modifications to the decibel (dB) requirements of filters without compromising their operation. According to [33], the designer of a high-level digital filter needs to first encode a single character (7 bits) as a watermark. The high-level filter's design is then broken into seven segments, each of which is employed as a modulation signal for one of the bits. It entails breaking the filter into seven segments and using each portion as a carrier signal with little dB change if the bit is '1' or none if the bit is '0'.

The authors of [34] divided the problem into two parts. Watermarking has been implemented at both the algorithmic and architectural levels to improve robustness. They've used a similar approach to [33] on the algorithmic level, where seven bits are added. However, they employed a static technique at the architectural level to watermark the transpose of the finite impulse response (FIR) filter [34]. Their solution is based on employing different filter building block architectures according to the bits that need to be embedded. A pipelined structure is used to create the transposed form FIR filter, with each pipeline stage consisting of basic multiplier-adder-delay block functions. Watermarking at the architectural level is based on making circuit modifications utilizing the fundamental blocks of each pipeline step without modifying the filter's transfer function. Two of these fundamental block changes are shown in Fig. 6. Both structures are capable of encoding a two-bit code: 00, 01, and 10.

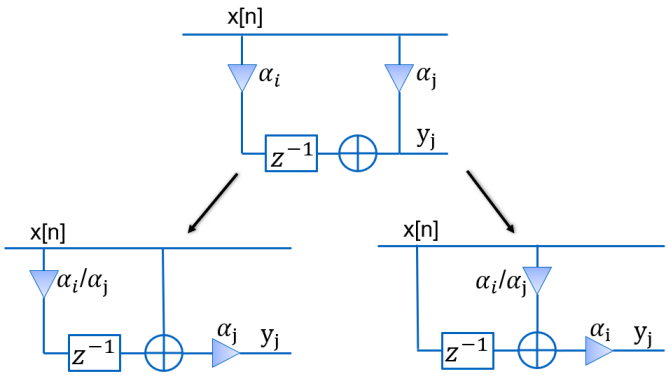

Fig. 6: Architecture level circuit transformations [34].

Both methods have a low embedding overhead and a low design overhead. However, as long as the DSP filter is not completely hidden in the design, they are immediately recognizable. Both techniques can be applied at the DSP algorithmic level, which is a relatively high level in the design flow. The authors did not take into account the robustness of their technique or any other criteria. Furthermore, the methods rely on a relatively low data rate, just one character (7 bits), making them unfeasible for application in an industrial setting. The watermark is especially susceptible to design variations at lower levels because of the low bit rate. Furthermore, such a watermark is quite vulnerable to masking attacks since even minor modifications in the filter function may readily hide or even delete the watermark.

C. FSM-based Watermarking

The FSM-based watermark is embedded at the behavioral level by introducing additional FSM states or transitions which does not interfere with the normal chip functionality. FSM based watermarking is a fairly researched area of hardware security: one reason behind the popularity of this technique is that FSMs in a design provide an easy way to extend the state space to hide watermark into it. In general, FSM-based watermarking techniques should meet the following requirements:

- Watermarking states or transitions should be well hidden into the existing state space so that the newly introduced states/transitions do not become easy target to the removal attack.
- Watermarking states or transitions must not change the original functionality of the IP.

- Newly introduced states/transitions should have reasonable impact on the original design in terms of overhead.
- It should not be affected by design optimization techniques during synthesis or physical design.

FSM watermarking techniques embed watermarking signature into the original design by modifying the state transition graph (STG). State transition graphs consist of states and transitions and thus, FSM watermarking techniques can be classified into two types: transition-based watermarking [45–47] and state-based watermarking [35, 36]. State-based watermarking schemes involve adding new states or changing state encoding. On the other hand, transition-based watermarking schemes add new transitions to the FSM or utilize unused transitions. The following sections review some of the related solutions of each type.

1) State-based FSM Watermarking: The first state-based watermarking strategy was proposed by Oliveira [35], where the watermark is inserted as a new property. The strategy involves manipulation of the STG by introducing redundancy so that it exhibits the chosen watermarking property. The user needs to choose a copyright text of arbitrary length of string. The string is then encrypted and hashed to generate the signature. The technique proposed in [35] breaks this signature into a combination of input sequence. Extra states and transitions are added into the STG in such a way that the input sequence, when applied, satisfies the property. The property itself is purely topological and does not depend on the state encoding. In this approach, modification of the STG is performed by duplicating existing STG. Next, one or more states and transitions are copied from the original STG and used to connect the previously duplicated STG to the original STG. This approach of STG modification adds large overhead to the design. Moreover, this approach is weak against the state minimization attack as it will remove all redundant states. Also, it uses a counter to detect expected input sequence. This is a weak point as identifying and removing this counter will render the watermarking scheme useless.

Lewandowski et al. [36] proposed another state-based watermarking scheme by embedding a watermark signature via state encoding. In this approach, a watermark graph is first constructed from the binary representation of the signature. The binary string is divided into blocks of a chosen size. Each node of the watermark graph will represent a block of the binary string. Then the algorithm inserts directed edge from one node to the next. These steps are repeated until entire binary string is encoded into the nodes of the watermark graph. Once the watermark graph is built, it is then embedded into the design STG. The proposed algorithm looks for isomorphism between the two STGs using a greedy heuristic. If the algorithm fails to find a sub-graph isomorphic to the watermark STG, a sub-graph which is most similar to the watermark graph is chosen and extra transitions are added to it to make it isomorphic to the watermark graph. Then, the nodes of watermark graph are mapped into the nodes of the sub-graph and the state encoding of the sub-graph is set by the state encoding of watermark graph. This approach of IP watermarking is susceptible to state recoding attack. State recoding attack can change the state encoding of the watermarked FSM and corrupt/change the watermark signature.

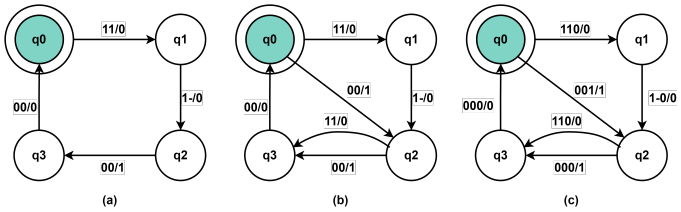

Fig. 7: FSM watermarking example from [45]: (a) original FSM, (b) adding new transitions, and (c) extending input and adding new transitions.

2) Transition-based FSM Watermarking: One of the earliest papers on FSM watermarking is [45], where the authors proposed an algorithm to extract the unused transitions in an FSM. The algorithm visits each state of the FSM to look for unused transitions and uses them for watermarking. If the algorithm fails to find any unused transition in the design, new input-output pins are inserted to expand the STG.

The state transition diagram shown in Fig 7(a) is the original FSM. The input is 2-bit and output is 1-bit. In this example, the watermark length is 2-bit and it is represented as ((00/1), (11/0)). Fig 7(b) shows the watermarked FSM without augmenting the input. If the current FSM is completely specified, then the input is extended by 1-bit and then watermarking transitions are added as shown in Fig 7(c).

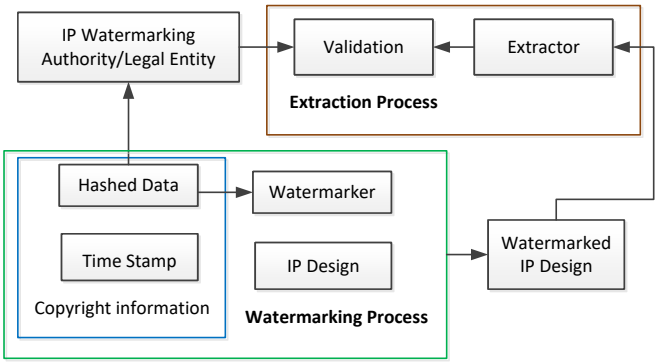

Fig. 8: Watermarking with third party keeping a time-stamped signature [47].

Utilizing existing transitions as well as unused transition for embedding watermark in an FSM was proposed in [46]. The authors break the watermarking process into three parts: generating the signature, embedding it into the FSM, and detecting the watermark. Fig 8 shows the watermarking framework with third party keeping a time-stamped signature. The technique utilizes a third party, namely watermarking authority, to generate the time-stamped watermark signature. This signature is stored in the database and reused during extraction process to provide authorship proof. For the watermark insertion, the paper [46] proposes two algorithms. The first algorithm works based on the input bits comparison. It first generates random inputs and associates watermark signature with these random inputs. Then it looks for a match between this randomly

generated input and existing input of a randomly chosen state of the STG. If there is a match between the inputs, the output of the state is compared with the signature. If the output matches, the state will be used as watermarking state. If there is no match between the outputs, a new transition is added to the state. And, if there is no match found with the inputs of the STG, the inputs are extended using an extra bit and the newly generated states are used for watermarking. This algorithm adds new input bits even if the STG is not completely specified which adds unnecessary overhead. Another algorithm works based on the mapping of signature bits into existing FSM outputs. The algorithm visits different states and tries to match signature bits with each state's output. If there is a match, that state is used as watermarking state. But if there is no match, it tries to add new transition using free state space. If there is no free state space, the algorithm extends the input similar to the other approach. Both of these algorithms perform poorly when the existing FSM has outputs of large size. In this case, probability of matching/mapping with the signature bits become far less.

In [48], the authors have proposed a watermarking scheme inspired by the stealthy nature of hardware Trojans (HTs). This technique is interesting since it provides very low overhead in the target IP and very good resiliency against the known attacks since Trojans are very difficult to detect by existing testing methods [49, 50]. In this approach, the designer of the IP can verify the watermark by loading the test vector twice through the scan-chain (which triggers the watermark) and by observing the payload output. The payload response is normal when the trigger test vector is loaded for the first time. However, when the trigger test vector is loaded for the second time, the payload bit is flipped in the response. These normal and HT-affected payload responses form the proof of ownership. However, in [48], hardware Trojans have been inserted in gate-level abstraction only. It can be enhanced to design and embed watermarking in other abstraction levels like RTL and physical design level [3, 50]. This proposition requires further investigation and research efforts to materialize properly.

The problems with the watermark signature mapping into the existing FSM output is addressed in [47]. The proposed FSM watermarking technique also uses existing transitions for watermarking. But the signature is not directly mapped into the outputs of states rather the watermark appears at certain points in the output sequence under the specific input. These specific positions are generated using a pseudo-random number generator [51, 52]. Similar to the previous approaches, if existing transitions cannot be used for watermarking, new transitions are introduced. If the FSM is completely specified, the FSM input is extended.

None of the FSM watermarking techniques discussed above discuss about the watermark extraction process when the IP is integrated deep into an SoC. In this case, the output of the watermarked FSM may not be directly observable using the primary outputs of the SoC. This problem is addressed in [53], where the authors proposed a hybrid scheme combining FSM watermarking with test-based watermarking. Additionally, the synthesis process can introduce security risks in the implemented circuit by inserting extra don't care states and transitions [54]. The IP is watermarked in two phases: adding watermark into existing FSM and re-ordering the scan cells when the IP is integrated in an SoC. The scan chain provides an easy watermark extraction process. This approach also does not consider a scenario where the IP is illegally used in a different SoC and that the test access may not be available. The attacker, in this scenario, may not provide open access to the IP or the access may be restricted which blocks the designer from proving their authorship.

D. Test Structure-based Watermarking

The core concept for testing-based watermarking techniques is embedding a watermark into a test sequence at the behavior design level. After integrating the IPs into the full SOCs, test signals have to be traceable. Using this fact, the authors combined this test sequence with the watermark generating circuit, so that any IP in the chip may be observed and tested even after the chip has been packaged [13, 55]. In the test mode, the selected IP sends output test patterns and watermark sequences. We can determine the identity of the IP provider according to the watermark sequence. Several post-fabrication watermark verification techniques based on test structures have been proposed in literature. Despite utilizing the sophisticated sequential automated test pattern generation (ATPG), scan design accomplishes the testability of sequential design by achieving the controllability and observability of the flip-flops present in the design [56]. For decades, scan chain-based testing methodology has been the backbone of design for test (DFT) in the EDA industry, and it remains as the most prevalent technique. D-type flip-flops (DFFs) are substituted by scan flip-flops to enable a design having scan chain capabilities (SFFs). SFFs are made up of a DFF, a multiplexer, and two new signals: scan-data SD and test control TC. As demonstrated in 9, SFFs can be chained by connecting the Q output of one SFF to the SD input of the next SFF. The Q-SD connection style [15] is used for this. The core under test (CUT) is switched between normal and testing mode via TC. Test data from the new primary input SI will be applied to the scan chain in the testing mode, and response data will be gathered from the new primary output SO [15].

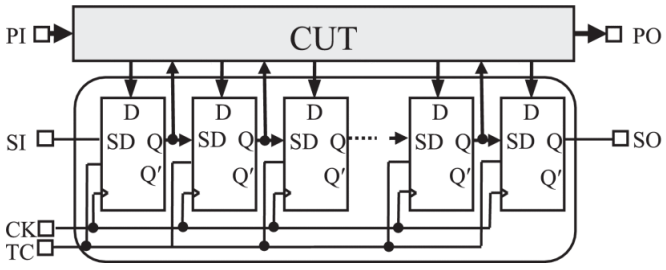

Fig. 9: Scan chain design used in DFT [15].

In [15], a scheme was proposed to insert watermark by controlling the connection style between two connected scan cells, to minimize various overhead due to watermarking. As a result, under the validation vector, the scan chain will output an output response that contains watermark information at

certain positions. This watermarking can be implemented by alternating the local routing and only negligible overhead on the test power is incurred, as shown in the middle part of Fig. 10. It can be applied to protect hard IP cores. If some hard IP cores provide some space for one to change the local routing, the watermark can be easily removed as the watermarked connection style may conflict with that determined by the test power optimization. An alternative scheme was proposed that the dummy scan cells will be inserted in scan chain when the optimized connection style conflicts with the watermarked style. Such dummy scan cells are merged in the common scan cells but play no role in testing and they also enable all connection patterns to satisfy the optimization criteria, as shown in the lower part of Fig. 10. Although some extra scan cells are incurred, it is robust against the possible removal attack. Designers can determine the amount of resilience they want from the flexible framework, reducing design overhead.

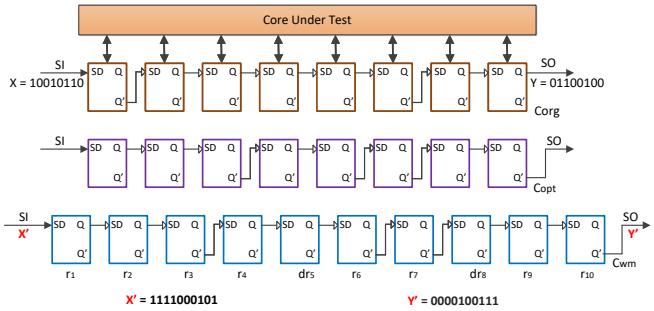

Fig. 10: DFT-based watermarking scheme [15].

An approach for watermarking at the gate-level while selecting the chain of scan registers for sequential logic test generation was suggested in [37]. The IP's watermark is transformed into user-specific constraints for selecting the scan register chain. The watermark is inserted using a set of standards for uniform circuit ordering. For this methodology, the watermark verification method is relatively straightforward and can be readily conducted by introducing a particular set of test vectors and returning a set of scan chain outputs uniquely tied to the watermark signature. However, ordering the scan cells is an NP-hard problem. In order to minimize the test power, a similar method was proposed in [38] which introduces the addition of the constraints originated by the owner's digital signature. Fault coverage and test application time remain unchanged since only the order of the scan cell get changed. In all of these proposed schemes, the watermark pattern can be observed in test mode. As a result, field verification of the ownership of the IP is possible. However, all of these schemes are susceptible to removal attacks since the watermarked test core structure is independent of the functional logic. The attacker can easily redesign the test structure for completely removing or corrupting the watermark partially.

A persuasive technique called synthesis-for-testability (SFT) was proposed in [39] for IP watermarking. The watermark is embedded as hidden constraints to the scan chain ordering mechanism very similar to [38]. However, the SFT watermarking mechanism inserts the watermark into the chain before synthesis, but the DFT based watermarking methods proposed in [37, 38] are performed after logic synthesis. The most promising fact of this technique is that test functions can be mingled with core functions, making the attempts to remove or alter the watermark a thousand times harder. If the attacker attempts to modify or remove the watermark now, there is a high possibility that design specification and optimal characteristics will get impacted. This method is quite promising, but in terms of wide acceptability, it has a long way to go since the SFT technique has not been widely adopted as a testing method in the current industry.

E. Side Channel-based Watermarking

Side-channel analysis can utilize the physical information leaked from a cryptographic device and is frequently used to retrieve the secret keys and their security issues [57–60]. The side-channel based watermark is that instead of leaking out secret information, the side-channel is engineered to contain a watermarking signal. The main idea is to use a side-channel, e.g., power consumption, to embed a watermark into an IP core. Then the verifier uses that side-channel information to extract the watermark and prove ownership. In [61], a power signature-based watermarking technique was introduced for protecting IP cores which uses the power supply pins to extract the watermark in FPGAs. It utilized the fact that supply voltage to the IP core depends on the switching of shift registers located in the IP core. The main idea is to convert the watermark into a specific voltage signal using a pattern generator, and then decode the extracted signature to prove ownership. Fig. 11 shows a watermark verification via power analysis.

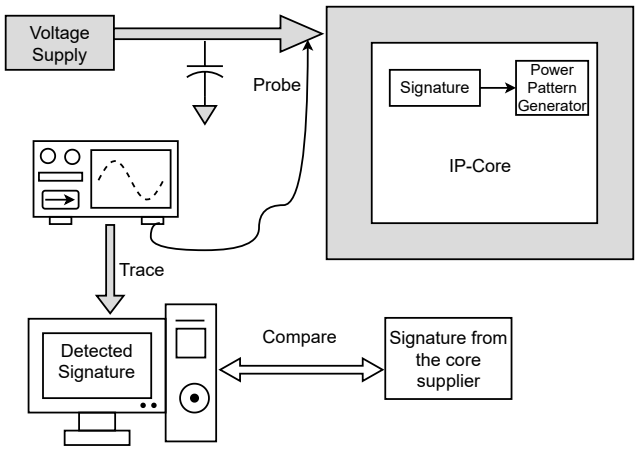

Fig. 11: Watermark verification via power analysis [61].

As the signature is hidden below the noise floor, it is difficult to perform removal attack, but this method has reliability issues while decoding the watermark signature. Also, modern SoCs contain tens of IP cores, hence, there is a high probability that in a functional chip the power consumption of the other IP cores can alter the watermark signature of the target IP. To solve this issue, [62] proposes a technique that involves

embedding ring oscillators into IP core to make it more distinguishable during parallel testing and operation. A group of ring oscillators (ROs) can cause a huge amount of switching activity which results in a significant amplification of the power consumption. Thus, an additional RO block controlled by an input vector can lead to a successful core detection [63], but it increases the area overhead of circuit, and could be susceptible to removal attacks.

In [64], authors proposes a watermark verification scheme using a correlation analysis based on the measurement of the power consumption of an IC. In addition, the authors of [65] discussed two different power-based watermark approaches such as spread spectrum-based watermark and input-modulated watermark. In the spread spectrum-based watermark a PRNG (pseudo random number generator) is used to generate a bitstream and then a leakage circuit maps the bitstream to a physical power consumption. The leakage circuit produces additional leakage or no additional leakage depending on the bitstream. In the input-modulated watermark, a leakage generating circuit is implemented in such a way that it results in power consumption when a known input pattern is applied. To implement this watermark, the verifier should have knowledge about some bits of the IP core which can vary for different measurements. Both of these techniques have low overhead, but it is possible to reverse engineer the chips and remove the watermark.

III. Assessment of Existing Watermarks

A. General Requirements for Hardware IP Watermarking

The general requirements for IP watermarking technology are almost similar to the requirements of multimedia watermarking technology. However, multimedia watermarking technology has more freedom to alter the cover media and embed the watermark [2]. Such alteration is restricted in hardware IP watermarking technology because the watermarked IP core must remain functionally correct. Based on the requirements for a hardware IP watermarking technique presented in [2], the following design criteria are briefly outlined:

- *Credibility*: The watermark should be promptly detectable for the proof of authorship. No third party (i.e., other than IP owner) should be able to claim the watermark by chance, i.e. the probability of occurrence of a particular watermark on a non watermarked IP core must be very low. The watermark insertion is half the process only, tracking and detection is the remaining important process in any watermarking technique. It is advantageous to ease the detection of watermark and enable the origin of fraudulence to be traced after possible attacks.
- *Low overhead*: The performance of the IP core in terms of area, speed and power should not be degraded/affected after embedding watermark. Both the computational time and cost needed for the watermark embedding should be kept low.
- *Resiliency (robustness)*: The robustness measures based on the strength of the hidden signature against different types of attacks. The watermark of the IP core should be impossible or difficult to remove or tamper without the complete knowledge of the software or design.
- *Invisibility*: The embedded watermark should not affect the functionality of the IP core. The watermark of the IP core should not be readily detectable by third parties. It should be well-concealed such that only the IP owner should be able to reveal it.
- *Granularity*: The watermark implementation should be distributed in a hardware design in order to protect the watermark from removal or masking.

In Table I, performance comparison of existing watermarking techniques in terms of credibility, overhead, resiliency, invisibility, granularity and resiliency to attacks (i.e., masking, forging, tampering, removal) have been illustrated.

B. Attack Analysis for IP Watermarking

In general, there exists four main types of attacks: removal, tampering, forging and reverse attacks. The shared prerequisite of these attacks is that they should not degrade the design performance. That is, an evidently deteriorated design is not what an attacker wants to steal.

1) Removal attacks: For removal attacks, the main goal of the adversary is to remove the watermark entirely. This task is normally very difficult to succeed with the shared prerequisite [66]. The complexity and feasibility of this attack depends on the type of watermarking approach as well as on how well the watermarking scheme is hidden inside the original design. This task is usually very difficult to succeed with the shared prerequisite. There is currently no known metrics which can be used to determine how deeply a watermark is hidden into a design. In general, the watermark should be tangled with the existing features or functionalities of the design in such a manner that confuses the attacker to perform removal attack.

2) Tampering attacks: Performing a successful removal attack is typically very difficult. As a result, the attacker can tamper with the watermarked design to hide the watermark's presence, known as masking attack. The minimal number of watermark bits that must be changed to result in a successful masking attack varies depending on the detection system. A probability of masking is defined as the chance that an attack would modify or erase enough information to make the watermark invisible without degrading the design's performance excessively [66].

3) Forging attacks: In forging attacks, the adversary implants his own watermark in the watermarked IP to claim his ownership to the design. The attacker may redo the watermark insertion process using his/her own signature or simply perform a ghost search for inserting the watermark. A ghost search is a way of creating an ostensibly genuine but distinct watermark depending on the detection mechanism of the targeted watermarked design and using it as the adversary's signature. The likelihood of a successful ghost search is the same as the probability of coincidence, as defined in [66].

4) Reverse engineering (RE) attacks: Another type of attack on embedded watermark of an IP is reverse engineering attack. This attack is not so common since it depends on a large number of factors. However, reverse engineering is a

TABLE I: Performance evaluation and comparison of the existing watermarking techniques.

	Abstraction Level / Type	Existing Techniques	Performance Evaluation					Resistance to Attacks			
			Credibility	Overhead	Resiliency	Invisibility	Granularity	Removal	Forging	Tampering	RE
Constraint-based Watermarking	System Synthesis	DSP Code Partioning [43]	Low	Low	High	Low	Low	Low	Low	High	Low
		Cache-line Coloring [24]	Low	Low	High	Low	Low	Low	Low	High	Low
	Behavioral Synthesis	Don't care based [25]	Low	Low	Medium	Low	High	Low	Low	Low	Low
		Register Allocation [26]	Low	Low	Low	Low	Low	Low	Low	High	Low
	Logic Synthesis	Schmitt Trigger [27]	Medium	Medium	Low	Low	Low	Low	High	High	Low
		Combinational Logic [28]	Low	High	Low	Low	Low	High	Low	Low	Low
		Tech. Mapping [29]	Low	High	Low	Low	Low	High	Low	Low	Medium
		Critical Path [30]	Medium	Medium	Low	Low	Low	High	Low	High	Medium
	Physical Synthesis	Path-time Constraint [40]	Medium	High	Low	Low	Low	Low	Low	Low	Medium
		Standard-cell Routing [40]	Low	High	Low	Low	Low	High	Low	Low	High
		Row-based Placement [40]	Low	High	Low	Low	Low	High	Low	Low	High
		Routing Grid [31]	Medium	High	Low	Low	Low	High	Low	High	High
		FPGA [32]	Medium	High	Low	Low	Low	High	Low	High	Medium
DSP-based Watermarking	Algorithmic	Character Encoding[33]	Medium	Low	Low	Low	Low	Low	Low	Medium	Medium
	Algorithmic-Architectural	Character Encoding and Static[34]	Medium	Low	Low	Low	Low	Low	Low	Medium	Medium
FSM based Watermarking	State-based	Watermark as a Property [35]	Low	High	Low	Low	Low	Low	Low	Medium	Low
		State Encoding-based [36]	Medium	High	Medium	Low	Low	Low	Low	Low	Low
	Transition-based	Unused Transition-based [45]	Medium	Medium	Medium	Low	Medium	Medium	High	Medium	Low
		Existing Transition-based [46]	Medium	Low	High	Medium	Medium	High	Low	Medium	Medium
		Robust Watermarking [47]	High	Low	High	Medium	High	High	High	Medium	High
	Hybrid	FSM and Test-based [53]	High	Low	High	Medium	High	High	High	High	Medium
Test Structure-based Watermarking	After Logic Synthesis	Scan chain reordering[37]	Medium	Low	Low	Low	Low	Low	Low	Low	Low
		Scan reordering with constraints [38]	Medium	Low	Low	Low	Low	Low	Low	Low	Low
	Before Logic Synthesis	Synthesis-for-Testability[39]	Medium	Low	High	High	Low	High	High	High	Medium
Power-based Watermarking	Physical Synthesis	Power trace analysis[61]	Medium	Low	Low	High	Low	High	Low	Low	Medium
		Viability-based power signature[62]	Medium	High	High	Low	Low	Low	Low	Low	Low
		Input Modulated[65]	Medium	Low	High	High	High	High	Low	High	High
		Spread spectrum based[65]	Low	Low	High	High	High	High	Low	High	High

strong attack on embedded watermark. Firstly, it needs to be ensured that the attacker has sufficient knowledge, facilities, time and money to make it possible to perform reverse engineering an IP. Secondly, given the attacker has sufficient knowledge, capabilities, time and money, the attacker needs to assess whether more profit will be earned or not by selling the illegal copies of the IP than the cost of performing such reverse engineering attacks. Finally, removal of the watermark is as difficult as completely designing a specific functionality. Reverse engineering attacks is not impossible, nevertheless, it is implausible to occur in normal designs. It targets complex expensive designs and security-critical designs dedicated to cryptographic applications for instance. Moreover, all available IP watermarking solutions do not work everywhere and more robust security solutions are required such as ciphering the FPGA bitstream [67]. It needs to be kept in mind that watermarking is a solution against illegal copying of IPs specifically, but not against IP reverse engineering which is considered to be a serious threat.

5) Countermeasures: Security analysis and countermeasures of an IP watermarking technique against masking and removal attacks are dependent on the watermark insertion and detection mechanism used, and they differ from case to case. In [2], authors reported a countermeasure to defend against the forging attack. If the watermarked design is forged by simply the addition of watermark, the IP owner is able to provide an IP core with only his watermark while the attacker has only the IP core with both watermarks. The IP core clearly belongs to the IP owner. The FSM-based watermarking scheme proposed in [47] provides good resiliency against the removal and masking attacks since the watermark bits are dispersed arbitrarily and hidden in the existing transitions of the FSM. This method to the state assignments of pseudo-input variables makes it almost infeasible to attack the watermarked FSM. The length of the watermark verification pattern can be altered without reducing the watermark strength. The scan chain-based watermarking technique proposed in [15], based on ordering of the connections between some pairs of scan flip-flops in the scan-chain, provides high watermark strength. However, this method is highly vulnerable to removal attacks.

Resiliency against the removal attack increases if the watermarking scheme is inseparable from the design and the attacker cannot distinguish it from the original design. One way to achieve this is by watermarking different parts of the design separately. Kirovski et al. [68] proposed that multiple watermarks can be inserted into different locations of the IP. Rather than insering a large watermark, the authorship signature is broken down into a series of small watermarks, each of which is randomly supplemented into a different region of the design and can be verified separately of the rest of the design. According to the authorship information, the small watermarks are translated into sets of extra constraints and allocated to pseudo-randomly selected locations. Since just a particular location of the design is required to decode the stego constraints owing to the localized watermark in that location, the approach allows parts of the watermarked IP to be secured individually. Furthermore, because the local watermarks are independent of one another, an attacker would have to change a significant portion of the IP to remove the copyright information. Charbon [42] proposed that multiple watermarks can be inserted at multiple levels of the design, establishing the concept of hierarchical watermarking. Multiple watermarks implanted in different abstraction levels, each independent of the other, give more powerful protection for the IP. The authorship information can only be removed if the attacker is able to remove all of the watermarks in each design level. One issue with the hierarchical watermarking is that the design overhead becomes large with increased number of watermarked levels. A similar concept to localized watermarking can be used to solve this problem. In [69], the authors have proposed another interesting approach for implementing secure DSP circuits for consumer electronics applications, considering a double line of defense: (1) key-based structural obfuscation for preventing attacks and (2) physical-level watermarking for detecting attacks. This methodology seems interesting since it provides almost zero overhead with very high tamper

TABLE II: Threat model for watermarking at different phases of design.

Attributes	Design House	SoC Integrator	Design Service Provider (e.g., DFT)	Foundry	OSATs	End User
Asset	RTL/Netlist/Layout	RTL/Netlist/Layout	Netlist/Layout	Layout	IC	IC
Objective	Piracy Reuse Counterfeit Over-produciton	Piracy Reuse Counterfeit Over-produciton	Piracy Reuse	Piracy Reuse Over-produciton	Piracy Reuse Over-produciton	Piracy
Capability	Tampering Removal Reverse Eng.	Tampering Removal Reverse Eng.	Tampering Removal Reverse Eng.	Removal Reverse Eng.	Reverse Eng.	Reverse Eng.

resistance. The key-based structural obfuscation (i.e, act as the first line of defense) ensures security against malicious intents for counterfeiting and Trojan insertion. Embedded watermark in the physical level (i.e, acting as the second line of defense) helps detect fake ICs or IPs and removes them from the supply chain.

IV. Comprehensive Threat Models

In order to develop appropriate watermarking approach to provide proof of IP ownership, understanding an attacker's objective, the assets she has access to, the capabilities she can use to exploit the vulnerability, and the different attack methods she may employ are important. To exploit vulnerabilities, the adversary must identify the objectives, assets, and capabilities available. The IP owner, design house, SoC integrator, untrusted foundry, third-party design service provider, outsourced semiconductor assembly and tests (OSATs), and end-users can be potential antagonists against watermarking [14, 17, 70]. Based on their capabilities, we have developed a threat model consisting of the objective, asset holding, and capability for each of the entities mentioned above in Table II. The following discussion defines these entities and briefly discusses their objective, capabilities, and asset holding in the threat model against watermarking.

In the supply chain of the SoC design, the 'IP owner' is the person who owns the legitimate rights to the intellectual property. Various abstraction levels can be used to describe the design, from layout to HDL code. Owners of the IP provide them to design houses and permit them to use the pre-designed IP cores under certain conditions. The 'design house' is the entity that owns the whole system-on-chip. It purchases licenses of different third-party IPs (3PIP) from IP providers and integrate them with in-house IPs to design the complete SoC [14, 70]. In some cases, the design house outsources the SoC integration part to other companies, termed 'SoC integrators', which are responsible for designing the final product for the design house. They are responsible for connecting the design house, the manufacturer, and the market. An SoC integrator can tamper, remove, or reverse engineer the watermark from the 3PIP cores if they have access to both the soft and hard IP cores as well as knowledge of each IP's functionality. Additionally, the design is subjected to a thorough functional analysis to identify bugs. Rogue designers may be able to tamper with DFT structures such as scan chains. In addition to these tools, the SoC integrator can use state-of-the-art reverse engineering software to extract netlists. These tools allow the integrator to analyze how each gate in the core is implemented and functional. Malicious SoC designers attack 3PIP with the primary purpose of stealing IP. A rogue design house may remove the original watermark of the 3PIP owner and insert its own watermark to claim ownership proof. As a result, 3PIP vendors have always had trust issues with SoC integrators [13, 14, 55, 70].

The SoC integrator hires third-party design service providers to perform specific design, implementation, and verification tasks. Their access to gate-level netlists and the device's scan chain makes them capable of launching several attacks [13, 55]. Their capability may also include netlist reverse engineering and access to failure analysis (FA) labs. The goal for attacking the hardware for a 3rd party service provider is IP piracy and reuse.

Due to the increased demand in consumer electronics, fabless semiconductor companies and integrated device manufacturers (IDMs) outsource their fabrication and manufacturing test services to offshore foundries and OSATs (outsourced semiconductor assembly and test) [19, 70]. The foundry requires the physical layout (GDSII) to manufacture the device, which makes them a significant suspect for IP infringement in the supply chain. They may have access to the manufacturing test patterns as well. In addition to the latest FA tools, each foundry is also capable of reverse engineering. Another asset available to the foundry is access to the DFT structures that will detect and analyze any failure in the die [17, 55]. A foundry equipped with the capabilities mentioned above can reverse engineer the chip and localize the key-storage element, key-delivery unit, key-gates, interconnect, and DFT distribution, bypassing a design's security. Researchers are being urged to re-examine the threat of IP piracy by end-users due to advancements in reverse engineering over the past few years. End-users have access to the unlocked chip and any related documentation. She can access any FA tools in any industrial or academic FA lab and learn about the chip's functionality, algorithm, and implementation [17, 55].

IP authors use watermarking schemes to embed authorship information into the IP. When developing different IP watermarking schemes, designers often restrict themselves within the boundary of the IP. The IP could be used legally or illegally within an SoC with some other IPs. Verifying the watermark signature embedded into the IP will require access to the IP within the SoC [62]. Watermark designers often consider that they have access to the IP. This assumption is valid but only within a specific context. If the IP owner contracts with an SoC integrator, the IP owner can assume that some form of IP access will be given for authorship verification. IP owners

TABLE III: Assumptions for different types of threat models.

Watermarking Methods	Access to IP	Access to IC	Prove Ownership	Abstraction Level	Threat Model
Digital Watermark	Yes	Yes / No	Within Contract	RTL	Removal RE
	Yes /No	Yes	Outside Contract	RTL / Gate	Removal RE
Universal Watermark	No	Yes	Outside Contract	RTL / Gate	Removal RE
Active Watermark	Yes /No	Yes	Outside Contract	RTL / Gate	Removal RE
WM for Trojan Detection	Yes /No	Yes	Outside Contract	RTL / Gate	Removal RE

can also force the SoC integrator for this access to authorship verification when forming the contract. Providing the access will benefit the SoC design house as proof of ownership of the IPs used in the SoC will eventually provide proof of ownership of the SoC (i.e., prove ownership within the contract). But the assumption may not be valid in the context of IP theft. The rogue SoC design house, which stole the IP, is not within any legal contract with the IP author and is not bound to provide easy access to the IP within the SoC to prove ownership (i.e., prove ownership outside contract). Hence, we should expect the rogue SoC integrator to actively block any path or access to the authorship verification scheme. The majority of the existing watermarking techniques rely on a challenge or specific input pattern to generate the watermark signature containing proof of ownership. When input access to the watermarked IP for the specific pattern remains ambiguous or blocked, the watermarked scheme will be rendered useless.

Based on the threat models discussed above for different entities involved in the supply chain and to meet the requirements for next-generation watermarking in resisting IP piracy, we envision four emerging watermarking methods. A summary of the threat model and assumptions of these emerging watermarking techniques are outlined in Table III. Further details on these envisioned watermarking techniques are discussed in the following section.

V. Research Ahead

In this section, we provide a roadmap for further research on IP watermarking. Designing a robust and efficient IP watermarking method is necessary to make reusable IP watermarking secure against piracy/theft, tampering, and reverse engineering in practical scenarios. In the past two decades, the literature introduced several watermarking solutions for providing proof of ownership for mostly single module IPs as described in Section II. However, the assumptions made to leverage these approaches are incompatible with the modern SoC threat landscape (see Table III) and must be addressed. The following aspects should be taken into consideration as high-level guidance for potential future research directions to maximize the applicability and impact of watermarking techniques in modern SoC.

A. Digital Watermark

For digital watermarking approach, the ability of an IP owner to detect and prove IP overuse in a suspect SoC can be coarsely defined by three scenarios: i) the IP owner has physical access to the SoC and access to the IP, ii) the IP owner has physical access to SoC, but lacks access to the IP, and iii) IP owner does not have physical access to the SoC but has access to the IP through cloud environment. Here, each watermark evaluation scenario is further defined and explored.

1) Access to the IP and IC: In this scenario, the IP owner has access to the IP and the IC. So, the IP owner can easily verify the watermark. This is an optimistic scenario for the IP owner because if the IP is stolen, the attacker may not allow the IP owner an easy access to the IP. Nonetheless, this scenario is prevalent in cases where the IP owner and the SoC developer are within a contract, and the SoC developer has agreed to provide access to the IP owner to be able to extract the watermark.

In this scenario, the IP owner has physical access to the SoC and control of the IP. A software code, IP primary inputs and outputs (I/Os) and/or the scan chains, may be available for use in watermark detection and verification. Additionally, time-dependent information with physical access only (e.g., power or electromagnetic side-channel information) is also available for use. Most existing watermarking techniques (see Section II) assume full access to the SoC and the target IP, resulting in the highest opportunity of watermark generation and extraction from a silicon design. However, it is improbable that a skilled attacker misusing a pirated IP would allow easy access to the IP through standard I/O and/or scan-chain operation. Watermarks that aim to address this unlikely scenario should consider the following design requirements:

- The watermark can be implemented in register-transfer level (RTL), gate-level, or layout level. If integrated into scan chain, the watermark can be embedded at the gate level during DFT insertion.
- The watermark must be detectable with primary I/O, scan-chain access, or especially developed software code that runs on the processing unit.
- It should not be affected by circuit modification when optimized for power, area, and/or performance.
- Although embedded in the IP, the watermark should not be affected by logic synthesis, place and routing process, timing closure, and power closure.
- The watermark logic must be disjoint from the IP functionality. Therefore, during the watermark insertion, IP functionality must not be altered.
- The performance of the IP, where the watermark is inserted, should not be impacted. Therefore, performance overhead after watermark insertion should be negligible.
- The watermark insertion, generation, and extraction process need to be stealthy. The watermark should be extremely hard to detect, change, or remove.

2) Access to the IC but not the IP: This is a more likely scenario where the IP owner has access to the IC but lacks access to the IP. We believe this is a more practical case because the attacker, who steals the IP or overuses it, should not be expected to allow the IP owner easy access to the IP. In general, this is a scenario where the IP is pirated by an attacker or a rogue SoC integrator without any contract with the IP owner. In this case, the IP owner has access to the suspect IC only, and not the input-output of the embedded IP.

As a result, only the inputs and outputs of the IC, incorporating the test structure, is available for proving ownership of the IP. Existing watermarking techniques (as described in Section II) do not explicitly discuss the watermark extraction process in the scenario where the IP owner has access to the IC but lacks access to the IP. In this scenario, access to the IP may not be available when the IP is illegally used in a different IC due to the fact that attacker knowingly blocks access to the stolen IP. It may be possible to extract the watermark signature from the IP if the watermarking signature is side-channel based [61]. However getting the signature could be more difficult for some techniques such as FSM-based and test-based, because the attacker may actively block the observability. Hence, attention and innovative research is required to design watermarks that prove ownership outside of the contract. In this case, following design requirements should be considered while designing and embedding watermark signatures:

- As access to the IP is not granted in this scenario, the watermark signature should be detectable without direct access to the IP.
- The watermark signature extraction could be side-channel based [61] when access to the IP is no longer viable. Therefore, watermark signature generation circuitry must be resilient against removal and reverse-engineering attacks.
- The watermark can be implemented in RTL, gate-level, or layout level abstraction. However, the change in abstraction level must not create any additional vulnerability to the underlying watermarking technique.
- Integrated circuits go through several design, implementation, integration, and optimization steps during the IC design flow. The security of the watermarking technique should not be impacted by the SoC design flow.

3) Access to the IP but not the IC: In this scenario, we assume that the verifying party (e.g., IP owner) has control over the IP but does not have physical access to the SoC. In other words, IP owner is not able to use I/Os or scan chains to apply inputs to the SoC to extract the embedded watermark. Therefore, one possible option could be accessing to the IP through cloud environment. Another possible option could be to activate the watermark circuit externally using electromagnetic (EM) pulse, laser, or alike. Moreover, IP owner can use the side channel concept [61, 65] to detect the watermark sequencing when externally activated during verification. There has been significant amount of research regarding remotely activated hardware Trojans reported in the literature such as Power Distribution Network (PDN) Trojan based on side-channel [71]. This technique can be used to activate the watermark generation circuit remotely. Furthermore, other techniques such as EM fault injection (EMFI) and laser fault injection (LFI) can be used to activate the watermark generation circuit externally [72]. Another possible option is to obtain access to the IP watermark through embedded HOST IP in the SoC. Such access requires co-operation between IP owner and the SoC designer. This case is similar to the previous one mentioned in subsection V-A1. In case of utilizing EM, laser, etc., watermark designer should consider the following requirements:

- The watermark can be implemented in RTL, gate-level, or layout level abstraction.
- A controller engine is needed inside SoC to extract watermarks from IPs and transfer them to the cloud environment through HOST IP, Ethernet, or WiFi.
- In other scenarios, the watermark must be detectable only when it is externally activated (i.e using EM, laser, or optical probe etc.)
- Similar to the previous scenarios, the watermark should not be affected by circuit synthesis, placement, routing, and optimization process.
- Additionally, the watermark circuitry must not alter the functionality of the IP core.
- The watermark signature should be easy to detect but difficult to remove.

B. Universal Watermark

In designing a universal watermark, the IP owner should be able to detect and prove IP overuse in a suspect SoC when the IP owner has physical access only to global I/O signals (i.e., clock, Vdd, or reset) of the IC, but lacks control of the IP. In this scenario, we assume that the IP owner does not have any legal contract with the SoC integrator, i.e., the SoC integrator has pirated or reverse engineered the IP. In general, access through scan is not readily available because of multiple scan chains connected to decompressor and compactor [73], making it difficult to locate and access an IP deeply embedded into the SoC. For example, a rogue SoC integrator/an attacker may change scan cell order in the IP or deactivate the JTAG interface from the board. Then, the IP owner cannot access the IP or cannot get correct stimuli into the scan chain if cells are reordered. In that case, IP watermark detection and verification is not possible through standard I/O or scan chain operation. Most of the existing watermarking techniques (see Section II) require access to the IP to prove ownership. Therefore, we need an alternative solution for watermark design, detection and verification without any IP access. In this scenario, watermark designers should consider the following design requirements:

- Considering the conditions for universal watermark, the signature must be detectable without primary I/O or scan-chain access.
- The watermark should be detectable only when activated using global I/O signals (i.e clock, Vdd or reset) of the IC, possibly with side-channel information.
- The watermark should be intertwined with the existing design in such a way that the watermark circuitry is resistant against removal and tampering attacks.
- The watermark can be implemented in RTL, gate level, or layout level based on the abstraction level of integration (soft, firm, or hard IP).
- The watermark should not be affected by synthesis, integration, design, and optimization process.
- The watermark signature should be easy to detect by the IP owner but hard to identify or remove by the SoC integrator who is out of contract.

We provide an example solution for IP owners (e.g., verifying party) to detect and prove IP overuse in a suspect SoC when the IP owner has physical access to SoC but not access to her IP. In this context, an optical analysis such as photon emission or electro-optical methods can be utilized during watermark verification. First, watermarked signature can be embedded to the IP core at the netlist level. A suspect SoC can be evaluated by activating the watermark generation circuit using global signals (i.e., clock, Vdd or reset) for the watermark signature generation. Then, we can use the electro-optical frequency map (EOFM) probing system [74] to extract the watermark from retrieving a reference map of the IC. In this approach, the optical contactless probing is used to measure magnitude and phase of EOFM markers for identifing the IP core layout localization and bit sequences that represent watermark signature. Based on this information, the watermark signature can be reconstructed and ownership can be verified by comparing reconstructed watermark signature with the inserted watermark signature. Although, the proposed approach may sound innovative and promising, it has a number of challenges before it can be used in practice. Some of those challenges include (1) reliable measurement of photon emissions, (2) addressing errors in the collected bit sequences that represent watermark signature, (3) ensuring attacker will not able to manipulate the IP such that the signature is impacted.

C. Active Watermark

An IP watermark has always been considered "passive" as it does not stop the attacker from stealing the IP. The watermarked IP also does not change its functionality if it has been stolen and used in another SoC. The IP owners can only prove their authorship if they can have access to the IP in the SoC. If the embedded watermark is of active nature, i.e., it prevents IP piracy or changes IP functionality, it would deter IP theft in a very effective way. Hence, we envision an active watermark, when inserted into an IP, should be aware of its surrounding in a SoC. When the IP owner is making a contract with a design house, they can plan for an IP ecosystem encircling the watermarking strategy. The watermarked IP and the neighborhood IPs form an ecosystem, which should function correctly only when they maintain a certain functionality within the same SoC. For example, the SoC integrator can make the communication between the watermarked IP and the neighborhood IPs dependent on the correct signature generated by the watermarked IP. The watermarked IP will also get a response/feedback from its neighborhood IPs so that it is aware of its surrounding. So, the SoC would function correctly only when the watermarked IP generates the correct signature. Therefore, this kind of watermarking strategy can make IP theft much more difficult. Moreover, if the watermarked IP is stolen and used in a different SoC, the IP would not function correctly as it cannot verify its neighborhood IPs. This forces the attacker to steal not only the watermarked IP but also the neighborhood IPs. Furthermore, the attacker also needs to place these IPs in an SoC properly and maintaining their interdependent relationship within the same SoC. All these should make stealing the watermarked IP near impossible. When developing active watermarking techniques, the designers should also consider the availability of access to the IP. Assuming easy access to the IP even when the IP is stolen may not be correct in reality as discussed in earlier sections.

D. Watermark for Trojan Detection

Watermarking techniques add some degree of overhead to the design, and IP designers may need to relax the design constraints to accommodate this overhead. These added features remain hidden, unused, and non-functional until they are activated or decoded to provide proof of authorship [50]. It will be an outstanding accomplishment if the features added for watermarking schemes can also thwart other security vulnerabilities. One possible approach could be to use the watermarking features to detect Trojans in the design. Although this idea mostly remains unexplored in the literature, Shukry et al. [75] proposed a watermarking strategy that can also detect Trojans in the design. The proposed watermarking scheme is particularly sensitive to manipulative design modifications, to the point where any change in the original design will impact the watermark output, hence called fragile watermarking (see Fig. 12). Thus, the watermarking strategy can effectively detect purposeful tampering attacks and Trojans insertion. The authors selected system FSMs to insert the fragile watermark because they are usually in charge of system control. Hardware Trojans usually modify them to change how the system works. Although the fragile watermarking technique is an excellent way to detect design alterations, it cannot detect a skillfully-inserted Trojan to manipulate the function output. In [76], a challenge-response based watermarking technique has been proposed. This method is very sensitive to small modifications in the design, thus providing great resiliency against tampering. It also exhibits a high watermark verification accuracy with reliable detection of malicious alterations or hardware Trojans. In both of these cases, it is assumed that both the IP and IC are accessible for watermark verification. Although these research efforts perform an excellent job showing that detecting Trojans with watermarking techniques is feasible, this problem domain requires more research and development to make it more practical.

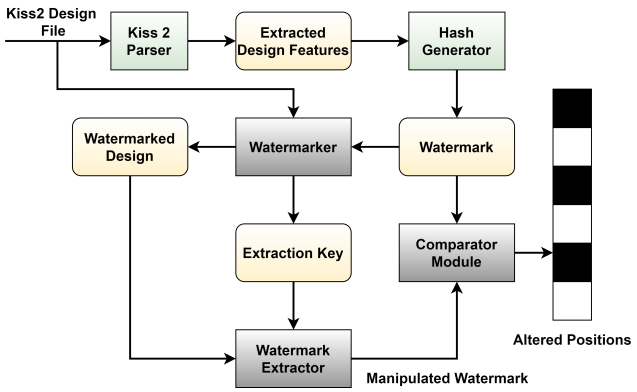

Fig. 12: Fragile FSM watermarking for Trojan detection [75].

E. Side Channel-based Watermark

In the side channel-based watermark, a unique signal is embedded into a side-channel of the device that serves as a watermark. Common examples of side-channel parameters that can be exploited to insert or verify the watermark include power consumption, execution time, and electromagnetic emission. So far, most of the IP watermarking techniques use the power consumption signature of the IP core to watermark the IP (see Section II-E). Other side-channel measures such as execution time, electromagnetic radiation, or contactless optical techniques can also be utilized to insert or verify watermarks. Since the embedded watermark system can cause power or area overhead, the watermark should not always be ON. Instead, it should be activated nondeterministically and for a brief amount of time. Furthermore, in the power signature–based watermark methods, the IP developers utilized the testing mode of the SoC to detect unauthorized use of their cores beacuse they assumed that the inputs and the outputs of each IP core are not available at the finished SoC. Therfore, the IP developers by putting SoC into test mode and observing its power signature, if the core under question is contained in the SoC will observe a very high correlation between the power signature of the core and that of the SoC [58, 59, 62]. However, using the power signature of a designer core as a watermark in the modern SoC can not provide definite results on core detection, because the power consumed by the rest cores in the SoC alters the designer core's signature. Therefore, the watermark verification in modern SoC using power signature analysis need a further study.

F. Emerging Topics in AMS Watermarking

Analog and mixed-signal (AMS) chips are the most counterfeited parts in the semiconductor industry. Different types of attacks such as removal, RE, counterfeiting, etc. can be performed at the system-level, circuit-level, or layout-level assuming that the attacker has access to both IP and SoC. SoCs may contain different IPs including memory, analog, & digital IPs [51, 77]. However, there are not many techniques that have been proposed to protect IPs in this domain compared to the digital domain. Existing analog and mixed-signal watermarking techniques [78, 79] require modifying the original layout. These watermarking approaches are somewhat difficult to verify and also prone to removal attacks. In the watermarking scheme proposed in [78], the watermark is embedded into the number of fingers of the transistors at the layout level. It can cause performance degradation due to a lack of proper implementation. The watermark extraction requires reverse engineering of a suspected chip which can be time-consuming. The analog watermarking technique requires addressing the unique characteristics of analog and mixed signal designs while considering the functional differences between various classes of designs. It should be universal and applicable to AMS chips to prove the ownership, distinguish the IPs in the SoC, and detect IP piracy and overuse. For example, an encrypted secret bitstream can be converted into an analog waveform using quantization and error correction. Then, this watermarked output can be generated from an analog chip by applying a secret input dependent on that chip behavior and transfer function. In this way, the designer does not need to modify the original design for the watermarked waveform generation. They need to develop an algorithm based on the chip behavior, which can perform an exhaustive search for input signals to generate the watermarked output from that chip.

G. Automatic Cost Effective IP Watermarking

Watermarking of embedded IPs should be capable of preserving the original functionality of the IP core with minimal area overhead. Therefore, IP watermarking approaches should imply a low overhead on the design process and the final watermarked product. Moreover, the watermark extraction time and cost should be kept low. Thus, the watermark embedding and reconstruction should be as fast, cost-effective, easy to evaluate, and ubiquitous as possible to be of any practical use. Furthermore, the watermark generation, embedding, and extraction should be easily integrated into any IP cores in modern SoCs, such as soft, firm, or hard. Therefore, it requires sufficient generic IP watermarking techniques with the very low area and timing overheads. Such techniques must be implemented in automatic design flows using CAD tools for rapid component tagging in a designer-friendly process.

H. Watermark Security Assessment

Several attacks against IP watermarking have been reported in the literature, such as removal attack, tampering attack, forging attack, and reverse-engineering attack (see Section III-B). However, few researchers have presented defense mechanisms to these attacks in the literature (see Section III-B5). Most of these defense mechanisms do not have high practical significance because they cannot provide comprehensive protection for IP watermarking against piracy. In this context, watermark methods should be integrated with advanced security techniques for secure IP protection. For example, in [80], authors have proposed the traceable IP protection algorithm in the Blockchain environment to improve the robustness of copyright information, where an IP core can be protected by inserting a watermark. As an attacker intends to destroy the content of IP core, first she should attack the watermark in the IP core, given that copyright requires verification. Moreover, if IP watermark techniques are ever to be deployed in critical security systems, they must resist removal, forging, and reverse-engineering attacks. So far, very little has been done in this area. This is an area that requires significant research and development. Furthermore, there is a need to develop computer-aided design (CAD) frameworks for the robustness assessment of IP watermarking against all types of attacks, such as removal, forging, tampering, reverse-engineering, etc.

VI. Conclusion

In this paper, we provided the most comprehensive overview of the current state-of-the-art of watermarking techniques and their pros and cons. Furthermore, we briefly discussed IP watermarking evaluation criteria and attack analysis. Then, we

discussed current threat models for IP piracy and overuse in modern SoC designs. There is no unique solution to detect and prove IP overuse in a suspect SoC. We discussed the need for future IP watermarking approaches in the research roadmap to effectively detect watermark and prove IP overuse in a suspect modern SoC. We hope this article will inspire others to focus future studies on IP watermarking in modern SoCs, and develop innovative solutions to be used in practice.

REFERENCES

[1] T. Nie, "Performance Evaluation for IP Protection Watermarking Techniques," *Watermarking: Volume 2*, p. 119, 2012.

[2] C.-H. Chang, M. Potkonjak, and L. Zhang, *Hardware IP Watermarking and Fingerprinting*. Cham: Springer International Publishing, 2016, pp. 329–368.

[3] M. Tehranipoor, H. Salmani, and X. Zhang, "Integrated circuit authentication: Hardware trojans and counterfeit detection," *Switzerland: Springer, Cham. doi*, vol. 10, pp. 978–3, 2014.

[4] E. Alliance, "ESD Alliance Electronic Design Market Data," 2021. [Online]. Available: https://www.semi.org/en/news-media-press/semi-press-releases/esda-q4-reports

[5] M. S. U. I. Sami, F. Rahman, F. Farahmandi, A. Cron, M. Borza, and M. Tehranipoor, "Invited: End-to-End Secure SoC Lifecycle Management," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*, 2021, pp. 1295–1298.

[6] J. Villasenor and M. Tehranipoor, "Chop shop electronics," *IEEE Spectrum*, vol. 50, no. 10, pp. 41–45, 2013.

[7] M. Potkonjak, G. Qu, F. Koushanfar, and C.-H. Chang, "20 Years of research on intellectual property protection," in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2017, pp. 1–4.

[8] W. Hu, C.-H. Chang, A. Sengupta, S. Bhunia, R. Kastner, and H. Li, "An Overview of Hardware Security and Trust: Threats, Countermeasures, and Design Tools," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 6, pp. 1010–1038, 2021.

[9] J. Wurm, Y. Jin, Y. Liu, S. Hu, K. Heffner, F. Rahman, and M. Tehranipoor, "Introduction to Cyber-Physical System Security: A Cross-Layer Perspective," *IEEE Transactions on Multi-Scale Computing Systems*, vol. 3, no. 3, pp. 215–227, 2017.

[10] D. Forte, S. Bhunia, and M. M. Tehranipoor, *Hardware Protection through Obfuscation*, 1st ed. Springer Publishing Company, Incorporated, 2017.

[11] P. Mishra, S. Bhunia, and M. Tehranipoor, *Hardware IP security and trust*. Springer, 2017.

[12] N. N. Anandakumar, M. S. Hashmi, and M. Tehranipoor, "FPGA-based Physical Unclonable Functions: A comprehensive overview of theory and architectures," *Integration*, vol. 81, pp. 175–194, 2021.

[13] X. Wang, D. Zhang, M. He, D. Su, and M. Tehranipoor, "Secure scan and test using obfuscation throughout supply chain," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 9, pp. 1867–1880, 2017.

[14] B. Colombier and L. Bossuet, "Survey of hardware protection of design data for integrated circuits and intellectual properties," *IET Computers & Digital Techniques*, vol. 8, no. 6, pp. 274–287, 2014.

[15] A. Cui, G. Qu, and Y. Zhang, "Ultra-Low Overhead Dynamic Watermarking on Scan Design for Hard IP Protection," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 11, pp. 2298–2313, 2015.

[16] S. Bhunia and M. Tehranipoor, *Hardware security: a hands-on learning approach*. Morgan Kaufmann, 2018.

[17] M. T. Rahman, M. S. Rahman, H. Wang, S. Tajik, W. Khalil, F. Farahmandi, D. Forte, N. Asadizanjani, and M. M. Tehranipoor, "Defense-in-depth: A recipe for logic locking to prevail," *Integr.*, vol. 72, pp. 39–57, 2020.

[18] D. Mehta, N. Mondol, F. Farahmandi, and M. Tehranipoor, "AIME: Watermarking AL Models by Leveraging Errors," in *2022 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2022, p. preprint.

[19] M. T. Rahman, D. Forte, Q. Shi, G. K. Contreras, and M. Tehranipoor, "CSST: Preventing distribution of unlicensed and rejected ICs by untrusted foundry and assembly," in *2014 IEEE International symposium on defect and fault tolerance in VLSI and nanotechnology systems (DFT)*. IEEE, 2014, pp. 46–51.

[20] https://www.computerworld.com/article/2578617/cisco-sues-huawei-over-intellectual property.html, "Cisco sues huawei over intellectual property," Computer World.

[21] https://www.mazzarellalaw.com/blog/2018/12/intel-files-trade-secret-theft lawsuit/, "Intel files trade secret theft lawsuit," Mazzarella Mazzarella LLP.

[22] https://techcrunch.com/2019/05/22/semiconductor-startup-cnex-labs-alleged-huaweis-deputy-chairman-conspired-to-steal-its-intellectual property/, "Semiconductor startup cnex labs alleged huawei's deputy chairman conspired to steal its intellectural property." TechCrunch.

[23] J. Robertson and M. Riley, "The big hack: How china used a tiny chip to infiltrate us companies," *Bloomberg Businessweek*, vol. 4, no. 2018, 2018.

[24] G. Qu and M. Potkonjak, "Analysis of watermarking techniques for graph coloring problem," in *Proceedings of the 1998 IEEE/ACM international conference on Computer-aided design*, 1998, pp. 190–193.

[25] G. Qu and L. Yuan, "Secure hardware IPs by digital watermark," in *Introduction to hardware security and trust*. Springer, 2012, pp. 123–141.

[26] F. Koushanfar, I. Hong, and M. Potkonjak, "Behavioral synthesis techniques for intellectual property protection," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 10, no. 3, pp. 523–545, 2005.

[27] T.-B. Huynh, T.-T. Hoang, and T.-T. Bui, "A constraint-based watermarking technique using Schmitt Trigger insertion at logic synthesis level," in *2013 International Conference on Advanced Technologies for Communications (ATC 2013)*. IEEE, 2013, pp. 115–120.

[28] D. Kirovski, Y.-Y. Hwang, M. Potkonjak, and J. Cong, "Protecting combinational logic synthesis solutions," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 12, pp. 2687–2696, 2006.

[29] A. Cui, C. H. Chang, and S. Tahar, "IP watermarking using incremental technology mapping at logic synthesis level," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 9, pp. 1565–1570, 2008.

[30] S. Meguerdichian and M. Potkonjak, "Watermarking while preserving the critical path," in *Proceedings of the 37th Annual Design Automation Conference*, 2000, pp. 108–111.

[31] M. Ni and Z. Gao, "Constraint-based watermarking technique for hard IP core protection in physical layout design level," in *Proceedings. 7th International Conference on Solid-State and Integrated Circuits Technology, 2004.*, vol. 2. IEEE, 2004, pp. 1360–1363.

[32] J. X. Zheng and M. Potkonjak, "Securing netlist-level FPGA design through exploiting process variation and degradation," in *Proceedings of the ACM/SIGDA international symposium on Field Programmable Gate Arrays*, 2012, pp. 129–138.

[33] R. Chapman and T. Durrani, "IP protection of DSP algorithms for system on chip implementation," *IEEE Transactions on Signal Processing*, vol. 48, no. 3, pp. 854–861, 2000.

[34] A. Rashid, J. Asher, W. Mangione-Smith, and M. Potkonjak, "Hierarchical watermarking for protection of DSP filter cores," in *Proceedings of the IEEE 1999 Custom Integrated Circuits Conference (Cat. No.99CH36327)*, 1999, pp. 39–42.

[35] A. Oliveira, "Techniques for the creation of digital watermarks in sequential circuit designs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 20, no. 9, pp. 1101–1117, 2001.

[36] M. Lewandowski, R. Meana, M. Morrison, and S. Katkoori, "A novel method for watermarking sequential circuits," in *2012 IEEE International Symposium on Hardware-Oriented Security and Trust*, 2012, pp. 21–24.

[37] P. M. Kirovski D., "Intellectual property protection using watermarking partial scan chains for sequential logic test generation," *High Level Design, Test Verification*, 1998.

[38] A. Cui and C.-H. Chang, "Intellectual property authentication by watermarking scan chain in design-for-testability flow," 06 2008, pp. 2645 – 2648.

[39] C.-H. Chang and A. Cui, "Synthesis-for-Testability Watermarking for Field Authentication of VLSI Intellectual Property," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 57, no. 7, pp. 1618–1630, 2010.

[40] A. B. Kahng, J. Lach, W. H. Mangione-Smith, S. Mantik, I. L. Markov, M. Potkonjak, P. Tucker, H. Wang, and G. Wolfe, "Constraint-based watermarking techniques for design IP protection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 20, no. 10, pp. 1236–1252, 2001.

[41] M. Tehranipoor and C. Wang, *Introduction to hardware security and trust*. Springer Science & Business Media, 2011.

[42] E. Charbon, "Hierarchical watermarking in IC design," in *Proceedings of the IEEE 1998 Custom Integrated Circuits Conference (Cat. No. 98CH36143)*. IEEE, 1998, pp. 295–298.

[43] I. Hong and M. Potkonjak, "Techniques for intellectual property protection of DSP designs," in *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181)*, vol. 5. IEEE, 1998, pp. 3133–3136.

[44] H. Dogan, D. Forte, and M. M. Tehranipoor, "Aging analysis for recycled FPGA detection," in *2014 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, 2014, pp. 171–176.

[45] I. Torunoglu and E. Charbon, "Watermarking-based copyright protection of sequential functions," *IEEE Journal of Solid-State Circuits*, vol. 35, no. 3, pp. 434–440, 2000.

[46] A. Abdel-Hamid, S. Tahar, and E. Aboulhamid, "Finite State Machine IP Watermarking: A Tutorial," in *First NASA/ESA Conference on Adaptive Hardware and Systems (AHS'06)*, 2006, pp. 457–464.

[47] A. Cui, C.-H. Chang, S. Tahar, and A. T. Abdel-Hamid, "A Robust FSM Watermarking Scheme for IP Protection of Sequential Circuit Design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 5, pp. 678–690, 2011.

[48] M. Shayan, K. Basu, and R. Karri, "Hardware Trojans Inspired IP Watermarks," *IEEE Design Test*, vol. 36, no. 6, pp. 72–79, 2019.

[49] H. Salmani, M. Tehranipoor, and J. Plusquellic, "A layout-aware approach for improving localized switching to detect hardware trojans in integrated circuits," in *2010 IEEE International Workshop on Information Forensics and Security*. IEEE, 2010, pp. 1–6.

[50] C. Lamech, R. M. Rad, M. Tehranipoor, and J. Plusquellic, "An experimental analysis of power and delay signal-to-noise requirements for detecting Trojans and methods for achieving the required detection sensitivities," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 3, pp. 1170–1179, 2011.

[51] M. T. Rahman, K. Xiao, D. Forte, X. Zhang, J. Shi, and M. Tehranipoor, "TI-TRNG: Technology independent true random number generator," in *2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC)*. IEEE, 2014, pp. 1–6.

[52] N. Nalla Anandakumar, S. K. Sanadhya, and M. S. Hashmi, "FPGA-Based True Random Number Generation Using Programmable Delays in Oscillator-Rings," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 67, no. 3, pp. 570–574, 2020.

[53] A. Cui, C.-H. Chang, and L. Zhang, "A hybrid watermarking scheme for sequential functions," in *2011 IEEE International Symposium of Circuits and Systems (ISCAS)*, 2011, pp. 2333–2336.

[54] A. Nahiyan, K. Xiao, K. Yang, Y. Jin, D. Forte, and

M. Tehranipoor, "AVFSM: A framework for identifying and mitigating vulnerabilities in FSMs," in *2016 53nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. IEEE, 2016, pp. 1–6.

[55] Y.-C. Fan, "Testing-Based Watermarking Techniques for Intellectual-Property Identification in SOC Design," *IEEE Transactions on Instrumentation and Measurement*, vol. 57, no. 3, pp. 467–479, 2008.

[56] M. L. Bushnell and V. D. Agrawal, *Essentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits*. Springer, 2002.

[57] P. C. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," in *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, ser. Lecture Notes in Computer Science, vol. 1666. Springer, 1999, pp. 388–397.

[58] N. Ahmed, M. Tehranipoor, and V. Jayaram, "Supply Voltage Noise Aware ATPG for Transition Delay Faults," in *25th IEEE VLSI Test Symposium (VTS'07)*, 2007, pp. 179–186.

[59] J. Ma, J. Lee, and M. Tehranipoor, "Layout-Aware Pattern Generation for Maximizing Supply Noise Effects on Critical Paths," in *2009 27th IEEE VLSI Test Symposium*, 2009, pp. 221–226.

[60] N. N. Anandakumar, "SCA Resistance Analysis on FPGA Implementations of Sponge Based MAC-PHOTON," in *8th International Conference, SECITC 2015, Bucharest, Romania, June 11-12*. Cham: Springer International Publishing, 2015, pp. 69–86.

[61] D. Ziener and J. Teich, "FPGA core watermarking based on power signature analysis," in *2006 IEEE International Conference on Field Programmable Technology*, 2006, pp. 205–212.

[62] G. Blanas and H. T. Vergos, "Extending the viability of power signature-Based IP watermarking in the SoC era," *2016 IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, pp. 281–284, 2016.

[63] X. Zhang and M. Tehranipoor, "Design of on-chip lightweight sensors for effective detection of recycled ICs," *IEEE transactions on very large scale integration (VLSI) systems*, vol. 22, no. 5, pp. 1016–1029, 2013.

[64] C. Marchand, L. Bossuet, and E. Jung, "IP watermark verification based on power consumption analysis," in *2014 27th IEEE International System-on-Chip Conference (SOCC)*, 2014, pp. 330–335.

[65] G. T. Becker, M. Kasper, A. Moradi, and C. Paar, "Side-channel based watermarks for integrated circuits," in *2010 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*. IEEE, 2010, pp. 30–35.

[66] A. T. Abdel-Hamid, S. Tahar, and E. M. Aboulhamid, "A Survey on IP Watermarking Techniques," *Des. Autom. Embedded Syst.*, vol. 9, no. 3, p. 211–227, Sep. 2004.

[67] L. Bossuet, G. Gogniat, and W. Burleson, "Dynamically configurable security for SRAM FPGA bitstreams," in *18th International Parallel and Distributed Processing Symposium, 2004. Proceedings.*, 2004, pp. 146–.

[68] D. Kirovski and M. Potkonjak, "Local watermarks: methodology and application to behavioral synthesis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 22, no. 9, pp. 1277–1283, 2003.

[69] A. Sengupta and M. Rathor, "Enhanced Security of DSP Circuits Using Multi-Key Based Structural Obfuscation and Physical-Level Watermarking for Consumer Electronics Systems," *IEEE Transactions on Consumer Electronics*, vol. 66, no. 2, pp. 163–172, 2020.

[70] U. Guin, D. DiMase, and M. Tehranipoor, "A comprehensive framework for counterfeit defect coverage analysis and detection assessment," *Journal of Electronic Testing*, vol. 30, no. 1, pp. 25–40, 2014.

[71] F. Schellenberg, D. R. Gnad, A. Moradi, and M. B. Tahoori, "An inside job: Remote power analysis attacks on FPGAs," in *2018 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2018, pp. 1111–1116.

[72] J. Lee, S. Narayan, M. Kapralos, and M. Tehranipoor, "Layout-Aware, IR-Drop Tolerant Transition Fault Pattern Generation," in *2008 Design, Automation and Test in Europe*, 2008, pp. 1172–1177.

[73] P. Srinivasan and R. Farrell, "Hierarchical DFT with Combinational Scan Compression, Partition Chain and RPCT," in *2010 IEEE Computer Society Annual Symposium on VLSI*, 2010, pp. 52–57.

[74] A. Stern, D. Mehta, S. Tajik, F. Farahmandi, and M. Tehranipoor, "SPARTA: A Laser Probing Approach for Trojan Detection," in *2020 IEEE International Test Conference (ITC)*, 2020, pp. 1–10.

[75] S. M. Hussein Shukry, A. T. Abdel-Hamid, and M. Dessouky, "Affirming Hardware Design Authenticity Using Fragile IP Watermarking," in *2018 International Conference on Computer and Applications (ICCA)*, 2018, pp. 1–347.

[76] A. Nair, P. SLPSK, C. Rebeiro, and S. Bhunia, "Signed: A challenge-response based interrogation scheme for simultaneous watermarking and trojan detection," 10 2020.

[77] U. Guin, X. Zhang, D. Forte, and M. Tehranipoor, "Low-cost on-chip structures for combating die and ic recycling," in *2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC)*. IEEE, 2014, pp. 1–6.

[78] D. L. Irby, R. D. Newbould, J. D. Carothers, J. J. Rodriguez, and W. T. Holman, "Low level watermarking of VLSI designs for intellectual property protection," in *Proceedings of 13th Annual IEEE International ASIC/SOC Conference (Cat. No. 00TH8541)*. IEEE, 2000, pp. 136–140.

[79] R. D. Newbould, D. L. Irby, J. D. Carothers, J. J. Rodriguez, and W. T. Holman, "Mixed signal design watermarking for IP protection," *Integrated Computer-Aided Engineering*, vol. 10, no. 3, pp. 249–265, 2003.

[80] L. Xiao, W. Huang, Y. Xie, W. Xiao, and K.-C. Li, "A Blockchain-Based Traceable IP Copyright Protection Algorithm," *IEEE Access*, vol. 8, pp. 49 532–49 542, 2020.