scientific reports

OPEN

Bio-mimicking DNA fingerprint profiling for HLS watermarking to counter hardware IP piracy

Anirban Sengupta✉, **Nabendu Bhui, Aditya Anshul & Vishal Chourasia**

The multifaceted, multivendor-based global design supply chain induces hardware threats of intellectual property (IP) piracy for modern computing and electronic systems. Current hardware watermarking techniques fall short either in terms of watermark strength (size of covert constraints generated) or number of security layers/variables involved in the security constraints generation process. This paper presents a novel approach for high level synthesis (HLS) watermarking by bio-mimicking DNA fingerprint profiling to counter hardware IP piracy. The proposed approach effectively captures the vital DNA fingerprint profiling phases such as DNA sequencing, DNA fragmentation, fragment replication, DNA ligase, etc. and bio-mimics them to generate a digital watermarking framework. The presented approach has been demonstrated on convolutional layer and JPEG compression-decompression (CODEC) algorithms that are widely used in several medical and machine learning applications. The proposed approach has been thoroughly compared with several state-of-the-art approaches. The proposed approach depicts superior security in the probability of coincidence of up to ~ 10^4 and tamper tolerance of up to ~ 10^{368} at 0% overhead as compared to the prior approaches.

Keywords Piracy, Hardware security, HLS, DNA fingerprinting, Encryption

Dedicated hardware IPs are indispensable for consumer electronics system design owing to their accelerated performance and higher efficacy. This efficacy is crucial for applications like real-time image processing or high-definition video decoding, which require performing computationally intensive tasks. The convolutional layer in convolutional neural network (CNN) or joint photographic experts group (JPEG)-compression-decompression (CODEC) are the crucial application frameworks for image/video processing[1,2]. These applications involve computationally intensive tasks; therefore, it is crucial to design their dedicated hardware IPs for achieving efficient system design[3–7]. Thus, the wide applicability of CNN and JPEG-CODEC makes it imperative to design dedicated hardware IPs. However, the global supply chain of electronic system design may pose several points of vulnerability, where hardware security threats can arise during various stages of the entire semiconductor life cycle. The possible threats include: (i) IP theft/piracy or unauthorized claim of IP ownership (which might occur within the SoC integration and fabrication house), (ii) netlist attacks (executed by the attacker during the fabrication phase), and (iii) insertion of backdoor logic through third-party IP cores[8–10]. Pirated/Counterfeited IPs may harbor malicious logic, leading to issues such as data leaks, functional failures, excessive heat, and reputational damage for vendors. Additionally, IPs/integrated circuits (ICs) can be fraudulently claimed or overproduced over the original licensing limit. This highlights the importance of security threats such as IP piracy and false claim of IP ownership, from an IP vendor's perspective[8–10]. Therefore, to ensure the security of hardware IPs, genuine IP vendor incorporates robust watermark-based security constraints in the design, which acts as sturdy digital evidence in case of piracy/ownership conflict. It has been standard practice in the hardware security community[10–14] to exploit the author's/owner's signature/biometric as watermark-based digital evidence. A person who is a highly trustworthy insider in the vendor's house can be selected for watermark embedding. The most trustworthy insider (typically the owner of the IP design house) acts as the representative on behalf of the entire IP design/vendor house comprising several employees. Therefore, it has been a standard de facto practice in the hardware security scientific community to exploit the owner's signature/biometric watermark as secret digital evidence for handling IP piracy.

There are various design abstraction levels for digital system design such as (a) algorithmic level, (b) system level, (c) RT-Level, (d) gate level, and (e) layout level. HLS is the design process that automatically converts a design representation (application) from algorithmic level to RT-level. Watermarking can be applied at various design levels, such as algorithmic, register transfer level (RTL), gate, or physical level. Using high-level synthesis

Department of Computer Science and Engineering, Indian Institute of Technology Indore, Indore, India. ✉email: asengupt@iiti.ac.in

(HLS) for watermarking (which provides security at higher abstraction levels at lower complexity and cost) enhances IP security, detects piracy, and protects against false claims by embedding security features early in the design process[15–20]. Employing hardware security through high level synthesis (HLS) provides a powerful approach to detect IP piracy as well as also to safeguard against false ownership claim. This is because HLS provides designers with flexibility in embedding watermark constraints during the register allocation/binding phase, which results in minimal design overhead. Further, handling hardware security during HLS enables propagating the security constraints into lower levels of design abstraction, thereby protecting soft IP versions, firm IP versions, and hard IP versions, respectively.

Watermarking, steganography, and cryptography are crucial in protecting hardware IP cores. Hardware watermarking[21–24] embeds unique identifiers (signature) in designs to trace ownership and provide detective control against piracy. Hardware steganography[25] hides IP-related information within a design to combat IP piracy and false IP ownership claim. Further, cryptography primitives are responsible for encryption or producing hash digest. They are used in conjunction with hardware watermarking to generate robust watermark signature, which can be embedded as digital evidence within the design. The existing techniques such as watermarking[21–24], steganography[25] still lack in generating a significant number of watermarking constraints for embedding, resulting into weaker digital evidence for detective countermeasure against IP piracy and fraud IP ownership conflict. In existing watermarking techniques[21–24], the signature encoding variable can be either compromised or leaked. These techniques do not have sufficient layers of security to provide robust digital evidence, like the proposed approach. Further, in IP steganography[25], the number of security constraints is very limited, unlike the proposed approach which generates a large size watermark constraints. All these deficiencies have been removed by the proposed approach in this paper.

Threat model

The modern integrated circuit (IC) design supply chain introduces potential points of vulnerability in the form of unwanted adversary/attacker. For example, an adversary can be present in the SoC design house, who can unlawfully perform IP piracy, create inferior quality pirated versions as well as falsely claim IP ownership of the original IP designed by the IP vendor. Similarly, a rouge designer present in the foundry can also act as a threat actor/adversary, who is potentially capable of performing IP piracy of the netlist design version. Please see Table 1 for summary. Therefore, an attacker can unauthorizedly tamper or modify the IP design without the consent of the authentic IP owner. It becomes crucial to counter this threat as the global supply chain has exposed IP designs to hardware threats, such as an adversary within the SoC design house attempting to illegally perform IP piracy and falsely claim IP ownership. A potential attacker at the SoC integrator design house could pirate and falsely claim IP ownership. Pirated IPs may be tampered with, potentially incorporating malicious logic that degrades performance, causes unreliable behavior, or leaks confidential information[8,26–28]. It has been established in the literature that piracy can result into significant financial damage, loss of brand value/reputation to the IP vendor, if not handled efficiently using detective countermeasure techniques. Since this threat model (through an untrustworthy SoC integrator as a threat actor) can potentially introduce inferior/malicious versions of the original IP, hence traceability of such IP versions is very vital. This paper addresses this threat by proposing a novel IP vendor's DNA fingerprint profiling-based hardware watermarking technique as detective countermeasure against IP piracy/false IP ownership. The objective is to provide robust detection against IP Piracy by an adversary in the SoC integrator house. In order to do so, the proposed approach exploits DNA fingerprint profiling of IP vendor for HLS watermarking. Table 2 highlights the acronyms and symbols used in the paper along with their description.

Novel contributions of the paper

a. The paper proposes a novel hardware watermarking technique using IP vendor's DNA fingerprint profiling to combat IP piracy and false IP ownership claim. The technique aims to provide robust detection against IP piracy by adversary in SoC integrator house.
b. The proposed approach incorporates DNA fingerprint profiling processes, including DNA sequencing, fragmentation, fragment replication, and DNA ligase, to create a bio-mimicked robust watermark signature.
c. The proposed approach demonstrates the complete DNA fingerprint based encrypted watermarking on convolutional layer and JPEG-CODEC IP design.

The rest of the paper is organized as follows: section "Related work" discusses the related works, section "Details of proposed methodology" presents the details of the proposed methodology, section "Results and analysis" presents the results and analysis, while section "Conclusion" concludes the paper.

Related work

The techniques for hardware security against IP piracy include hardware watermarking[11,21–23,29], steganography[25], facial biometric[12], and DNA biometric methods[30]. The watermarking method in[21] uses a binary encoding

Protector and threat actors	IP vendor	SoC design house/integrator	Foundry
Threat scenario and solution	Defender	Attacker	Attacker
	Hardware watermarking	IP piracy and fraud IP ownership claim	IP piracy and fraud IP ownership claim

Table 1. Typical IP piracy attack scenario in global design supply chain cycle.

Acronym/symbols	Description
IP	Intellectual property
HLS	High level synthesis
IC	Integrated circuit
DNA	Deoxyribonucleic acid
RTL	Register transfer level
RAT	Register allocation table
CNN	Convolutional neural network
JPEG-CODEC	Joint photographic experts group-compression-decompression
SoC	System-on-chip
HDL	Hardware description language
DFG/CDFG	Data flow graph/control data flow graph
FU	Functional unit
RE	Restriction enzymes
SDFG	Scheduled data flow graph
bp^{F1}	First DNA fragment sequence
bp^{F2}	Second DNA fragment sequence
bp^{F3}	Third DNA fragment sequence
DNA^{S1}	Final first DNA fragment sequence after DNA profiling
DNA^{S2}	Final first DNA fragment sequence after DNA profiling
DNA^{S3}	Final first DNA fragment sequence after profiling
Y_i	Probability of coincidence
Z_m	Control steps required by multiplier
Z_a	Control steps required by adder
L_m	Latency/delay of multiplier
L_a	Latency/delay of adder
TT	Tamper tolerance

Table 2. Acronyms and symbols used in the paper along with their description.

scheme to embed watermarking constraints, converting the vendor's signature into constraints integrated into the design as additional edges. In[22], the method embedded digital signature bits within the hardware description language (HDL) design, using message digest 5 (MD5) and secure hashing algorithm (SHA-1) however it results into design area overhead. The strategy in[23] also uses SHA1 and RSA at the HDL level, but it is vulnerable if the RSA key is compromised. This approach is only dependent on the RSA key, as it is the only security layer in the watermarking approach. On the contrary, the proposed DNA fingerprint-based watermarking employs the following multiple security layers, which an attacker needs to completely decode to regenerate the embedded watermark constraints to falsely claim IP ownership: (a) type of restriction enzymes used by IP vendor, (b) fragmentation process of the DNA sequence employed by IP vendor, (c) number of base pairs used for DNA fragment replication by IP vendor, (d) DNA fusion process to generate final DNA signature sequence, (e) DNA encoding rule of the IP vendor, (f) AES encryption using IP vendor key, and (g) watermark constraints embedding rule of the IP vendor. Therefore, the usage of AES as one of the security layers in the proposed framework is only used for augmenting the strength of the DNA fingerprinting based watermark signature. In order to regenerate the final DNA watermark constraints, an attacker needs to decode the above security layers (a)-(g). Without successful decoding of all layers, an attacker is unsuccessful in their adversarial attempt. The technique in[11] employs functional unit binding for watermarking, though it incurs significant design overhead. In[29], a protection scheme uses logic encryption and watermarking, implementing FSM obfuscation for IP theft detection. Authors in[31] propose a technique using dated handwritten signature to be used as watermark for protecting hardware. Furthermore, steganography-based security in[25] embeds stego-constraints as a secret mark, though it becomes less effective if design data or the stego-encoding process is compromised. Additionally, generating stego-constraints is complex, introducing implementation complexity. Further, facial biometric-based hardware watermarking in[12] derives security constraints from the IP vendor's facial features, while the approach in[30] exploits IP vendors DNA biometric to produce hardware watermark. Moreover, authors in[11] have presented a pragma insertion-based hardware watermarking approach using the functional unit allocation phase of the HLS process. Despite its robustness[11], is only applicable to commercial HLS tools. Further[11], is not capable of generating large-size constraints due to the pragma insertion-based watermarking approach, and there is no security analysis, such as analysis of watermark collision, brute-force analysis, standard attacks analysis, etc. Additionally, there are chances of the final resource profile and performance profile being affected (overhead) due to the insertion of pragma as watermark constraints. These biometric-based methods[30,31] offer more robust security than previous techniques[11,21–23,25,29] however, they still lack in generating a significant number of watermarking constraints for embedding, resulting into weaker digital evidence for detective countermeasure against IP piracy and fraud IP ownership conflict. In contrast, the proposed approach embeds a strong DNA fingerprint-based watermark using a DNA profiling process, making it more robust and tamper-tolerant compared to existing techniques. Moreover, due to its massive watermark strength, it provides robust digital evidence (author credibility proof) against IP piracy and false claim IP ownership. The proposed approach can withstand standard threats of ghost insertion search attack (watermark collision), tampering attack (brute force), forgery attack and watermark removal attack. The empirical evidence of the superior quality results obtained for the probability of coincidence and tamper tolerance, compared to prior works, is shown later in section IV (Table 6).

Details of proposed methodology

Overview

This paper introduces a novel security methodology to fortify IP designs, leveraging DNA fingerprint profiling as watermark security provided by an IP vendor. This detective countermeasure acts as a protective barrier against security threats, particularly those posed by untrustworthy design houses seeking to pirate the netlist/register transfer level (RTL) representation. The proposed approach works by fragmenting DNA information followed by replicating the fragmented DNA sequence, which is subsequently subjected to DNA fusion (ligase). The fused DNA sequence is subjected to post-processing to generate a DNA signature, which is then further subjected to AES encryption. The encrypted DNA signature is converted into watermarking constraints. The proposed approach is capable of generating robust DNA fingerprint as watermark, compared to other contemporary techniques[11,21–23,25,29]. The proposed method has been specifically analyzed/tested on CNN convolutional layer and JPEG-CODEC applications because these applications are widely used in several machine learning accelerators and image processing cores for multimedia and consumer electronics systems. Further, these applications are complex and data-intensive by nature and therefore require sophisticated hardware security (watermarking) techniques to secure them robustly against IP piracy/false IP ownership claim. However, the proposed approach is applicable for any other data intensive applications such as discrete cosine transform (DCT), fast fourier transform (FFT), finite impulse response (FIR) filter, digital filters, etc. during designing secure watermarked IP core.

Figure 1 illustrates the data flow graph (DFG) of the proposed approach, which integrates the watermark HLS design flow with the DNA fingerprint profiling process. In the proposed methodology, the high-level description of hardware application (DFG), resource constraints (also known as 'allocation pragma-based directive') and module library are taken as input. Here, the watermark HLS design flow (which is an automated process) translates a high-level behavioral description (DFG) of an application framework into a secure RTL design. The input is a data flow graph (DFG) of the convolutional layer in the CNN or JPEG-CODEC application. The watermark HLS design flow typically involves the following key steps (as shown in Fig. 1): scheduling (LIST algorithm), functional unit (FU) allocation, binding, register allocation and watermark embedding process. Firstly, the scheduling and FU allocation is performed, followed by generating a register allocation table (RAT) for a scheduled CNN/JPEG CODEC design that involves mapping the storage variables (responsible for holding the intermediate computed values) to specific registers in the scheduled design. The RAT includes the total number of required registers, control steps, and storage variables of the scheduled design. This initial RAT of the scheduled design is later used to embed the IP vendor's DNA fingerprint as a watermark. The next major block of the proposed approach includes DNA fingerprint profiling and its conversion into HLS watermark. The sub-steps include the following: capturing human body samples of IP vendor (blood, hair, skin tissue, saliva, etc.). From the human body sample of the IP vendor, DNA sequence is extracted. *Note*: the DNA sample data has to be extracted ethically with the consent of the respective entity (IP vendor). Since, the DNA data is kept preserved in a secret database for analysis; therefore there is no issue with privacy. Further, this extracted DNA sequence is used to perform DNA fingerprint profiling. DNA fingerprint profiling is composed of DNA fragmentation using restriction enzymes (RE). Next, the fragmented DNA sequence is replicated (by mimicking the biological process), followed by DNA fusion (mimicking the ligase process). After this, the final DNA sequence is generated, which is subjected to post-processing to generate the DNA signature of the IP vendor. This generated DNA signature is encrypted to generate the final encrypted DNA signature. Thereafter, the encrypted DNA signature is encoded (using the IP vendor's specified encoding rule) for conversion into watermark constraints. These watermark (security) constraints are embedded into the register allocation table (RAT) obtained from the HLS process. Finally, from the DNA fingerprint-based watermark RAT, the secure RTL IP design of the respective application using HLS is produced.

Proposed secure HLS design flow

The details of the proposed DNA fingerprint profiling-based hardware watermarking approach are depicted in Fig. 1. As shown in Fig. 1, the proposed watermarking approach consists of two major blocks: (a) secure HLS design flow block and (b) proposed DNA fingerprint profile-based watermark signature and embedding block. The primary input of the first block, i.e., secure HLS design flow block, comprises of (i) a high level description of the input hardware application in the form of C/C++code/control data flow graph (CDFG)/transfer function, (ii) a module library containing area and delay information of used functional resource, such as adder, multiplier, etc., and (iii) IP vendor's defined resource constraints (allocation pragma-based directive). At first, the secure HLS design flow accepts the mathematical transfer function of hardware applications, such as the convolutional layer filter for CNN. Subsequently, the input transfer function is converted into its intermediate representation to generate its corresponding control data flow graph (CDFG). The generated CDFG is fed as input to the scheduling, allocation, and binding modules of the HLS process, along with the resource constraints, to generate its corresponding scheduled data flow graph (SDFG). The generated SDFG contains scheduling information for the different operations of CDFG into designated control steps along with its corresponding register assignments. Finally, an initial register allocation table (RAT) is generated from the obtained SDFG,

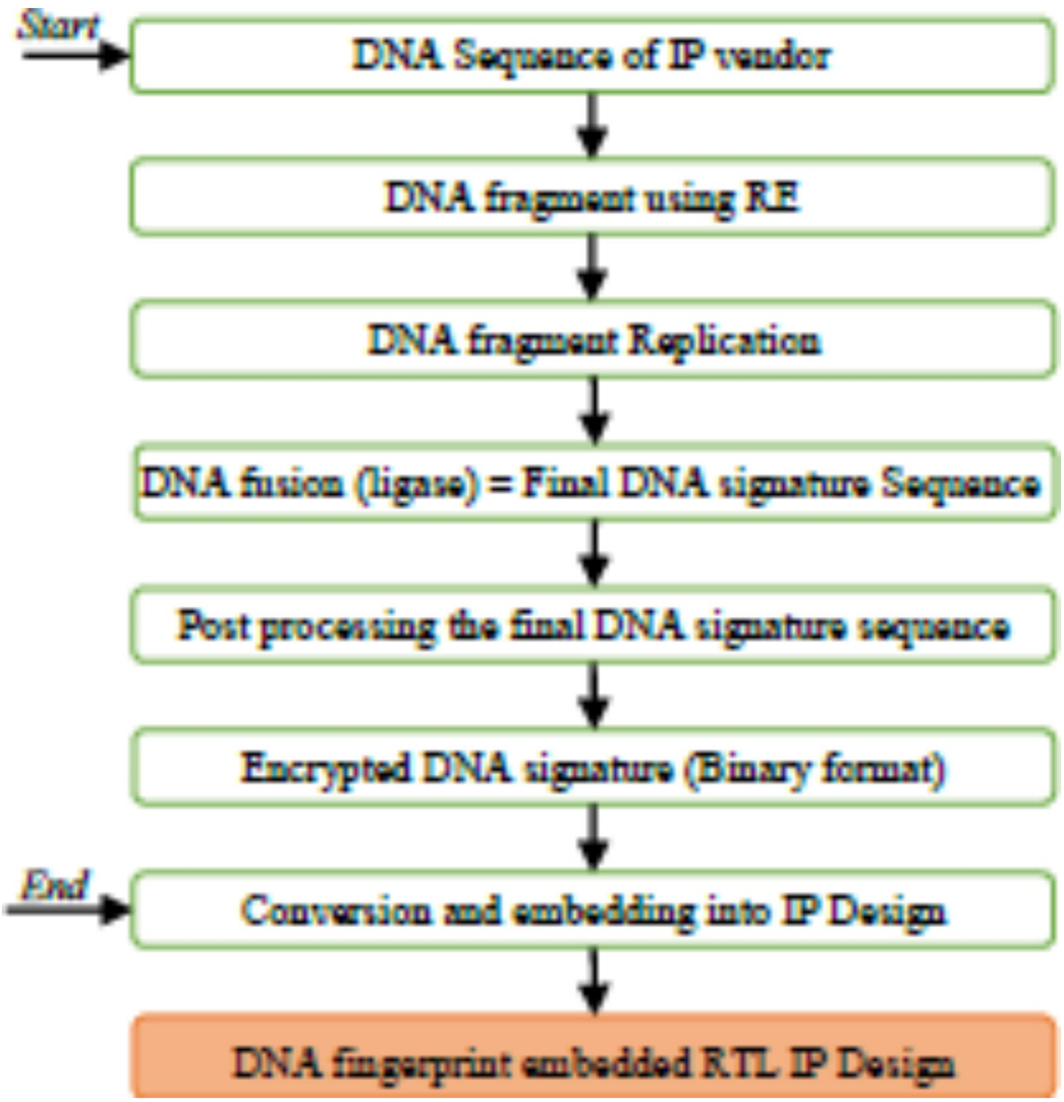

Fig. 1. Proposed hardware watermarking methodology using HLS based DNA fingerprint profiling.

which is fed as input to the proposed DNA fingerprint profiling block for watermark constraints embedding. The embedded DNA fingerprint profile-based watermark constraints act as a detective countermeasure against potential scenarios of IP piracy and false IP ownership assertion. The details regarding the generation of the IP vendor's DNA fingerprint profile-based watermark signature are discussed in the next section.

Proposed DNA fingerprint profiling process

Figure 1 also depicts the proposed DNA fingerprint profile-based watermark signature generation process. The proposed DNA fingerprint profiling block accepts the IP vendor's DNA sequence as the primary input. The DNA sequence can be extracted using the IP vendor's body sample, such as saliva, blood, skin tissue, etc[30]. The DNA sequence used in the proposed approach can be generated using[32–34]. Next, the input IP vendor's DNA sequence is used for its corresponding fingerprint profiling. Figure 2 depicts DNA fingerprinting process. The DNA sequence is composed of two different types of base pairs (BP)- Thymine (T) & Adenine (A) pair and Guanine (G) & Cytosine (C) pair. Figure 3 shows a sample DNA sequence of an IP vendor. Next, in order to perform DNA fingerprint profiling, the DNA sequence is fragmented using restriction enzymes (RE). By mimicking the biological process, the DNA fragmentation is performed using RE chemical composition. The RE performs purification of the input DNA sequence. DNA sequence molecules are cut at specific locations called "restriction sites". The REs bio-mimicked in the proposed approach are (a) EcoRI and (b) EcoRV. EcoRI is responsible for cutting the DNA sequence in a staggered way, resulting in "sticky" ends, and EcoRV cuts DNA with "blunt" ends.

- In the first step of the proposed methodology, restriction enzymes are applied to the DNA sequence; in the second step, DNA fragmentation is performed. Figure 4 depicts the fragmented DNA sequence using EcoRI (as Enzyme 2) and EcoRV (as Enzyme 1). As shown in Fig. 4, three fragments are generated post-RE enzyme application over the DNA sequence.

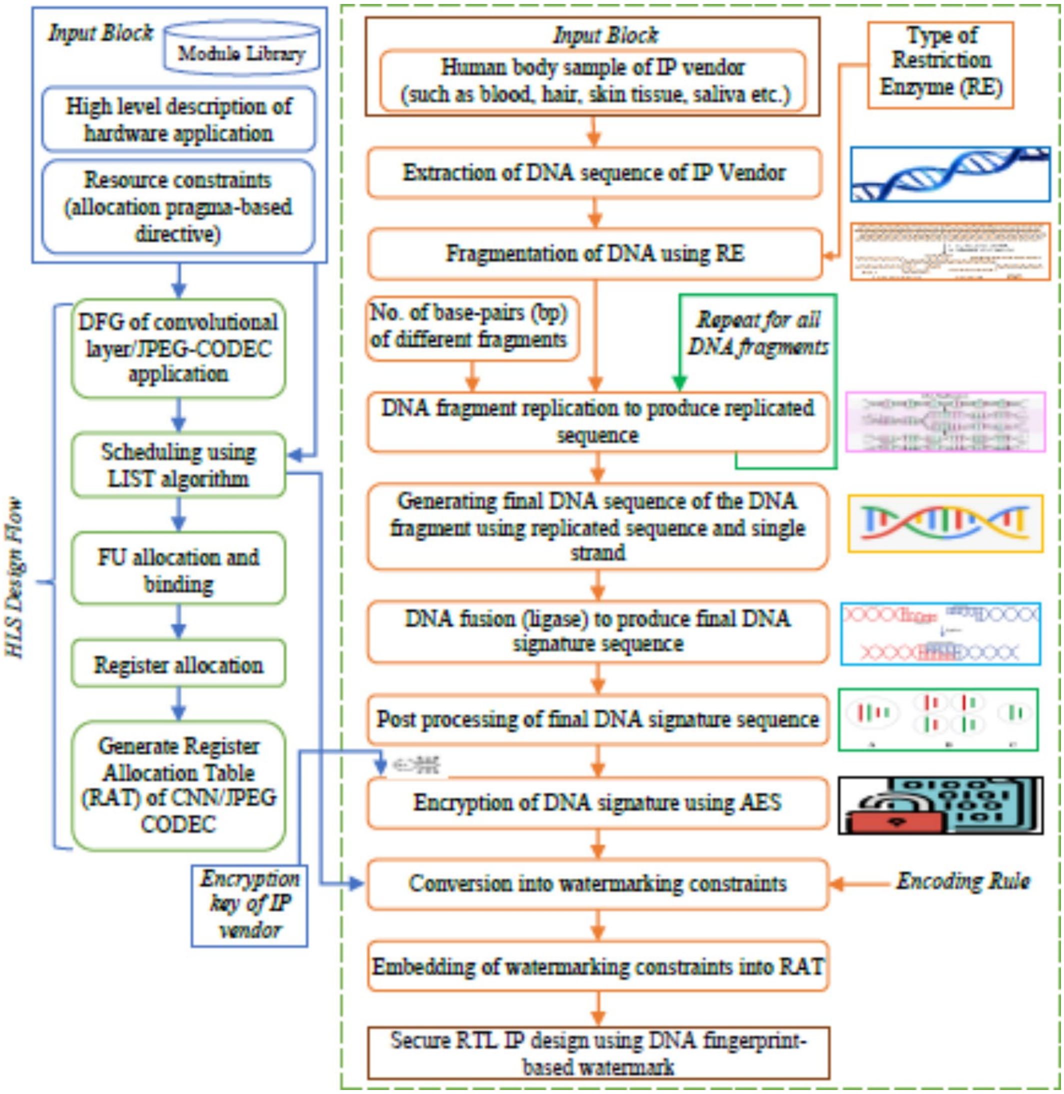

Fig. 2. DNA fingerprint process.

5	G A C T T A C G T A C C C T T A A T C C A T G G A A T T C C G A G C A T G	3`
3`	C T G A A T G C A T G G G A A T T A G G T A C C T T A A G G C T C G T A C	5`

Fig. 3. Sample DNA sequence.

- Next, in the third step, the base pairs and the remaining single strands are generated. Figure 5 depicts the corresponding base pairs of first (bp^{F1}), second (bp^{F2}), and third (bp^{F3}) DNA fragments, respectively, along with single strands (SS) corresponding to the second and third DNA fragments, respectively. Next, Fig. 6 graphically represents the base pairs strength of the different DNA fragments.
- Further, in the fourth step, the obtained DNA fragments are replicated N-number of times by bio-mimicking the DNA replication process. Here N is bp^{F1}, bp^{F2} and bp^{F3} for the first, second and third DNA fragments, respectively. Post replication, the single strands of respective DNA fragments are merged with the replicated DNA sequence as a suffix or prefix according to the RE type. Figure 7a–c depict the merging process of replicated DNA fragments with their corresponding single strands. Figure 8a–c highlight the finally generated merged DNA sequence corresponding to different DNA fragments.
- Further, in the fifth step, the final DNA signature sequence is generated post-DNA fusion (i.e., DNA^{S1} # DNA^{S2} # DNA^{S3}, where # is the fusion operator).
- Subsequently, post-processing of the final DNA signature sequence is performed in the sixth step, where each alphabet of the DNA signature sequence is encoded into its digit equivalent as per the IP vendor's encoding rule. Figure 9 depicts the generated final DNA signature sequence.

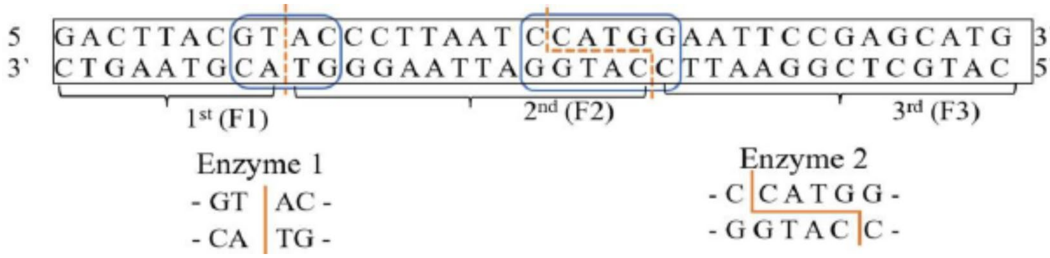

Fig. 4. Application of restriction enzymes on DNA sequence.

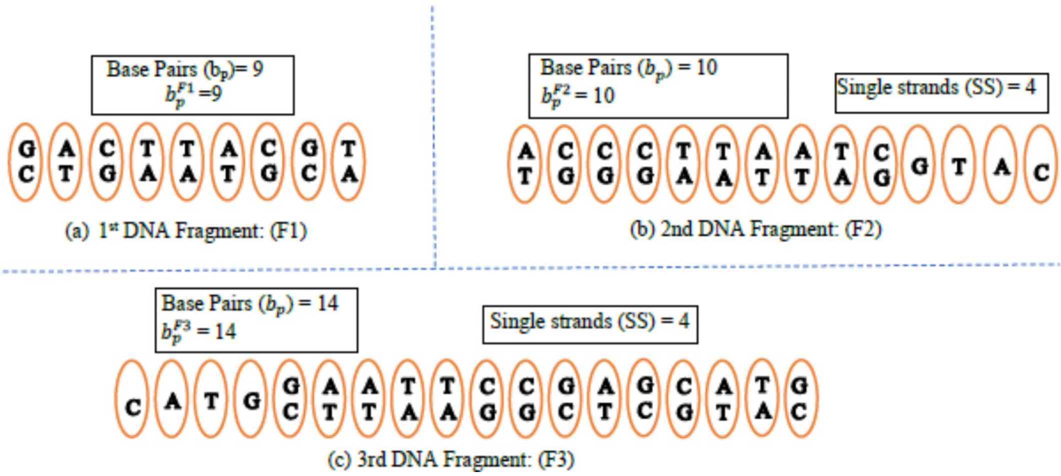

Fig. 5. Generated DNA fragments of the DNA sequence post-applying restriction enzymes.

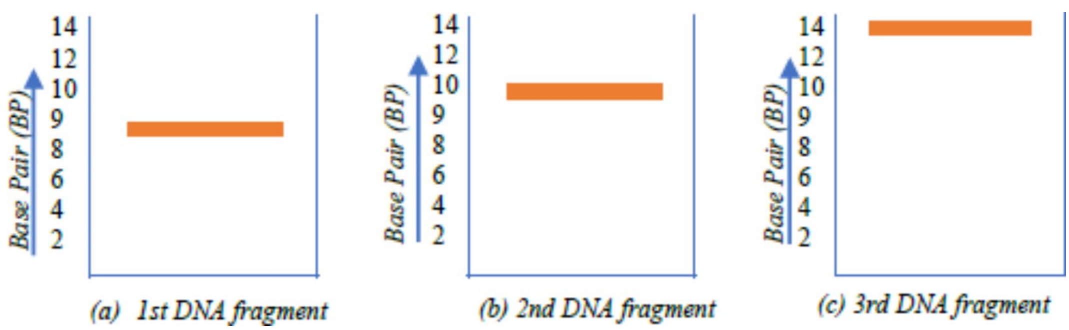

Fig. 6. Graphical representation of the base pairs strength of the different DNA fragments.

DNA encoding rule

Element 'A' (alphabet value = 1) is encoded in binary as '1', 'B' (alphabet value = 2) is encoded in binary as '10', 'C' (3) as '11', 'G' (7) as '111', and 'T' (20) as '10100'.

After post-processing, the final DNA fingerprint profile-based watermark signature sequence is generated in step 8. Figure 10 shows the complete DNA fingerprint watermark signature sequence after applying the DNA encoding rule. The obtained DNA watermark signature sequence is fed as input to the AES encryption block to obtain its corresponding encrypted DNA watermark signature. The final encrypted DNA fingerprint watermark signature is shown in Fig. 11.

Note: The size of the final DNA watermark signature sequence is dependent on the following inputs of the IP vendor: (a) type of RE enzymes used for generating DNA fingerprint profile, (b) number of DNA fragments produced post applying RE enzymes on the extracted DNA sequence of the IP vendor, and (c) number of base pairs and single strands present in each DNA fragments. Therefore, the strength of the DNA watermark signature sequence can vary depending on the above-discussed inputs. For the sake of demonstration, the paper has shown that the extracted DNA sequence of IP vendor post-applying RE enzymes has resulted in a DNA watermark signature sequence of 2048 bits.

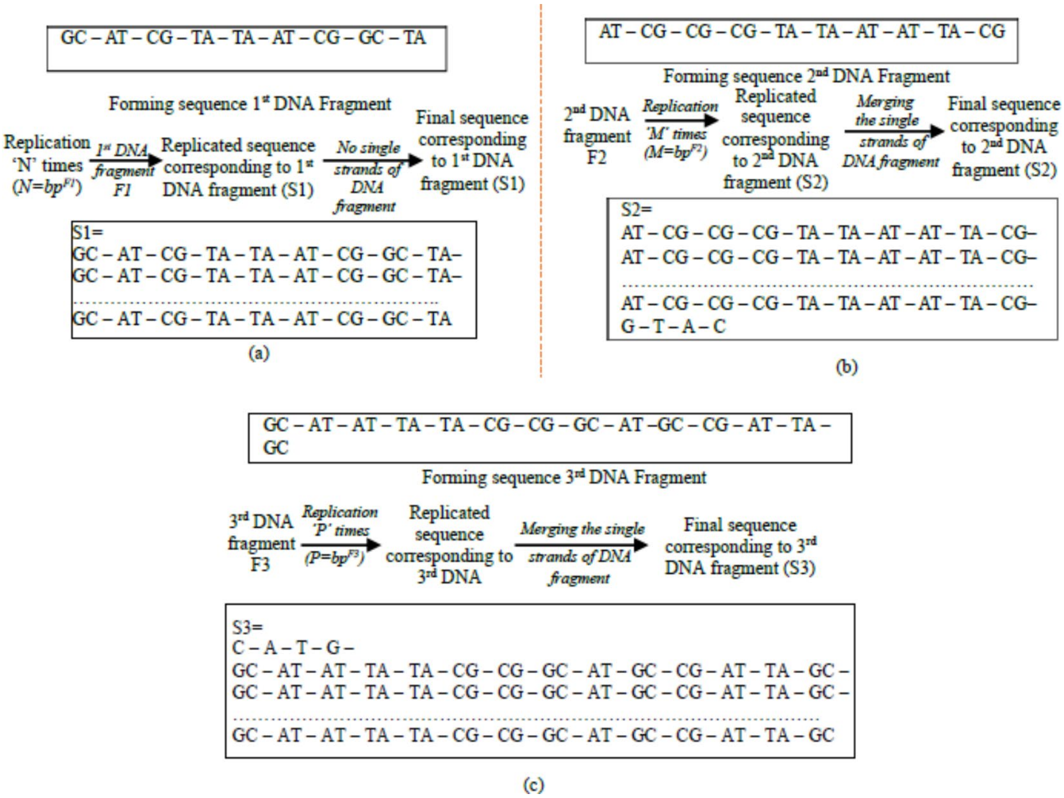

Fig. 7. (**a**) First DNA fragment replication process, (**b**) Second DNA fragment replication and merging process, and (**c**) Third DNA fragment replication and merging process.

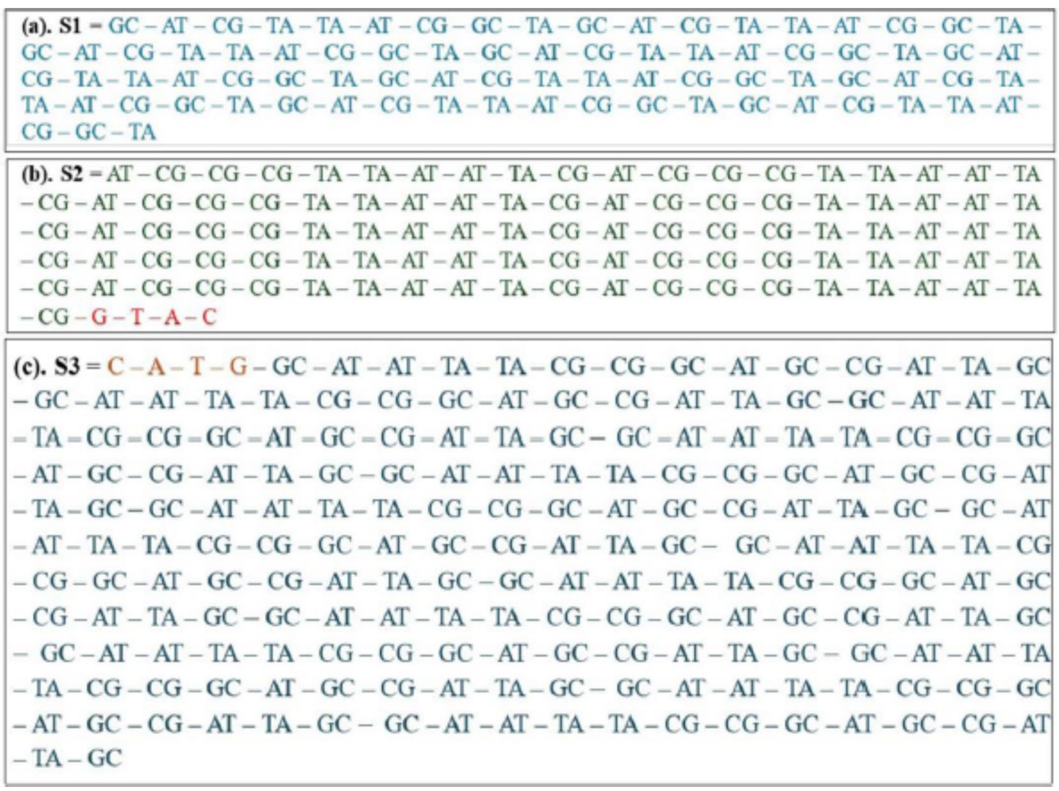

Fig. 8. (**a**–**c**) Final generated merged DNA sequence corresponding to different DNA fragments.

Generating encrypted DNA signature using AES

The proposed hardware watermarking methodology employs the Advanced Encryption Standard (AES) technique to produce an encrypted DNA watermark signature (as shown in Fig. 1). This process begins with the generated DNA signature as the primary input, along with the IP vendor's AES encryption key. Initially, the DNA signature is fed as input into the AES encryption module. AES encryption starts with the substitution phase, utilizing substitution boxes (s-boxes) to perform non-linear bit transformations. This substitution process introduces confusion, making it difficult to predict the relationship between the input and the output bits. The second phase of AES encryption involves row shifting and column mixing, which provides diffusion, spreading the influence of each input bit over many output bits. After these initial transformations, an XOR operation is executed between the transformed output and the AES encryption key. The result of this XOR operation is then fed into the subsequent round of the AES encryption module. Ultimately, this multi-round process yields an encrypted DNA watermark signature as the final output. *Note*: The AES phase used in the proposed DNA fingerprint based watermarking methodology is just one of the several security layers employed for embedding the watermark constraints into the IP design. The proposed DNA fingerprint-based watermarking employs the following security layers, which an attacker needs to completely decode to regenerate the embedded watermark constraints to falsely claim IP ownership: (a) type of restriction enzymes used by IP vendor, (b) fragmentation process of the DNA sequence employed by IP vendor, (c) number of base pairs used for DNA fragment replication by IP vendor, (d) DNA fusion process to generate final DNA signature sequence, (e) DNA encoding rule of the IP vendor, (f) AES encryption using IP vendor key, and (g) watermark constraints embedding rule of the IP vendor. Therefore, the usage of AES as one of the security layers in the proposed framework is only used for augmenting the strength of the DNA fingerprinting based watermark signature (and not for strengthening the encryption process). In order to extract the generated DNA sequence, an attacker, therefore, needs to decode the above five security layers (a)-(e) before he/she can launch an attack on the AES module. Since breaking/decoding five consecutive layers (a)-(e) is highly daunting and challenging for an attacker due to the involvement of several intricate security variables, hence an attacker would be unsuccessful in reaching the sixth security layer of AES (layer (f) mentioned above). Therefore, even though AES is vulnerable to certain attacks, however, from the perspective of overall security, an attacker would fail to regenerate the embedded watermark constraints to falsely claim IP ownership.

Embedding encrypted DNA signature on convolutional layer for secure IP design

This subsection demonstrates the embedding of the obtained encrypted DNA watermark signature on the convolutional layer application for generating a secure IP design. The convolutional layer process in CNN is highly data and computation-intensive in nature[35]. The convolutional layer in CNN is highly data and computation-intensive because it performs numerous complex operations over large input volumes with multiple filters, which involves extensive matrix multiplications and additions. It involves applying kernel filters over the input image to generate a feature map, which results in the computation of output pixels. Consider an input image (X) with dimensions S x T, where S and T denote the number of rows and columns, respectively. Additionally, let's consider a kernel (K) with dimensions w x z, where w and z represent the rows and columns in the filter kernel matrix. The convolution operation of the kernel filter over the input image can be mathematically expressed as follows, adopted from[35]:

$$\mathrm{C_y} = \sum_{\mathrm{S,w=lower\ value}}^{\mathrm{S,w=upper\ value}} \left(\sum_{\mathrm{T,z=lowervalue}}^{\mathrm{T,z=upper\ value}} X_{\mathrm{ST}} \times \mathrm{K_{wz}} \right) \tag{1}$$

GC – AT – CG – TA – TA – AT – CG – GC –TA – GC – AT – CG – TA – TA – AT – CG – GC –TA – GC – AT – CG – TA
– TA – AT – CG – GC –TA – GC – AT – CG – TA – TA – AT – CG – GC –TA – GC – AT – CG – TA – TA – AT – CG –
GC –TA – GC – AT – CG – TA – TA – AT – CG – GC –TA – GC – AT – CG – TA – TA – AT – CG – GC –TA – GC – AT –
CG – TA – TA – AT – CG – GC –TA – GC – AT – CG – TA – TA – AT – CG – GC –TA # AT – CG – CG – CG – TA – TA
– AT – AT – TA – CG – AT – CG – CG – CG – TA – TA – AT – AT – TA – CG–AT – CG – CG – CG – TA – TA – AT – AT
– TA – CG – AT – CG – CG – CG – TA – TA – AT – AT – TA – CG – AT – CG – CG – CG – TA – TA – AT – AT – TA –
CG – AT – CG – CG – CG – TA – TA – AT – AT – TA – CG – AT – CG – CG – CG – TA – TA – AT – AT – TA – CG – AT
– CG – CG – CG – TA – TA – AT – AT – TA – CG – AT – CG – CG – CG – TA – TA – AT – AT – TA – CG – AT – CG –
CG – CG – TA – TA – AT – AT – TA – CG – G – T – A – C # C – A – T – G – GC – AT – AT – TA – TA – CG – CG – GC
– AT – GC – CG – AT – TA – GC –GC – AT – AT – TA – TA – CG – CG – GC – AT – GC – CG – AT – TA – GC – GC –
AT – AT – TA – TA – CG – CG – GC – AT – GC – CG – AT – TA – GC – GC – AT – AT – TA – TA – CG – CG – GC – AT
– GC – CG – AT – TA – GC –GC – AT – AT – TA – TA – CG – CG – GC – AT – GC – CG – AT – TA – GC – GC – AT –
AT – TA – TA – CG – CG – GC – AT – GC – CG – AT – TA – GC – GC – AT – AT – TA – TA – CG – CG – GC – AT –
GC – CG – AT – TA – GC – GC – AT – AT – TA – TA – CG – CG – GC – AT – GC – CG – AT – TA – GC – GC – AT –
AT – TA – TA – CG – CG – GC – AT – GC – CG – AT – TA – GC – GC – AT – AT – TA – TA – CG – CG – GC – AT –
GC – CG – AT – TA – GC – GC – AT – AT – TA – TA – CG – CG – GC – AT – GC – CG – AT – TA – GC – GC – AT –
AT – TA – TA – CG – CG – GC – AT – GC – CG – AT – TA – GC – GC – AT – AT – TA – TA – CG – CG – GC – AT –
GC – CG – AT – TA – GC – GC – AT – AT – TA – TA – CG – CG – GC – AT – GC – CG – AT – TA – GC

Fig. 9. Final DNA signature sequence.

Pre encrypted DNA fingerprint signature:

```
111111101001111110100110100111010011111111111010011111111101
001111110100110100111010011111111111010011111111010011111110
100110100111010011111111111010011111111010011111110100110100
111010011111111111010011111111010011111101001101001110100111
111111110100111111110100111111010011010011101001111111111
010011111111101001111110100110100111010011111111111010011111
111010011111101001101001110100111111111111010011111111010011
11110100110100111010011111111111101001#11010011111111111111
101001101001110100110100101001111111101001111111111111111101
001101001110100110100101001111111101001111111111111111101001
101001110100110100101001111111101001111111111111111101001101
001110100110100101001111111101001111111111111111101001101001
110100110100101001111111101001111111111111111101001101001110
100110100101001111111101001111111111111111101001101001110100
110100101001111111101001111111111111111101001101001110100110
100101001111111101001111111111111111101001101001110100110100
101001111111101001111111111111111101001101001110100110100101
00111111111110100111#111101001111111111101001101001010011010 0
111111111111111110100111111111111101001010011111111111111010
011010010100110100111111111111111111101001111111111110100101
001111111111111010011010010100110100111111111111111111101001
111111111110100101001111111111111101001101001010011010011111
111111111111101001111111111111101001010011111111111111101001101
001010011010011111111111111111101001111111111110100101001111
111111110100110100101001101001111111111111111111101001111111
111110100101001111111111111101001101001010011010011111111111
111111010011111111111110100101001111111111111101001101001010 0
110100111111111111111111110100111111111111101001010011111111111
111010011010010100110100111111111111111111101001111111111110
100101001111111111111101001101001010011010011111111111111111
101001111111111111101001010011111111111111101001101001010011010
011111111111111111101001111111111110100101001111111111111101
001101001010011010011111111111111111101001111111111111010010
100111111111111110100110100101001101001111111111111111110100
1111111111110100101001111111111111110100110100
```

Fig. 10. Pre-encrypted DNA fingerprint watermark signature.

Here, C_y represents the output pixel value obtained after convolving the kernels over the input image, where y denotes the output pixel number. For example, consider a sample kernel of dimensions 3×3 being convolved over the input image. The elements of the matrix of the input image (X) are denoted as X_{00}, X_{01}, …, X_{22} and the elements of the 3×3 filter kernel matrix (K) are denoted as $k_{00}{}^{1}$, $k_{01}{}^{1}$, …, $k_{22}{}^{1}$. Consequently, the computation for the output pixel can be expressed as follows (adapted from[35]):

Encrypted DNA fingerprint signature:

0010101011001000001111001011100100001100110110001001010100100
0010100110111101100001111011101110101011011101010001100001
0011101001101100101110111110011011100000001000000110001111 1
1111011111111011010101010011100110000111010101011100101101 0
0010101011011100101110111111010010110011010101001000101011 1
0001001011101101110111001101100000011001100011000110100111 1
0101110011001010000011110000101000101001011110011111011110 1
1100111101011010110100001100110111000011010111111001111100 1
0110100101010101100010001010110010011110011111011101110100 1
0001100111111010111110011001111010100011111000010000110000 1
1010110001100011100101011011101111111111010011010000011010 0
1011000011110000000101111001110110000011101000100001000011 00
1011001110111111100000000000010101111000100110111001000100101
1000110101100000101101100000101101111100101000101100110001 0
1110100000111101001001000000111111100100001011101011001010
0010000010100110010101111010000110001000001110100110011100 1
0010011101111010110010100101000010001001001000101010110011 0
0001011001000100010100011100010001001100011000011010011111 1
0110111101000001110110010100110101000011101111010000101010 1
0110101111010001000111000001001011011100100011010101110101 1
0011111100001111110100000110110011110111100001000111010001 1
1110010010110001000111010111010111011011100101000001110010 1
1100101011010100011001111010110101111010011011100111000011 1
0101100101110010111001110001011001001100111010011100100000 1
1011101110100110101011010111111101010100100101000011000101 0
1101001010110100100100101001000000100000110000100110100011 0
0110101000101110111011000101000000010101100010100011011111 0
0110111011010011000100101110101010101111011011101011111110 0
1101111001111101110010110001111100011101101110010001000101 1
0110100100101000000000110101010001100100000011110000100101 1
1110111011011010111001101101100000000011110111100001011000 0
0001001110110101000101010111111011110000010111001001111000 0
0010011011011111010010101111001001101110010001101010111010 1
1001111110000111111010000011011001111011110000100011101000 1
1111001001011000100011101011101011101101 11

Fig. 11. Post-encrypted DNA fingerprint watermark signature.

$$
\begin{aligned}
C_1 = & \left[\left(X_0 \times k_{00}^1\right) + \left(X_1 \times k_{01}^1\right) + \left(X_2 \times k_{02}^1\right)\right] \\
& + \left[\left(X_{10} \times k_{10}^1\right) + \left(X_{11} \times k_{11}^1\right) + \left(X_{12} \times k_{12}^1\right)\right] \\
& + \left[\left(X_{20} \times k_{20}^1\right) + \left(X_{21} \times k_{21}^1\right) + \left(X_{22} \times k_{22}^1\right)\right]
\end{aligned} \tag{2}
$$

$$
\begin{aligned}
C_2 = & \left[\left(X_1 \times k_{00}^1\right) + \left(X_2 \times k_{01}^1\right) + \left(X_3 \times k_{02}^1\right)\right] \\
& + \left[\left(X_{11} \times k_{10}^1\right) + \left(X_{12} \times k_{11}^1\right) + \left(X_{13} \times k_{12}^1\right)\right] \\
& + \left[\left(X_{21} \times k_{20}^1\right) + \left(X_{22} \times k_{21}^1\right) + \left(X_{23} \times k_{22}^1\right)\right]
\end{aligned} \tag{3}
$$

The control data flow graph (CDFG) corresponds to the behavioral description of the convolutional layer in a CNN for two parallel pixel computations. After the generation of the CDFG, a scheduled dataflow graph (SDFG) is created based on the given resource constraints. In the SDFG shown in Fig. 12, storage variables are indicated from Q0 - Q69 and X1 - X36, which indicates different registers used for storage variables. For demonstration purposes, the SDFG for the convolutional layer filter, scheduled with the IP vendor's allocation pragma directive of one adder and six multipliers, is illustrated in Fig. 12. This SDFG is then used to produce the corresponding register allocation table (RAT). RAT is the representation of optimal allocation/assignment of storage variables into registers (based on register sharing). *Note: Storage variables in the SDFG are used to temporarily hold the primary and intermediate input/outputs of the computations*. Table 3 presents the RAT corresponding to the generated SDFG.

Mapping rule
If the encrypted watermark signature bit is 0, then even-even storage variable pairs (*Qi*, *Qj*) from the SDFG are generated as watermarking constraints, where *i* and *j* are the storage variable numbers. In contrast, for an encrypted watermark signature bit 1, the generated constraints are odd-odd storage variable pairs in the SDFG.

The resulting watermark constraints for convolution layer include pairs such as (*Q0*, *Q2*), (*Q0*, *Q4*),---, (*Q0*, *Q68*), (*Q2*, *Q4*),---, (*Q2*, *Q68*),---, (*Q66*, *Q68*), (*Q1*, *P3*), (*Q1*, *Q5*),---, (*Q1*, *Q69*), (*Q3*, *Q5*), (*Q3*, *Q7*),---, (*Q67*, *Q69*). These constraints are embedded into the convolutional layer RAT during the register allocation phase of the HLS process. To ensure storage variable pairs corresponding to watermarking constraints are not assigned to the same register (color), registers are either locally swapped or a new register is allocated[12]. This approach leads to unique register allocations corresponding to storage variable pairs of watermarking constraints[12]. Table 3 displays the RAT for the convolutional layer before and after watermark constraints integration, with changes depicted in red. The final RAT, with embedded watermark constraints, can be used to generate the secure convolutional layer IP core using classical HLS. The presence of the IP vendor's DNA fingerprint profile-based watermark constraints acts as a detective countermeasure against potential threats of IP piracy and false IP ownership assertion.

Detection of IP piracy and nullification of false IP ownership claim

An attacker may try to engage in IP piracy or falsely claim IP ownership using various threats, including brute-force attack, tampering attack and watermark collision based attack. To counter these threats, the proposed integrated encrypted DNA watermark must provide robust security against IP piracy and fraudulent claim. In cases of IP ownership disputes, the ownership conflict is resolved with the help of integrated IP vendor's encrypted DNA watermark constraints within the design as digital evidence. Ownership validation involves extracting hidden watermark constraints from the IP's RTL file and matching them with the original constraints (which can be regenerated using the proposed DNA fingerprint profiling algorithm, AES key, and mapping rule). Only the legitimate IP owner can successfully match these constraints, preventing the adversary from successfully claiming ownership in IP court. Additionally, for detecting IP piracy, the original watermarking constraints can be compared with those extracted from a suspected chip; a match would confirm the occurrence of IP piracy. *Note*: while creating and/or regenerating the original DNA based watermark constraints, AES is used only once (using an automated tool) to produce the encrypted constraints. Therefore, AES does not add

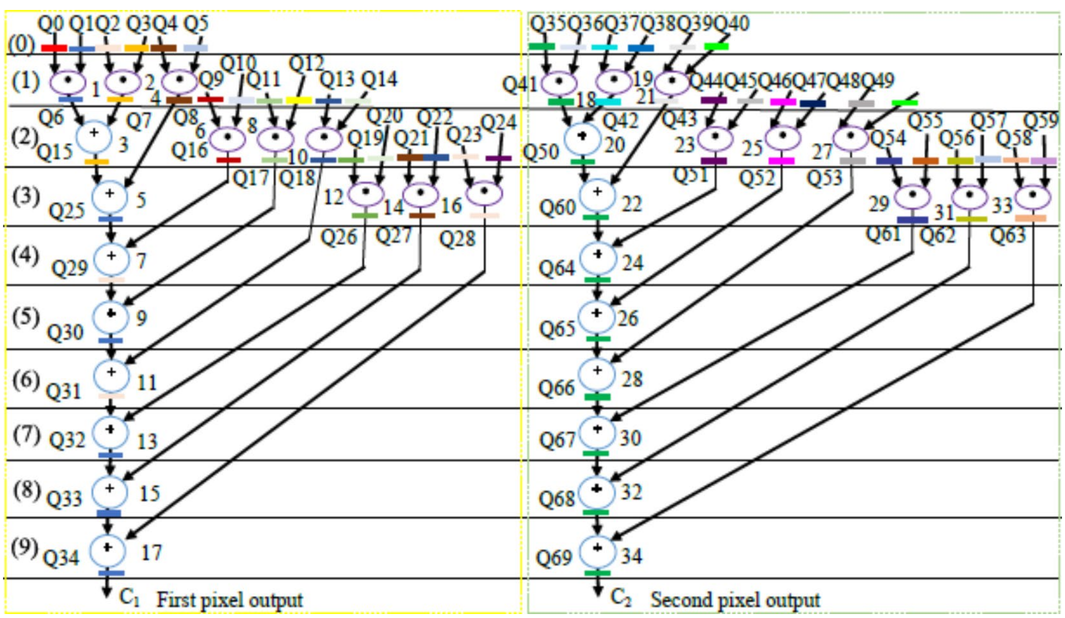

Fig. 12. Post watermarking constraints embedded SDFG of loop unrolled CNN using 6 M (*) and 2 A (+).

C.S	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	X13	X14	X15	X16	X17	X18	X19	X20	X21	X22	X23	X24	X25	X26	X27	X28	X29	X30	X31	X32	X33	X34	X35	X36
0	Q0	Q1	Q2	Q3	Q4	Q5	-	-	-	-	-	-	-	-	-	-	-	-	Q35	Q36	Q37	Q38	Q39	Q40	-	-	-	-	-	-	-	-	-	-	-	-
1	~~Q6~~	Q6	Q7		~~Q8~~	Q8	Q9	Q10	Q11	Q12	Q13	Q14	-	-	-	-	-	-	~~Q41~~	Q41	Q42	-	~~Q43~~	Q43	Q44	Q45	Q46	Q47	Q48	Q49	-	-	-	-	-	-
2	Q15	-	-	-	-	-	Q16	-	~~Q17~~	Q17	Q18	-	Q19	Q20	Q21	Q22	Q23	Q24	Q50	-	-	-	~~Q43~~	Q43	Q51	-	Q52	-	Q53	-	Q54	Q55	Q56	Q57	Q58	Q59
3	~~Q25~~	-	-	-	Q25		Q16	-	~~Q17~~	Q17	Q18	-	Q26	-	~~Q27~~	Q27	Q28	-	~~Q60~~	-	-	-	Q60	-	Q51	-	Q52	-	Q53	-	Q61	-	Q62	-	Q63	-
4	~~Q29~~	-	-	-	-	-	-	Q29	~~Q17~~	Q17	Q18	-	Q26	-	~~Q27~~	Q27	Q28	-	~~Q64~~	-	-	-	-	-	-	Q64	Q52	-	Q53	-	Q61	-	Q62	-	Q63	-
5	~~Q30~~			Q30	-	-	-	-	-	-	Q18	-	Q26	-	~~Q27~~	Q27	Q28	-	~~Q65~~	-	-	Q65	-	-	-	-	-	-	Q53	-	Q61	-	Q62	-	Q63	-
6	~~Q31~~	-	-	-	-	-	-	-	-	-	-	Q31	Q26	-	~~Q27~~	Q27	Q28	-	~~Q66~~	-	-	-	-	-	-	-	-	Q66	-	-	Q61	-	Q62	-	Q63	-
7	~~Q32~~	-	-	-	-	-	-	-	Q32	-	-	-	-	-	~~Q27~~	Q27	Q28	Q67	~~Q67~~	-	-	-	-	-	-	-	-	-	-	-	-	-	Q62	-	Q63	-
8	~~Q33~~	-	-	-	-	-	-	-	-	-	-	-	-	Q33	-	-	Q28	-	~~Q68~~	-	-	-	-	-	-	-	-	-	-	Q68	-	-	-	-	Q63	-
9	~~Q34~~	-	-	-	-	-	-	-	-	-	-	-	-	-	Q34	-	-	-	Q69	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-		-

Table 3. Convolution layer RAT pre and post embedding DNA watermark security constraints.

much performance overhead during watermark validation process. The watermark validation time remains acceptable and lower.

Results and analysis

The experimental evaluation of the proposed methodology is carried out on a 2.30 GHz processor and 4GB memory. For area and latency estimation, the proposed methodology employs a 15 nm technology based on the NanGate library[36]. The NanGate library is an open-source standard library used by researchers to evaluate watermarked IP design in terms of design overhead cost. The cell characterized information of each gate is used to generate the area and delay of each RTL component such as unit adder, multiplier etc. The area/delay of each RTL component is then fed into the HLS tool library. The HLS framework builds the watermarked datapath of the IP design and computes its area and schedule latency for design cost evaluation. This has been a standard practice in hardware security community and adopted in several prior works[23,30,31]. For experimental analysis, the proposed approach is integrated in publicly available industrial standard HLS tool[37] and executed on Intel Xeon processor with the following operating conditions: (a) pragma directive: resource constraints – adders, multipliers (b) scheduling: LIST algorithm, (c) Module library: 15 nm NanGate Open CL containing information of unit delay and unit area of each adder and multiplier. The benchmarks are adopted from[35,38]. The proposed approach is empirically evaluated in terms of security and design cost overhead analysis. The security analysis of the proposed approach has been performed using two security metrics: (a) probability of coincidence (Y_i) and (b) tamper tolerance (TT). The runtime of the proposed watermark embedding process is 363 ms for the CNN convolutional layer and 848 ms for the JPEG-CODEC.

Analysis of security strength and challenges/limitations of this work

Challenges/limitations

(a) Bio-mimicking DNA fingerprint profiling requires careful examination and selection of appropriate restriction enzymes, which are pivotal in generating the DNA fragments. In the proposed methodology, an IP vendor needs to carefully select and apply the RE enzymes to determine the exact number of base pairs and single strands in each DNA fragment.
(b) The embedding process of the proposed DNA fingerprint-based watermark requires careful swapping/local alterations of storage variables to registers during HLS. Without careful swapping, there are chances of register conflicts and violation of watermark constraints during the generation of HLS-based secure IPs.
(c) Further, during watermark constraint conversion process from watermark signature during HLS, the storage variable pairs corresponding to each signature bit, needs to be carefully generated. Without careful generation of storage variable pairs as watermark constraints, there are chances of incorrect mapping of signature bits to storage variable pairs. These are certain precautionary measures for an IP designer in order to generate the accurate watermark constraints corresponding to the DNA fingerprint.

<u>Challenges during IP piracy detection</u> During the process of IP piracy detection, regeneration of the original watermark constraints is needed for comparing with those extracted from the suspected chip. However, the regeneration of the original watermark constraints can be a challenging process as it requires multiple security information such as (a) the type of restriction enzymes used by the IP vendor, (b) the DNA fusion process to generate the final DNA signature sequence, (c) DNA encoding rule of the IP vendor, (d) AES encryption using IP vendor key, and (e) watermark constraints embedding rule of the IP vendor.

Security analysis

The probability of coincidence is an important metric for evaluating the security robustness of a system, particularly in terms of how likely an IP vendor's watermark may appear in an unsecured IP design. This metric, denoted as Y_i, measures false positive and the presence of digital evidence in the final secured hardware design. A lower Y_i value indicates stronger security, which signifies a lesser likelihood of coincidentally matching the watermark constraints in an unsecured design. Y_i is represented as[12,21,39]:

$$Y_i = \left(1 - \frac{1}{n}\right)^w \tag{4}$$

Where w is the embedded IP vendor's DNA-based watermarking constraints, and n is the total register required in the SDFG before embedding watermarking constraints. Tables 2 and 3 report the value of the probability of

Watermark strength	Registers	Y_i	TT
1280 bits	137	8.46E−5	2.08E+385
1408 bits	137	3.31E−5	7.08E+423
1536 bits	137	1.29E−5	2.41E+462
1664 bits	137	5.07E−6	8.20E+500
1792 bits	137	1.98E−6	2.79E+539
1920 bits	137	7.78E−7	9.49E+577
2048 bits	137	3.04E−7	3.23E+616

Table 4. Computed values of Y_i and TT for proposed approach corresponding to JPEG-CODEC application.

Watermark strength	Registers	Y_i	TT
384 bits	36	2.18E−16	2.08E+385
512 bits	36	5.94E−18	7.08E+423
640 bits	36	1.61E−19	2.41E+462
768 bits	36	4.38E−21	8.20E+500
896 bits	36	1.19E−22	2.79E+539
1024 bits	36	3.23E−24	9.49E+577
1152 bits	36	8.78E−26	3.23E+616

Table 5. Computed values of Y_i and TT for proposed approach corresponding to convolution layer application.

Approaches	Watermark strength	Y_i	TT
Proposed approach	2048	3.04E−7	3.23E+616
Watermarking, 2005[21]	240	1.72E−1	1.76E+72
Automated watermarking,2008[22]	160	3.09E−1	1.46E+48
Watermarking, 2011[23]	256	1.53E−1	1.57E+77
Pragma based watermarking, 2021[11]	71	5.94E−1	NA
FSM watermarking, 2022[29]	128	3.91E−1	3.40E+38
IP steganography, 2019[25]	203	2.26E−1	NA
Facial biometric, 2021[12]	83	5.44E−1	9.67E+24
DNA biometric, 2022[30]	128	3.91E−1	3.04E+38
Hard sign watermarking, 2024[31]	826	2.35E−3	4.47E+248

Table 6. Comparison of achieved Y_i and TT through the proposed approach with other related techniques.

coincidence for the proposed approach with variation in the watermark signature bits (strength) corresponding to JPEG-CODEC and convolutional layer applications. As shown in Tables 4 and 5, the value of Y_i for the proposed approach decreases with an increase in watermark signature size, which is desirable in the case of a robust security methodology. Further, Table 6 depicts the comparison of Y_i among the proposed approach and[11,12,21–23,25,29–31]. The proposed approach depicts more robust security with a lower value of Y_i than all similar prior approaches. This is due to the generation and embedding of a larger number of watermarking constraints, thereby lowering the Y_i value and increasing the difficulty for attackers to locate the same security constraints in an unsecured design.

Next, tamper tolerance (TT) is a critical measure of a system's robustness against brute-force and tampering attacks. An increase in TT value signifies the generation of more signature combinations (search space), complicating an attacker's task of decoding the precise signature combination and identifying the secret watermarking constraints. This higher TT value enhances the system's resilience against tampering. The TT metric is represented as[12,21,39]:

$$TT = E^w \tag{5}$$

Here, E denotes the number of different encoding variables and w is the total embedded watermark constraints. Tables 4 and 5 report the value of the tamper tolerance for the proposed approach with variation in the watermark signature bits (strength) corresponding to JPEG-CODEC and convolutional layer applications. As shown in Tables 4 and 5, the value of TT for the proposed approach increases with an increase in watermark signature size, which is desirable in the case of a robust security methodology. Further, comparisons in Table 6 illustrate that

Watermark strength	Registers pre embedding	Registers post embedding	Design cost pre embedding	Design cost post embedding	% Overhead
384 bits	137	137	0.157	0.157	0
512 bits	137	137	0.157	0.157	0
640 bits	137	137	0.157	0.157	0
768 bits	137	137	0.157	0.157	0
896 bits	137	137	0.157	0.157	0
1024 bits	137	137	0.157	0.157	0
1152 bits	137	137	0.157	0.157	0

Table 7. Design cost pre and post embedding watermark in JPEG-CODEC IP design.

Watermark strength	Registers pre embedding	Registers post embedding	Design cost pre embedding	Design cost post embedding	% Overhead
384 bits	36	36	0.31	0.31	0
512 bits	36	36	0.31	0.31	0
640 bits	36	36	0.31	0.31	0
768 bits	36	36	0.31	0.31	0
896 bits	36	36	0.31	0.31	0
1024 bits	36	36	0.31	0.31	0
1152 bits	36	36	0.31	0.31	0

Table 8. Design cost pre and post embedding watermark in convolutional layer IP design.

the proposed method achieves a higher *TT* value compared to other similar prior approaches, i.e[11,12,21–23,25,29–31], thereby providing superior tamper tolerance. This higher value indicates a robust DNA watermark signature (with a greater signature search space and size), making it harder for an attacker to guess the exact embedded watermark signature. Notably[11,25], do not report the *TT* value due to non-involvement of encoding variables.

Design cost analysis

The design cost analysis of the proposed methodology is performed using the design cost function depicted in Eq. (6)[12,21,30,31].

$$Design\ cost = \left(t_1\left(\frac{Latency}{L_{max}}\right)\right) + \left(t_2\left(\frac{Area}{A_{max}}\right)\right) \tag{6}$$

Where $t_1 = t_2 = 0.5$ for giving equal weightage to watermark embedded design latency and area, *Latency* and *Area* are design latency and area corresponding to watermarked hardware IP. Further, L_{max} and A_{max} are their corresponding maximum latency and area, respectively. Note: The design metrics area and latency of the watermarked IP designs reflect the design area and latency post-embedding of the watermark signature of the IP vendor. It reflects if any design area/latency overhead has occurred due to the embedding of the watermark signature.

The *area* and *latency* are computed using Eqs. (7) and (8)[12,21,30,31].

$$Area = \sum_{i=1}^{2} \left(A\left(S\right) * (\mathrm{S})\right) \tag{7}$$

$$\mathrm{Latency} = \left(\left(Z_m * L_m\right) + \left(Z_a * L_a\right)\right) \tag{8}$$

Where *A(S)* denotes the area of resource type *(S)* and (*S*) denotes the quantity of utilized instances for a specific resource type, Z_m and Z_a denote control steps required by multiplier and adder, and L_m and L_a denote latency/delay of multiplier and adder. *Note: The other possible factors used in the design cost function are power dissipation and execution time.* Tables 7 and 8 report the registers needed in the IP design before embedding watermarking constraints, registers needed in the IP design after embedding watermarking constraints and design cost for the proposed approach corresponding to JPEG-CODEC and convolutional layer applications. It has been a standard practice in the scientific community[11,12,21–23,25,29–31] to only report the register count needed for storing the storage variables in the data flow graph execution. The offline storage requirement for other purposes such as DNA signature is never considered for reporting, as this is not part of the IP design/SoC design. All prior wor ks[11,12,21–23,25,29–31] have reported through similar practice. The proposed watermarking methodology does not add any extra design cost overhead (in terms of register count) post-integration of watermarking constraints. The register count reflects the number of storage hardware used in the HLS register binding stage (shown in Table 3 earlier reflecting the requirement of 36 registers viz. X1-X36) of the proposed methodology, which in turn indicates the storage hardware count in the HLS-generated register transfer level (RTL) datapath of the watermarked IP core. These usually store the primary inputs, intermediate outputs of the datapath computation,

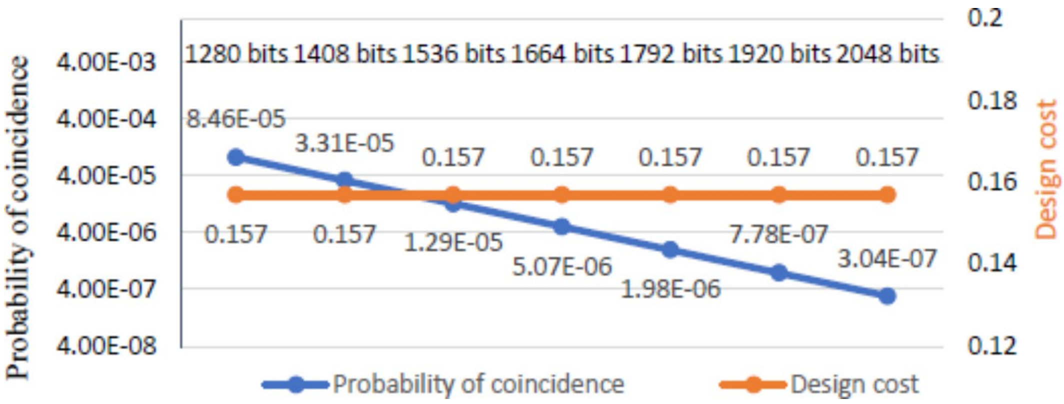

Fig. 13. Probability of coincidence vs. design cost tradeoff for JPEG-CODEC application.

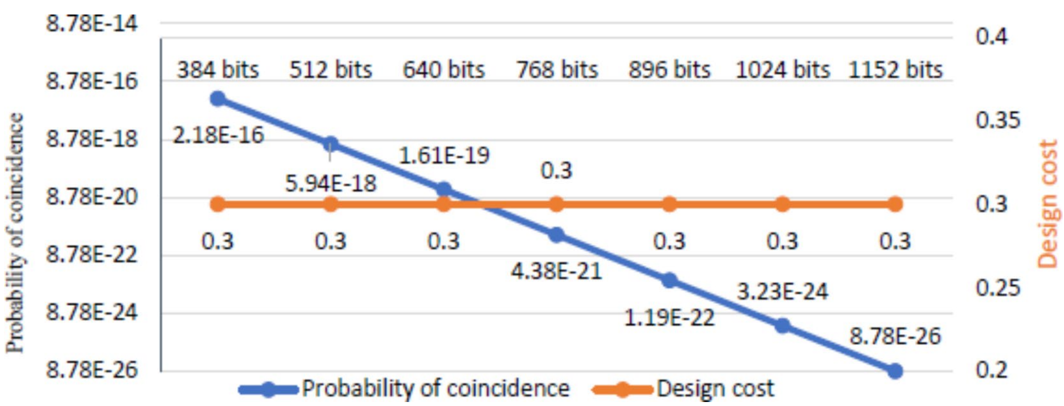

Fig. 14. Probability of coincidence vs. design cost tradeoff for convolutional layer application.

and primary output. Since the proposed approach does not store the encryption bits and DNA watermark signature, these do not consume any register space for their own storage.

Figures 13 and 14 demonstrate the probability vs. design cost tradeoff curve for the proposed approach corresponding to JPEG-CODEC and convolution layer applications. As shown in Figs. 13 and 14, the Y_i value decreases with an increase in watermark signature size without incurring any design cost overhead. Since, in the proposed approach, all the generated watermark constraints are embedded through local swapping/alteration of storage variable assignment to registers, therefore it does not result into any register overhead. Further, during the proposed watermark embedding process, the functional units (FUs), such as adders and multipliers of the design, are kept intact (preserved). Hence, there is no FU overhead in the watermarked design. Moreover, the proposed watermark only embeds its respective constraints during the register binding stage and does not embed during scheduling, interconnect design or FU assignment phases. Therefore, there is no delay/latency overhead due to watermark embedding. In summary, the proposed approach, therefore, does not incur area or latency overhead, thereby resulting in zero design overhead (as shown in Figs. 13 and 14). It is, therefore, evident from these figures that an increase in watermark strength from 384 to 2048 bits results in enhanced security at zero design cost overhead.

Table 9 shows the comparison of the proposed watermarking approach with the existing hardware watermarking methods in terms of various security/design properties. As evident from Table 9, the proposed approach overcomes various defects/shortcomings of the existing approaches, viz. hardware watermarking approaches[21,23,31], hardware steganography[25], facial biometric[12], and DNA biometric[30], in terms of enhanced security, lower design overhead, and greater resilience against standard threats for IP piracy/false ownership. Additionally, the comparison of different hardware watermarking techniques with the proposed hardware watermarking approach is shown in Table 10.

Security against forgery attack

An attacker may attempt to forge the IP vendor's DNA sequence to falsely prove IP ownership. However, he/she cannot use the forged DNA sequence to successfully generate the final encrypted DNA signature. The proposed DNA fingerprint profile-based hardware watermarking approach includes multiple security factors/layers (only known to the original IP vendor), which are crucial for generating the final encrypted watermarking constraints. The multiple security factors are as follows: (a) the usage of restriction enzyme types to perform the fragmentation

Security/design properties	Proposed DNA fingerprint-based watermark	Hardware watermarking[21,23,31]	Hardware steganography[25]	Facial biometric[12]	DNA biometric[30]
Numbers of security features generated	Very high	Lower	Lower	Lower	Moderate
Strength of ownership proof	Very high	Lower	Lower	Moderate	Moderate
Security design overhead	Zero	Lower	Lower	Lower	Lower
Resiliency against forgery attack/tampering	Strong	Moderate	Lower	Moderate	Moderate
Strength of the generated security constraints	Very high	Lower	Lower	Moderate	Moderate
Number of security layers for generating hardware security constraints	Very high	Lower	Lower	Moderate	Moderate
Robust authentication using IP vendor's natural identity	Yes	No	No	Yes	Yes
Usage of external hardware resources (such as camera, scanner, etc.)	No	No	No	Yes	No
Tamper tolerance ability	Very high	Lower	Lower	Moderate	Moderate

Table 9. Comparison of various watermarking methods with proposed watermarking approach.

Watermarking approaches	Features/characteristics of watermarking approaches				
	Crypto-logic for signature storage/generation	Usage of CIG for signature embedding	Usage of IP vendor/seller biometric	Signature embedding during FU allocation/FU binding/ Scheduling/Register binding	Tamper tolerance and probability of coincidence analysis
Dynamic watermarking[21], 2005	Yes	Yes	No	No	Yes
Numeric data-driven watermark[1], 2012	No	No	No	No	Yes
Single-phase watermarking[24], 2016	No	Yes	No	No	Yes
Pragma based watermarking[11], 2021	No	No	No	Yes	No
Code transformation-based watermarking[20], 2021	No	No	No	Yes	No
Facial biometric based watermarking[12], 2021	Yes	Yes	Yes	No	Yes
Palmprint biometric based watermarking[14], 2021	Yes	Yes	Yes	No	Yes
Proposed DNA fingerprint profile-based watermarking	Yes	Yes	Yes	Yes	Yes

Table 10. Comparison of hardware watermarking approaches based on characteristics/features.

of DNA sequence, (b) number of base pairs generated corresponding to the different DNA fragments, (c) number of single strands generated corresponding to the different DNA fragments, (d) replication factor of generated fragmented DNA sequence, (e) DNA alphabet encoding rule used to generate digit equivalent, (f) encryption key, and (g) mapping rule. All these security factors are completely unknown to the attacker. Therefore, the proposed approach effectively neutralizes forgery attack.

Security against Brute-Force attack (tamper Tolerance Analysis)

An attacker may attempt to tamper with or remove the original implanted encrypted DNA signature-based watermark constraints to evade IP piracy detection. The security algorithm's resistance to such an attack is known as tamper tolerance. A higher tamper tolerance value is particularly desirable because it signifies a larger signature search space with numerous possible combinations, thereby complicating the attacker's task of accurately guessing the embedded watermark signature. The proposed methodology reports a high tamper tolerance value, indicating its strong defense against tampering attack. This robustness is demonstrated in Section IV.A and illustrated in Tables 4 and 5, highlighting the proposed method's effectiveness in withstanding tampering attacks. The higher tamper tolerance ensures that the watermark remains intact and the IP design stays secure, providing security against unauthorized modifications.

Security against ghost signature search attack and false positive/watermark collision (probability of coincidence)

To confirm original authorship, the embedded secret watermark must be easily verifiable, ensuring that no unauthorized third party can accidentally claim it (watermark collision or ghost signature search attack). A crucial metric is the probability of coincidence, which measures the likelihood of a false positive match in an unsecured IP design with the original watermark constraints. Thus, a lower probability of coincidence signifies stronger security and credibility. The effectiveness of the proposed method is illustrated in Section IV. A and detailed in Tables 4 and 5, where the proposed approach demonstrates a lower value for the probability of coincidence as compared to similar prior approaches.

Conclusion

This paper presented a novel hardware watermarking methodology, which bio-mimics DNA fingerprint profiling framework to generate a robust IP vendor's DNA watermark signature. The proposed approach uses the IP vendor's DNA sequence to generate its corresponding watermark through bio-mimicking DNA fingerprint profiling steps such as DNA sequencing, DNA fragmentation, fragment replication, DNA ligase, etc. The generated watermark signature is then converted into watermarking constraints and implanted into the hardware application (such as JPEG-CODEC and convolutional layer applications) during the register allocation phase of the HLS process. The proposed approach depicts robust and stronger security in terms of security metrics such as probability of coincidence and tamper tolerance than prior similar approaches. The proposed approach depicts improvement in the probability of coincidence of up to $\sim 10^4$ and tamper tolerance of up to $\sim 10^{368}$ at 0% design cost overhead as compared to the prior approach. The embedded IP vendor's DNA watermarking constraints provide sturdy detective countermeasure against IP piracy and false IP ownership claim.

Data availability

The datasets used and/or analyzed during the current study is available from the corresponding author on reasonable request.

Received: 20 June 2024; Accepted: 13 September 2024
Published online: 28 September 2024

References

1. Le Gal, B. & Bossuet, L. Automatic low-cost IP watermarking technique based on output mark insertions. *Des. Autom. Embed. Syst.***16**, 71–92 (2012).
2. Schneiderman, R. DSPs evolving in consumer electronics applications. *IEEE Signal. Process. Mag*. **27**(3), 6–10 (2010).
3. Albawi, S., Mohammed, T. A. & Al-Zawi, S. Understanding of a convolutional neural network. in *Proc. ICET*, pp. 1–6. (2017).
4. Dai, D. An introduction of CNN: Models and training on neural network models. in *International Conference on Big Data, Artificial Intelligence and Risk Management (ICBAR), Shanghai, China, 2021*, pp. 135–13. (2021).
5. Guo, K. et al. Angel-Eye: A Complete design flow for mapping CNN onto customized hardware. in *Proc. ISVLSI*, pp. 24–29. (2016).
6. Putra, R. E. et al. Implementation of convolutional neural network in the development of object recognition system. in *Sixth International Conference on Vocational Education and Electrical Engineering (ICVEE), Surabaya, Indonesia, 2023*, pp. 286–290. (2023).
7. Bai, L., Zhao, Y. & Huang, X. A CNN accelerator on FPGA using depthwise separable convolution. in *IEEE Trans. Circuits Syst., II, Exp. Briefs*. **65**(10), pp. 1415–1419 (2018).
8. Rostami, M., Koushanfar, F. & Karri, R. A primer on hardware security: Models, methods, and metrics. in *Proceedings IEEE*, **102**(8), pp. 1283–1295 (2014).
9. Tehranipoor, M. & Knapp, C. T1A: Opportunities and challenges for secure hardware and verifying trust in integrated circuits. in *27th IEEE International System-on-Chip Conference (SOCC), Las Vegas, NV, USA* (2014).
10. Tehranipoor, M., Guin, U. & Forte, D. Hardware IP watermarking. in *Counterfeit Integrated Circuits*. (Springer, 2015). https://doi.org/10.1007/978-3-319-11824-6_10.
11. Chen, J. & Schafer, B. C. Watermarking of behavioral IPs: A practical approach. in *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE), France*, pp. 1266–1271 (2021).
12. Sengupta, A. & Rathor, M. Facial biometric for securing hardware accelerators. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **29**(1), 112–123 (2021).
13. Kahng, A. B. et al. Watermarking techniques for intellectual property protection. in *Proceedings 1998 Design and Automation Conference. 35th DAC. (Cat. No.98CH36175), San Francisco, CA, USA*, pp. 776–781 (1998).
14. Sengupta, A., Chaurasia, R. & Reddy, T. Contact-less palmprint biometric for securing DSP coprocessors used in CE systems. in *IEEE Transactions on Consumer Electronics*, vol. 67, no. 3, pp. 202–213 (2021).
15. Wang, X., Zheng, Y., Basak, A. & Bhunia, S. IIPS: Infrastructure IP for secure SoC design. in *IEEE Trans. Comput*. **64**(8), pp. 2226–2238 (2015).
16. Koushanfar, F. et al. Can EDA combat the rise of electronic counterfeiting? *DAC Des. Autom. Conf.***2012**, 133–138 (2012).
17. Arafin, M. T., Stanley, A. & Sharma, P. Hardware-based anti-counterfeiting techniques for safeguarding supply chain integrity. in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–4 (2017).
18. Hroub, A. & Elrabaa, M. E. S. SecSoC: A secure system on chip architecture for IoT devices. in *2022 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 41–44 (2022).
19. Colombier, B. & Bossuet, L. Survey of hardware protection of design data for integrated circuits and intellectual properties. *IET Comput. Digit. Tech.***8**(6), 274–287 (2015).
20. Badier, H., Pilato, C., Lann, J. C. L., Coussy, P. & Gogniat, G. Opportunistic IP birthmarking using side effects of code transformations on high-level synthesis. in *2021 IEEE Design, Automation & Test in Europe Conference & Exhibition (DATE), Grenoble, France.* (2021).
21. Koushanfar, F., Hong, I. & Potkonjak, M. Behavioral synthesis techniques for intellectual property protection. *ACM Trans. Des. Autom. Electron. Syst.***10**(3), 523–545 (2005).
22. Castillo, E. et al. Automated signature insertion in combinational logic patterns for HDL IP core protection. in *2008 4th Southern Conference on Programmable Logic, Bariloche, Argentina*, pp. 183–186 (2008). https://doi.org/10.1109/SPL.2008.4547753
23. Yu, T. & Zhu, Y. A new watermarking method for soft IP protection. in *International Conference on Consumer Electronics, Communications and Networks (CECNet), Xianning, China, 2011*, pp. 3839–3842 (2011).
24. Sengupta, A. & Bhadauria, S. Exploring low cost optimal watermark for reusable IP cores during high level synthesis. *IEEE Access.* **4**, 2198–2215 (2016).
25. Sengupta, A. & Rathor, M. IP core steganography for protecting DSP kernels used in CE systems. *IEEE Trans. Consum. Electron.***65**(4), 506–515 (2019).
26. Rizzo, S., Bertini, F. & Montesi, D. Fine-grain watermarking for intellectual property protection. *EURASIP J. Info Secur.* 10 (2019).
27. Potlapally, N. Hardware security in practice: Challenges and opportunities. in *IEEE International Symposium on Hardware-Oriented Security and Trust, San Diego, CA, USA, 2011*, pp. 93–98 (2011).
28. Islam, S. A., Sah, L. K. & Srinivas Katkoori High-level synthesis of Key-obfuscated RTL IP with Design Lockout and Camouflaging. *ACM Trans. Des. Autom. Electron. Syst.***26**, 1 (2020).
29. Karmakar, R., Jana, S. S. & Chattopadhyay, S. A cellular automata guided finite-state-machine watermarking strategy for IP protection of sequential circuits. *IEEE Trans. Emerg. Top. Comput.*, **10**(2), 806–823 (2022).

30. Sengupta, A. & Chaurasia, R. Securing IP cores for DSP applications using structural obfuscation and chromosomal DNA impression. *IEEE Access.***10**, 50903–50913 (2022).
31. Rathor, M. & Rathor, G. P. Hard-sign: A hardware watermarking scheme using dated handwritten signature. *IEEE Des. Test.***41** (2024).
32. University of California. Riverside, http://www.faculty.ucr.edu/~mmaduro/random.htm. Last accessed on June 2024.
33. Bioinformatics, D. N. A. sequence suite. https://www.bioinformatics.org/sms2/random_dna.html. Last accessed on June 2024.
34. Universitet, A. https://users-birc.au.dk/~palle/php/fabox/random_sequence_generator.php. Last accessed on June 2024.
35. Sengupta, A. & Chaurasia, R. Secured convolutional layer IP core in convolutional neural network using facial biometric. in *IEEE Trans. Consum. Electron.*, **68**(3), pp. 291–306 (2022).
36. NanGate 15 nm Open Cell Library. [Online]. Available: June. (2024). http://www.nangate.com/?pageid=2328.
37. CAD for Assurance, Crypto-Steganography, H. L. S. & Tool, F. IEEE CEDA and IEEE HSTTC, (2024). https://cadforassurance.org/tools/ip-icprotection/crypto-steganography-tool/
38. University of California Santa Barbara Express Group, accessed on [Online]. Available: May. (2024). http://express.ece.ucsb.edu/benchmark
39. M. Potkonjak. Methods and systems for the identification of circuits and circuit designs, USPTO, US7017043B1 (2006).

Acknowledgements
This work is technically and financially supported by CSIR grant no. 22/0856/23/EMR-II.

Author contributions
Anirban Sengupta - ideation, development of idea, experiments, paper writing and supervisionNabendu Bhui - implementation workAditya Anshul - experiments and paper writingVishal Chourasia - demonstration and experiments.

Declarations

Competing interests
The authors declare no competing interests.

Approval for human experiments
We confirm that all methods were carried out in accordance with relevant guidelines and regulations.We confirm that all experimental protocols were approved by Indian Institute of Technology Indore.We confirm that informed consent was obtained from all subjects and/or their legal guardian(s).

Additional information
Correspondence and requests for materials should be addressed to A.S.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.