Low overhead symmetrical protection of reusable IP core using robust fingerprinting and watermarking during high level synthesis

Dipanjan Roy, Anirban Sengupta*

Discipline of Computer Science and Engineering, Indian Institute of Technology Indore, India

HIGHLIGHTS

- Proposed approach offers symmetrical protection at less than 1% area overhead.
- Proposed approach offers symmetrical protection at less than 1.1% latency overhead.
- Our work leverages HLS steps to embed vendor watermark and buyer fingerprint in IP design.

ABSTRACT

Intellectual Property (IP) core used in computing system-on-chip provides a unique blend of yielding enhanced design productivity with reduced design cycle time. However, leveraging benefits of IP core require protection against threats from both seller’s and buyer’s perspective. This paper proposes a novel symmetrical IP core protection methodology that embeds a buyer fingerprint and seller watermark simultaneously during high level synthesis (HLS). The proposed work leverages major HLS steps to concurrently embed buyer fingerprint signature and seller watermark signature into a reusable IP core design. The proposed signature encoding for fingerprint and watermark is multi-variable in nature offering strong robustness, low embedding cost and low design overhead. Results on standard benchmarks indicated that the proposed symmetrical approach satisfies all the major protection features of a watermark and fingerprint such as strong robustness to both seller & buyer, low overhead, low runtime and low embedding cost. Further on comparison with baseline design (no protection), the proposed approach offers symmetrical protection (both buyer and seller) at less than 1% area overhead and less than 1.1% latency overhead. Additionally on comparison with a recent unsymmetrical approach, the proposed approach offers symmetrical protection (both buyer and seller) at 0% area overhead and less than 1.1% latency overhead.

© 2017 Elsevier B.V. All rights reserved.

1. Introduction

With the sky-rocketing progress of application and implementation technology in the domain of system-on-chip (SoC) based products, designs have become more sophisticated and innovative. In order to address these complex design challenges, a balance between design productivity and time-to-market is required, which is where the use of reusable intellectual property (IP) cores [1] designed through high level synthesis is essential. High Level Synthesis [2] is a process of converting a behavioral/algorithmic description of an IP into its equivalent register transfer level counterpart consisting of several sub-steps like compilation, transformation, scheduling, allocation and binding. Reusable IP cores not only expedite the productivity of complex design process but also assist in cost reduction. However for sustainable employment of reusable IP cores in complex designs, its protection against threats is highly critical. In case of a reusable IP core there are two entities involved viz. buyer and seller. Let us look at the protection aspects from both buyer’s and seller’s perspective: In a reusable IP core, an IP buyer may demand exclusive user right as a buyer i.e. would not want the same IP copy to be resold/redistributed in the market. This happens when an IP buyer procures an IP based on his custom specifications from an IP seller, thus creating an unique mapping between an IP seller and an IP buyer. Buyer fingerprint facilitates tracing of illegally resold/redistributed copies of an IP core by a dishonest IP seller [3]. Similarly an IP core seller, when selling his design to a buyer, must protect his work from piracy/jorgery and false claim of ownership [4–11].
Proposes threat model for seller: Proposes a symmetrical IP core protection methodology offers presents our multi-variable fingerprinting methodology that embeds a symmetrical IP core protection during high level synthesis that incorporates seller watermark and buyer fingerprint encoding simultaneously.

Proposes symmetrical protection methodology obtains extremely low overhead design in terms of hardware area and latency.

Proposes a symmetrical IP core protection methodology offers higher robustness, lower design overhead/embedding cost, fault tolerance, and faster signature encoding/decoding.

The above differences also highlight the novelties over our previous work [14]. In this paper, we propose a novel low overhead symmetrical protection methodology for reusable IP core during high level synthesis from both buyer’s and seller’s standpoint by employing robust fingerprint and watermarking respectively.

Threat model: The proposed work protects a reusable IP core from threats for two different parties: buyer and seller.

- Threat model for buyer: Tracing illegally resold/redistributed copies of a reusable IP core by a dishonest IP seller, thus providing exclusive user right to the buyer. Accomplished through buyer fingerprint in proposed work.
- Threat model for seller: Protection of ownership of the seller against false claim of ownership. Also protection against IP piracy and IP counterfeiting. Accomplished through seller watermark in proposed work.

Target technology/platform: Our proposed symmetrical IP protection methodology can be easily integrated with any EDA tools of current generation. Hardware description language (HDL) or any high level language used for IP generation can easily merge with proposed technique in the design tools.

Why symmetrical IP core protection at architecture level (during HLS)? HLS for digital ICs is way matured starting from performance optimization, power optimization, to process variation optimization, conducted at the architecture level [15, 16]. In the current era of smart devices, reusable IP core based designs are essential to meet the time to market demand where HLS techniques can play more crucial role for the design engineers [17]. So, a natural progression of HLS research is to equip protection feature for reusable IP cores along the other challenges at the architecture level. Symmetrical IP core protection through embedding buyer fingerprint and seller watermark during HLS (during pre-synthesis phase) not only protects the lower level designs but also offers low overhead and lesser complexity/effort in implementation. Therefore, symmetrical IP protection during HLS in the form of concurrently embedded buyer fingerprint and seller watermark not only offers robust protection, low design overhead and low implementation runtime, but simultaneously provides necessary protection to both concerned entities involved.

2. Proposed approach: threat models, target platform and motivation

The novel contributions of the current paper are as follows:

- Proposes multi-variable fingerprinting methodology that embeds buyer’s signature during scheduling and register allocation phases of HLS to protect the reusable IP core from buyer’s perspective.
- Proposes a symmetrical IP core protection during high level synthesis that incorporates seller watermark and buyer fingerprint encoding simultaneously.
- Proposes symmetrical protection methodology obtains extremely low overhead design in terms of hardware area and latency.
- Proposes a symmetrical IP core protection methodology offers higher robustness, lower design overhead/embedding cost, fault tolerance, and faster signature encoding/decoding.

3. Related prior works

3.1. Background on fingerprint and watermark

Both watermark and fingerprint are the signature provided by two different parties to preserve their right in protection of a reusable IP core. While watermark carries the signature of the IP seller, fingerprint is meant for the IP buyer. In the domain of IP protection a watermark and/or fingerprint should satisfy the following major properties as hidden signature:

Fig. 1. Motivation for symmetrical IP core protection.

This necessitates robust symmetrical protection of reusable IP cores from both buyer’s and seller’s perspective. A diagrammatic overview of proposed symmetrical IP core protection is shown in Fig. 1. It is well acknowledged that most of these IPs need substantial time and effort to be designed and verified, yet they can be easily resold, copied, or modified. Buyer and seller of IP designs thus demand assurances that their content will not be illegally redistributed and falsely claimed. Protection against such complex challenges is implemented by embedding dual entity robust signature scheme during high level synthesis in the form of hidden buyer signature and hidden seller signature in a reusable IP core unit.

Existing intellectual property protection schemes [12] like copyrights, patents, trademarks, trade-secret, industrial design rights etc. are quite scattered, limited in possibilities and inadequate in protecting reusable IP cores from buyer and seller perspective. The ultimate two objectives of reusable IP core protection is: (a) from buyer’s perspective: trace illegally resold/redistributed copies by a dishonest seller (b) from seller’s perspective: protect against piracy/forgery and false claim of ownership [13]. In past few decades various signature hiding schemes have been proposed in different contexts of multimedia, like text, audio, image, video etc. But, all these techniques are inapplicable to reusable IP core protection as it may change functionality and accuracy during signature insertion. Thus buyer fingerprint and seller watermark signature insertions in the context of IP core protection must not only provide necessary protection to both entities but also preserve the correct functionality of the design. Although it is a well acknowledged that it may not be easy to embed protection mechanisms at zero design overhead (area and latency), however the goal is to minimize it as much as possible without compromising the protection capability of an IP. This paper proposes a novel low overhead symmetrical protection methodology for reusable IP cores based on robust multi-variable buyer fingerprint and seller watermark that incorporates hidden constraints during scheduling and register allocation steps of high level synthesis process. (Note: Latency/Area overhead is the extra delay/area incurred in a design due to insertion of watermarking or fingerprinting constraints).

The rest of the paper is as arranged as follows: Section 2 discusses novel contributions of this paper. Section 3 discusses about the background on fingerprint and watermark, related major approaches on IP core protection, while Section 4 presents our proposed low cost symmetrical IP protection HLS methodology. Further, experimental results have been presented in Section 5, followed by conclusion in Section 6.
Should be resilient against tampering. Should be robust/fault tolerant and difficult to detect/remove by an attacker. Should be resilient against tampering. Runtime of embedding process should be low. Latency and hardware overhead after embedding signatures should be minimal. For genuine owner with complete knowledge of encoding rules, detection process should be smooth [18,19]. Should preserve correctness and functionalities of an IP core after embedding signatures.

The last property imposes an additional restriction on the signature insertion process of an IP compared to artifacts in case of images. The proposed symmetrical, multi-variable signature encoding approach during HLS not only satisfies the aforementioned properties, but also explores a low cost optimal watermarked–fingerprinted design solution.

3.2. Prior works on IP protection

The literature suggests that there is no previous work on symmetrical protection (for buyer and seller) of reusable IP core during high level synthesis (i.e. at architecture level). The prior works on IP core protection only focuses on seller watermark, thus new innovations are needed that provide symmetrical IP core protection from both buyer and seller perspective during high level synthesis (at architecture level). For example, single entity protection [20–22] (though seller watermark) have been proposed at lower abstraction level of design process, which not only does not provide design protection at higher abstraction level and incurs overhead, but also does not protect the buyer. On the contrary, symmetrical protection through dual entity signature scheme i.e. buyer fingerprint and seller watermark at higher abstraction layer, not only protects lower design level and incurs lower overhead, but also protects both buyer and seller. Therefore, proposed symmetrical IP protection in the form of concurrently embedded buyer fingerprint and seller watermark during High Level Synthesis (HLS) offers enhanced protection, low overhead and low implementation complexity, as well as simultaneously provides protection to both concerned parties (see Fig. 2).

Signature of watermarking technique in [20] is embedded at the netlist and bit-file level. Using this technique, seller’s signature can be identified at the power supply pins of the FPGA. After measuring the supply voltage of the core a signature detection algorithm is used to detect the watermark. In [21], a hierarchical watermarking technique for FPGA IP protection is presented. The seller’s watermark is inserted into the netlist using a look-up table-SRL transformation, thereafter watermark is embedded into the bitstream of the same design. Authors’ in [22] detected the signature from a voltage trace, which is inserted into the functional parts of the watermarked core. The voltage is sampled, analyzed and decoded using signature detection algorithm before comparing with the original signature. All the aforementioned watermarking techniques are performed at lower abstraction level (not at behavioral/architectural level). However, there are few works [18,23,14,24,25] that have performed seller watermarking based protection at higher abstraction level. In non-symmetrical IP protection approaches [18,23], the watermarks are embedded in the form of additional edges in a colored interval graph during register allocation phase of HLS. The encoding scheme of the watermark uses two variables (0 and 1) for signature encoding. The watermark has fault tolerance capability as the watermarking constraints are evenly distributed throughout the design. Further, it is covert and does not damage the functionality of the design. However since the encoding on only based on two variables, thus it may be easy for attackers to tamper with the signature if both variable encoding rules are known somehow. On the contrary, in [25,14], the proposed approach provided non-symmetrical IP core protection in the form of embedded watermark in register allocation phase of HLS which is based on multi-variable signature encoding. Multi-variable signature encoding provides better robustness as the encoding process is more complex resulting into more watermarking constraints in the design. Further, the aforesaid works have proposed a low cost approach during register allocation phase of HLS using particle swarm optimization (PSO) driven exploration process. PSO based exploration performs a trade-off between latency and area overhead achieved during watermarking and yields an optimal low cost solution. Moreover, authors’ in [24] have presented an in-synthesis IP watermarking scheme. The approach uses marking scheme based on mathematical relationships between numeric values as inputs and outputs at specified time. The embedded watermark protects the sellers right while satisfying the user constraints in terms of design area and latency.

All aforesaid watermarking techniques [18,23–25,14] provide non-symmetrical IP core protection from seller’s perspective, thereby does not provide protection of right from buyer’s perspective. The only work [26] for symmetrical IP protection that exists is at the lower abstraction level. The approach provided protection from seller’s as well as buyer’s perspective and thus protects the rights for both the parties however, it does not provide protection of designs generated at higher abstraction level. Additionally, neither of the aforesaid approaches have embedded buyer fingerprint during scheduling phase in HLS. Therefore there is absolutely no work has been done so far which provided symmetrical IP core protection at higher abstraction level viz. behavioral/architectural level.

Along with the signature embedding techniques for IP core protection, few non-signature type IP core protection mechanisms are also available. Obfuscation is one of them. It accepts a circuit as an input and generate a new representation of it, having same functionality as the original [27]. The newly generated circuit makes the reverse engineering process more complex and hard for attackers. In [28] authors’ reported a design scheme using netlist-level obfuscation to protect the hardware IP. Selected node of the design and state transition function is modified in the aforesaid approach to achieve a key-based obfuscation. Computational forensic engineering (CFE) is another non-signature based IP protection mechanism. It tries to judge the likelihood whether it was generated by a well-known source [29] or not. IP metering [30], is another non-signature type IP core protection mechanism which identify unauthorized duplicate of an IP core. In hardware metering, [31], a unique ID is inserted through programming into the design of the IP core, which helps to identify the unauthorized copy. This unique ID is generated using a different register allocation or scheduling. As a result, the IP controller changed but, the functionalities and correctness remains same. This unique reconfigured controller is programmed at the end to provide a unique ID for every manufactured chip. This paper advances state-of-the-art by proposing a novel low overhead symmetrical protection methodology for reusable IP cores based on robust buyer fingerprint and seller watermark that incorporates hidden constraints during scheduling and register allocation steps of HLS.
4. Proposed low overhead symmetrical IP core protection during HLS

The proposed approach provides symmetrical IP core protection to the buyer as well as seller by covertly inserting fingerprinting and watermarking constraints during high level synthesis. An IP seller besides inserting his own watermark, additionally inserts buyer fingerprint (based on the request of the buyer) into the design during HLS. This protects the rights of both the seller and buyer against threats (discussed in Sections 1 and 2). As shown in Fig. 3, the buyer fingerprint signature is decoded into constraints before embedding into an HLS design. The HLS engine then accepts the decoded constraints of the sellers watermark signature. The combination of seller watermark and buyer fingerprint is implanted into the design in successive phases within the HLS engine, after both have been decoded to its equivalent constraints. Firstly, the buyer fingerprint is decoded and embedded in scheduling phase inside HLS engine. Once fingerprint embedded schedule is obtained, finally the seller watermark is decoded and embedded in register allocation phase inside HLS engine. Thus the HLS design now embeds both buyer fingerprint and seller watermark besides its regular design constraints (such as resource constraints, library etc.). Based on the watermarked and fingerprinted schedule and register allocation information, the multiplexing scheme is developed for each resource to determine the interconnectivity details, type/size of multiplexer and inputs/outputs. Based on the multiplexing scheme developed for each resource, the final datapath block diagram of an IP is created, followed by timing specification of the controller. The final output is a symmetrically protected RTL description (datapath and controller designs) of a reusable IP core. The RTL description of a reusable IP core of an application is in the form of a datapath hardware description language (HDL) or schematic and control HDL. Then the HDL files are imported into commercial CAD logic synthesis tools for RTL synthesis. RTL synthesis yields optimized gate level structure of the symmetrically protected design which is further implemented, simulated and converted to respective bitstreams for emulation on a reconfigurable platform such as field programmable gate array (FPGA). Once verified in emulation it is further imported to physical design tools for floorplanning, routing and placement, followed by layout generation. Finally, the design file of the application specific integrated circuit generated is sent to foundry for fabrication.

4.1. Problem formulation

For a given data flow graph (DFG) determine an optimal symmetrical IP core protected (in the form of embedding buyer fingerprint and seller watermark) design solution within buyer given resource constraints, \( X_i = N(R_1), N(R_2), \ldots, N(R_D) \), at behavioral level; where \( N(R_D) \) is the number of resource type \( R_D \). The problem can be formulated as: Minimize: Low embedding cost \( A_T, L_T \) for an optimal design solution \( X_i \), Subject to: \( A_T \leq A_{\text{com}} \) and \( L_T \leq L_{\text{com}} \) and IP protection through robust watermarking and fingerprinting.

4.2. Proposed area model

Total area \( A_T \) consumed by watermark and fingerprint embedded solution can be expressed by the following model:

\[
A_T = \sum_{i=1}^{m} A(R_i) \times R_i + A(\text{mux}) \times N(\text{mux}) \\
+ A(\text{buffers}) \times N(\text{buffers}).
\]
Total area is the sum of area occupied by the hardware resources, interconnecting units and storage units. The area is evaluated with respect to module library according to CMOS 90 nm technology node.

4.3. Proposed delay model

Total delay \( L_T \) of a secured IP (in the form of watermark and fingerprint) is expressed as:

\[
L_T = \sum_{c=1}^{n} \text{Max}(D(op_1), \ldots D(op_r))
\]

where, \( 1 \leq i \leq r \) (\( i \) represents the respective operation and \( n \) is the maximum number of nodes). The delay model is based on the latency values of each functional hardware described at 90 nm technology scale.

4.4. Proposed cost function

The cost of each solution is evaluated through the following function:

\[
C_f(X_i) = w_1 \frac{L}{L_{\text{max}}} + w_2 \frac{X_i}{A_{\text{max}}}
\]

where, \( C_f(X_i) \) is the cost of the solution with resource configuration \( X_i \), \( L_{\text{max}} \) and \( A_{\text{max}} \) are the maximum possible area and latency in the design space, \( w_1 \) and \( w_2 \) are user defined weights both kept 0.5 to maintain equal preference.

4.5. Proposed buyer signature encoding and seller signature encoding

Embedding signature during HLS process can be achieved by imposing hidden additional constraints during its design steps. Scheduling and register allocation steps have been selected to demonstrate insertion of buyer fingerprinting and seller watermarking constraints in covert form. (Note: In the rest of the paper, while explanation we play the role of an IP seller).

In the proposed work, buyer fingerprint constraints have been embedded in both scheduling and register allocation step to increase robustness, while seller watermarking constraints embedded only in register allocation step. In the proposed approach, adding buyer fingerprint constraints (by an IP seller) in scheduling step is achieved by forcing specific operations in specific control step (CS) during schedule conflict resolution process. In other words, instead of employing priority resolver functions or random break, we employ fingerprint encoding rule to select which operation to assign to which CS. This is a very covert way of inserting buyer fingerprinting, as during regular operation scheduling conflict the fingerprint is inserted with zero hardware and minimal latency overhead. Further, the fingerprint and watermarking constraints during register allocation step in HLS are realized with the concept of colored interval graph (CIG), where the nodes of the graph represent storage variables and the edges represent overlapping lifetime between corresponding storage variables. In another words, if two storage variables exist in the same control step of a scheduler, their lifetime overlap, there will be an edge between them. Adding additional edges as constraints will force storage variables of a CIG to execute through distinct registers. The flow diagram of proposed symmetrical IP protection is shown is Fig. 4. The detailed encoding details is presented in upcoming section.

4.5.1. Fingerprint encoding

In proposed fingerprint encoding mechanism of our symmetrical approach, additional constraints are inserted both in the scheduling and register allocation steps of HLS (to strengthen robustness). Buyer fingerprint constraints in scheduling step is embedded by an IP seller by forcing specific operations in specific control steps (CS) during schedule conflict resolution process. The definition of proposed fingerprint signature encoding is as follows:

- \( x = \) Force even operation in odd control step while resolving scheduling conflict in scheduling phase.
- \( y = \) Force odd operation in even control step while resolving scheduling conflict scheduling phase.
- \( z = \) Encoded value of edge with node pair as (odd, odd) in colored interval graph.

In order to apply the fingerprint encoding for \( x \) and \( y \), operations are kept in a sorted list. Similarly to apply fingerprint encoding for \( z \), the storage variables in the scheduling must be kept in sorted order (Note: demonstration of proposed fingerprint approach through an example is shown in next section). The strength of buyer fingerprint increases with its signature size. Our proposed approach is flexible for accepting any large size fingerprint. As evident, proposed fingerprint signature scheme comprises of three unique variables and each variable is mapped with a unique encoding that does not add much overhead to the design while preserving robustness.

The fingerprint signature combination to be chosen by the buyer is recommended to have more \( x \) and \( y \) digits and less \( z \) digits. This is because both \( x \) and \( y \) fingerprint digit encodings are such that it imposes zero hardware area overhead and negligible latency overhead. This is possible because both fingerprint digits \( x \) and \( y \) are inserted during operation scheduling conflict resolution i.e. fingerprint encoding rule is used as conflict resolver rule. But many \( z \) digits in fingerprint signature may result in hardware overhead, however, it is crucial as it adds another protection layer by embedding fingerprint constraint in the register allocation stage.

4.5.2. Watermark encoding

In the proposed symmetrical approach, watermarking is applied using methodology in [14]. In [14], additional constraints for watermark are inserted in the register allocation step of HLS by adding additional edges between the nodes of a CIG. Adding these additional edges will force storage variables of a CIG to execute through distinct registers. This approach is flexible for adding any number of additional edges in the register allocation step. The strength of the watermark increases with the number of additional edges (i.e. watermark signature size). Though robustness increases with increase in size of watermark signature, a large signature may incur significant hardware overhead.

Therefore it is necessary to employ a robust watermarking scheme that provides substantial protection with minimal overhead. Our proposed symmetrical approach employs a multi-variable watermark creation mechanism from our recent work [14] as discussed in the following section.

As proposed in [14], the mechanism of watermark creation consist of four different variables. Each variable carries a different encoding rule. The seller is free to choose any random combination of these four variables as his watermark. Each variable adds an additional edge between two specific nodes, encoded as the rule of that variable. In other words, selection of node pair from CIG to add an additional edge is defined in the variable as encoding mechanism. The definition of each variables in the watermark scheme is as follows:
Fig. 3. Proposed symmetrical IP core protection during high level synthesis.

Fig. 4. Proposed symmetrical IP core protection behavioral synthesis methodology with buyer fingerprint and seller watermark.

- \( i \) = encoded value of edge with node pair as \((\text{prime, prime})\)
- \( I \) = encoded value of edge with node pair as \((\text{even, even})\)
- \( T \) = encoded value of edge with node pair as \((\text{odd, even})\)
- \( ! \) = encoded value of edge with node pair as \((0, \text{any integer})\)
Although ‘z’ digit of buyer fingerprint and ‘i’, ‘l’, ‘T’ and ‘!’ digits of seller watermark both embed constraints in register allocation phase, however their encoding rules are different resulting into completely different constraints.

4.6. Proposed process for symmetrical protection of IP core using multi-variable fingerprint and watermark

The broad and detailed steps for fingerprint and watermark creation process to achieve symmetrical IP protection in HLS are described below:

4.6.1. Fingerprint embedding process during HLS

**Broad steps:**

1. Select desired buyer signature.
2. Decode buyer signature to its equivalent constraints.
3. Use the decoded constraints to perform scheduling during operation conflict.
4. Assign storage variables to registers from the schedule using the concept of colored interval graph.
5. Insert additional edges in colored interval graph based on decoded constraints and perform re-assignment of register allocation.

**Detailed steps:**

1. Accept a preferred signature for fingerprint as arbitrary combination of: x, y, z from buyer.
2. Create a sorted list of operations of the DFG as per their number in increasing order.
3. Accept the watermark of the seller in the form of i, l, T, ! where these characters hold the encoded value of additional edges to be inserted.
4. Create a list of additional edge pairs corresponding to its encoded values by traversing the sorted nodes in step (1).
5. Insert the additional edges in the colored interval graph of fingerprint embedded design as watermark constraints.
6. Modify the timing table design of register allocation based on created watermark.

After performing the above steps, a seller watermark embedded IP design is also generated. Thus now the IP design embeds both buyer fingerprint and seller watermark. This indicates a reusable IP core with symmetrical protection where buyer’s fingerprint and seller’s watermark are both inserted.

4.7. Motivational example for proposed approach

The first subsection illustrates the proposed fingerprint creation process and the second subsection illustrates the proposed watermark creation process. As mentioned earlier, buyer fingerprint constraints of ‘x’ and ‘y’ are embedded in the scheduling step of HLS by assigning specific operations to specific control steps (CS) during schedule conflict resolution process, while variable ‘z’ adds additional edges in register allocation phase by forcing specific storage variables to execute in distinct registers. On the other hand, seller watermarking constraints in the form of ‘i’, ‘l’, ‘T’ and ‘!’ are embedded only in the register allocation step of HLS by forcing specific storage variables to execute in distinct registers.

4.7.1. Example of fingerprint embedding

Assuming MESA data flow graph and resource configuration (2 adders and 3 multipliers) provided as inputs. Next a desired buyer signature is chosen (using arbitrary combination of ‘x’, ‘y’ and ‘z’) as discussed in Section 4.5. For the purpose of demonstration, a 7-digit buyer fingerprint signature is selected: ‘x x y x y z z’. The fingerprint signature is then decoded. The corresponding decoded meaning is shown in Table 2. The decoded constraints are obtained using the fingerprint embedding process proposed in Section 4.6. Next, the MESA data flow graph needs to be scheduled using the resource constraint specified. In the first odd CS, there are four ready multiplication operations viz. opn 1, opn 2, opn 3 and opn 4, but only 3 multiplier resources are available. This indicates existence of schedule operation conflict scenario. The decoded fingerprint constraints will now be inserted in the schedule while resolving the operation conflict. For example, first digit ‘x’ forces even multiplication operation (opn 2) into odd control step (CS 1) during schedule conflict resolution. Similarly the next digit ‘x’ forces multiplication opn 4 in odd CS 1 during schedule conflict resolution. The third multiplication is scheduled in CS 1 by simply choosing the first operation in the sorted list. Thus opn 1 is selected and scheduled in CS 1. Similarly odd opn 5 was scheduled in even CS 2 after resolving operation conflict through fingerprint encoding of digit ‘y’. Further even opn 6 was scheduled in odd CS 3 after resolving operation conflict through fingerprint encoding of digit ‘y’. This process of embedding fingerprint constraints in scheduling continues until all operation conflicts are resolved. Since no further schedule operation conflict exists, hence the last three fingerprint digits could not be embedded. However scheduling is normally continued until all operations of DFG are scheduled based on resource constraint. The final schedule with buyer fingerprint...
embedded is shown in Fig. 5. Its schedule counterpart where no fingerprint embedding is performed during conflict resolution is shown in Fig. 6. As evident with absolutely zero hardware area and zero latency overhead, the buyer fingerprint is successfully embedded in the design without distorting the functionality.

Next the storage variables (v1–v17) are assigned in the obtained schedule and its corresponding CIG is created. Two additional edges corresponding to two ’z’ fingerprint digits are to be added as constraints in the CIG (representing register allocation phase). Although two additional edges have to be added, but coincidently an edge between (v1, v3) already exists by default. Thus, only new edge between (v1, v5) needs to be added (marked by dotted line in CIG of Fig. 7). The information of this new edge added in the CIG, needs to be accommodated in the timing table of register allocation (shown in Table 1). This is accommodated by executing

storage variables v1 and v5 in distinct registers (Red and Green). The modified controller with watermarking constraints embedded is shown in Table 3.

4.7.2. Watermark creation process

After embedding buyer fingerprint in the schedule, it now acts as an input for embedding seller watermark. For demonstration, a 7-digit watermark signature is selected: ‘i i i I i T !’. Its corresponding decoded constraints as additional edges is shown in Table 4. As evident from the CIG in Fig. 8, only two additional edges ((v2, v7), (v2, v9)) as constraints needs to be inserted in the CIG. This is because the remaining five edges are already present in the graph by default. Therefore, due to an extra edge inserted between v2 and v9, both are forced to execute through distinct registers. v2 and v7 are anyways executed through distinct registers as evident in Table 4. The final modified timing table for register allocation is shown in Table 5. It is clear that zero hardware area and latency overhead are incurred after embedding seller watermark as well.

Thus for MESA benchmark, embedding both buyer fingerprint and seller watermark for symmetrical IP protection incurred absolutely zero area and latency design.
Table 4
Watermark and its meaning.

<table>
<thead>
<tr>
<th>Desired watermark (7-digit)</th>
<th>Corresponding additional edges to insert between nodes of colored interval graph</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>(v2,v3)</td>
</tr>
<tr>
<td>i</td>
<td>(v2,v5)</td>
</tr>
<tr>
<td>i</td>
<td>(v2,v7)</td>
</tr>
<tr>
<td>l</td>
<td>(v2,v4)</td>
</tr>
<tr>
<td>i</td>
<td>(v2,v9)</td>
</tr>
<tr>
<td>T</td>
<td>(v1,v2)</td>
</tr>
<tr>
<td>l</td>
<td>(v0,v1)</td>
</tr>
</tbody>
</table>

Table 5
Final timing table for register allocation after embedding fingerprint and watermark.

<table>
<thead>
<tr>
<th>Control step</th>
<th>Red (R)</th>
<th>Green (G)</th>
<th>Blue (B)</th>
<th>Yellow (Y)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>V0</td>
<td>V1</td>
<td>V2</td>
<td>V3</td>
</tr>
<tr>
<td>1</td>
<td>V5</td>
<td>V4</td>
<td>V2</td>
<td>V6</td>
</tr>
<tr>
<td>2</td>
<td>V5</td>
<td>V7</td>
<td>V11</td>
<td>V9</td>
</tr>
<tr>
<td>3</td>
<td>V10</td>
<td>V8</td>
<td>V14</td>
<td>V9</td>
</tr>
<tr>
<td>4</td>
<td>V12</td>
<td>V13</td>
<td>–</td>
<td>V16</td>
</tr>
<tr>
<td>5</td>
<td>V15</td>
<td>V13</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>6</td>
<td>V17</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
</tbody>
</table>

Fig. 8. Colored interval graph embedded with buyer fingerprint constraints (additional edges in blue dots) and seller watermarking constraints (additional edges in red dotted line). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

4.9. Properties of generated fingerprint and watermark through proposed approach

In the proposed symmetrical IP core protection approach, the embedded signatures have the following properties:

- **Low embedding cost**: The watermarked–fingerprinted design solution should impose low overhead in terms of hardware area and latency. An ideal signature must have minimal hardware (register count and functional units) and latency overhead. In the proposed approach, a low cost watermarked–fingerprinted design solution is explored.

- **Resiliency**: A signature should be so strong and robust that it is difficult to remove or tamper. The proposed watermark and fingerprint are both equipped with strong protection. This makes the watermark and fingerprint extremely hard to remove or tamper without complete knowledge of both encoding rules. In addition to this, it is difficult for an attacker to locate the HLS step where the signature embedding scheme has been employed. Besides, we have distributed our signatures’ constraints throughout the IP design which makes it resilient to attacks.

- **Fault tolerance**: As our proposed signature scheme distribute the additional constraints throughout the design, ownership remains preserved even after partial removal or tampering by an adversary.

- **Adaptability to any CAD Tool**: The proposed signature embedding methodology is compatible to any new generation CAD tool. It can be integrated with any modern CAD tool flow seamlessly.

- **Signature creation and detection time**: Signature creation process should not be so complex that it takes too much to generate a signature. In our proposed approach, the signature creation time for both watermark and fingerprint is very less (our experimental results confirm this). Besides, the signature detection process is simple and straightforward for a genuine buyer who has the complete knowledge of encoding rules.

5. Experimental results

Our proposed approach provides a low overhead robust symmetrical IP core protection during high level synthesis. Following metrics are used to confirm the robustness and low overhead of proposed approach:

- Cost of embedding seller watermark and buyer fingerprint in IP core design which signifies quality of the solution.
- Hardware and storage overhead in terms of functional hardware units and extra register required after embedding seller watermark and buyer fingerprint mark.
- Time consumed to create signature.
- Quality of signature in terms of security/strength.
- Impact of large seller signature and buyer signature on hardware area, latency and embedding cost.
- Scalability of the approach to handle a large benchmark.

The result section is therefore divided into two sub-sections: (i) Result of proposed symmetrical protection approach in terms of latency, hardware area, cost and security metric (ii) Comparison of the proposed symmetrical IP core protection with unsymmetrical approach (Note: as no symmetrical IP core protection approach exists during HLS).
Table 6
Result for fingerprint and watermark solution through the proposed approach \((F = 30\) and \(W = 30\)).

<table>
<thead>
<tr>
<th>Benchmarks</th>
<th>Proposed solution</th>
<th>Total area ((\mu \text{m}^2))</th>
<th>Total latency (ns)</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARF ((\text{nodes} = 28))</td>
<td>(2(+), 4(*))</td>
<td>196.61</td>
<td>2.59</td>
<td>0.8393</td>
</tr>
<tr>
<td>DCT ((\text{nodes} = 42))</td>
<td>(4(+), 2(*))</td>
<td>223.35</td>
<td>3.80</td>
<td>0.8343</td>
</tr>
<tr>
<td>IDCT ((\text{nodes} = 42))</td>
<td>(4(+), 2(*))</td>
<td>224.13</td>
<td>3.73</td>
<td>0.8270</td>
</tr>
<tr>
<td>BPF ((\text{nodes} = 29))</td>
<td>(2(+), 2(*))</td>
<td>202.9</td>
<td>3.77</td>
<td>0.8787</td>
</tr>
<tr>
<td>MPEG ((\text{nodes} = 28))</td>
<td>(3(+), 5(*))</td>
<td>224.13</td>
<td>2.38</td>
<td>0.6645</td>
</tr>
<tr>
<td>JPEG ((\text{nodes} = 33))</td>
<td>(4(+), 4(*))</td>
<td>724.3</td>
<td>14.24</td>
<td>0.7349</td>
</tr>
</tbody>
</table>

Table 7
Measuring probability of coincidence \((P_c)\) as strength of watermark.

<table>
<thead>
<tr>
<th>Benchmarks</th>
<th># of registers before fingerprint</th>
<th>(P_c) # of watermark constraints ((W))</th>
<th>Area ((\mu \text{m}^2))</th>
<th>Latency (ns)</th>
<th>Cost</th>
<th>Area ((\mu \text{m}^2))</th>
<th>Latency (ns)</th>
<th>Cost</th>
<th>Area ((\mu \text{m}^2))</th>
<th>Latency (ns)</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARF</td>
<td>8</td>
<td>0.26308</td>
<td>0.06921</td>
<td>0.01821</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DPR</td>
<td>8</td>
<td>0.26308a</td>
<td>0.06921</td>
<td>0.01821</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IDCT</td>
<td>9</td>
<td>0.30795</td>
<td>0.09483</td>
<td>0.0292</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BPF</td>
<td>7</td>
<td>0.21406</td>
<td>0.04582</td>
<td>0.00981</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPEG</td>
<td>14</td>
<td>0.47560</td>
<td>0.22715</td>
<td>0.10826</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JPEG</td>
<td>12</td>
<td>0.41890</td>
<td>0.17548</td>
<td>0.07351</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 8
Variation of area, latency and cost with the increment of fingerprint size.

<table>
<thead>
<tr>
<th>Benchmarks</th>
<th>Resource configuration</th>
<th>Fingerprint size ((F))</th>
<th>Area ((\mu \text{m}^2))</th>
<th>Latency (ns)</th>
<th>Cost</th>
<th>Area ((\mu \text{m}^2))</th>
<th>Latency (ns)</th>
<th>Cost</th>
<th>Area ((\mu \text{m}^2))</th>
<th>Latency (ns)</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARF</td>
<td>(2(+), 4(*))</td>
<td>(F = 10)</td>
<td>195.82</td>
<td>2.47</td>
<td>0.8211</td>
<td>195.82</td>
<td>2.52</td>
<td>0.8288</td>
<td>195.82</td>
<td>2.59</td>
<td>0.8391</td>
</tr>
<tr>
<td>DCT</td>
<td>(4(+), 2(*))</td>
<td>(F = 20)</td>
<td>222.56</td>
<td>3.75</td>
<td>0.8283</td>
<td>222.56</td>
<td>3.80</td>
<td>0.8340</td>
<td>222.56</td>
<td>3.80</td>
<td>0.8340</td>
</tr>
<tr>
<td>IDCT</td>
<td>(4(+), 2(*))</td>
<td>(F = 30)</td>
<td>224.56</td>
<td>3.72</td>
<td>0.8246</td>
<td>224.56</td>
<td>3.73</td>
<td>0.8268</td>
<td>224.56</td>
<td>3.73</td>
<td>0.8276</td>
</tr>
<tr>
<td>BPF</td>
<td>(2(+), 2(*))</td>
<td>(F = 40)</td>
<td>202.11</td>
<td>3.74</td>
<td>0.8764</td>
<td>202.11</td>
<td>3.74</td>
<td>0.8764</td>
<td>202.11</td>
<td>3.77</td>
<td>0.8784</td>
</tr>
<tr>
<td>MPEG</td>
<td>(3(+), 5(*))</td>
<td>(F = 50)</td>
<td>224.13</td>
<td>2.36</td>
<td>0.6639</td>
<td>224.13</td>
<td>2.36</td>
<td>0.6639</td>
<td>224.13</td>
<td>2.38</td>
<td>0.6645</td>
</tr>
<tr>
<td>JPEG</td>
<td>(4(+), 4(*))</td>
<td>(F = 60)</td>
<td>724.3</td>
<td>14.24</td>
<td>0.7349</td>
<td>724.3</td>
<td>14.24</td>
<td>0.7349</td>
<td>724.3</td>
<td>14.24</td>
<td>0.7349</td>
</tr>
</tbody>
</table>

Table 9
Variation of area, latency and cost with the increment of watermark size after embedding fingerprint.

<table>
<thead>
<tr>
<th>Benchmarks</th>
<th>Resource configuration</th>
<th># of watermark constraints ((W)) after embedding fingerprint</th>
<th>Area ((\mu \text{m}^2))</th>
<th>Latency (ns)</th>
<th>Cost</th>
<th>Area ((\mu \text{m}^2))</th>
<th>Latency (ns)</th>
<th>Cost</th>
<th>Area ((\mu \text{m}^2))</th>
<th>Latency (ns)</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARF</td>
<td>(2(+), 4(*))</td>
<td>(F = 30, W = 10)</td>
<td>196.61</td>
<td>2.59</td>
<td>0.8391</td>
<td>196.61</td>
<td>2.59</td>
<td>0.8391</td>
<td>196.61</td>
<td>2.59</td>
<td>0.8391</td>
</tr>
<tr>
<td>DCT</td>
<td>(4(+), 2(*))</td>
<td>(F = 30, W = 20)</td>
<td>223.35</td>
<td>3.80</td>
<td>0.8343</td>
<td>223.35</td>
<td>3.80</td>
<td>0.8343</td>
<td>223.35</td>
<td>3.80</td>
<td>0.8343</td>
</tr>
<tr>
<td>IDCT</td>
<td>(4(+), 2(*))</td>
<td>(F = 30, W = 30)</td>
<td>224.13</td>
<td>3.73</td>
<td>0.8268</td>
<td>224.13</td>
<td>3.73</td>
<td>0.8268</td>
<td>224.13</td>
<td>3.73</td>
<td>0.8268</td>
</tr>
<tr>
<td>BPF</td>
<td>(2(+), 2(*))</td>
<td>(F = 40, W = 10)</td>
<td>202.11</td>
<td>3.77</td>
<td>0.8784</td>
<td>202.11</td>
<td>3.77</td>
<td>0.8784</td>
<td>202.11</td>
<td>3.77</td>
<td>0.8784</td>
</tr>
<tr>
<td>MPEG</td>
<td>(3(+), 5(*))</td>
<td>(F = 40, W = 20)</td>
<td>224.13</td>
<td>2.38</td>
<td>0.6645</td>
<td>224.13</td>
<td>2.38</td>
<td>0.6645</td>
<td>224.13</td>
<td>2.38</td>
<td>0.6645</td>
</tr>
<tr>
<td>JPEG</td>
<td>(4(+), 4(*))</td>
<td>(F = 50, W = 30)</td>
<td>724.3</td>
<td>14.24</td>
<td>0.7349</td>
<td>724.3</td>
<td>14.24</td>
<td>0.7349</td>
<td>724.3</td>
<td>14.24</td>
<td>0.7349</td>
</tr>
</tbody>
</table>

5.1. Result of the proposed approach in terms of embedding cost, security metric and implementation complexity

Table 6 shows the result of proposed symmetrical IP core protection approach in terms of hardware area, latency and embedding cost (for fingerprint strength \((F) = 30\) and watermark strength \((W) = 30\)). For example for IDCT benchmark, the corresponding area, latency and embedding cost is 224.13 sq. micro meter \((\mu \text{m})\), 3.73 pico-seconds \((\text{ps})\) and 0.827 respectively (Note: module library comprises of resources with respect to 15 nm technology scale adopted from NanGate open source library). The cost of each watermarked–fingerprinted IP core solution is evaluated using Eq. \((3)\).

Table 7 reports the probability of coincidence \((P_c)\) for the watermark in the proposed approach. It measures the probability of generating the same colored solution with the watermark signature and indicates the proof of ownership. The function for evaluation of \(P_c\) is derived from \([18]\) is defined as:

\[
P_c = (1 - \frac{1}{c})^f
\]

where, \(P_c\) is the probability of coincidence, \(c\) is the number of color used, \(f\) is the fingerprinting constraints. As evident from the Table 7, as signature fingerprint strength increases, the \(P_c\) decreases. So, in order to obtain a stronger proof of ownership, a larger size signature is to be chosen.

Table 8 shows the impact of incrementing buyer fingerprint size on hardware area, latency and embedding cost. In this experiment, the buyer fingerprint size is varied to gauge its relative effect on various design metrics. Three different values of fingerprint size is chosen viz. \(F = 10\), \(F = 20\) and \(F = 30\). As evident
from the results, the increment of hardware area with increase in fingerprint strength is zero is most cases. This is because in our fingerprint signature encoding no variable except ‘z’ embeds fingerprint constraint during register allocation phase (which in turn may add register overhead). Thus likelihood of hardware area increase is very minimal with increase in fingerprint size. On the contrary, the minimal increment of latency with increase in fingerprint strength can be noted for some benchmarks (either for $F = 20$ or $F = 30$). This increment is a result of adding more fingerprint constraints during scheduling phase. However, the increment is very slight as fingerprint constraints are embedded during operation conflict resolution in scheduling. Consequently, minimal increase in cost with increase in fingerprint strength is also noted for some benchmarks.

Table 9 shows the impact of incrementing seller watermark size (post-embedding buyer fingerprint) on hardware area, latency and embedding cost. Since seller watermark is embedded after buyer fingerprint, hence different values of watermark strength are indicated by $W$ and $F$ both. In this experiment, the seller watermark size is varied (by keeping the buyer fingerprint strength to maximum i.e. $F = 30$) to gauge its relative effect on aforesaid design metrics. Three different values of watermark size is chosen viz. $W = 10 @ F = 30$, $W = 20 @ F = 20$ and $W = 30 @ F = 30$. As evident from the results, the increment of latency with increase in watermark strength is zero for all cases. This is because in our watermark signature encoding does not embed watermarking constraint during scheduling phase (which in turn adds no latency overhead). Thus latency increase with increase in watermark size is nil for any size of $W$ (i.e. $W = 10$, $W = 20$ and $W = 30$). On the contrary, the minimal increment of hardware area with increase in watermark strength can be noted for some benchmarks (either for $W = 20$ or $W = 30$). This increment is a result of adding more watermarking constraints during register allocation phase. Consequently, minimal increase in cost with increase in watermark strength is also noted for some benchmarks.

5.2. Comparison of the proposed symmetrical IP core protection technique with a non-symmetrical technique

To the best of our knowledge in the literature and industry, there exists no symmetrical IP core protection methodology during high level synthesis, besides the proposed work. We have compared IP designs generated through proposed symmetrical protection approach with baseline IP design (with no protection feature i.e. no embedded seller watermark and buyer fingerprint) in terms of hardware area, latency and embedding cost. As evident from Table 10 in proposed approach, the proposed approach provides symmetrical IP protection to both buyer and seller at minimal area overhead (less than 1%). Further, the proposed approach provides symmetrical IP protection to both buyer and seller minimal latency overhead (avg. 1.02%). Consequently, the proposed approach provides symmetrical IP ownership protection at extremely low cost overhead compared to baseline.

As mentioned earlier, there exists no symmetrical IP core protection methodology during high level synthesis. Thus we have compared proposed symmetrical protection approach with an unsymmetrical protection approach [18] (embeds only seller watermark) in terms of hardware area, latency and embedding cost. As evident from Table 10 in proposed approach, the proposed approach provides symmetrical IP protection to both buyer and seller at minimal area overhead (less than 1%). Further, the proposed approach provides symmetrical IP protection to both buyer and seller minimal latency overhead (avg. 1.02%). Consequently, the proposed approach provides symmetrical IP ownership protection at extremely low cost overhead compared to baseline.

Table 12 shows the comparison of proposed symmetrical protection approach with unsymmetrical protection approach [18] and baseline in terms of storage hardware (registers). As evident from the results, the storage hardware overhead of proposed symmetrical approach compared to unsymmetrical approach (only seller watermark and no buyer fingerprint) [18] is zero. Further, the storage hardware overhead of proposed symmetrical protection

### Table 10
Comparison of proposed symmetrical IP core protection methodology with the baseline IP design.

<table>
<thead>
<tr>
<th>Benchmarks</th>
<th>Resource configuration</th>
<th>Area (μm²)</th>
<th>Latency (ns)</th>
<th>Cost</th>
<th>Baseline Proposed Overhead (%)</th>
<th>Baseline Proposed Overhead (%)</th>
<th>Baseline Proposed Overhead (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARF</td>
<td>2 (+1, 4)</td>
<td>195.82</td>
<td>2.46</td>
<td>0.8185</td>
<td>0.8363</td>
<td>2.48</td>
<td>18</td>
</tr>
<tr>
<td>DCT</td>
<td>4 (+1, 2)</td>
<td>222.56</td>
<td>3.73</td>
<td>0.8266</td>
<td>0.8343</td>
<td>0.95</td>
<td>30</td>
</tr>
<tr>
<td>IDCT</td>
<td>4 (+1, 2)</td>
<td>222.56</td>
<td>3.73</td>
<td>0.8246</td>
<td>0.8267</td>
<td>0.25</td>
<td>18</td>
</tr>
<tr>
<td>BPF</td>
<td>2 (+1, 2)</td>
<td>202.11</td>
<td>3.69</td>
<td>0.8702</td>
<td>0.8787</td>
<td>0.97</td>
<td>30</td>
</tr>
<tr>
<td>FIR</td>
<td>4 (+1, 4)</td>
<td>179.31</td>
<td>1.80</td>
<td>0.7370</td>
<td>0.7526</td>
<td>2.07</td>
<td>18</td>
</tr>
<tr>
<td>MPEG</td>
<td>3 (+1, 5)</td>
<td>224.13</td>
<td>2.36</td>
<td>0.6639</td>
<td>0.6645</td>
<td>0.09</td>
<td>30</td>
</tr>
<tr>
<td>JPEG</td>
<td>4 (+1, 4)</td>
<td>724.3</td>
<td>14.24</td>
<td>0.7349</td>
<td>0.7349</td>
<td>0</td>
<td>18</td>
</tr>
</tbody>
</table>

### Table 11
Comparison of proposed symmetrical IP core protection methodology with [18].

<table>
<thead>
<tr>
<th>Benchmarks</th>
<th>Resource configuration</th>
<th>Area (μm²)</th>
<th>Latency (ns)</th>
<th>Cost</th>
<th>Baseline Proposed Overhead (%)</th>
<th>Baseline Proposed Overhead (%)</th>
<th>Baseline Proposed Overhead (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARF</td>
<td>2 (+1, 4)</td>
<td>196.61</td>
<td>2.46</td>
<td>0.8187</td>
<td>0.8363</td>
<td>2.48</td>
<td>18</td>
</tr>
<tr>
<td>DCT</td>
<td>4 (+1, 2)</td>
<td>223.35</td>
<td>3.73</td>
<td>0.8266</td>
<td>0.8343</td>
<td>0.95</td>
<td>30</td>
</tr>
<tr>
<td>IDCT</td>
<td>4 (+1, 2)</td>
<td>223.35</td>
<td>3.73</td>
<td>0.8246</td>
<td>0.8267</td>
<td>0.25</td>
<td>18</td>
</tr>
<tr>
<td>BPF</td>
<td>2 (+1, 2)</td>
<td>202.90</td>
<td>3.69</td>
<td>0.8702</td>
<td>0.8787</td>
<td>0.97</td>
<td>30</td>
</tr>
<tr>
<td>FIR</td>
<td>4 (+1, 4)</td>
<td>180.09</td>
<td>1.80</td>
<td>0.7370</td>
<td>0.7526</td>
<td>2.07</td>
<td>18</td>
</tr>
<tr>
<td>MPEG</td>
<td>3 (+1, 5)</td>
<td>224.13</td>
<td>2.36</td>
<td>0.6639</td>
<td>0.6645</td>
<td>0.09</td>
<td>30</td>
</tr>
<tr>
<td>JPEG</td>
<td>4 (+1, 4)</td>
<td>724.3</td>
<td>14.24</td>
<td>0.7349</td>
<td>0.7349</td>
<td>0</td>
<td>18</td>
</tr>
</tbody>
</table>

### Table 12
Comparison of storage hardware (registers) used in a baseline, [18] and proposed solution.

<table>
<thead>
<tr>
<th>Benchmarks</th>
<th># of storage hardware (register) used</th>
<th>Storage hardware (register) overhead with respect to baseline</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARF</td>
<td>8</td>
<td>0</td>
</tr>
</tbody>
</table>
### Table 13
Comparison of proposed approach and [18] in terms of signature creation time.

<table>
<thead>
<tr>
<th>Benchmarks</th>
<th>Resource configuration</th>
<th>Signature creation time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARF</td>
<td>2(+), 4(*)</td>
<td>19</td>
</tr>
<tr>
<td>DCT</td>
<td>4(+), 2(*)</td>
<td>17</td>
</tr>
<tr>
<td>IDCT</td>
<td>4(+), 2(*)</td>
<td>14</td>
</tr>
<tr>
<td>BPF</td>
<td>2(+), 2(*)</td>
<td>11</td>
</tr>
<tr>
<td>FIR</td>
<td>4(+), 4(*)</td>
<td>31</td>
</tr>
<tr>
<td>MPEG</td>
<td>3(+), 5(*)</td>
<td>14</td>
</tr>
<tr>
<td>JPEG</td>
<td>4(+), 4(*)</td>
<td>61</td>
</tr>
</tbody>
</table>

**Fig. 9.** Comparison of area between baseline approach, unsymmetrical approach [18] and proposed symmetrical approach. (Note: As evident proposed approach (with seller watermark and buyer fingerprint) provides complete (for both parties) protection with minimal area overhead).

**Fig. 10.** Comparison of latency between baseline approach, unsymmetrical approach [18] and proposed symmetrical approach. (Note: As evident proposed approach (with seller watermark and buyer fingerprint) provides complete (for both parties) protection with minimal latency overhead).

approach with respect to baseline (no seller watermark and buyer fingerprint) is very minimal (less than 0.2%). Figs. 9–11 indicate comparison of area, latency and cost between baselines design, unsymmetrical approach [18] and proposed symmetrical approach. As evident, the proposed approach provides complete IP core protection to both parties at minimal hardware area (overall less than 1%), latency (overall less than 1.1%) and cost (overall less than 2%) overhead.

Table 13 reports comparison of proposed symmetrical protection approach with [18] in terms of signature creation time. For proposed approach signature creation time includes creation time of seller watermark and buyer fingerprint, while only seller watermark for [18]. As evident from the results, the proposed approach while embedding both seller multi-variable watermark and buyer fingerprint, nominally increases the runtime overhead compared to [18] (that only embeds seller dual-variable watermark). This trend is observed for all size benchmarks, thus demonstrating the scalability of the proposed approach.

Additionally, Fig. 12 shows the variation of cost overhead obtained for various benchmarks. As evident, with the increase in size (parallelism i.e. number of parallel operations) of the benchmarks, cost overhead % sharply decreases. In other words, cost overhead of proposed approach is highest for ARF which comprises of minimum number of parallel operations (parallelism possible) while is least for JPEG which comprises of maximum number of parallel operations (parallelism possible). This is possible because the proposed approach embeds fingerprint signature during operation conflict resolution in scheduling which occurs more often for bigger applications (comprising of large number of parallel operations) working under a resource constraint. Therefore, the proposed approach offers lower (negligible) overhead for larger applications. This demonstrates the scalability of the proposed approach.

### 6. Conclusion and future work

This paper presents a novel approach for symmetrical IP core protection during high level synthesis using multi-variable encoding based fingerprinting and watermarking. The proposed approach is capable of yielding a low overhead symmetrically protected IP core design with buyer fingerprint and seller watermark embedded. The fingerprint and watermark for symmetrical IP protection proposed in this paper are highly robust in nature. Besides yielding a low overhead IP design our approach also provides robustness, fault tolerance, low signature embedding time and adaptability to any CAD tool.

We intend to extend our current research work of symmetrical IP core protection to other HLS steps like binding, allocation. This would enable us to perform a comparative study between various HLS steps to find the better suited phase for signature embedding process. We also plan to make our encoding process more robust by adding more variable.

### Acknowledgment

The authors would like to thank Indian Institute of Technology Indore for their support.
Dipanjan Roy is a research scholar in Discipline of Computer Science and Engineering at Indian Institute of Technology (I.I.T) Indore. His area of research is IP core protection during HLS, Hardware protection and security during HLS, etc. He received his M. Tech. Degree in Banking Technology & Information Security from University of Hyderabad, India. His Master's thesis was from Institute for Development and Research in Banking Technology (IDRBT), Hyderabad. He worked as an software development engineer in “Amazon Development Center”, Bangalore before joining I.I.T, Indore.

Anirban Sengupta (Ph.D., M.A.Sc.) is currently an Assistant Professor in Discipline of Computer Science and Engineering at Indian Institute of Technology (I.I.T) Indore. He directs the research lab on Behavioural Synthesis and Trust, Springer, New York, NY, USA, 2012, pp. 103–122.

He has around 107 Publications & Patents (which includes 11 Patents). Some of his recent patents were approved for commercialization in electronics industry. In the past, his Patents generated funding from Ontario Center of Excellence (OCE), Canada. He had performed industry interactive research extensively for more than 2 years with Calyptra, Bluespec, BEECube, Huawei Canada during development of his Ryerson Design Space Exploration Tool arising from his Patent.

He currently serves in editorial positions of 10 IEEE & IET, Elsevier Transactions and Journals in capacities such as Associate Editor, Executive Editor and Guest Editor. He is currently serving as Associate Editor of IEEE Transactions on Aerospace and Electronic Systems, Editor of Elsevier Microelectronics Journal, Associate Editor of IET Journal on Computer & Digital Techniques, Associate Editor of IEEE ACCESS Journal, Associate & Executive Editor of IEEE Consumer Electronics Magazine and Associate Editor of IEEE VLSI Circuits & Systems Letter. He further serves as Guest Editor of IEEE Transactions on Consumer Electronics, IEEE Transactions on VLSI Systems and IEEE ACCESS Journals. He is a regular reviewer of IET Journal on Computers and Digital Techniques and IEEE Transactions on VLSI. He has served as Technical Program Chair of 15th IEEE International Conference on Information Technology (ICT) 2016 as well as Technical Program Chair of 3rd IEEE NIH 2017. He regularly serves in TPC of ISVLSI, GLSVLSI, ICCSE, ICAS, CECCE etc.

References