Fast Robust Intellectual Property Protection for VLSI Physical Design

Debasri Saha and Susmita Sur-Kolay
A.C.M. Unit, Indian Statistical Institute, Kolkata, India. {debasri_r, ssk}@isical.ac.in

Abstract

In deep sub-micron VLSI technology, design reuse has become essential due to more integration on a single chip in shorter time. Design reuse however is susceptible to misappropriation of the Intellectual Property (IP) of the design. There may be illegal reselling or unauthorized reuse of the design, creating false charges on legal buyer by the IP owner, false claim for IP ownership, tampering of watermarks present in the design for IP protection (IPP). While identifying IP owner and legal IP buyer through copy detection remains an exhaustive method, public and convincing watermark verification for IP ownership is not still safe. Our proposed algorithm $ROBUST\_IP$ tackles all the problems from an entirely new viewpoint. It facilitates faster extraction of signatures of IP owner and buyer, whereas removing or tampering the watermarks by an attacker remains infeasible, even if public verification is allowed. The scheme is effectively applied for IPP in both ASIC and FPGA designs. It has been tested on various MCNC benchmarks. The experimental results are quite encouraging and the overhead incurred by our technique on the design is negligible.

Keywords: Intellectual property, watermarking, fingerprinting, VLSI physical design, electronic design automation.

1 Introduction

In deep sub-micron VLSI technology, shrinking time-to-market and increased design productivity have led to widespread design reuse. Circuit components, available in electronic form, are known as virtual components and these constitute the Intellectual Property (IP) of a VLSI physical design [1]. Instead of designing from scratch, appropriate IPs are assembled to meet the specifications of the customers in time. However, infringement of IP is more probable [2] with some typical attacks:

- illegal reselling of design IP by a buyer or unauthorized reuse by someone other than legal buyer without paying proper royalty fee to the authentic IP owner;
- repudiation attack by the IP owner on the legal purchaser – challenging the validity of legal purchaser and creating false charges of illegal reselling;
- additive attack – claiming false IP ownership by someone other than authentic IP owner by insertion of his watermark overriding that of genuine IP owner.

For IPP, various kinds of marks (watermarks, fingerprints, fingermarks) are embedded into the design. Watermark based on IP owner's signature is for identification of design IP owner, whereas buyer's fingerprint derived from buyer's signature is for identification of the legal IP buyer. In case of any dispute on IP ownership, public mark verification is performed, but it provides hints to a potential attacker for the following kinds of IP mark tampering:

- removing or altering marks containing IP owner's identity for destruction of IP ownership;
- removing or altering marks containing buyers identity safeguards the buyer in illegal reselling.

This motivates us to develop a robust IP protection scheme.

The paper is organized as follows. Section 2 assesses prior approaches. Section 3 redefines tasks of IP protection (IPP) team. Our proposed algorithm for IPP in VLSI physical design and its analysis are given in Section 4. Section 5 presents the experimental results and the concluding remarks appear in Section 6.

2 Prior Approaches

Design IP protection has immense significance in electronic design automation. Constraint-based watermarking [1] maps the signature of a designer into a set of additional constraints. In [2], topological information of a design is uniquely mapped into a topological signature. Both approaches [1, 2] donot include buyer's identity in the design to check illegal reselling. Buyer's fingerprint is embedded in the LUTs of FPGA in [3] and fingerprints for VLSI CAD optimizations are developed in [4]. But both approaches [3, 4] apply exhaustive search to find out fingerprints in the design using copy detection. Possible locations of buyer's

fingerprint can be restricted, but such reduction in solution space enhances the probability of fingerprint tampering by the attacker. Moreover, none of these works highlight the challenges faced during convincing public mark verification. The method for public detection of marks discussed in [5] is not effective, as the marks are likely to be tampered. Furthermore it fails to identify the first illegal reseller. So, none of the existing schemes can avoid exhaustive copy detection to identify legal buyer and IP owner, or provide robust and convincing public mark verification.

3 Redefining Tasks of IPP Team

An independent IPP team is needed not only for mark verification but also for embedding marks during design selling, because the IP owner may also not be trustworthy (follows from repudiation attack).

During selling, IPP team extracts and verifies

S1: buyer's signature from the design to ensure buyer's protection against repudiation attack;

S2: IP owner's signature from the design to (i) convince the buyer that the product is from a bonafide IP owner; (ii) convince the IP owner that his signature is properly embedded for future proof of IP ownership.

During verification, IPP team

V1: extracts buyer's signature from the design to find legal buyer, i.e., the first illegal reseller for illegally resold copy.

V2: extracts IP owner's signature from the design to find out the true IP owner.

V3: publicly highlights certain marks based on IP owner's company name for convincing verification in case of any dispute over IP ownership.

Master key – private to IPP team:
This key, unknown to IP owner and buyer, is essential for
a) avoiding exhaustive search for marks during verification; if the master key is used to generate location of marks embedding, it can also be used for faster extraction of marks;
b) for encrypting the signatures or their hash values or both.

A buyer's key cannot serve the purpose of a master key as a buyer can locate the marks easily and can suitably tamper the marks for his benefit. Similarly IP owner's key cannot be used for this purpose, as without scanning the design, it is not always possible for the IPP team to identify IP owner.

Generation of identical design copies [Figure 1]- *the threat to public mark verification:* Although buyer's signature is included to make each legally sold copy of the same design distinct, additional copies generated through illegal reselling or unauthorized reuse are entirely identical. Hence public mark verification, if allowed for such a copy, leaves potential clues to identical copy-holder.

Public verification count C_{PV} – private to IPP team:
This parameter specifies the number of times a design copy (unique to a particular buyer) or any identical copy of it, has been publicly verified for IP ownership proof.

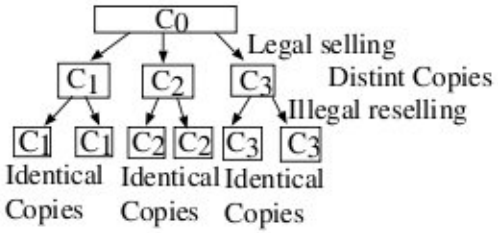


Figure 1. Generation of identical copies

For a design IP, the value of C_{PV} is likely to differ for each distinct design copy and hence depends on buyer's signature S_B. For each design IP, C_{PV} is to be stored against S_B in a secret file maintained by IPP team. During a public verification of a distinct copy, the corresponding C_{PV} value is incremented and depending on the incremented value of C_{PV}, a distinct subset of watermarks is publicly highlighted to make each public verification distinct.

4 The Proposed Scheme ROBUST_IP

4.1 An Overview

IP Mark Inclusion: [Figure 2]
For IP protection of a VLSI design, the following two types of marks are embedded into the design during selling:
a) a fingermark containing both IP buyer's signature and IP owner's signature, this fingermark is a combination of buyer's fingerprint and owner's watermark;
b) certain watermarks for public verification, each is based on IP owner company's name.

At the time of selling, IPP team concatenates IP creator's (or owner's) signature S_C and buyer's signature S_B to generate combined signature S_{CB}, which is encrypted using the master key K_M to generate S'_{CB}. The first part of S'_{CB} is embedded into specific locations in the entire design, determined from K_M and a design parameter, e.g, the total number of modules n, or the total number of nets e in the design. The location of rest of S'_{CB} is determined using K_M, n or e and the already embedded part of S'_{CB} (This value differs for different buyers of the same design).

A one-way function is applied on S_C and S_B to generate h_{S_C} and h_{S_B}, the hash values of S_C and S_B respectively. These hash values are embedded into specific locations of the design, determined from K_M and the design parameter. Our scheme embeds an additional set of n watermarks (each contains Message Authentication Code i.e., MAC value [6] generated from IP owner company name using K_M) and their locations are selected using K_M, S_B and n or e. During selling of a distinct design to a distinct buyer, the corresponding C_{PV} is set to zero in the secret file.

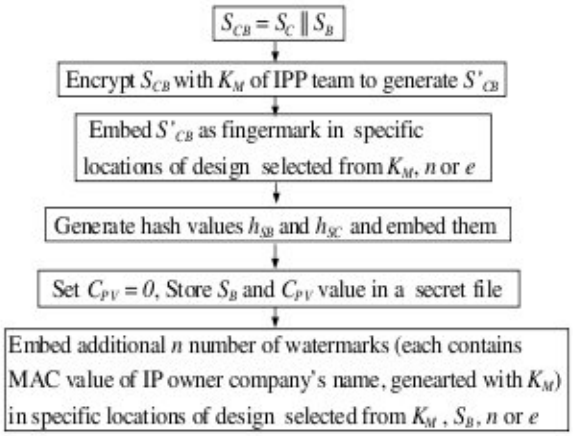


Figure 2. Flowchart for IP mark inclusion

IP Mark Verification: [Figure 3]
The proposed scheme determines the first fingermark location in the design by using K_M and the design parameter used. The first part of S'_{CB} is extracted from that location. It is then used along with K_M and the design parameter to find the rest of the mark locations. The encrypted form of entire combined signature, S'_{CB} is extracted. Subsequently it is decrypted with K_M to recover combined signature S_{CB}, and hence the buyer's signature and the IP owner's signature. To detect any tampering of fingermarks, the hash values of the recovered signatures are computed. On the other hand, the location of already embedded hash values are found out using K_M and design parameter. If the hash values of the recovered signatures are identical with the embedded hash values, it is ascertained that the fingermarks remain virgin, and the recovered buyer's and creator's signatures are original values of S_B and S_C respectively. S_B helps to identify the legitimate buyer of that distinct design copy and S_C helps to identify the genuine IP owner.
In the event of any dispute on IP ownership, the value of C_{PV} corresponding to the extracted S_B is incremented by 1 in the secret file. Among the $(2^n - 2)$ proper subsets (excluding null set) of the set of n watermarks, a distinct subset containing watermarks corresponds to $'1'$ bits of n-bit binary representation of the current value of C_{PV} is selected. The locations of that subset of marks are found out using K_M, S_B and n or e as before and highlighted publicly.

4.2 ROBUST_IP for VLSI Physical Design

$ROBUST\_IP$ is applicable for IPP in VLSI physical design, if an encryption algorithm, a method for selecting the locations of marks, nature of marks and a specific design phase for IP marking are chosen appropriately.
Marks are inserted in the post-placement phase of physical design. DES algorithm is used to encrypt S_{CB} with K_M. As one-way hash function, Lagged Fibonacci Generator

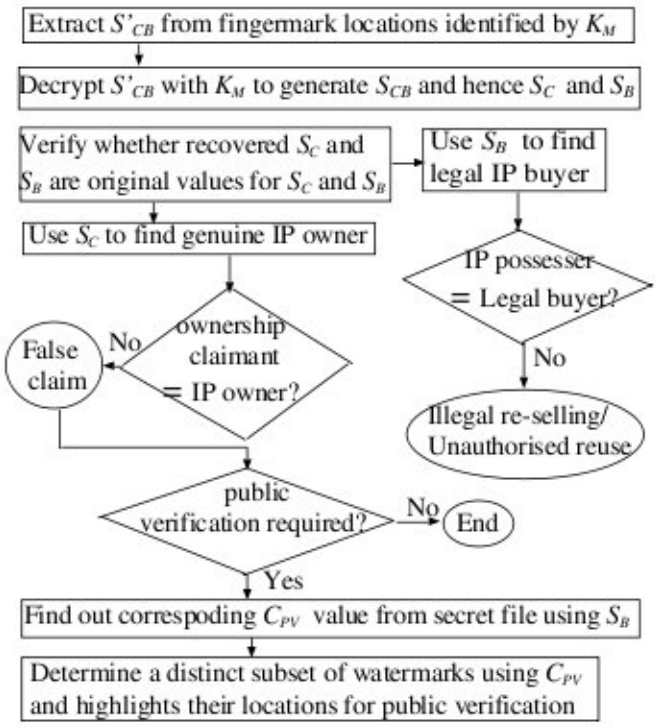


Figure 3. Flowchart for IP mark verification

(LFG) can be applied on signatures to generate the corresponding hash values. LFG can also be used as Pseudo Random Number Generator (PRNG) with K_M, S'_{CB} and n or e as seed values to select locations of marks to embed signatures as well as their hash values. The locations of the marks are allowed to depend on a part of S'_{CB}, if those are visually imperceptible. A trie data structure is used to store signature of all buyers along with corresponding value of C_{PV} in the secret file.

IP Marking in ASIC Layout

For high performance ASIC, repeaters have important role in minimizing delay. Marks can be embedded as minute modification in the size of a few repeaters in either (a) uniform optimal repeater system or (b) initial cascaded buffer system. Without the precise knowledge of mark locations, any attempt to remove or tamper the marks by removing or modifying the entire repeater system, can degrade the performance of ASIC immensely. K_M is used to select a set of nets for mark insertion through the repeater size modification. The bit string to be stored for IP marking, is split into number of 3 or 4-bit substrings, the decimal equivalent of which is denoted as x. Let C_0 and R_0 be the input capacitance and output resistance of minimum-sized repeater, R_{int} and C_{int} the total interconnect resistance and capacitance respectively.
(a) Uniform optimal repeater system: [Figure 4(a)]
k = # repeaters for the selected net, h = size of an optimal

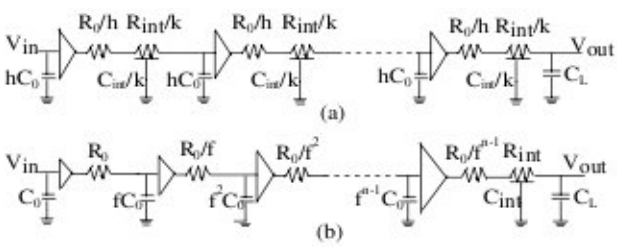


Figure 4. (a) Optimal repeater system (b) Cascaded buffer system

size repeater w.r.t the minimum-sized one. The total delay

$$T = k(2.3\frac{R_0}{h}(\frac{C_{int}}{k} + hC_0) + \frac{R_{int}}{k}(\frac{C_{int}}{k} + 2.3hC_0)) \quad (1)$$

where $k = \sqrt{\frac{R_{int}C_{int}}{(2.3R_0C_0)}}$ and $h = \sqrt{\frac{R_0C_{int}}{(R_{int}C_0)}}$ for minimum delay [7]. Now, x is stored by reducing the size of each repeater by x to $h' = h - x$. The output resistance and input capacitance of each repeater is changed from R_0/h and C_0h to R_0/h' and C_0h' respectively. The modified total delay

$$T' = k(2.3\frac{R_0}{h'}(\frac{C_{int}}{k} + h'C_0) + \frac{R_{int}}{k}(\frac{C_{int}}{k} + 2.3h'C_0)) \quad (2)$$

The increase in delay
$\Delta T = T' - T = 2.3x(\frac{C_{int}R_0}{h(h-x)} - R_{int}C_0)$, is very small.

(b) Initial cascading phase: [Figure 4(b)]
n = # repeaters in the initial cascading phase of selected net, f = size ratio of successive repeaters, then the total delay

$$T = 2.3(n-1)fR_0C_0 + \frac{2.3R_0C_{int}}{f^{n-1}} + R_{int}C_{int} \quad (3)$$

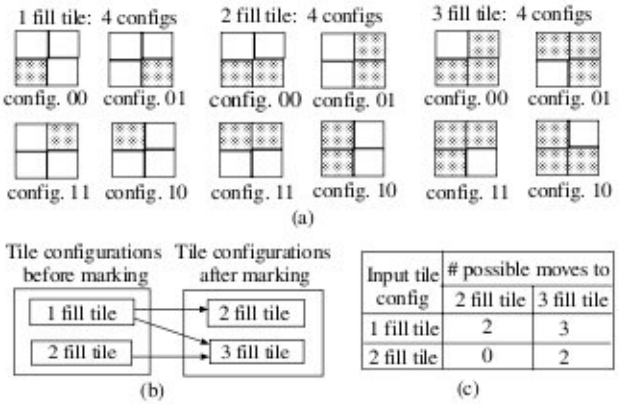
where $n = ln(C_{int}/C_0)$ and $f = e$ for minimum delay [7]. Now to store x, the size of first repeater is changed from S_1 to $S_1' = S_1 + x$ and the size of the other repeaters except the last one in cascading phase are changed accordingly to maintain a constant size ratio f' where $f'^{(n-1)} = f^{(n-1)}/f''$ and $f'' = S_1'/S_1$. The output resistance and the input capacitance of the first repeater become $R_0' = R_0/f''$ and $C_0' = C_0f''$ respectively. Now the total delay

$$T' = 2.3(n-1)f'\frac{R_0}{f''}C_0f'' + \frac{2.3(R_0/f'')C_{int}}{f'^{n-1}} + R_{int}C_{int} \quad (4)$$

$\Delta T = T' - T = 2.3(n-1)R_0C_0(f' - f)$ which is negative, as $f' < f$ if $S_1' > S_1$.

Remark: It is assumed that all the integral buffer sizes are commercially available, otherwise sizes are to be altered in available steps, appropriately choosing the value of x and keeping change in delay in permissible limit.

Example 1: Let S_B=$MYSELF$ with length L_B=6 and S_C=$SECRET$ with length L_C=6. If A,B,...,Z are represented as 1,2,...,26 respectively and each of these integer form is converted to 5-bit string, then S_{CB} is a



Input tile config	# possible moves to	
	2 fill tile	3 fill tile
1 fill tile	2	3
2 fill tile	0	2

(c)

Figure 5. (a) Config.s for 1-fill, 2-fill, 3-fill tiles (b) Tile config.s before and after marking (c) Moves from each input tile config.

60-bit binary string which is padded to 64 bit with zeros. Let $K_M = SOLUTION$ with length L_M=8, which used as 56-bit key (ASCII representation) for DES encryption. Hence S_{CB}' is a 64-bit string, which is split into 3 or 4-bit substrings with decimal equivalent x.

IP Marking in FPGA Layout

For FPGA, the IP marks are included as bitstream in the LUTs of unused cells. A FPGA chip is represented as a set of 2×2 tiles. Experimental observations with VPR reveals that, all unused cells are neighboring to each other in a good placement. So a tile with all 4-cells unused is not a good selection for marking. Hence marks are included in a partially filled tile, and also the tile remains partially filled after marking [Figure 5(b)]. Using certain bits of K_M, we uniquely specify moves from one tile configuration to another due to IP marking, and can uniquely recover unmarked tile configuration during verification.

F_f = a specific bit of the master key, which determines number of used cell in the tile after marking.

U = a specific bit of the master key, which determines number of cells to be marked to reach fill level F_f. From F_f and U, the initial fill level F_i of the tile is determined. First, a particular tile is selected using PRNG. From the selected tile, the next tile with fill level F_i is found out for marking and any tile with fill level F_f on the path from initially selected tile to the tile selected for marking is filled up completely with randomly generated bit string.

Let M be two specific bits of the master key, which determine the move from the initial tile configuration C_i at fill level F_i to the final tile configuration C_f at fill level F_f due to marking [Figure 5(a) and 5(c)]. Then, $C_f = C_i \bigoplus M$. After mark inclusion, the unused outputs of neighboring

cells are connected to the inputs of the marked cells and the outputs of the marked cells are connected to the don't care inputs of neighboring cells.

For verification, the marked tile with fill level F_f next to the selected one with PRNG is determined and final configuration C_f is obtained. Using the F_f bit and the U bit of K_M, the initial fill level F_i is determined. Then using the M bits of the master key, the initial tile configuration C_i at F_i can be uniquely recovered as $C_i = C_f \bigoplus M$. Comparing the initial and final tile configurations, the marked cell(s) can be identified.

Example 2: The 64-bit string S'_{CB} is computed similarly as for ASIC design. For FPGAs, as 4-input LUT is used, 4 cells are required, each containing 16 bits. The hash values h_{S_B} and h_{S_C}, each of 8 bit, are stored in a LUT of another logic cell. Then, 6 more cells are selected, each containing MAC value generated from IP owner company's name ARM using K_M. So, there are $(2^6 - 2)$ distinct public verifications are possible for the design copies generated from a distinct design of a particular legal buyer.

4.3 Analysis of ROBUST_IP

It discusses time complexity and strength of the scheme.

Lemma 1. *Inclusion of IP marks and verification of those marks, each takes $O(max(L_C, L_B, L_M))$ time, where L_B=length of buyer's signature and L_C=length of creator's signature and L_M=length of master key. For FPGA, an additional prepossessing time of $O(n)$ is required for IP marking, n= number of logic cells in the design.*

Proof. The length of the composite signature S_{CB} is $O(L_C + L_B)$. Encryption of S_{CB} with the master key takes $O(L_C + L_B + L_M)$ time. Finding of the fingermark locations and their embedding takes constant time. For FPGA design, in the worst case, prepossessing time of $O(n)$ is required for mark embedding in the design to determine tile configurations from the logic cell descriptions. The timing requirement for verification of marks is similar. □

Let us define the following probability values.
p_{C1} = probability failing to detect false IP ownership claim, if attacker correctly guesses S_C; $p_{C1} = (26)^{-L_C}$
p_{C2} = probability of failing to detect false IP ownership claim, if attacker correctly guesses K_M, hence find out locations of fingermarks and retrieve S_C; $p_{C2} = (26)^{-L_M}$.
p_B = probability to find out and tamper any location of fingermark through random guessing. This is relevant in illegal reselling and raising false claim for IP ownership.

Lemma 2. *For a design with n number of logic cells and k number of inputs, the strength against*
(a) brute-force attack to raise false claim IP ownership is $O((26)^{L_C})$, L_C = length of creator's signature S_C.
(b) additive attack to raise false claim IP ownership is $O((26)^{L_M})$, L_M = length of master key K_M.
(c) cryptanalysis attack to tamper the fingermarks is $O(max(n^3, nk^2 3^k))$.

Proof. For the proof of (a), S_C can have $(26)^{L_C}$ possible values. Additive attack is based on guessing the value of K_M having $(26)^{L_M}$ possible values, hence the proof for (b). For (c), for each of the n cells, an attacker may apply logic extraction from the FPGA bitmap ($O(n^2)$ time) followed by its modification corresponding to elimination of a particular FPGA cell (const. time) and verification of functional correctness of the design ($O(k^2 3^k)$ time)[8]. □

Theorem 1. *$ROBUST\_IP$ is a linear time algorithm and robust enough against typical attackes.*

Proof. Follows from Lemmata 1 and 2. □

5 Results

For inserting fingermarks and watermarks in ASIC layouts, 0.18μ technology is considered. According to NTRS'97 for this technology, $R_0 = 2$ KΩ, $C_0 = 1$fF, $R_{int} = (68 * l)$ Ω, $C_{int} = (.12 * l)$pF where l is wirelength. For various lengths of interconnects, in Table 1, the delay before and after marking with mark digit x=8 are computed using eqn. (1) and (2) respectively for uniform optimal repeater system. For initial cascaded repeater system, Table 2 gives the corresponding values for mark digit x=16, computed using eqn. (3) and (4) respectively.

Table 1. Effect of mark inclusion (x=8) in ASIC design with uniform optimal repeaters

Wirelength (mm)	delay before marking (ps)	delay after marking (ps)	% increase in delay
2	61.65	62.03	0.62
5	154.18	155.15	0.62
7	215.85	217.21	0.63
10	308.36	310.31	0.63
12	370.03	372.37	0.63
15	462.53	465.45	0.62
17	524.22	527.52	0.63
20	616.73	620.62	0.63

Mark inclusion in the initial cascading phase of buffer reduces the delay at the cost of little area overhead whereas IP marking in optimal repeater system has no area overhead but causes very small increase in delay. Although signal delay of a number of nets, selected for marking increases, in the worst case, if all the nets of a path from input to output are selected, the % increase in delay of the path is the maximum of these values for the selected nets.

For FPGA designs, $ROBUST\_IP$ is implemented in C

Table 2. Effect of mark inclusion (x=16) in ASIC design with initial cascaded repeaters

Wirelength (mm)	delay before marking (ps)	delay after marking (ps)	% decrease in delay
2	102.88	77.87	23.93
5	285.22	254.07	10.89
7	484.44	456.20	5.83
10	904.71	876.47	3.12
12	1266.49	1238.25	2.23
15	1931.55	1903.32	1.46
17	2454.32	2425.18	1.18
20	3361.59	3332.45	0.86

on 1.2 GHz Sun Blade 2000 machine using Sun OS 9. Table 3 shows, ultimate area overhead is 0% in all cases. The variable number of dummy filling prevents the attacker to be ascertained that all the marked cells have been removed by a deletion attack. Table 5 reveals, the smaller the value of

Table 3. Impact of fingermark and watermark inclusion for MCNC FPGA benchmarks

Circuits	# cells	# unused cells	# Marked cells (fingermarked + watermarked)	Dummy filling	% Rise in cell usage
Alu4	1522	78	$5+6$	2	0.85
Apex4	1262	34	$5+6$	1	0.95
Apex2	1878	58	$5+6$	3	0.74
Ex5p	1064	25	$5+6$	2	1.22
Elliptic	3604	117	$5+6$	3	0.38
Pdc	4575	49	$5+6$	3	0.30
Bigkey	1707	57	$5+6$	3	0.82
Des	1591	90	$5+6$	2	0.81

Table 4. CPU time for inclusion and verification of marks for FPGA MCNC benchmarks

Circuits	CPU time (msecs) for		CPU time (msecs) for	
	fingermark inclusion	watermarks inclusion	fingermark verification	public verification
Alu4	1.472	0.214	1.453	0.202
Apex4	1.388	0.210	1.351	0.190
Apex2	1.576	0.215	1.554	0.200
Ex5p	1.436	0.204	1.419	0.182
Elliptic	1.876	0.221	1.847	0.204
Pdc	1.927	0.226	1.898	0.207
Bigkey	1.488	0.220	1.185	0.203
Des	1.426	0.213	1.238	0.198

p_B, p_{C1}, p_{C2}, the greater is the strength of $ROBUST\_IP$. The strength of the scheme enhances with the increase of the values of each L_B, L_C, L_M and the size of the design.

6 Conclusion

Our scheme efficiently extracts signatures of legitimate buyer and IP owner right from the design in absence of both IP buyer and owner. It is faster as it elegantly bypasses exhaustive copy detection of the IP marks throughout the design. Additive attack and repudiation attack can also be efficiently tackled. For high performance ASICs, $ROBUST\_IP$ for the fist time effectively uses the repeater system to embed tamper-resistant marks and it adopts localized embedding of signatures to ensure more robust IPP for FPGA layout. It ensures convincing but more secured public mark verification. $ROBUST\_IP$ has negligible overhead on VLSI physical design.

Table 5. Probability values p_{C1}, p_{C2} for various lengths of K_M and K_C.

L_M, length of K_M	L_C, length of K_C	p_{C1}	p_{C2}
5	5	$8.41e^{-8}$	$8.41 * 10^{-8}$
8	8	$4.78e^{-12}$	$4.78 * 10^{-12}$
10	10	$7.08e^{-15}$	$7.08 * 10^{-15}$
12	12	$1.04e^{-17}$	$1.04 * 10^{-17}$
15	15	$5.96e^{-22}$	$5.96 * 10^{-22}$
18	18	$3.39e^{-26}$	$3.39 * 10^{-26}$
20	20	$5.01e^{-29}$	$5.01 * 10^{-29}$

References

[1] A. B. Kahng, J. Lach, H.Mangione-Smith. Constraint-Based Watermarking Techniques for Design IP Protection, *Proc. of IEEE Transactions on CAD/ICAS*, vol 20, No. 10, Oct 2001, pp. 1236-1251.

[2] E. Charbon. Hierarchical Watermarking in IC Design, *Proc. of the CICC*, 1998, pp. 295-298.

[3] J. Lach, W. H. Mangione-Smith, M. Potkonjak. Robust FPGA Intellectual Property Protection Through Multiple Small Watermarks, *Proc. of IEEE/ACM Design Automation Conf.*, June 1999, pp. 831-836.

[4] A. E. Cladwell, H. Choi, A. B. Kahng. Effective Iterative Technique for Fingerprinting Design IP, *Proc. of IEEE Transactions on CAD/ICAS*, vol 23, No 2, 2004.

[5] Gang Qu. Publicly Detectable Techniques for the Protection of Virtual Components, *Proc. of DAC*, June 2001, pp. 474-479.

[6] W. Stallings. *Cryptography and Network Security*, PHI, 2005.

[7] H. Bakoglu, J. Meindl. Optimal interconnection citcuits for VLSI, *IEEE Transactions on Electron Devices*, vol. ED. 32, No. 5, 1986, pp. 903-909.

[8] S. Friedman, K. Supowit. Finding the Optimal Variable Ordering for Binary Decision Diagrams, *Proc. of ACM/IEEE DAC*, 1987, pp. 348-356.