[10] J. M. Daga and D. Auvergne, "A comprehensive delay macromodeling for submicron CMOS logics," *IEEE J. Solid-State Circuits*, vol. 34, pp. 42–55, Feb. 1999.
[11] S. Turgis and D. Auvergne, "A novel macromodel for power estimation for CMOS structures," *IEEE Trans. Computer-Aided-Design*, vol. 17, pp. 1090–1098, Nov. 1998.
[12] A. Chatzigeorgiou and S. Nikolaidis, "Collapsing the transistor chain to an effective single transistor," in *Design Automation and Test Conf. Proc.*, Paris, France, Feb. 1998, pp. 2–6.
[13] A. Nabavi-Lishi and N. C. Rumin, "Inverter models of CMOS gates for supply current and delay evaluation," *IEEE Trans. Computer-Aided Design*, vol. 13, pp. 1271–1279, Oct. 1994.
[14] Q. Wang and S. B. K. Vrudhula, "A new short circuit power model for complex CMOS gates," in *Proc. IEEE Alessandro Volta Memorial Workshop Low Power Design (Volta99)*, Como, Italy, Mar. 4–5, 1999, pp. 98–106.
[15] K. Nose and T. Sakurai, "Analysis and future trend of short circuit power," *IEEE Trans. Computer-Aided Design*, vol. 19, pp. 1023–1030, Sept. 2000.
[16] Y. Jun and K. Jun, "An accurate and efficient delay time modeling for MOS logic circuits using polynomial approximation on CAD," *IEEE Trans. Computer-Aided Design*, vol. 8, Sept. 1989.
[17] A. Chatzigeorgiou, S. Nikolaidis, and I. Tsoukalas, "A modeling technique for CMOS gates," *IEEE Trans. Computer-Aided Design*, vol. 18, May 1999.
[18] A. A. Hamoui and S. C. Rumin, "An analytical model for current, delay, and power analysis of submicron CMOS logic circuits," *IEEE Trans. Circuits Syst.*, vol. 47, Oct. 2000.
[19] H. Kriplani, F. Najm, and I. Hajj, "Improved delay and current models for estimating maximum currents in CMOS VLSI," in *IEEE Custom Integrated Circuits Conf.*, San Diego, CA, May 1–4, 1994, pp. 429–432.
[20] B. S. Cherkauer and E. G. Friedman, "Channel width tapering of serially connected MOSFET's with emphasis on power dissipation," *IEEE Trans. VLSI Syst.*, vol. 2, pp. 100–113, Mar. 1994.
[21] J. Meyer, *Semiconductor Device Modeling for CAD*, Herskowitz and Schilling, Eds. New York: McGraw Hill, 1972, ch. 5.
[22] Y. H. Shih and S. M. Kang, "Analytic transient solution of general MOS circuits primitive," *IEEE Trans. Computer-Aided Design*, vol. 11, pp. 719–731, June 1992.
[23] Y. H. Shih, L. Leblebici, and S. M. Kang, "A fast timing and reliability simulator for digital MOS circuits," *IEEE Trans. Computer-Aided Design*, vol. 12, pp. 1387–1402, Sept. 1993.
[24] Q. Wang and S. K. Vrudhula, "A new short circuit power model for complex CMOS gates," in *Proc. IEEE Alessandro Volta Memorial Workshop Low Power Design*, Como, Italy, Mar. 4–5, 1999, pp. 98–106.
[25] D. S. Kung and R. Puri, "Optimal P/N ratio selection for standard cell libraries," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, San Jose, CA, Nov. 7–11, 1999, pp. 178–184.
[26] C. Mead and L. Conway, "Introduction to VLSI systems," in *Addison-Wesley Ser. Computer Sci.*, 2nd ed., Oct. 1980.

Publicly Detectable Watermarking for Intellectual Property Authentication in VLSI Design

Gang Qu

***Abstract*—Highlighted with the newly released intellectual property (IP) protection white paper by VSI Alliance, the protection of virtual components or IPs in very large scale integration (VLSI) design has received a great deal of attention recently. Digital signature/watermark is one of the most promising solutions among the known protection mechanisms. It provides desirable proof of authorship without rendering the IP useless. However, it makes the watermark detection, which is as important as watermarking, an NP-hard problem. In fact, the tradeoff between hard-to-attack and easy-to-detect and the lack of efficient detection schemes are the major obstacles for digital signatures to thrive. In this paper, the authors propose a new watermarking method which allows the watermark to be publicly detected without losing its strength and security. The basic idea is to create a cryptographically strong pseudo-random watermark, embed it into the original problem as a special (which the authors call *mutual exclusive*) constraint, and make it public. The authors combine data integrity technique and the unique characteristics in the design of VLSI IPs such that adversaries will not gain any advantage from the public watermarking for forgery. This new technique is compatible with the existing constraint-based watermarking/fingerprinting techniques. The resulting public–private watermark maintains the strength of a watermark and provides easy detectability with little design overhead. The authors build the mathematical framework for this approach based on the concept of mutual exclusive constraints. They use popular VLSI CAD problems, namely technology mapping, partitioning, graph coloring, FPGA design, and Boolean satisfiability, to demonstrate the public watermark's easy detectability, high credibility, low design overhead, and robustness.**

***Index Terms*—Authentication, copy detection, data integrity, information hiding, intellectual property protection, mutual exclusive constraints, virtual component, watermarking.**

I. Introduction

The advances in very large scale integration (VLSI) semiconductor technology and the system-on-a-chip design paradigm, coupled with the shrinking time-to-market window, have changed the traditional system design methodology. Design reuse and intellectual property (IP), also called virtual component, based designs have become more and more important. IP trading plays a central role in the design-for-reuse methodology and the potential of infringement is growing fast. However, the global awareness of IP protection remains low. The goals of IP protection are to enable IP providers to protect their IPs against unauthorized use and to detect and to trace the use of IPs [22].

According to the IP protection white paper released recently by VSIA, there are three approaches to the problem of securing an IP: deterrent approaches like patents, copyrights, and trade secrets; protection via licensing agreements or encryption; detection mechanisms such as physical tagging, digital watermarking, and fingerprinting [22]. Of the early efforts on IP protection, only detection mechanisms enable designers to do the so-called self-protection. Other approaches require either time, or money, or both. Using the detection methods, designers embed digital signatures or other traceable marks into IPs

Manuscript received September 18, 2001; revised March 23, 2002. This paper was recommended by Associate Editor R. Gupta.

The author is with the Electrical and Computer Engineering Department and Institute of Advanced Computer Study, University of Maryland, College Park, MD 20742 USA(e-mail: gangqu@eng.umd.edu).

Digital Object Identifier 10.1109/TCAD.2002.804205

during the design and implementation phases, such that unauthorized usage can be detected and the source of theft can be traced. There are three main detection mechanisms: 1) *tagging and tracking techniques*, where labels are attached to IPs in the manufacturing phase for the purpose of tracing [23], [27]; 2) *digital watermarking techniques*, where digital signatures are hidden in the design such that they can be later revealed by IP owners to show authorship [3], [8], [10], [14]; and 3) *fingerprinting techniques*, where a unique copy of IP is created to trace each individual user [2], [12].

Clearly the success of digital signatures relies on their detectability and traceability. Therefore, developing efficient detection techniques is essential to the protection mechanism and is as important as developing watermarking techniques. However, the general copy detection process is equivalent to the problems of pattern matching or subgraph isomorphism which are well-known NP-hard and most of the existing watermarking/fingerprinting literature leave this as an open challenging problem [7], [10]. Compared to watermarking and fingerprinting, we see the research on copy detection lacking both in breadth and in depth. To date, three different approaches for copy detection in VLSI designs have been reported.

For several instances (namely scheduling, graph coloring, and gate-level layout), Kahng *et al.* [9] choose signatures selectively and develop fast comparison schemes to detect such signatures. To enable these fast comparison algorithms, one has to first identify a common structural representation of IPs and what constitutes an element of the IP structure; secondly, one has to determine a means of calculating locally context-dependent signatures for such elements. Although this approach is generic, it is not always easy to find such a common structure and to design fast and accurate pattern matching algorithms.

Charbon and Torunoglu [4] discuss copy detection under a design environment that involves IPs from multiple sources and that requires IP providers to register their IPs in a trusted agent. They first generate a compact signature from every IP block independently and make it public. Then they perform the IP integration process in a way such that one can extract the signatures from the final design. This approach requires every IP provider to deposit his signature into a "bank" maintained by a third party. And again, matching algorithms need to be developed to detect signatures from a circuit.

More recently, Kirovski *et al.* [11] propose a forensic engineering technique to identify solutions generated by strategically different algorithms. They first run each algorithm on a large number of instances to collect statistical data, then these algorithms are clustered based on the properties of solutions they find. To detect which algorithm is applied to obtain a given solution, they simply check its properties based on which the algorithm clustering has been performed.

In the broader area of information hiding, most of the reported literature on detection focuses on how to extract and recover the embedded data with the secret key from the stego-data [15]. There are only two existing approaches to make watermarks publicly detectable. One is based on the so-called public-key watermarking [6], the other relies on zero-knowledge protocols [5].

In this paper, we propose a new watermarking technique to solve the copy detection problem. The core concept is to divide the watermark into two parts: the public part which is made visible to the public, and the private part which is only visible for authorized people. Both the public and private watermark are in the form of additional design constraints. Their difference is that the public watermark is embedded in designated locations with known methods to guarantee public detectability, while the private part is embedded in a secret way as in the traditional constraint-based watermark. We use cryptographic techniques for data integrity to deter any attempt of removing or modifying the public watermark. The separation of public watermark and private watermark provides the following advantages.

- It facilitates easy public copy detection. A relatively convincing authorship can be verified by end users, in constant time, without forensic experts. This to a great extent deters illegal redistribution.
- The public watermark is hard to forge because it is generated by a data integrity technique and embedded in the design process of VLSI IPs.
- The new technique is compatible with all existing watermarking/fingerprinting methods. The performance overhead to gain easy and public detectability is little.
- Each public watermark is guaranteed to be distinct, which means that a 100% credibility will be achieved (from the public part only) if this method is adopted by all IP providers.

The rest of the paper is organized as follows. In Section II, we explain the creation, embedding, and detection of the public–private watermark. We introduce the concept of *mutual exclusive constraints* and build the theoretical background for the generic public watermarking technique in Section III. This approach can be applied to many well-studied problems in the context of VLSI such as Boolean satisfiability (SAT), partitioning, field-programmable gate array (FPGA) layout, technology mapping, and graph coloring. We validate the public watermarking technique and report experimental results on SAT and graph coloring before we draw conclusions in Section V.

II. Public-Private Watermarking Technique

Watermarking and fingerprinting are indirect protection schemes in that they provide a deterrent to infringers by providing the ability to demonstrate ownership of an IP to its originator [22]. However, establishing the ownership is challenging as we have discussed earlier. In this section, we propose the public–private watermarking technique that simultaneously provides credibility as high as any traditional constraint-based watermark and the public detectability that no other watermarks can.

Watermark Creation: The proposed public–private watermark consists of two parts: public and private, which are selected separately. We inherit the private part of the watermark from the traditional digital watermark discussed in early works [2], [7], [8], [10], [14]. A typical watermark is a cryptographically strong pseudo-random bit stream created by crypto systems using designer's digital signature as the secret key. For example, we can hash the plain text message to get a 128-bit or higher hash result. We then apply a stream cipher to make the same plain text message pseudo random using the above hash result as the key. The public watermark has a header and a body that are both derived from a *short* plain text message such as the four- or three-letter symbol for the design company. The ASCII code of this short text is used as the public watermark header. The public watermark body, which is a pseudo-random bit stream, is created exactly the same way as we build the private watermark.

Watermark Embedding: Watermark embedding is the process of translating the watermark, a pseudo-random bit stream, into design constraints. The public and private watermark can be embedded by either the same encoding scheme or different ones. The development of such schemes requires us to explore the characteristics of the given problem and we will discuss this in Section IV on several specific VLSI CAD problems. However, to ensure the detectability of public watermark, we must make the following public: 1) the one-way hash function being used in the construction of public watermark; 2) the (public) watermark encoding scheme being used to create the constraints; and

3) the place where we embed these (public watermark related) constraints. We keep the secret key out of the reach of public to make the private watermark secure[1].

Watermark Detection: The public watermark can be detected from the following public information: 1) the hash function; 2) the watermark scheme; and 3) the place that hosts the (public) watermark. First, we check for the existence of constraints in the hosts for public watermark and obtain the entire public watermark message. Next, we extract the message header (since its length is known), which is the designer's public signature in ASCII. Then, we can hash this message header for further verification of the authorship. A match of the message header with designer's publicly known signature gives the first level proof of authorship. A match of the message body with the new hash result will establish stronger proof. Finally, further evidence can be shown when the secret key (for the private watermark) is available or forensic tools can be used to detect the private watermark by the existing copy detection techniques [4], [9], [11].

Watermark Robustness: Unlike the private watermark, which is as secure as before, the public watermark is visible to the public and may be vulnerable against attacks. In most known watermarking techniques, attackers will have a great amount of advantage if they can detect the watermark. The public watermark exploits the concept of *data integrity* to prevent this. To be more specific, a forgery is successful if the adversary can replace the original public watermark by his public information. The adversary can create his own public watermark using the published hash function. Then he can alter the constraints based on the public-known watermark encoding scheme and embed them in the specific places. Finally, he modifies the known solution, which satisfies the original public watermark, to meet his faked public watermark constraints. Now a successful forgery is built! However, this is unrealistic[2], if not impossible, due to two facts: 1) the faked hash will be different from the original in half of the bits statistically even if the message header is changed only by one bit, which is significant if we make the message body sufficiently long. 2) the design integrity implies that even one small local change may alter the behavior of the design, which means that any forgery will require some level of local modification.

III. Theory of Public Watermarking

Public watermarking differs from traditional (or private) watermarking in the way the watermark is embedded into the design as additional constraints. It is the basic assumption that watermark should be "invisible" [3], [7], [8], [18]. It is also this assumption that makes watermark detection hard. In the public watermarking approach, the watermark is embedded into a special type of constraints, which we call *public watermark holder*, using a known encoding scheme. We have already discussed in the previous section about the creation, embedding, detection, and robustness of public watermark. In this section, we give a summary of the mathematic foundation for constructing public watermark holders. A detailed description can be found in the technical report [16].

We embed the public watermark by adding a special type of constraint: mutual exclusive constraints. Given a problem $\mathcal{P}$, a set of $n \geq 2$ constraints $\{C_1, C_2, \ldots, C_n\}$ are *mutual exclusive* if any solution $\mathcal{S}$ satisfies at most one constraint C_i, $(1 \leq i \leq n)$. A mutual exclusive set of constraints is *complete* if any solution $\mathcal{S}$ satisfies exactly one constraint. A mutual exclusive set is *strongly mutual exclusive* if for any constraint C_i, there exists a solution $\mathcal{S}$ that satisfies C_i and violates C_j $(j \neq i)$.

Existence Theorem: A complete strongly mutual exclusive set exists for all problems with more than two different solutions.

Cutting Space Theorem: A set of complete strongly mutual exclusive constraints partitions the solution space as the union of nonempty disjoint subsets.

Data Hiding Theorem: n different pieces of information (of any length) can be hidden with a (complete) strongly mutual exclusive set of n constraints.

Intuitively, each user will be assigned one constraint from a set of complete strongly mutual exclusive constraints for his public watermark holder. When the watermarked problem is solved, solution will only come from the subset that satisfies this constraint. From the cutting space theorem, we conclude that there will not be any collision[3] among different watermark holders. This essentially provides the ultimate 100% proof of the authorship and it is independent of the length of the public watermark. Clearly, it is of our interest to find large complete strongly mutual exclusive sets to accommodate the possible large number of public watermarks. We now introduce a constructive method to construct large complete mutual exclusive sets.

Two constraints are *independent* if any solution's satisfiability to one constraint has no impact on its satisfiability to the other one. Two sets of constraints, $\{C_1, C_2, \ldots, C_n\}$ and $\{C'_1, C'_2, \ldots, C'_m\}$, are *independent* if C_i and C'_j are independent for any $1 \leq i \leq n$, and $1 \leq j \leq m$. The *join* of two sets of constraints, $\{C_1, \ldots, C_n\}$ and $\{C'_1, \ldots, C'_m\}$, is defined as the set $\{C_1 \wedge C'_1, \ldots, C_1 \wedge C'_m, C_2 \wedge C'_1, \ldots, C_n \wedge C'_m\}$, where constraint $C_i \wedge C'_j$ is satisfied if and only if both constraints C_i and C'_j are satisfied.

Join Theorem: If two complete strong mutual exclusive sets are independent and have n and m constraints, respectively, then their join is a complete strong mutual exclusive set with $n \cdot m$ constraints.

We summarize the public watermarking approach as follows.

1) *Build a public watermark holder:*
   1) *Obtain a group of complete strongly mutual exclusive and independent constraints* $\{C_1, C'_1\}, \{C_2, C'_2\}, \ldots, \{C_k, C'_k\}$.
   2) *Construct their join* $\{\bar{C}_{i_1} \wedge \bar{C}_{i_2} \wedge \cdots \wedge \bar{C}_{i_k} : \bar{C}_{i_j} = C_{i_j}$ or $C'_{i_j}\}$.
2) *Create a public watermark:*
   1) *Convert (plain text) public signature into an ASCII bitstream* $\mathcal{PS}$.
   2) *Hash* $\mathcal{PS}$ *by a one-way hash function to get the hash result* $\mathcal{H}(\mathcal{PS})$.
   3) *Encode* $\mathcal{H}(\mathcal{PS})$ *by a stream cipher with* $\mathcal{PS}$ *as the key to get* $\mathcal{SC}_{\mathcal{PS}}(\mathcal{H}(\mathcal{PS}))$.
   4) public watermark $= \mathcal{PS} \cdot \mathcal{SC}_{\mathcal{PS}}(\mathcal{H}(\mathcal{PS}))$, *where* $\cdot$ *is the string concatenation.*
3) *Embed the public watermark:*
   1) *For public watermark* $p_1 p_2 \ldots p_k$, *we choose public watermark holder* $\bar{C}_1 \wedge \cdots \wedge \bar{C}_k$, *where* $\bar{C}_i = C_i$ *if* $p_i = 0$ *and* $\bar{C}_i = C'_i$ *if* $p_i = 1$.
   2) *Embed constraints* $\{\bar{C}_1, \ldots, \bar{C}_k\}$ *into design.*

The stego-problem is obtained by adding *THE* constraints that correspond to the public watermark under the embedding scheme in Step 3. The strongly mutual exclusiveness guarantees the existence of stego-solution (or watermarked solutions) to the stego-problem. Different

[1]The security of the cryptographic function depends on the secret key, not on which hash function or stream cipher we use to encrypt the message. Also, it is the digital signature, which is independent of the watermark encoding schemes, that carries the proof of authorship.

[2]By *unrealistic*, we mean that the performance degradation of the modified IP is so large that one will not accept it and the design loses its value.

[3]A collision occurs when one solution meets more than one public watermark. In such a situation, one cannot identify the real author(s) and the watermark fails.

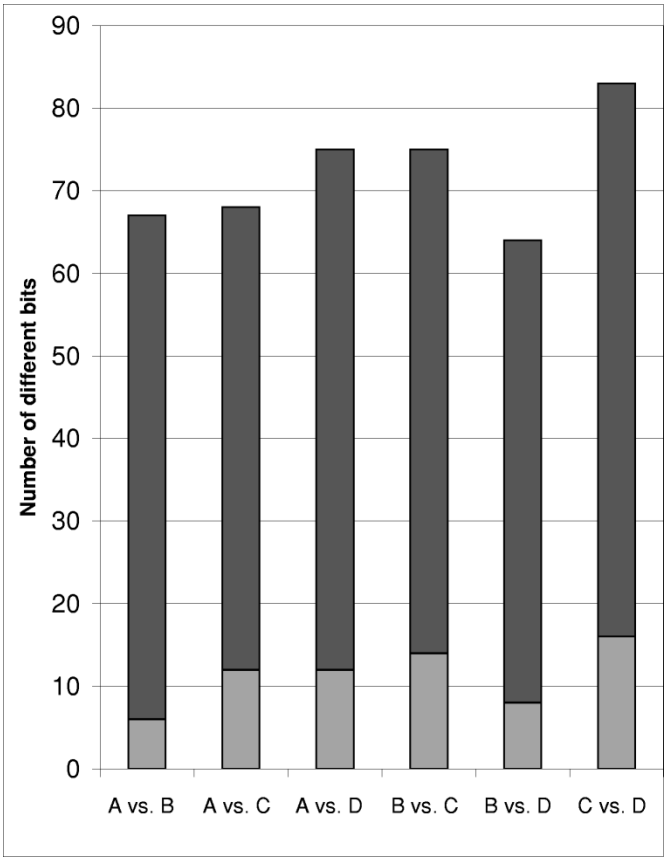

Fig. 1. Hamming distance among the four public watermark message. The bottom half comes from the message header (plain text part), and the top half comes from the message body (results of RC4).

watermarks will be mapped to different constraints of the strongly mutual exclusive set. Therefore, all stego-problems will be different and the property of mutual exclusiveness guarantees their solutions will be distinct. In sum, we have the following.

Theorem (Correctness of the Approach): If the constraints are strongly mutual exclusive, there always exist (stego-) solutions for the stego-problem. Furthermore, different stego-problems will have different (stego-)solutions that are all solutions to the original problem.

IV. Validation and Experimental Results

We have explained how to create the public–private watermark which is a pseudo-random bit stream (except the header of public watermark). We have conducted case studies on several well-known VLSI CAD problems to validate this approach. In this section, we report the results on SAT and graph coloring problems to demonstrate the public watermark's robustness and its impact to the system's performance (or quality of the solution). Detailed results on other problems, such as partitioning, standard cell place and route, technology mapping, and FPGA layout can be found in the technical report [16].

A. Boolean Satisfiability

The Boolean SAT seeks to decide, for a given formula, whether there is a truth assignment for its variables that makes the formula true. SAT appears in many contexts in the field of VLSI CAD, such as automatic pattern generation, logic verification, timing analysis, delay fault testing, and channel routing. We necessarily assume that the SAT instance to be protected is satisfiable and that there is a large enough solution space to accommodate the watermark.

Given a formula $\mathcal{F}$ on a set of Boolean variables V, the simplest watermarking technique for public detectability is to hide the public watermark behind a known subset of variables $\{v_1, v_2, \ldots, v_k\}$. Suppose the public watermark message is $m_k \ldots m_2 m_1$, we embed it by forcing $v_i = m_i$ in the solution. This can be done by adding a single-literal clause v_i (if $m_i = 1$) or v_i' (if $m_i = 0$)[4] to the formula $\mathcal{F}$.

We pick four four-letter messages A, B, C, and D. We use MD5 [19], [25] as the one-way hash function to obtain four 128-bit messages H(A), H(B), H(C), and H(D). Next we use RC4 [26] to encrypt these messages using their ASCII codes as the encryption keys. The resulting pseudo-random bit streams are appended to the ASCII codes of the corresponding plain text to form the four public watermark messages. Fig. 1 shows the Hamming distance for each of the six pairs among these four public watermark message. A and B, B and D are relatively close because each pair has one letter in common accidentally.[5]

We now embed these public watermark messages to DIMACS SAT benchmarks, where the instances are generated from the problem of inferring the logic in an eight-input, one-output "blackbox" [28]. We first select 32 variables for the message header, then choose 128 (or 64 for instances of small size, e.g., with less than 600 variables) more variables for the message body. We then assign values to these variables based on the public watermark and solve for the assignment of the rest variables to get the original solution.

With the given solution (and variables that carry the public watermark), an adversary retrieves the public message header, modifies it, and computes the new message body. He then embeds this forged message and resolves the problem. Our goal is to show that there is little correlation between the original solution and adversary's new solution, i.e., attacker has little advantage from the original solution or it is equally difficult to obtain a solution.

Table I shows our experimental results, where messages A, B, C, D are embedded to the four SAT instances, respectively. The second column gives the number of variables N in these instances. We consider the adversary changes randomly 4, 8, 16, and 24 bits in the 32-bit message header. We repeat each trial five times, the columns labeled "body" show the average number of bits changed in the faked message body from the original. We solve each instance with this faked message (both header and body) embedded and calculate the Hamming distance between the new solution and the original solution. The average distances (rounded to the nearest integer) are reported in columns with the label "sol.".

The last two rows report these average distances percentage-wise. The first is the distance in public domain, which is very close to 50% if we exclude the mandatory header part. It is independent of the number of bits being modified in the header and shows the robustness of our cryptographic tools in generating pseudo-random bit streams. The last row shows that the new solutions are not close to the original solution. (When we solve the original instances for multiple solution, their average distance is also about 45%.) Therefore, we can conclude that the new solutions are independent of the given solution, which means that once the public watermark has been modified, the adversary loses almost all the advantage from the given solution. This is further verified by the fact that the run time difference for resolving the problem and solving from scratch is so small (within 5%) that we consider they are the same.

[4]A single-literal clause imposes a very strong constraint to the formula. Statistically it will cut the entire solution space by one-half. Therefore, we may use a short public watermark message, in particular for instances with not so many variables. However, the credibility can always be enhanced by adding private watermark using other techniques, such as those proposed in [8].

[5]The ASCII codes for messages A, B, C, and D are: "01 010 011 01 000 111 01 001 001 00 100 000", "01 000 011 01 000 100 01 001 110 00 100 000", "01 010 011 01 001 110 01 010 000 01 010 011", and "01 001 101 01 000 101 01 001 110 01 010 100".

TABLE I
AVERAGE NUMBER OF DIFFERENT BITS IN PUBLIC MESSAGE BODY ("BODY"), AVERAGE DISTANCE (ROUNDED TO INTEGER) FROM THE ORIGINAL SOLUTION ("SOL.") WHEN 4-, 8-, 16-, AND 32-BIT FORGERY IS CONDUCTED TO THE PUBLIC MESSAGE HEADER ON SAT BENCHMARKS

		4 bits in header		8 bits in header		16 bits in header		24 bits in header	
$\mathcal{F}$	N	body	sol.	body	sol.	body	sol.	body	sol.
ii8b1	336	31.2	148	32.8	150	31.8	168	32.6	170
ii8b2	576	33.6	260	30.6	258	32.4	265	32.0	272
ii8b3	816	62.2	363	64.0	376	67.4	358	61.6	387
ii8b4	1068	65.8	489	66.2	472	63.4	492	62.6	513
Ave. Dist. (%)		40.2%	-	43.5%	-	50.5%	-	56.2%	-
Ave. Dist. (%)		-	44.9%	-	44.9%	-	46.5%	-	48.3%

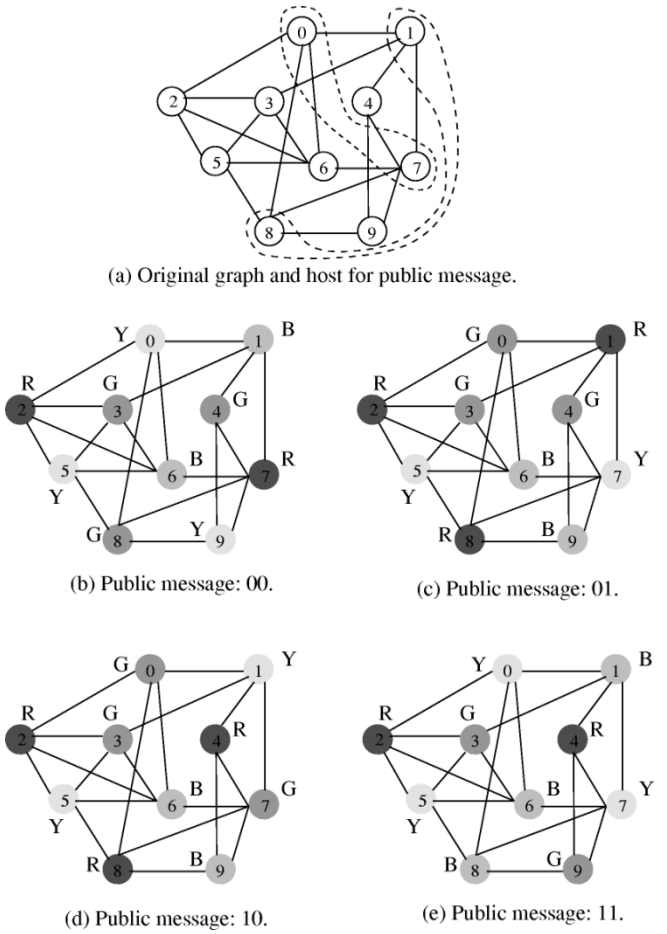

Fig. 2. Four GC solutions with different public watermarks added to the same graph. Letters beside the nodes stand for the colors: Y(ellow), B(lue), R(ed), G(reen).

B. Graph Coloring

The NP-hard graph vertex coloring optimization seeks to color a given graph with as few colors as possible, such that no two adjacent vertices receive the same color. We propose the following public–private watermarking technique for the graph coloring problem and use it to demonstrate the impact of our approach on the quality of the solution.

For a given graph, we select pairs of vertices that are not connected directly by an edge. We hide one bit of information behind each pair as follows: adding one edge between the two vertices and thus making them colored by different colors to embed 1; collapsing this pair and thus forcing them to receive the same color to embed 0.

Consider Fig. 2, where two pairs of unconnected vertices, nodes 0 and 7, and nodes 1 and 8, are selected as shown in the dashed circles in 2(a). The rest of Fig. 2 shows four different coloring schemes with a 2-bit public watermark message embedded. To detect such watermark, one can simply check the colors received by nodes 0, 1, 7, and 8. For example, in Fig. 2(c), nodes 0 and 7 are colored by G(reen) and Y(ellow), respectively, which means the first bit (the most significant bit) is 0. Similarly, the observation that nodes 1 and 8 are both colored by R(ed) tells us the second bit of the message is 1. Therefore, we detect a public message "01".

TABLE II
EMBEDDING PUBLIC WATERMARK TO REAL LIFE GRAPH AND RANDOMIZED GRAPH

original instance			32-bit message		64-bit message	
	vert.	opt.	overhead	best	overhead	best
fpsol2.i.1	496	65	0.2	65	0.7	65
fpsol2.i.2	451	30	0.1	30	0.5	30
fpsol2.i.3	425	30	0.1	30	0.5	30
inithx.i.1	864	54	0.0	54	0.2	54
inithx.i.2	645	31	0.9	31	1.8	32
inithx.i.3	621	31	1.1	31	1.9	32
DSJC1000	1000	85.8	0.5	86	2.0	87

To evaluate the tradeoff between protection and solution degradation (in the case of graph coloring, the number of extra colors), we first color the original graph, then color the watermarked graph and compare the average number of colors required. We consider two classes of real life graphs (the *fpsol2* and *inithx* instances from [29]) and the DIMACS on-line challenge graph [28].

Table II shows the number of vertices in each graph, the optimal solutions except the DSJC1000 problem which is still open (the number in the table for this problem is the average of ten trials with 85-color solutions occur several times), and the overhead introduced by public watermark messages of various length. For each instance, we create ten 32-bit and ten 64-bit public watermark messages randomly. We add the message to the graph and color the modified graph. The average number of colors and the best solution we find are reported. One can easily see that the proposed approach causes little overhead for real life instances but loses best solutions for the randomized DSJC1000 graph. The reason is that there exist localities in real life graphs of which we can take advantage. However, such localities do not exist or are very difficult to find in random graphs.

V. CONCLUSION

Our work is motivated by the proliferation of IP reuse in VLSI design and the potential of it being illegally redistributed and misused. We propose a public–private watermarking method, the first that allows the IP's authorship to be established easily and publicly. We achieve this by allowing part of the watermark to be public. We use cryptographic techniques, in particular techniques for data integrity, to protect the public watermark from forgery. Using the traditional constraint-based watermark as private part, this public–private watermarking scheme is capable of providing public detectability with no degradation on the

watermark's strength. We explain the basic approach and develop specific techniques for various classes of VLSI CAD problems. The new approach is compatible with all the existing watermarking techniques. With the help from organizations pushing for design standards, for example VSIA, this method has the potential of solving eventually the IP protection problem.

REFERENCES

[1] A. Adelsbach, B. Pfitzmann, and A. Sadeghi, "Proving ownership of digital content," in *Proc. 3rd Int Information Hiding Workshop*, Sept 1999, pp. 126–141.

[2] A. E. Caldwell, H. Choi, A. B. Kahng, S. Mantik, M. Potkonjak, G. Qu, and J. L. Wong, "Effective iterative techniques for fingerprinting design IP," *36th ACM/IEEE Design Automation Conf. Proc.*, pp. 843–848, June 1999.

[3] E. Charbon, "Hierarchical watermarking in IC design," in *Proc. IEEE 1998 Custom Integrated Circuits Conf.*, May 1998, pp. 295–298.

[4] E. Charbon and I. Torunoglu, "Copyright protection of designs based on multi source IP's," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, Nov. 1999, pp. 591–595.

[5] S. Craver, "Zero knowledge watermark detection," in *Proc. 3rd Int. Information Hiding Workshop*, Sept. 1999, pp. 102–115.

[6] F. Hartung and B. Girod, "Fast public-key watermarking of compressed video," in *Proc. IEEE Int. Conf. Image Processing*, Oct. 1997, pp. 528–531.

[7] I. Hong and M. Potkonjak, "Behavioral synthesis techniques for intellectual property protection," *Proc. 36th ACM/IEEE Design Automation Conf. Proc.*, pp. 849–854, June 1999.

[8] A. B. Kahng, J. Lach, W. H. Mangione-Smith, S. Mantik, I. L. Markov, M. Potkonjak, P. Tucker, H. Wang, and G. Wolfe, "Watermarking techniques for intellectual property protection," *35th ACM/IEEE Design Automation Conf. Proc.*, pp. 776–781, June 1998.

[9] A. B. Kahng, D. Kirovski, S. Mantik, M. Potkonjak, and J. L. Wong, "Copy detection for intellectual property protection of VLSI design," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, Nov. 1999, pp. 600–604.

[10] D. Kirovski, Y. Hwang, M. Potkonjak, and J. Cong, "Intellectual property protection by watermarking combinational logic synthesis solutions," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, Nov. 1998, pp. 194–198.

[11] D. Kirovski, D. Liu, J. L. Wong, and M. Potkonjak, "Forensic engineering techniques for VLSI CAD tools," *37th ACM/IEEE Design Automation Conf. Proc.*, pp. 581–586, June 2000.

[12] J. Lach, W. H. Mangione-Smith, and M. Potkonjak, "FPGA fingerprinting techniques for protecting intellectual property," *Proc. IEEE 1998 Custom Integrated Circuits Conf.*, pp. 299–302, May 1998.

[13] ——, "Signature hiding techniques for FPGA intellectual property protection," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, Nov. 1998, pp. 186–189.

[14] A. L. Oliveira, "Robust techniques for watermarking sequential circuit designs," *36th ACM/IEEE Design Automation Conf. Proc.*, pp. 837–842, June 1999.

[15] B. Pfitzmann, "Information hiding terminology," in *1st Int. Information Hiding Workshop*, May 1996, pp. 347–350.

[16] G. Qu, "Publicly detectable watermarking for intellectual property authentication in vlsi design," Univ. Maryland Inst. Advanced Computer Studies (UMIACS), UMIACS-TR-2002-17, 2002.

[17] G. Qu and M. Potkonjak, "Analysis of watermarking techniques for graph coloring problem," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, Nov. 1998, pp. 190–193.

[18] G. Qu, J. L. Wong, and M. Potkonjak, "Optimization-intensive watermarking techniques for decision problems," *36th ACM/IEEE Design Automation Conf. Proc.*, pp. 33–36, June 1999.

[19] R. L. Rivest. (1992) The MD5 Message-Digest Algorithm. [Online]. Available: http:// www.cis.ohio-state.edu/htbin/rfc/rfc1321.html

[20] N. A. Sherwani, *Algorithms for VLSI Physical Design Automation*, 3rd ed. New York: Kluwer, 1999.

[21] "Architecture Document Version 1.0," Virtual Socket Interface Alliance, 1997.

[22] "Intellectual property protection white paper: Schemes, alternatives and discussion Version 1.0," Virtual Socket Interface Alliance, 2000.

[23] "Virtual component identification physical tagging standard Version 1," Virtual Socket Interface Alliance, 2000.

[24] G. Wolfe, J. L. Wong, and M. Potkonjak, "Watermarking graph partitioning solutions," *38th ACM/IEEE Design Automation Conf. Proc.*, pp. 486–489, June 2001.

[25] [Online]. Available: ftp://ftp.sunet.se/pub3/vendor/sco/skunkware/uw7/fileutil/md5/src

[26] [Online]. Available: ftp://ftp.ox.ac.uk/pub/crypto/misc/rc4.tar.gz

[27] [Online]. Available: http://www.siidtech.com/

[28] [Online]. Available: http://dimacs.rutgers.edu/

[29] [Online]. Available: http://mat.gsia.cmu.edu/COLOR/instanc5pes.html

Preferred Direction Steiner Trees

Mehmet Can Yildiz and Patrick H. Madden

***Abstract*—The planar rectilinear Steiner tree problem has been extensively studied. The common formulation ignores circuit fabrication issues such as multiple routing layers, preferred routing directions, and vias between layers. In this paper, the authors extend a previously presented planar rectilinear Steiner tree heuristic to consider layer assignment, preferred routing direction restrictions, and via minimization. They use layer-specific routing costs, via costs, and have a minimum cost objective. Their approach combines the low computational complexity of modern geometry-based methods with much of the freedom enjoyed by graph-based methods. When routing costs mirror those of traditional planar rectilinear Steiner problems, the authors' approach obtains close to 11% reductions in tree lengths, compared to minimum spanning trees; this is on par with the performance of the best available Steiner heuristics. When via costs are significant and layer costs differ, they observe average cost reductions of as much as 37%. Their method can also reduce the number of vias significantly.**

***Index Terms*—Interconnect synthesis, optimization, routing, Steiner tree.**

I. INTRODUCTION

The Steiner problem has a long history in very large scale integration (VLSI) computer-aided design; global and detail routers use Steiner trees to define routes for signal nets, and many interconnect optimization approaches begin with Steiner constructions and then apply wire sizing, driver sizing, and buffer insertion to improve performance. Steiner tree research has focused on *planar rectilinear* formulations, which are appropriate for single layer routing and are

Manuscript received June 5, 2001; revised February 1, 2002. This work was supported in part by the National Science Foundation under Grant CCR-9988222. This paper was recommended by Associate Editor T. Yoshimura.

The authors are with the Computer Science Department, State University of New York, Binghamton, NY 13092 USA (e-mail: pmadden@binghamton.edu).

Digital Object Identifier 10.1109/TCAD.2002.804105