Robust IP Watermarking Methodologies for Physical Design*

Andrew B. Kahng, Stefanus Mantik, Igor L. Markov, Miodrag Potkonjak,
Paul Tucker†, Huijuan Wang and Gregory Wolfe

UCLA Computer Science Dept., Los Angeles, CA 90095-1596
† UCSD Computer Science & Engineering Dept., La Jolla, CA 92093-0114

Abstract

Increasingly popular reuse-based design paradigms create a pressing need for authorship enforcement techniques that protect the intellectual property rights of designers. We develop the first intellectual property protection protocols for embedding design *watermarks* at the physical design level. We demonstrate that these protocols are transparent with respect to existing industrial tools and design flows, and that they can embed watermarks into real-world industrial designs with very low implementation overhead (as measured by such standard metrics as wirelength, layout area, number of vias, routing congestion and CPU time). On several industrial test cases, we obtain extremely strong, tamper-resistant proofs of authorship for placement and routing solutions.

1 Introduction

Due to rapidly growing device counts, shortened design cycle times and a compounding "design productivity shortfall" [9], core-based design and software reuse strategies are widely believed to be the only viable implementation alternatives for the next level of integrated circuits and their system integration. As a result, development of intellectual property protection (IPP) techniques and tools has emerged as a very prominent open research topic. In this work, we develop the first protocols for IPP at the *physical design* level, using the concept of constraint-based watermarking.

As defined in [5], a *design watermark* is an invisible (i.e., imperceptible to human or machine analysis) identification code that is permanently embedded as an integral part within a design. Various criteria for a given watermarking-based IPP technique, as determined by a leading industry organization, include [12]: (i) maintenance of functional correctness, (ii) transparency to existing design flows; (iii) minimal overhead cost; (iv) enforceability; (v) flexibility in providing a spectrom of protection levels; (vi) persistency; (vii) invisibility; and (viii) proportional component protection. In [5], we list other watermarking desiderata, and describe a canonical approach to watermarking-based IPP where additional *constraints* encoding the IP author's signature are added into a given design optimization instance. The solution of the optimization instance, in satisfying these constraints (which would be unlikely in a random solution to the original instance), implicitly contains a proof of authorship. The approach described in [5] is transparent to existing design flows in that it relies on *preprocessing* (of inputs) or *postprocessing* (of solutions) with respect to any given design optimization.

We center on the physical design phase for several reasons.

- Physical design is traditionally viewed as a "difficult" domain, where even a small percentage variation in solution quality can make or break a design, and where high-quality solutions are known to have strong structural resemblance to each other [4]. Devising a watermarking technique that can make a solution "unique", without compromising solution quality, is quite challenging in such a domain.
- With deep-submicron technology, many performance constraints (e.g., budgeted edge delays consistent with path timing bounds) cannot be considered satisfied until they are satisfied in the physical design. Thus, for example, it is disingenuous to "watermark" a design by constraining timing budgets, without verifying that such constraints are satisfied after physical design.
- Other trends – IP reuse methodologies, higher perceived valuation of "hard IP", increasing availability of multiple foundry sources, difficulty of performance validation before physical design, changing handoff models, etc. – all point to physical design as an appropriate juncture in the design cycle for watermarking.

Contributions of Our Work

To our knowledge, this work gives the first solution for IPP at the physical design level. For placement, we propose a *postprocessing* flow that encodes a signature as specified parity (i.e., odd- or even-index) of the cell row within which particular standard cells must be placed. For routing, we propose a *preprocessing* flow that encodes a signature as upper bounds on the wrong-way wiring used to route particular signal nets.[1] Using real industrial design examples and commercial layout tools, we demonstrate the effectiveness of both the preprocessing- and the postprocessing-based watermarking. In particular, strong signatures are achieved without compromising any of the standard metrics for solution quality (routability, wirelength, number of vias, CPU time, etc.). We also demonstrate that these signatures are tamper-resistant. We conclude that addressing IP protection at a lower level of abstraction has an advantage: designs inherently have orders of magnitude more components, allowing significantly stronger proofs of authorship as well as lower overhead. We also conclude that the postprocessing approach is not only feasible, but indeed quite attractive for several reasons: (i) it enables watermarking of already existing designs; (ii) it enables direct calculationg of the hardware overhead incurred by IPP;[2] and (iii) is may be likelier to find acceptance among designers and managers, since the complete design process is not altered in any way.

* Work by M. Potkonjak and G. Wolfe supported in part by DARPA under grant N66001-97-2-8901. Work by A. B. Kahng, S. Mantik, I. L. Markov, P. Tucker and H. Wang supported by a grant from Cadence Design Systems, Inc.

[1] The particular constraints used to encode the watermark affect the strength of the authorship proof as well as metrics of the layout solution. Devising constraint types such that strong signatures can be achieved transparently using existing flows and tools is, in our experience, a very nontrivial task.

[2] When preprocessing is used, two designs (one without additional signature constraints, and one with) must be realized in order to determine the hardware overhead incurred by IPP.

2 Related Work

Related work in artifact watermarking and cryptography is reviewed in [5].[3] We therefore focus our survey of related concepts within the physical design realm.

No previous work in the literature deals with watermarking of physical design solutions. However, constraint specification and management now receive close attention through all phases of chip implementation, including physical design. Standard chip implementation flows begin with such high-level constraints as clock cycle times and offsets, I/O boundary timing, power dissipation bounds, and choice of packaging and implementation technology. Derived constraints[4] then arise throughout the RTL floorplanning, block placement, and routing phases. Within physical design, the following constraint types are most common.

- Timing constraints. Path delay constraints are often expressed in some form of Standard Delay Format (SDF), with heuristic "path cover" techniques used to reduce data volume and improve convergence of timing-driven layout tools. A static timing analysis engine may operate directly from the clock cycle times/offsets and I/O boundary timing to evaluate timing correctness, without explicit enumeration of timing path constraints. For purposes of layout design, path delay constraints are typically budgeted into individual constraints on source-sink edges [11].
- Physical (floorplanning) constraints. To improve timing convergence of the design process, assumptions made during RTL floorplanning or block floorplanning must be propagated to downstream flow stages (e.g., placement and global routing). This is often accomplished via region constraints: a given cell must be located in a given region of the layout, a set of cells must be co-located as a "group", etc. Such constraints may be captured using PDEF or equivalent formats which allow specification of assumed routing topology, layer usage, etc. at the level of global routing.

The mechanisms by which physical design tools enforce such constraints vary widely. However, classic paradigms such as top-down min-cut placement synthesis or ripup-and-reroute interconnect synthesis generally do not support constraints well. One reason is that iterative search mechanisms used today were originally adopted for regimes with "smooth" cost surfaces, while adding constraints induces more 0-1 "discontinuities" in the cost surface. Another reason is that many performance constraints are "global" (e.g., path delay, power dissipation, EMC, signal integrity) while current layout approaches rely on local optimizations. Most importantly, "good" solutions to hard combinatorial problems are often quite similar.[5] The implications for watermarking in physical design are that (i) current tools do not easily support too many "extra" watermarking constraints, and (ii) introduction of too many watermarking constraints will likely degrade solution quality. These issues complicate the choice of watermarking technique.

[3] Our present techniques use well-established ingredients – namely, the cryptographic hash function MD5, the public-key cryptosystem RSA, and the stream cipher RC4 [8, 7] – on which many state-of-the-art commercial cryptographic programs are based.

[4] Derived constraints are of two basic types. *Inferred* constraints can often be viewed as "transformed", e.g., when a signal net's wirelength upper bound is inferred from a signal propagation delay upper bound. *Refined* constraints can often be viewed as created by a "budgeting" or "allocation" process, e.g., when a global path delay constraint is broken up into separate edge delay constraints.

[5] This has been generally characterized as a "big valley" [3] or "massif central" [6]; the phenomenon has also been specifically documented for standard-cell placements under the minimum wirelength objective [4].

3 Watermarking Standard-Cell Place and Route

We have considered a variety of mechanisms by which standard-cell physical design can be constrained. Our goal has been to develop a watermarking protocol that, beyond satisfying criteria listed above, is (i) consistent with existing design practices and tools, (ii) relatively easy to implement, and (iii) acceptable in terms of its impact on real-world layout metrics.

3.1 Row-Based Placement

For row-based placement, traditional physical embedding constraint types (i.e., region and grouping constraints) are straightfoward to realize. Region constraints are transparent to top-down placers, since iterative partitioners accomodate "fixed" preassignments (see [2] for a review). Annealing placers (see [10] for a review) also support such constraints by restricting move generation, and analytic placers support region constraints via inequalities or center-of-gravity constraints (see [1] for a review). Grouping constraints are typically enforced by inducing contracted netlists over clustered representations of the design. However, region and grouping constraints are not well-suited to placement-based watermarking: when made without any structural knowledge of the netlist or of good placement solutions, they can lead to substantial deterioration of solution quality.[6]

Our approach bases the watermarking constraints on the underlying fine-grain placement substrate, which is well-defined prior to the placement phase of design. By "placement substrate", we mean the row structure of legal site locations in the physical floorplan. In particular, we constrain individual cells to be placed with specified *cell row parity*. For example, cell INV4_10 might be constrained to be placed in a cell row that has EVEN index; cell RKPPX1Y might be constrained to an ODD-index row. Our approach has the following advantages:

- Very few constraints are needed to make a strong signature. E.g., if the signature constrains 50 cells with specific row parities, and if the placer realizes all 50 constraints, the chances are 2^{-50} that this could have occurred by accident. Typical placement instances have tens of thousands of standard cells.
- It is compatible with region and group constraint types, and can be applied as soon as a gate-level netlist exists (no specific row/site plan is needed). A priori, the only time failure is guaranteed is when a "watermark cell" is constrained to be in, e.g., an EVEN row, and is simultaneously constrained to be in a region that contains only a single ODD row.
- It is not easy to tamper with the signature via local perturbations: the small signature size implies that many cells must be perturbed before the signature becomes unrecoverable. Furthermore, local perturbations that shift cells between cell rows are difficult to make without worsening the solution quality.
- It is not affected by downstream stages of the design flow. Many current design methodologies do not significantly change the row assignments (let alone the locations) of existing cells during routing; hence, our proposed watermarking scheme will remain intact.
- It allows the watermark to be realized completely during the placement phase. (For schemes such as the watermarking of budgeted timing constraints, the realization remains incomplete until after routing [5].)

[6] The impact on solution quality is even worse when the design is performance-constrained. Imagine the effect of arbitrarily constraining a cell in a timing-critical path to be in the "wrong" region of the layout.

3.2 Routing

For standard-cell routing, applicable constraints usually involve performance (e.g., crosstalk and delay bounds) or reliability (antenna rules, electromigration and self-heat limits, hot-electron rules). How these constraints are represented, how they are enforced, and what degrees of freedom (e.g., shielding, tapering, spacing, repeater insertion, driver sizing, topology design, etc.) are exploited depends on the routing tool.

We considered constraint types involving segment widths, spacings, and choice of topology. These not only are difficult to enforce within current routing approaches, but also have potentially harmful interactions with performance constraints (e.g., a watermarking constraint might require a net to be routed at minimum separation from its closest neighbors; a crosstalk constraint might dictate otherwise). We also considered various "parity" watermarking schemes based on, e.g., the orientation of the "L" for two-pin connections, the parity of the number of segments, the parity of path lengths in the routing, etc. These were dismissed as highly unnatural (e.g., pin access clearly dictates which "L" the router will choose), difficult to enforce using known routing methodologies (e.g., parity of total tree length), or vulnerable to simple tampering (e.g., tampering by compaction would ruin length-parity schemes).

Our approach bases the watermarking constraints on the (per-net) *costing* of the underlying routing resource. Specifically, for each watermark net we impose unusual costs on "wrong-way" and/or via resources, and hope that the watermark nets are provably unusual in their utilization of such resources. Many commercial routers already accept such control of the routing cost structure on a per-net basis. Our approach has the following advantages:

- Very few constraints are needed to make a strong signature, assuming that the resource costing is reflected in the routing result for each watermark net.
- It is compatible with many existing routing constraints, e.g., those that are based on wire width, spacing or shielding.[7]
- It is not easy to disturb the signature with local perturbations: the small signature size implies that many nets will need to be rerouted before the signature is likely to be unrecoverable. Furthermore, as designs are increasingly limited in terms of the interconnect resource, the routing of watermark nets is likely to be "locked in" by the routing of the remaining non-watermark nets. Hence, destroying the watermark requires rerouting of the design.

3.3 Commercial Tools

To demonstrate practicality and allow evaluation with respect to real-world design metrics, we test watermarking protocols that are transparent to existing layout tools. Our implementation of the proposed placement and routing watermarking methodologies uses commercial tools from Cadence Design Systems: placement watermarking is built around QPlace v4.0.27 and WarpRoute v1.0.10, and routing watermarking is built around the IC Craftsman v2.1.3 router using a standard constraint type in this tool ("limit way" rule). We now give details of the experimental protocol.

4 Experimental Protocol

4.1 Placement

Our experimental methodology is designed to show how easily an existing tool can be modified to offer watermarking capability. The basic comparison is shown in Figure 1. A traditional non-watermarked placement flow reads library and design information

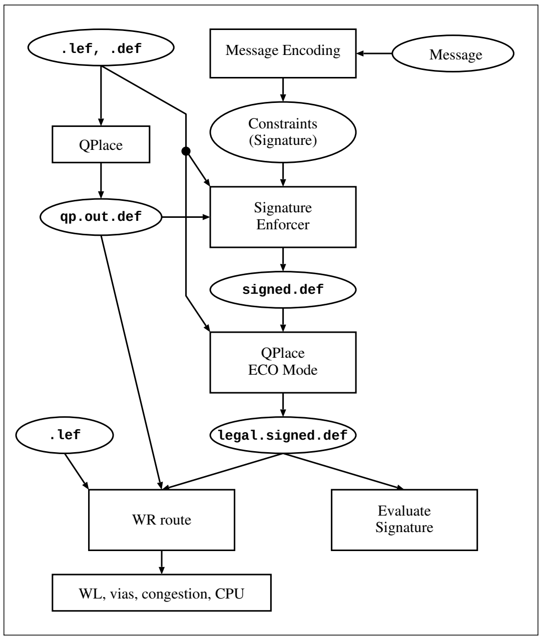

Figure 1: A postprocessing-based watermarking protocol for standard-cell placement, using the Cadence Design Systems QPlace v4.0.27 and WarpRoute v1.0.10 tools.

via LEF/DEF formats, executes QPlace, then executes WarpRoute to evaluate the placement quality. This is shown on the left side of Figure 1. Our *postprocessing-based* watermarking flow, shown on the right of the figure, consists of the following steps:

1. We read the default QPlace placement result (a DEF file with location data) and the LEF file into our internal design database.
2. We ask the user for a message (e.g., "Placed by QP on 10-10-97") which we then transform into row-parity constraints for some subset of the core (non-pad, standard) cells of the design.
3. We enforce all the row-parity constraints by local changes to the placement (e.g., pair-swap operations), generating a "signed DEF" file.
4. We ensure that the resulting placement is ready for routing, by re-running in "ECO mode"; this makes only minimal changes to the placement, and only if necessary (typically, to avoid illegal overlaps with fixed obstacles or other cells). The output of this step is a "legal signed DEF" file.
5. We execute WarpRoute, and evaluate the placement quality.

We make the following observations. (1) Our postprocessing approach is absolutely equivalent to what might be implemented in a modification of the actual commercial tool. Alternatively, our watermarking flow is trivially implemented by scripting and standard capabilities of the commercial placer (LEF/DEF manipulation, ECO placement, etc.). (2) We begin with a high-quality solution and retrospectively impose constraints. Not only is this a good approach to maintaining solution quality, but from the outside one cannot tell whether the watermarked placement is created from scratch or by postprocessing of a nonwatermarked placement. (3) The "final list of core cells" is a well-defined concept in all existing design flows including those that invoke "in-place optimization" or

[7] Some potential conflicts exist with respect to wrong-way routing, but these have not been an issue in our experience, particularly since we impose *upper* bounds on the use of wrong-way routing.

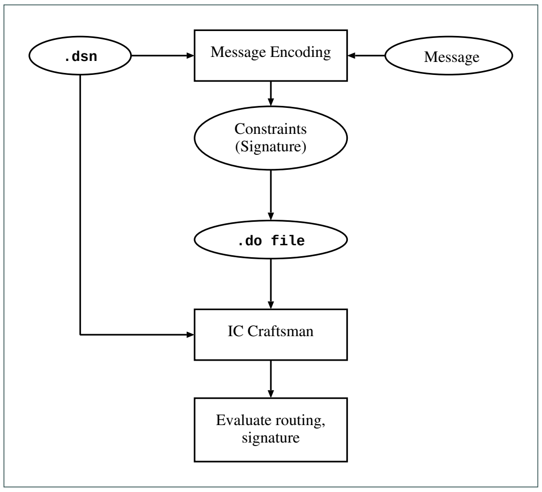

Figure 2: A preprocessing-based watermarking protocol for (standard-cell) gridless area routing, using the Cadence Design Systems IC Craftsman v2.1.3 tool. The tool is used in its standard context, controlled by a ".do" file using standard rules syntax.

"placement-based synthesis". Thus, generating a watermark based on core cell indices and row parities is also well-defined.[8]

4.2 Routing

A traditional non-watermarked routing flow using the IC Craftsman router reads library and placed design information via the .dsn file format, then executes the router under the control of a ".do file". This is shown on the left side of Figure 2. Our *preprocessing-based* watermarking flow, shown on the right side of the Figure, consists of the following steps:

1. We identify all unique signal net names in the .dsn file.
2. We ask the user for a message (e.g., "Routed by ICC on 10-10-97") which we then transform into a list of "watermark nets" (some subset of the net names in the design).
3. We then constrain the watermark nets using IC Craftsman rules in the "do file". Specifically, our methodology applies a "limit way = 1" rule to each of the watermark nets.
4. We execute the ICC router.

4.3 Evaluation of Signature Strength

Each constraint involves some "random" choice, e.g., choosing a random cell or signal net. (Such choices are not actually random, but use a cryptographically strong pseudorandom number generator that is seeded with a binary signature file.) The choices may occur either with or without replacement. If there is replacement, then constraints will be independent of each other. Even if there is no replacement, the constraints may very nearly act as if they were independent, especially if the pool of constraints to choose from is large relative to the number of constraints actually chosen. As long as the constraints are either independent or nearly so, the probability P_c of a solution carrying an author's watermark purely by coincidence can be computed by a simple binomial. We use P_c to measure the strength of the authorship proof.

Let X be the number of constraints imposed, let x be the number of these that are *not* satisfied, and let p be the probability of a constraint being satisfied purely by coincidence.[9] The probability that x or fewer out of X constraints are satisfied by coincidence is given by $P_c = \sum_{i=0}^{x}(C(X,i)\cdot(p)^{X-i}\cdot(1-p)^i)$.

- For our placement watermarks, the signature consists of a certain subset of cells, each constrained to be in a cell row with specified-parity index. We use $p = 0.5$ as the chance that a given cell will satisfy its constraint by coincidence.

- For our routing watermarks, the signature consists of a certain subset of signal nets, each with an "unusually low" limit on the amount of wrong-way wiring that can be used to route the net. We use the following methodology to establish a binary indicator of whether a given net has been "successfully watermarked". Given a routed design, we evaluate the total wirelength (WL_{tot}) and the wrong-way wirelength (WL_{way}) for each signal net. We then rank all nets in order of increasing value of the ratio WL_{way}/WL_{tot}. The watermark nets are expected to occur earlier in this ranking, while non-watermark nets are expected to occur later in this ranking. We then establish a cutoff rank below which a watermark net is considered "successfully watermarked", and above which a watermark net is considered "not successfully watermarked". In the routing experimental results reported below, we always set the threshold rank at the 40th percentile, i.e., $p = 0.4$. (Stronger results can be obtained by more carefully choosing the value of p; this is noted below.)

4.4 Resistance to Tampering Attacks

Another way to evaluate the strength of a given watermark is to assess its resistance to attacks (see [5] for a survey of prototypical attacks). Thus, in addition to reporting P_c values, we also report the resistance of our watermarking schemes to the following *tampering* attacks.[10] In these scenarios, the *attacker* is trying to erase the watermark by small layout perturbations.

- **Placement**
  - Assumptions: (i) the attacker has access only to an incremental ("legalizing") placement tool such as QPlace ECO mode;[11] (ii) the watermarking scheme is unknown to the attacker; and (iii) original design constraints are retained.
  - Attack: (i) select N random pairs of cells and swap the locations of each cell pair; and (ii) run the legalizing placer to legalize the design (continue with routing, etc.).
- **Routing**
  - Assumptions: (i) the attacker has access only to incremental (single-net) auto-routing; (ii) the watermarking scheme is unknown to the attacker; and (iii) original design constraints are retained.
  - Attack: select N random nets, then reroute these nets with only the original design constraints (if any).

[8] If one desires a netlist-dependent, *floorplan-independent* watermark, then one must assume a canonical row indexing (e.g., top-down for horizontal rows and left-right for vertical rows). One can also simply define the row parity constraints in terms of two equivalence classes of constrained cells. We also note that the implicit assumption of fixed cell names is also reasonable; any methodology allowing arbitrary renaming of cells would likely have some overhead for verification.

[9] If constraints are not sufficiently independent of each other, p is not well-defined. Overestimating the value of p always makes P_c larger, which we interpret as weaker strength of the authorship proof. Since overestimating p can never improve the supposed strength of our watermark, we can typically obtain useful estimates for p even when its exact value is not known.

[10] A tampering attack attempts to remove the rightful IP owner's signature, and possibly introduce the attacker's own signature into the IP.

[11] Recall that the attacker could always re-solve the placement problem from scratch to remove the IP owner's signature, but we assume this is too expensive.

5 Experimental Results

We applied our proposed physical design watermarking protocols to six industry test cases, four in placement and two in routing. Aspects of the test cases are given in Table 1. The routing test case *sc*2 has a relatively small number of nets relative to cells because many signals are pre-routed, and hence not included in the netlist.

	Placement				Routing	
	test1	test2	test3	test4	sc1	sc2
#cells	9011	12133	12857	20577	1653	4250
#nets	11962	11828	10880	25634	1802	1597

Table 1: Numbers of cells and nets in the six industry test cases.

5.1 Watermark Strength P_c

Results for the placement experiments are summarized in Table 2. We report five post-routing layout quality measures for each test case. These measures are: total wirelength, total number of vias, percentage of overcongested "global routing cells" (as reported by the placer), and CPU time in (mm:ss) required by the router (all CPU times are for a 140MHz Sun Ultra-1). Together, these measures provide a fairly complete picture of the utility of each placement. In the Table, the subscript *orig* indicates the default unwatermarked solution; the subscript $wm-x,y$ indicates a watermarked solution with x cells in the signature, of which y ended up being successfully watermarked. There is essentially no solution quality overhead to introducing the placement watermark. Our placement watermarking protocol also shows graceful degradation of solution quality if extremely strong signatures are required.

Test Case	WL	# Vias	Cong	CPU	P_c
$t1_{orig}$	6.38	86072	1.52%	11:38	
$t1_{wm-56,52}$	6.40	86595	1.52%	12:03	5.5e-12
$t1_{wm-112,96}$	6.40	86449	1.52%	12:13	2.2e-15
$t1_{wm-224,189}$	6.42	86712	1.54%	12:10	4.8e-27
$t1_{wm-448,389}$	6.44	87143	1.53%	12:08	5.7e-61
$t1_{wm-896,786}$	6.51	87716	1.52%	13:02	7.3e-127
$t1_{wm-1792,1526}$	6.62	88955	1.55%	13:25	8.3e-215
$t2_{orig}$	3.32	95601	0.86%	9:30	
$t2_{wm-56,53}$	3.33	95811	0.78%	9:20	4.1e-13
$t2_{wm-112,110}$	3.33	95978	0.80%	9:06	1.2e-30
$t2_{wm-224,208}$	3.34	95913	0.77%	9:11	4.5e-44
$t2_{wm-448,409}$	3.35	96554	0.91%	9:15	3.4e-79
$t2_{wm-896,837}$	3.38	97902	0.94%	9:28	3.2e-177
$t2_{wm-1792,1678}$	3.42	99467	1.13%	9:56	2.9e-357
$t3_{orig}$	3.13	52401	4.47%	13:58	
$t3_{wm-56,55}$	3.15	52433	4.57%	11:32	7.9e-16
$t3_{wm-112,110}$	3.14	52636	4.60%	11:45	1.2e-30
$t3_{wm-224,219}$	3.16	52529	4.65%	11:53	1.7e-58
$t3_{wm-448,443}$	3.17	53003	4.57%	11:44	2.1e-124
$t3_{wm-896,879}$	3.21	53559	5.00%	11:45	7.2e-235
$t3_{wm-1792,1740}$	3.25	54279	5.14%	11:58	3.2e-439
$t4_{orig}$	8.13	179526	0.02%	17:07	
$t4_{wm-56,50}$	8.14	179680	0.02%	17:35	5.1e-10
$t4_{wm-112,102}$	8.14	179678	0.02%	14:42	1.2e-20
$t4_{wm-224,201}$	8.16	180052	0.02%	15:45	5.7e-37
$t4_{wm-448,407}$	8.18	180590	0.02%	23:49	3.5e-77
$t4_{wm-896,797}$	8.25	182224	0.02%	19:59	1.6e-136
$t4_{wm-1792,1597}$	8.32	183783	0.02%	17:44	6.9e-274

Table 2: Watermarking results for placement. Wirelengths are scaled to 10^8 user database units.

Results for the routing experiments are summarized in Table 3. We report three post-routing layout quality measures: total wirelength (WL), total number of vias, and CPU time required by the IC Craftsman router. Since our watermarking strategy is based on limiting the length of acceptable wrong-way routing in watermarked nets, we also report total wrong-way wirelength (WW) in each solution. Finally, we report the value of P_c for each watermarked design. Increasing the signature size (i.e., the number of watermark nets constrained with the "limit way = 1" rule) improves the value of P_c without significantly degrading the routing performance.

Test Case	WL	WW	# Vias	CPU	P_c
$sc1_{orig}$	5.48	9.67	13440	422:27	
$sc1_{wm-20,20}$	5.46	9.56	13320	402:18	1.1e-8
$sc1_{wm-40,40}$	5.49	9.55	13426	672:19	1.2e-16
$sc1_{wm-80,78}$	5.48	9.23	13433	503:48	1.1e-28
$sc1_{wm-160,159}$	5.46	8.32	13681	641:32	5.2e-62
$sc1_{wm-320,315}$	5.47	7.51	13921	450:34	9.5e-117
$sc2_{orig}$	2.29	3.54	5590	23:52	
$sc2_{wm-20,20}$	2.29	3.60	5547	21:29	1.1e-8
$sc2_{wm-40,40}$	2.30	3.77	5716	18:58	1.2e-16
$sc2_{wm-80,79}$	2.30	3.47	5713	16:50	1.8e-30
$sc2_{wm-160,151}$	2.29	3.23	5650	20:43	1.3e-48
$sc2_{wm-320,294}$	2.30	2.93	5706	20:43	2.2e-85

Table 3: Watermarking results for routing. Total wirelengths (WL) are scaled to 10^7 user database units, and wrong-way wirelengths (WW) are scaled to 10^6 units.

As a side note, recall from above that the value of p (a consequence of the threshold rank) may be chosen to optimize the signature strength measure P_c. Table 4 shows how calculated P_c values can vary as p varies from 0.2 to 0.4. In the Table, the second column gives the size of the signature (number of watermark nets), and each entry $x(y)$ represents the P_c value (x) and the number of unsuccessfully watermarked nets (y). We observe that fine-tuning of p (e.g., choosing $p = 0.35$) could potentially improve our results.

5.2 Resistance to Tampering

Tables 5 and 6 present results of experiments where we attempt to tamper with placement and routing watermarks, respectively. respectively. In each Table, the second column indicates the original signature size, and the third column (Init) gives the original watermarked solution quality (total WL). Subsequent columns indicate the number of cell pair-swap (net ripup and reroute) operations performed in the placement (routing) tampering, expressed as a percentage of the total number of cells (nets) in the design. We report P_c values for the 10% column to show that the watermarks remain strong even after tampering. For placement (Table 5), the solution quality degrades much faster than the signature strength, even though we restricted all random pair swaps to occur over Manhattan

test case	# of nets	p values				
		0.2	0.25	0.3	0.35	0.4
sc1	20	1.1e-14(0)	9.1e-13(0)	3.5e-11(0)	7.6e-10(0)	1.1e-8(0)
	40	1.4e-24(2)	1.0e-22(1)	1.2e-21(0)	5.8e-19(0)	1.2e-16(0)
	80	3.0e-46(5)	8.9e-41(4)	2.6e-38(2)	3.7e-33(2)	1.1e-28(2)
	160	3.5e-91(10)	4.9e-82(7)	1.7e-75(4)	3.4e-71(1)	5.2e-62(1)
	320	6.2e-91(88)	2.2e-115(46)	3.2e-129(20)	8.9e-124(11)	9.5e-117(5)
sc2	20	1.5e-5(7)	1.6e-9(2)	3.8e-8(2)	7.6e-10(0)	1.1e-8(0)
	40	7.9e-10(14)	2.5e-15(6)	5.3e-18(2)	4.4e-17(1)	1.2e-16(0)
	80	1.9e-13(34)	7.0e-32(10)	2.5e-33(5)	3.7e-33(2)	1.8e-30(1)
	160	2.6e-17(80)	5.8e-32(49)	3.3e-47(24)	2.3e-45(18)	1.3e-48(9)
	320	2.5e-8(214)	2.4e-34(137)	7.7e-61(81)	1.5e-83(41)	2.2e-85(26)

Table 4: P_c values corresponding to different values of p. Each entry $x(y)$ represents the P_c value (x) and the number of unsuccessfully watermarked nets (y).

Test	# cells	Init	1 %	2 %	5 %	10 %	P_c
t3	56	3.15	3.24	3.34	3.61	4.00	4.1e-13
	112	3.14	3.25	3.33	3.60	4.03	4.9e-25
	224	3.16	3.26	3.34	3.61	4.02	4.5e-44
	448	3.17	3.27	3.36	3.62	4.04	9.1e-95
	896	3.21	3.30	3.39	3.65	4.08	1.1e-180
	1792	3.25	3.35	3.44	3.69	4.12	1.1e-334
t4	56	8.14	8.28	8.68	9.83	11.57	1.3e-7
	112	8.14	8.31	8.75	9.80	11.52	1.3e-14
	224	8.16	8.34	8.71	9.86	11.64	2.7e-29
	448	8.18	8.36	8.76	9.80	11.62	3.4e-48
	896	8.25	8.42	8.88	9.96	11.63	6.2e-94
	1792	8.32	8.48	8.88	9.94	11.64	6.1e-196

Table 5: Result from tampering with the placement watermark. Solution quality degrades much faster than signature strength; hence, tampering does not appear to be a viable form of attack.

Test	# nets	Init	1%	2%	5%	10%	P_c
sc1	20	5.46	5.47	5.47	5.49	5.53	1.1e-5
	40	5.49	5.50	5.50	5.51	5.55	1.3e-11
	80	5.48	5.48	5.48	5.49	5.54	4.2e-21
	160	5.46	5.46	5.47	5.48	5.54	3.1e-38
	320	5.47	5.48	5.48	5.49	5.54	1.4e-61
	CPU	534:06	28:30	51:58	118:26	127:25	
sc2	20	2.29	2.29	2.29	2.29	2.30	7.7e-8
	40	2.30	2.30	2.30	2.31	2.31	8.6e-13
	80	2.30	2.29	2.30	2.30	2.30	1.1e-23
	160	2.29	2.29	2.29	2.30	2.30	2.3e-36
	320	2.30	2.30	2.30	2.31	2.31	1.7e-67
	CPU	19:45	1:24	2:18	3:57	8:39	

Table 6: Result from tampering with the routing watermark. CPU times for ripup and reroute approach the time required to re-solve the problem from scratch; hence, tampering does not appear to be a viable form of attack.

distances less than twice the cell row height. (ECO placement CPU times were consistent and small, and we do not report them.) For routing (Table 6), the solution quality appears relatively immune to tampering (other measures such as number of vias also remained constant). However, the CPU time required to tamper with a large number of nets approaches the cost of redoing the entire solution from scratch (at which point tampering is not needed). We conclude that our watermarking schemes are quite robust with respect to random tampering.

Finally, Figure 3 shows the watermarked layout of test case *sc*1 (56 watermark nets), and Figure 4 shows the unwatermarked layout of the same design.

6 Conclusions

In conclusion, we have developed the first IPP protocols for embedding design watermarks at the physical design level. We have implemented these protocols transparently to existing design flows, using leading industrial tools. On real designs, we show strong proofs of authorship with very acceptable cost overhead for the watermarking. We also show the robustness of our watermarking scheme with respect to random tampering attacks. Our current research is aimed at extending these ideas to alternate watermarking strategies that are appropriate for various application scenarios in physical design, and that will remain robust under various specific forms of attack.

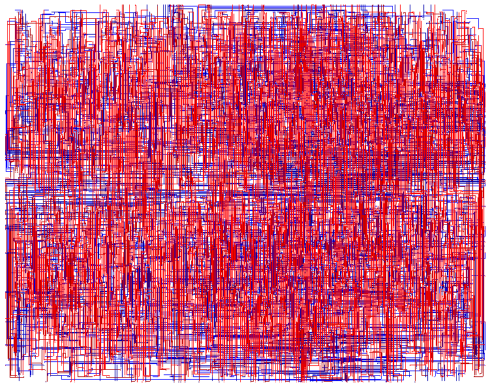

Figure 3: Routing solution (watermarked) for the sc1 test case.

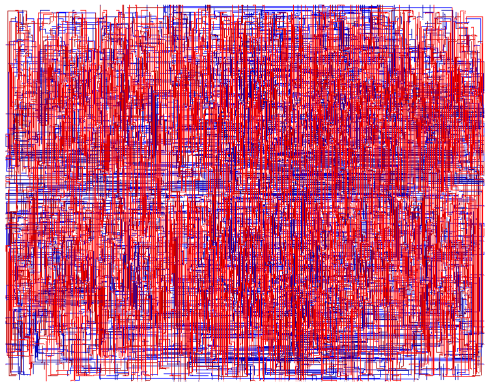

Figure 4: Routing solution (unwatermarked) for the sc1 test case.

References

[1] C. J. Alpert, T. Chan, D. J. Huang, A. B. Kahng, I. Markov, P. Mulet and K. Yan, "Faster Minimization of Linear Wirelength for Global Placement", in *Proc. ACM/IEEE Intl. Symp. on Physical Design*, Napa, April 1997, pp. 4-11.

[2] C. J. Alpert and A. B. Kahng, "Recent Directions in Netlist Partitioning: A Survey", *Integration: The VLSI Journal* 19 (1995), pp. 1-81.

[3] K. D. Boese, A. B. Kahng and S. Muddu, "New Adaptive Multistart Techniques for Combinatorial Global Optimizations", *Operations Research Letters* 16(2) (1994), pp. 101-113.

[4] D. C.-M. Chi, *Improving Upon Local Search Heuristics for VLSI Standard Cell Placement*, M.S. Thesis, UCLA Computer Science Department, 1995.

[5] 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", *Proc. ACM/IEEE Design Automation Conf.*, 1998.

[6] S. A. Kauffman, *At Home in the Universe: The Search for Laws of Self-Organization and Complexity*, New York, Oxford University Press, 1995.

[7] A. J. Menezes, P. C. van Oorschot and S.A. Vanstone, *Handbook of Applied Cryptography*, Boca Raton, CRC Press, 1997.

[8] B. Schneier, *Applied Cryptography : Protocols, Algorithms, and Source Code in C*, 2nd ed., New York, Wiley, 1996.

[9] Semiconductor Industry Association, *The National Technology Roadmap for Semiconductors: Technology Needs*, December 1997.

[10] W. Swartz and C. Sechen, "Timing Driven Placement for Large Standard Cell Circuits", *Proc. ACM/IEEE Design Automation Conf.*, June 1995, pp. 211-215.

[11] G. E. Tellez, D. A. Knol and M. Sarrafzadeh, "A Performance-Driven Placement Technique Based on a New Budgeting Criterion", *Proc. IEEE Intl. Symp. on Circuits and Systems*, May 1996, vol. 4, pp. 504-507.

[12] VSI Alliance, *Fall Worldwide Member Meeting: A Year Of Achievement* (guidelines proposed by VSIA development working group on intellectual property protection), Santa Clara, Oct. 1997.