Signature Hiding Techniques for FPGA Intellectual Property Protection

John Lach[1], William H. Mangione-Smith[1], Miodrag Potkonjak[2]
Departments of Electrical Engineering[1]and Computer Science[2]
The University of California, Los Angeles

Abstract – This work presents the first known attempt to leverage the unique characteristics of FPGAs to protect commercial investments in intellectual property. A watermark is applied to the physical layout of a digital circuit when it is mapped into an FPGA. This watermark uniquely identifies the circuit origin and yet is difficult to detect. While this approach imposes additional constraints, experiments involving a number of large complex designs indicate that the performance impact is small.

1 Introduction

We have developed and evaluated a method for applying cryptographically encoded watermarks to digital designs. The approach is shown to successfully encode long messages on existing designs of moderate to large complexity with little or no impact on circuit performance or resource requirements. By using these messages to encode authorship signatures, we can provide compelling evidence to establish design ownership.

1.1 Motivation

It is generally agreed that the most significant problem facing digital IC designers today is system complexity. Complex systems tend to be assembled using smaller components in order to reduce complexity as well as to take advantage of localized data and control flows. This trend toward partitioning enables design reuse, which is essential to reducing development cost and risk while also shortening design time. Design reuse has been employed by systems designers for years; what is new is that the boundaries for component partitions have moved inside of the IC packages. These reusable modules are commonly referred to as Intellectual Property (IP), as they represent the commercial investment of the originating company but do not have a natural physical representation.

Direct theft is a major concern of IP vendors. It is possible for customers, or a third party, to simply sell an IP block as their own without even reverse engineering the design. Because IP blocks are designed to be modular and integrated with other system components, the thief can simply repackage them without bothering to understand either the architecture or implementation.

This paper presents a novel solution to the risk of direct misappropriation. The essential idea involves embedding a digital watermark, which uniquely identifies the creator, in an IP block. This watermark allows the IP owner to verify the physical layout as their property, in a way that is likely to be much more compelling than the existing option of verifying the design against a registered database.

1.2 Motivational Example

While the concepts developed here can be applied to a wide range of FPGA architectures, all of the discussion and experimental work will be conducted in the context of the Xilinx XC4000 architecture [13]. These devices are composed of an array of configurable logic blocks (CLBs), each of which contains two flip-flops and two 16x1 lookup tables (LUTs). A hierarchical and segmented routing network is used to connect CLBs in order to form a specific circuit configuration.

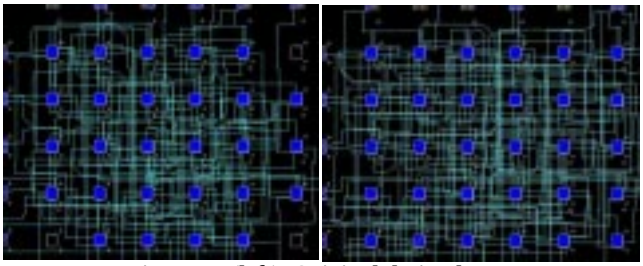


Figure 1a (left). Original design layout
Figure 1b (right). Watermarked design layout

Consider the case of PREP Benchmark #4, a large state machine, which can be mapped into a block of 27 CLBs. This mapping results in 3 unused CLBs, or 3*32=96 unused LUT bits. Each unused LUT bit is used to encode one bit of the signature. Figure 1a shows the layout of the original design as produced by the standard Xilinx backend tools, while Figure 1b shows the layout for the same design after applying the watermark constraints to the three unused CLBs and re-placing the design. The constrained CLBs are then incorporated into the design with unused interconnect and neighboring CLB inputs, further hiding the signature.

1.3 Technological Issues

The standard digital design flow generally follows these steps: behavioral HDL, synthesis to RTL, technology mapping, and finally physical layout involving place and route. Watermarking can be applied to any level of this design flow and, if developed properly, will propagate to later stages [2, 6]. However, because a watermark is fundamentally an optional component of a system design, any watermark can be removed by reverse engineering a design to a stage in the flow before the watermark has been applied. For example, the detailed approach developed here will be used to watermark a design at the physical level by manipulating LUTs and interconnect. The IP vendor will then deliver their technology in the form of a hard macro. If the macro can be reverse engineered to a netlist, the watermark will be removed, specifically because it is not a functional part of the circuit operation. Fortunately, most FPGA vendors have taken a business position that they will not reveal the specification of their configuration streams, specifically to complicate the task of

reverse engineering and thus protect the investment of their customers [10].

1.4 Contributions

This paper presents the first method for protecting Intellectual Property, in the form of reusable digital circuits, even after the IP has been delivered in commercial products. By manipulating hardware resources, we are able to encode relatively long messages in a manner that is difficult to observe by a third party, resists tampering, and has little impact on circuit performance or size. This capability provides three main benefits:

1. It reduces the risk that a watermarked circuit will be stolen, i.e. used illegally without payment or transferred to a third party.
2. It reduces the risk that any unmarked circuit will be stolen.
3. It can be used to identify the backend tool chain used to develop a design, and thus be part of the royalty mechanism used for CAD tools.

2 Reverse Engineering Techniques

While Xilinx and other FPGA vendors make some efforts to complicate the task of reverse engineering, it certainly is possible to recover the configuration specification with a concerted effort. NeoCAD was able to accomplish this for the Xilinx XC4000 series devices through a directed investigation of the bitstreams produced by the Xilinx backend tools. Given this information, it should be relatively straightforward to produce a Xilinx netlist file and then use commercial tools to move back up the design flow. Another possible line of attack involves removing the packaging material and then using a visual inspection tool to produce a circuit representation of the CLB. A similar approach has recently been used to produce a complete layout of a 386 microprocessor in approximately 2 weeks [1].

In response to the proven success of the reverse engineering attacks, we believe that hiding the watermark is necessary but not sufficient. Any effective watermarking scheme should make the signature appear to be part of the functional digital circuit to whatever extent is possible.

3 Related Work

Ad-hoc techniques for the watermarking of text and image documents have been manually practiced for many centuries. Modern techniques for signature data hiding in image, video, and audio signals have received a great deal of attention. A spectrum of steganography-based approaches for protection of digital images has been proposed [3, 9, 12].

Recently, a set of techniques for intellectual property protection through watermarking at the behavioral level down to the physical layout using superimposition of additional constrains in conjunction to those specified by the user has been proposed [2, 5, 6]. In this paper, we propose the first intellectual property protection technique for FPGA designs. Different design phases (physical synthesis of FPGA-based design vs. behavioral synthesis) result in very different sets of synthesis and optimization issues.

Cryptography also has a long history. Two decades ago, the public-key techniques introduced by Stanford researchers redefined the field [4]. Many techniques, from both a practical and theoretical viewpoint, have been summarized in [8].

We use cryptographic techniques to select a subset of FPGA physical design constraints from a set of constraints that are not already used for design specification. An additional benefit is that the cryptographic techniques also provide probabilistic randomization and therefore protection from added constraints. For this task, we use the standard cryptography tools from the PGP-cryptography suite, the secure hash function MD5, and the RSA/MIT stream cipher RC4 [8].

4 Approach

The global flow of our watermarking system is represented by the pseudo-code in Figure 2. First, the complete design passes through the vendor place and route tool in order to get an initial estimate of the resource constraints. The process terminates if the available resources are not sufficient to satisfy the watermark request. In this case, the IP developer has the option of either mapping into a larger physical area or requesting a smaller signature. Next, the signature is transformed in order to make it more difficult to detect and tamper with. Once the signature has been prepared, it is embedded into the input files of the place and route tools, through a combination of netlist modifications and physical constraints. Finally, the modified circuit again is passed through the vendor place and route tools. If the resulting physical layout achieves the system performance goals, then the watermarking process is complete.

1. Read in netlist and desired signature
2. Use vendor tools to place and route unmodified netlist
3. If (not enough spare resources for signature) then exit and retry with smaller signature
4. Process signature:
5. Pack 8-bit ASCII into continuous 7-bit characters
6. Encrypt signature to match "channel", i.e. typical design, spectrum
7. Add error correction coding
8. Interleave ECC-encrypted blocks to combat localized tampering
9. Embed properly-sized clique
10. Modify netlist and physical constraints to embed prepared signature
11. Execute vendor place and route tools on modified netlist
12. *If (performance is too low) retry with smaller signature else terminate with success*

Figure 2. Global flow of watermarking system

4.1 Signature Embedding

The first step in signature preparation involves transforming the signature so that it will appear to have the same statistics as an actual design. This process can be thought of as an application of encryption, which generally whitens a signal to match a channel with Gaussian white noise. However, in this case, the purpose of whitening the signal is not fundamentally to mask its content but rather its existence.

The next step in signature preparation involves adding error-correction coding (ECC). By doing so, we combat the malicious third party that manages to identify a part of a signature and attempts to modify or remove it. If the modification is small enough and localized, the ECC codes will be useful for retrieving the original signature and providing proof of design tampering.

The final step in signature preparation involves interleaving multiple ECC blocks. It is possible that a malicious third party would be able to identify a particular LUT that is non-essential to

the device function, and change its programming. If sixteen consecutive ECC blocks are interleaved, one bit at a time, over a set of LUTs, then each LUT will only contain one bit from any ECC block. This interleaving guarantees that the validation software can successfully retrieve the signature in the face of any single point fault, i.e. a LUT that has been tampered with.

Embedding the processed signature involves using free LUTs in an unmarked design. Each LUT in the XC4000 family encodes 16 bits of information, and from our experience most designs have a large number of unused LUTs. The signature is coded into LUTs defined by the designer's signature and a secure hash function, and the design is placed and routed around the signature. Since the actual signature is known only to the designers, they are also the only ones who know the location of the unused LUTs. Therefore, if the unused LUT location is disclosed for one design, designs with other signatures are still secure.

4.2 Validation

When the owners of an IP block believes their property has been misappropriated, they must deliver the configuration in question to an unbiased validation team. The IP vendor produces a seed that they claim was used to produce the block. With the seed and signature, the validation team reverses the signature preparation and embedding process: identify the CLBs used for hiding the signature using the functions defined by the secure hash function, reverse the block interleaving, apply the ECC if necessary, decrypt the message using a known key, and finally print out the resulting signature. If the signature matches that claimed by the IP vendor, then ownership has been established.

5 Experimental Results

We have evaluated the proposed approach by watermarking three large designs on FPGAs with various signature sizes, from an extremely small mark to the maximum size given unused LUT availability.

The overhead of the proposed approach comes in the form of area (physical resources) and timing. Area overhead is inevitable, as previously unused LUTs are used to encode the signature. However, in reality, area overhead does not increase linearly with the size of the signature. Rather, the calculation of area overhead involves the realization that place and route tools rarely pack utilized CLBs into a minimal area. Therefore, area overhead should be viewed in terms of the area used by the watermarked design minus the total area of the original design, including unused CLBs and LUTs.

Timing overhead may arise due to the constraints on placement as defined by the size and location of the signature. A LUT dedicated to the signature may impede placement of circuit components and lengthen the critical path. As the signature size grows, more constraints are made on the placement of the design, thus increasing the possibility for performance degradation.

The three designs used to evaluate the approach are a MIPS R2000 processor core designed for FPGAs, a reconfigurable Automatic Target Recognition (ATR) system [11], and a digital encryption standard (DES) design [7]. The MIPS core and the DES design were both implemented on the Xilinx XC4028EX-3-PG299, and the ATR system was implemented on the XC4062XL-3-PG475. For each design, the smallest possible device was used.

5.1 Results

Experimental results reveal that both area and timing overhead are low. After each design was placed and routed with no signature constraints, the number of unused LUTs was calculated and the circuit timing was noted. The original physical layout statistics are shown in Table 1. In each case, the designs were laid out such that the entire FPGA area was being used, with LUTs and entire CLBs being sporadically unused, illustrating that the place and route tools do not pack logic with optimum density. Therefore, there is essentially no area overhead required by the proposed approach. The approach utilizes free space in the original design and increases the density of occupied CLBs and LUTs. For tools that attempt to pack logic with increased density, area overhead may become apparent depending on signature size.

For each design, incrementally larger signatures were placed in the FPGA, and the design was placed and routed around the restricted resources. For each instance, the circuit timing was noted and compared to the original design. This process was repeated until the largest possible signature, i.e. one making use of all unused LUTs, was implemented. The results are shown in Tables 2-4.

For each table, the top two rows show the size of the watermark, first in bits and then in number of encoded ASCII characters. The next row for each design shows the percent resource increase in terms of the number of used CLBs. As mentioned above, the area increase for each instance is nearly 0%, but the table reflects the additional percentage of CLBs actually utilized in the watermarked design. Finally, the timing degradation for each instance is shown. Positive percentages indicate a decrease in performance. The table reveals that timing degradation is small and even negative in many instances. Relatively small changes in a circuit netlist or routing constraints can often result in a dramatically different placement and a corresponding change in speed. It appears that the impact of watermarking on performance is well below this characteristic variance, and thus the performance impact is non-monotonic with signature size.

Figures 3a and 3b are examples of DES layouts. Figure 3a is the original layout of the design with no watermark constraints. Note that the original placement does not achieve optimal logic density. Instead, unused CLBs are dispersed throughout the design. Figure 3b shows the layout with an embedded signature of 4768 bits.

6 Conclusion

As the market for reusable digital designs grows, issues concerning protection of proprietary designs come to the forefront. This paper has described a technique that takes advantage of FPGA flexibility to encode a watermark that is extremely difficult to detect and/or remove. The watermark uniquely identifies the design's origin, thus protecting designers against misuse or unauthorized distribution. Although the watermark is applied to the physical layout of the design by imposing constraints on the backend CAD tools, the area and timing overhead is extremely low. Experiments have shown that, even on very complex designs, a watermark can be applied and validated at this fine-grained level with little to no impact on design performance and area.

design	# used CLBs	# spare CLBs	min period (ns)
MIPS R2000	756	268	185.007
ATR	1876	214	424.542
DES	875	149	166.293

Table 1. Original physical layout statistics

mark size (bits)	800	1568	2592	3200	3872	4608	5408	6272	7200	8192
# ASCII chars	114.29	224.00	370.29	457.14	553.14	658.29	772.57	896.0	1028.6	1170.3
% resources	3.31	6.48	10.71	13.23	16.01	19.05	22.35	25.93	29.76	33.86
% timing	-1.04	-0.47	3.17	-7.15	-4.69	1.65	-11.53	2.47	11.95	-5.23

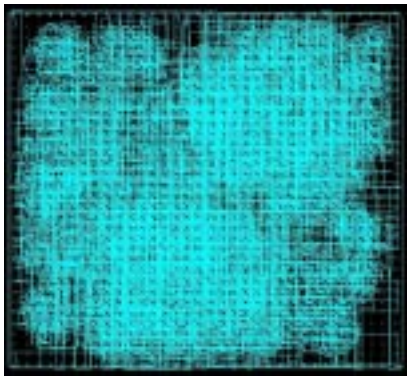
Table 2. MIPS R2000 – Impact of watermark size on resources and speed

mark size (bits)	32	800	1568	2944	4608	5984	6848
# ASCII chars	4.57	114.29	224.00	420.57	658.29	854.86	978.29
% resources	0.05	1.33	2.61	4.90	7.68	9.97	11.41
% timing	-10.74	3.46	-25.93	-7.99	-13.50	10.25	-1.57

Table 3. ATR - Impact of watermark size on resources and speed

mark size (bits)	32	800	1568	2528	3200	3872	4768
# ASCII chars	4.57	114.29	224.00	361.14	457.14	553.14	681.14
% resources	0.11	2.86	5.60	9.03	11.43	13.83	17.03
% timing	-22.98	-14.83	-5.07	-1.90	11.05	-11.93	-3.28

Table 4. DES - Impact of watermark size on resources and speed



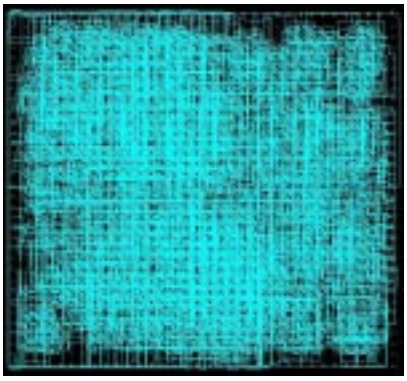


Figure 3a (left). DES original layout
Figure 3b (right). DES with 4768 bit watermark

Acknowledgements

The authors would like to thank Prof. Brad Hutchings and Peter Bellows for their assistance. This work was supported by the Defense Advanced Research Projects Agency of the United States of America, under contract F30602-96-C-0350 and subcontract QS5200 from Sanders, a Lockheed Martin company.

References

1. Anderson, R., and Kuhn, M. Tamper resistance - A cautionary note. *Proceedings of the Second USENIX Workshop on Electronic Commerce*. (1996), 1-11.
2. Charbon, E. Hierarchical watermarking in IC design. *Proceedings of the Custom Integrated Circuits Conference '98*. (1998).
3. Cox, I.J. et al. Secure spread spectrum watermarking for images, audio and video. *Proceedings of the Third International Conference on Image Processing*. (1996), 243-246.
4. Diffie, W. and Hellman, M. New directions on cryptography. *IEEE Transactions on Information Theory*. IT-22, 6 (Nov. 1976), 644-654.
5. Hong, I., and Potkonjak, M. Behavioral synthesis techniques for intellectual property protection. unpublished manuscript. (1997).
6. Kahng, A.B. et al. Watermarking techniques for intellectual property protection. *Proceedings of the Design Automation Conference '98*. (1998).
7. Leonard, J. and Mangione-Smith, W.H. A case study of partially evaluated hardware circuits: Key-Specific DES. *Field Programmable Logic*. London, England (1997).
8. Schneier, B. *Applied Cryptography: Protocols, Algorithms, and Source Code in C*. New York: John Wiley & Sons (1996).
9. Swanson, M.D. et al. Transparent robust image watermarking. *International Conference on Image Processing*. (1996), 211-214.
10. Trimberger, S. Personal communication. Xilinx Corporation. (1997).
11. Villasenor, J. et al. Configurable computing solutions for automatic target recognition. *Proceedings of IEEE Workshop on FPGAs for Custom Computing Machines*. Ed. Arnold, J. and Pocek, K.L. Napa, CA (1996), 70-79.
12. Wolfgang, R.B. and Delp, E.J. A watermark for digital images. *Applications of Toral Automorphisms*. 3 (1996), 219-222.
13. Xilinx. *The Programmable Logic Data Book*. San Jose, CA (1996).