A Survey on Protection of FPGA Based IP Designs

M .Meenakumari 1 & G. Athisha2

1Department of ECE, SNS College of Engineering, Coimbatore
2Department of ECE, PSNA College of Engineering & Technology, Dindigul
E-mail : mnakumari @ gmail.com, gathisha @ yahoo.com

Abstract – The size and complexity involved in designing of electronic devices and systems is continuously outpacing the designer productivity. So designers have to regularly strive for new solutions in terms of design tools & methodology. IP (Intellectual property) reuse methodology has been introduced to cope up with very large & complex designs. However, the IP core is vulnerable to many dangers such as copyright fraud, readback attack, cloning, reverse engineering etc. This paper provides a comprehensive review of current state-of-art of IP protection of FPGA based IP core.

Keywords – Bitstream, Intellectual property, Watermarking

I. INTRODUCTION

An SoC (System-On-Chip) usually contains reusable IPs, memory elements, clock circuits, test architecture and so forth. According to [1] a System-On-Chip (SoC) is defined as a complex IC that integrates the major functional elements of a complete end-product into a single chip. The use of pre-designed hardware modules, also called Intellectual Property (IP) cores brings a new revolution in the EDA design industry. Creators and owners of IP designs want assurance that their content will not be illegally redistributed by consumers. Today, thousands of designers are creating and exchanging IPs on an increasingly large scale. Watermarking IP cores is different from multimedia watermarking, because the user data, which represents the circuit, must not be altered since functional correctness must be preserved.

The VSI Alliance IP protection development working group [4] identifies three main approaches to secure IPs. First, a deterrent approach where the owner uses legal means trying to stop attempts for illegal distribution i.e using patents, copyrights and trade secrets. This method does not provide any physical protection to the IP. Second, a protective approach where the owner tries to prevent the unauthorized use of the IP physical by license agreements and encryption. Third, a detection approach where the owner detects and traces both legal and illegal usages of the designs as in watermarking and fingerprinting. This tracking should be clear enough to be considered as evidence in front of a court if needed. The VSI alliance proposed the usage of the three approaches for proper protection of IP designs. It is shown in Fig.1.

![Fig. 1 : FPGA based IP Core Protection](image)

The SOC design flow has three main IP blocks which are explained as follows: (i) Soft IP: They are delivered in the form of Hardware Description Language. They are more flexible and have increased Intellectual property risks because the RTL source code is required by the Integrator. (ii) Firm IP: They are delivered in the form of the full or partial netlist. They do not include routing. They are more optimized in structure and topology for area and performance. Risks are same as that of Soft IP. (iii) Hard IP: Hard IPs, delivered as GDSII files are optimized for power, size, or performance. From a security point of view, hard IP is the safest because they are hard to be reverse engineered or modified. Fig.2 describes the types of IP cores.
An IP core developer embeds a signature inside his core using a watermark embedder and sells the protected IP core. A third-party company may obtain an unlicensed copy of the protected IP core and use it in one of their products. If the IP core developer becomes suspicious that his core might have been used in a certain product without proper licensing, he can simply acquire the product and check for the presence of his signature. If this attempt is successful and his signature presents a strong enough proof of authorship, the original core developer may decide to accuse the product manufacturer of IP fraud and press legal charges.

**II. ATTACKS AGAINST IP WATERMARKING**

The goal of IP protection as a whole is to provide the necessary technologies to make enforcement a more manageable task while deterring infringement at all levels of a design. A potentially effective scheme requires the capability of effectively detecting and subsequently tracking IP infringement cases. This task can be accomplished by a process known as watermarking. The method consists of embedding a unique code, or watermark, which exploits the IP’s unique features. Digital watermarking attacks are categorized into four main classes [5]: unauthorized removal, unauthorized embedding, unauthorized detection, and system attacks. The same categorization applies to IP watermarking schemes; yet unauthorized detection is not considered a high risk for IPs.

**A. Removal attack**

Removal attacks aim at the removal of the watermark information [5]. This is tried without breaking the security of the watermark, i.e., without searching for the key used in the embedding. Removal attacks are divided into either elimination attacks or masking attacks. The intruder tries to eliminate the watermark completely in the elimination attacks. As an example, the intruder tries to estimate the watermark and subtract it from the watermarked design. On the other hand, masking attacks do not aim at removing the watermark itself, but rather aim at distorting the watermark detector such that it will not be able to sense the availability of the watermark. This attack is considered one of the main measures for robustness of a watermark.

**B. Embedding attack**

Embedding attacks (forging) aim of embedding another watermark in the design, this can be done either by ghost searching, where the intruder tries to find a ghost watermark and consider it as his watermark, or by re-embedding the watermark if he/she has the tools necessary to do this. System attacks aim of attacking the concept of watermarking itself, as an example, attacking the cryptographic base of the watermarking, or removing the chip that check the watermark physically in case of video for instance.

**C. Readback Attack**

Readback is a feature that is provided for most FPGA families. This feature allows to read a configuration out of the FPGA for easy debugging. An overview of the attack is given in [31]. The idea of the attack is to read the configuration of the FPGA through the JTAG or programming interface in order to obtain secret information (e.g., Keys, proprietary algorithm). The readback functionality can be prevented with a...
security bit. In some FPGA families, more than one bit is used to disable different features, e.g., the JTAG boundary.

D. Watermark Evaluation Criteria:

To evaluate watermark some criteria are defined in [29].

- Functional correctness: This is the most important criteria. If the watermark process destroys the functional correctness, it is useless to distribute the core.
- Resources overhead: Many watermark algorithms need some extra resources. Some for the watermark itself, some because of the degradation of the optimization results from the design tools.
- Transparency: The watermark procedure should be transparent to the design tools. It should be easy to integrate the watermarking step into the design flow, without altering the common design tools.
- Verifiability: The watermark should be embedded in such a way that simplifies the verification of the authorship. It should be possible to read out the watermark only with the given product without any further information from the design flow, which must be ordered from the accused company.
- Difficult to remove: The watermark should be resistant against removal attack. The effort to remove the watermark should be greater than an effort needed to develop a new core or removal of watermark should cause corruptness of the functionality of the core. Watermarks which are embedded into the function of the core are more robust against removal than additive watermarks.
- Strong proof of authorship: The watermark should identify the author with a strong proof. It should be impossible that other persons can claim the ownership of the core. The watermark procedure must be resistant against tampering.

III. IP CORE PROTECTION TECHNIQUES

Additive and constraint based methods are the different techniques available to protect IP core designs. Additive methods are watermarking procedures, where a signature is added to the functional core. That means that the watermark is not embedded into the function of the core. Constraint based watermarking presents a signature as a set of additional constraints which are applied on the hardware optimization problem.

A. IP Core Protection At Layout Level

Many watermarking techniques are available for the protection of IP cores at various design levels. John Lach et al. [9] Introduced fingerprinting techniques for protecting FPGA Intellectual property. It uses an FPGA design tiling and partitioning technique that greatly reduces the cost of generating many difficult functionally equivalent circuit instances. Gang Qu and Miodrag Potkanjak [10] investigated the effect of applying a watermark to the physical layout of a digital circuit when it is mapped into an FPGA.

W. H. Mangione Smith et al. [11] proposed FPGA watermarking technique by using post processing constraints. The essence of their approach is to encode the signature bits and embed them into the unused look-up tables (LUTs) such as that they do not affect the original designs, and then reroute the design around these LUTs. The disadvantage of this approach is that the watermark is not embedded as a functional part of the design. The watermark can be removed without affecting design functionality. Watermarked lookup tables do not reflect any functionality, thus they are prone to eliminate if optimization algorithms are used.

K. W. Yip et al. [12] proposed a partial encryption scheme in which the configuration bit stream is partially encrypted and then loaded onto the separate RAM built into the FPGA. This requires a decryption unit on the FPGA to read and decrypt the encrypted bitstream from the special purpose RAM and then load the decrypted bitstream into the main configuration RAM. The security of the techniques relies on the fact that bitstream file is hard to reverse engineering. The main disadvantage of this method is it requires additional hardware.

R.S. Chakraborty et.al [20] discussed a method for hardware IP protection using netlist level obfuscation. This methodology can be integrated in the System-on-Chip design and manufacturing flow to simultaneously obfuscate and authenticate the design.

Lin Yuan et al. [13] determined the delay on each net in the design and modified the delay by adding required timing constraints on that net in a user constraint file in ISE. This file is integrated with the design during implementation and eventually it affects place and route result. Delays along critical paths are more controlled in order to meet the system performance requirements.

Andrew B. Khan et al. [14] introduced IPP protocols for embedding designs watermarks at the physical design level is at placement and routing levels. Later on Andrew B. Khan et al. [15] introduced a new preprocessing approach that embeds watermarks as
constraints into the input of design tool and also a new post processing approach that embeds watermarks as constraints into an output of the design tool. The essence of their approach is to encode the signature bits and embed them into the unused look-up tables (LUTs) such that they do not affect the original design and then reroute the design around these LUTs. The disadvantage of this approach is that the watermark is not embedded as a functional part of the design; given enough information, the watermark can be removed without affecting the design functionality.

B. IP Core Protection at Netlist level

Dasko Kirovski et al. [16] developed two protocols for embedding tool specific information into a logic network while performing multilevel logic minimization and technology mapping. A copyright information is hashed using a cryptographically secure hash function to create a key used to seed a pseudo random number generator. The pseudo random number generator is used to generate a unique set of design constraints. By superimposing these constraints to original network, new input is generated. It proves authorship of the design at levels of abstraction equal to lower than logic synthesis.

Aijiao Cui and C.H. Chang [25] suggested a re-synthesis method for embedding the IP designer information into a distributed copy of master design. Moiz Khan et al. [22] embedded authorship information in the combinational circuit by rewiring circuit with one or more redundant addition/removal steps.

C. IP Core Protection at Behavioral Level

Arvindo Olivera [17] introduced a methodology for the watermarking of synchronous sequential circuits that makes it possible to identify the authorship of the designs by imposing a digital watermark on the state transition graph of the circuit. To watermark a design, the user should define an arbitrary long string that clearly describes his/her ownership rights. This data is considered the watermark information. After encrypting this message using a public key, the user should use a one-way hash function, such as MD5 to obtain a compact signature of this arbitrarily long sentence. The arbitrary sequence is then broken to input sequence combinations. For example, if the design has 16 inputs, and the sequence is 128 bits, it defines a unique sequence of 8 input combinations. The user then changes the STG in such a way that the sequence of states reached by this sequence of inputs exhibits a specific property, which is rare in non-modified STGs. This property is purely topological and does not depend on the specific encoding. If, later on, the watermark need to be uncovered, the designer shows this input sequence and the property he/she defined. In order to define the input sequence to change the STG properties, extra states are added in a systematic way to satisfy this property. The algorithm has allow overhead in the design flow, because it does not need to go through the FSM to find the unused transitions.

Torunoglu and Charbon [26] introduced the first approach on FSM watermarking. The algorithm is mainly based on extracting the unused transitions in a state transition graph (STG) of the behavioral model. These unused transitions are inserted in the STG associated with a new defined input/output sequence, which will act as the watermark. The approach starts with building the FSM representation of the function, then visiting every state and finding the unused state transitions (input/output pairs). In case the FSM is completely specified (CSFSM), new input/output pairs are added to expand the FSM. The minimum number of transitions needed is then calculated, and compared to the maximum number of free transitions to satisfy the probability that a non-watermarked design would carry this watermark by coincidence. If this probability cannot be satisfied, input/output pairs should be added to satisfy the watermark requirements.

A.T. Abdel Hamid [18] proposed the first public-key IP watermarking scheme at the FSM level. Aijiao Cui et al. [19] introduced a new approach for watermarking IP designs based on the embedding of the ownership proof as part of the IP design’s FSM without increasing the number of states in STG. This approach utilizes coinciding as well as, un-used transitions in the state transition graph of the design.

Encarnacion Castillo et al. [21] proposed a method to spread digital signature bits within memory structures or combinational logic that are part of the system at a high level description of the design.

Wei Liang et al. [23] discussed a technique to extract maximal delay set through state transformation and to add a watermark sequence to the maximal delay state set. Debapriya Basu Roy et al. [24] proposed an approach based on embedding ownership information as part of the IP design’s Finite State Machine. But this method added number of states in STG.

IV. FPGA DESIGN PROTECTION

SRAM FPGA based systems normally store the configuration file in a non-volatile memory outside the FPGA. In such a situation, an eavesdropper can easily retrieve the configuration file flowing through the port, and possibly clone the same design in other FPGAs.

In order to protect designs from reverse engineering, cloning, and overbuilding there is the new "Device DNA" feature available only in members of the
new Spartan-3A family from Xilinx. The Spartan-3A is the first low cost FPGA that addresses and increases the level of security at a design level. Xilinx offers Device DNA Security [32] in the Spartan-3A/3AN/3A DSP FPGA platforms to protect IP. The Device DNA is a 57-bit ID, unique to every Spartan-3A/3AN/3A DSP FPGA. This ID can be used to tie a design to a specific FPGA. The Security Algorithm reads the Device DNA and generates an Active Value. It then compares the value with the Check Value, stored during the initial setup. If the Check Value is equal to the Active Value, the normal operation can occur.

In Spartan 6, AES algorithm is used to encrypt the bitstream of the user design before it goes into production. Certain Spartan-6 devices, support storage of a 256-bit key in battery-backed RAM for secure operation.

The Virtex-5 family supports 256-bit AES encryption/decryption technology to achieve a very high degree of design security. The AES key is stored in dedicated memory, powered by either an auxiliary power supply or an externally connected battery. If any tampering is detected the key and the bitstream in the memory will be zeroized. Virtex-6 FPGAs are the first and only programmable devices to offer cryptographically strong bitstream authentication preventing attacks. An on-chip bitstream keyed-HMAC algorithm implemented in hardware provides additional security beyond that of using AES bitstream encryption alone. Without knowledge of the AES and HMAC keys, the bitstream cannot be loaded, modified, or cloned.

To provide design security, Stratix II and Stratix II GX devices [33] uses 128-bit advanced encryption standard and a non-volatile key for configuration bitstream encryption. In this series the input to the FPGA will be encrypted first and stored in a memory outside the FPGA. During configuration the input file will be decrypted by using a secret which stored in the FPGA.

A. Protection level of FPGA

The level of protection offered by actual integrated circuits is an interesting metric to identify works that must be carried out to improve the security level of one particular type of integrated circuit. Paper [11] defines the various security levels for modern electronic systems and the corresponding taxonomy of attackers. There are two nonvolatile FPGA technologies that are even more secure than competing ASIC technologies. They are antifuse-based FPGAs and Flash-based FPGAs. The security level of the classical integrated circuits is given in table I.

<table>
<thead>
<tr>
<th>FPGA Technologies</th>
<th>Security Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRAM FPGA</td>
<td>0</td>
</tr>
<tr>
<td>SRAM FPGA With Bitstream Encryption</td>
<td>3</td>
</tr>
<tr>
<td>Flash FPGA</td>
<td>4</td>
</tr>
<tr>
<td>Antifuse FPGA</td>
<td>4</td>
</tr>
</tbody>
</table>

B. Watermark Verification

As mentioned in [29], when considering a finished FPGA products, there are five potential information sources can be used for extracting a watermark: configuration bit file, ports, power consumption, electromagnetic (EM) radiation, and temperature. The bit file can be extracted by wire tapping the communication between the PROM and the FPGA. Some FPGA manufactures provide an option to encrypt the bitstream which makes communication monitoring useless. However, it is possible to read out some information stored in RAMs or lookup tables to finish verification.

Another approach is to employ unused ports which is limited only at top-level designs and impractical for IP cores. The method called “Power Watermarking” can force patterns on the power consumption of an FPGA as a covert channel to transmit data to the outside. Related works shown in [28, 29] indicate the clock frequency and toggling logic can be used to control such a power spectrum covert channel. The resulting change in power consumption can be extracted as the signature from the FPGA’s power spectrum.

With almost the same strategy it is also possible to extract signatures by raster scanning electromagnetic (EM) radiation of an FPGA with an EM sensor. Unfortunately, it becomes impractical since modern FPGAs are delivered in a packaged shape which decreases the EM radiation. Finally, a watermark might be read out by monitoring the temperature radiation which is similar to power and EM-field watermarking approaches.

V. CONCLUSION

Protecting copyrights of intellectual property providers and integrators has become a serious problem. It arises from the fact that electronic circuits are readily available in a form of virtual blocks at any abstraction levels, thus allowing for abuses and theft. Several
methods have been described to generate watermarks at various levels of hierarchy during electronic design flow. The protection of IP implemented in an FPGA should be proactively planned and implemented. Designers and engineers must determine the best FPGA to use and best methods of securing IP to protect the success of their products.

VI. REFERENCES


[27] www.opencores.com


[32] Spartan 3A DSP family data sheet

[33] AN 341: Using the design security feature in stratix II and stratix II GX devices.