A Survey of hardware protection of design data for integrated circuits and intellectual properties

Brice Colombier, Lilian Bossuet

A survey of hardware protection of design data for integrated circuits and intellectual properties

Brice COLOMBIER, Lilian BOSSUET

Hubert Curien Laboratory, UMR CNRS 5516, University of Lyon, Saint-Etienne, France

{brice.colombier,lilian.bossuet}@univ-st-etienne.fr

Abstract

This paper reviews the current situation regarding design protection in the microelectronics industry. Over the past ten years, the designers of integrated circuits and intellectual properties have faced increasing threats including counterfeiting, reverse-engineering and theft. This is now a critical issue for the microelectronics industry, mainly for fabless designers and intellectual properties designers. Coupled with increasing pressure to decrease the cost and increase the performance of integrated circuits, the design of a secure, efficient, lightweight protection scheme for design data is a serious challenge for the hardware security community. However, several published works propose different ways to protect design data including functional locking, hardware obfuscation, and IC/IP identification. This paper presents a survey of academic research on the protection of design data. It concludes with the need to design an efficient protection scheme based on several properties.

1. Introduction

The economic model on which most microelectronics companies are based has changed over the last decade [1]. This is because the cost of a manufacturing facility has increased exponentially to reach billions of dollars. As a consequence, many designers cannot afford to produce their own circuits, and as a result, the production system has split into two different entities. The same entity is no longer responsible for both the design and manufacture of the circuit. The firms who are only responsible for the design part are called *designers* and are referred to as *fabless* since they cannot produce the circuit themselves. In parallel, there are companies that own the means of production, and are called *foundries*.

Due to the distinction between the location of the design and production, fabless companies depend on foundries for the production facilities they need to build their circuits. Here we distinguish two design categories. On the one hand, companies who produce specifications for integrated circuit (IC) layouts that can be directly

used in the photo-lithography process. They can also provide RTL schematics of the design. On the other hand, firms specialize in selling VHDL/Verilog codes, part of a netlist or layout, and FPGA bitstreams. These pieces of design are then used as building blocks, called intellectual properties (IP), to be embedded in a more complex system, or used on their own.

As shown in Fig.1, in both cases, the design has to be transferred from the party who owns the rights to the manufacturer/integrator. In the case of ICs (top of Fig. 1), the fabless designer has to send the design layout to the foundry. In the case of IPs (bottom of Fig. 1), the IP designer has to send the VHDL/Verilog source, netlist or FPGA bitstream to the system integrator.

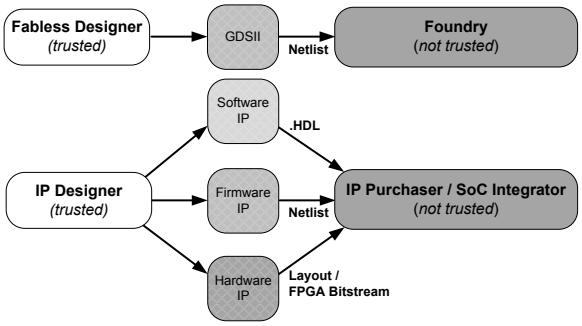

Figure 1 - The asymmetrical transfer of design data.

Clearly, there is a weight asymmetry between the two parties. Indeed, to get the system properly built, the rights owner has to give the entire design to the foundry or the system integrator, and thus is no longer the sole owner of the specifications. As a result, the original terms of use may not be respected, and the design may be misused by the foundry or the system integrator. In this way, the number of counterfeit electronic circuits collected by U.S. Customs between 2001 and 2011 increased 700-fold [2]. U.S. Customs collected 5.6 million counterfeit electronic products between 2007 and 2010 [3]. Overall, counterfeiting is estimated at about 7% of the semiconductor market [4] representing a loss of about $22 billion for the lawful industry in 2014.

This paper presents a survey of hardware protection schemes of the design data for ICs and IPs. We focus mainly on hardware protection, hardware identification techniques and specific protection schemes for IP licensing. The paper is organized as follows: In section 2, we discuss the threats to semiconductor devices we identified, from which we built a threat model. In section 3, we review published protection schemes. In sections 4 to 6, we provide detailed information on the different categories of protection schemes: hardware protection, hardware identification and specific protection schemes for IPs. In section 7, we compare the protection schemes using different metrics, and then outline the characteristics of a good solution. Finally, in section 8, we point out the gaps in existing schemes, and other possible ways to envisage protection.

2. Threat model

2.1 Parties involved

Before presenting the schemes used for the protection of design data by IC and IP designers, it is important to identify all the parties who could be involved in the process of exchanging designs. These parties are often mentioned in the literature, so it is important to understand what they represent. We list ten parties:

The **Design owner** is the person who owns the rights to the design. They can also be called the rights owner. The designated design can be described in different ways, from layout to HDL code. The design owner is the person who sells the design to the system integrator, and allows the system integrator to use it under specified conditions.

The **IP provider** is the person who sells IPs as building blocks intended to be used later by the system integrator.

The **Manufacturer** is also called the founder. He/she is the builder of the chip and the integrated circuit. To manufacture the device, the manufacturer requires the complete design specifications, as explained in Fig. 1.

The **System integrator** is the person who designs the final product and sells it to the customer. Their role is to link the design owner, the manufacturer and the market.

The **Tester** is the person whose responsibility is to check the circuit was correctly built. He/she guarantees the circuit complies with the original specifications.

A **Competitor,** in an economic context, is an adversary of the design owner. Their aim is to acquire market shares. Some competitors may use illegal methods to recover the design data from the design owner in order to produce a similar integrated circuit or IP without paying for the original design. In addition, using such illegal methods, the time-to-market could be shorter than for the design owner.

The **User** or **customer** is the one who originally provided the specifications. He/she is the person who will use the device once it is built.

A **Broker**, also called stockist, buys a large quantity of integrated circuits and sells them in smaller quantities to system integrators and/or direct users. The broker is a critical entity in the microelectronics supply chain; most of the counterfeiting cases in the last decade involved a broker [2, 4].

A **Trustworthy party** is a party who has no commercial interest in the transaction, but is only present to ensure the design and product exchanges are conducted correctly. The Trustworthy party is supposed to prevent theft or leaks during the transaction. He/she thus ensures each party that the other party is trustworthy, and to a certain extent, thus plays the role of a middleman.

A **Waste collector** is in charge of recycling electronic items, but in practice only around 17% of electronic waste is officially recycled [5]. This has led to the emergence of unofficial treatment facilities in Asia and Africa [6]. As a consequence, waste collectors play a significant role in counterfeiting schemes.

2.2 The main threats to design data

A threat model allows designers to identify possible threats to their design and to estimate the effort required to place the design under secure protection. The entry points for the attacker are the different descriptions produced for each design. From silicon to the entire ASIC, or from the logic gates to the IP, a system can be described in many different ways. All these ways can thus also be considered as potential entry points for an attacker to find out exactly what a design involves. We now detail the four main types of threats: reverse engineering, counterfeiting, theft (hardware theft and cloning) and hardware Trojans.

Reverse engineering is a major concern for both IP rights owners and IC designers. Reverse engineering is very commonly used to investigate how a device works [7]. Some companies are specialized in providing this kind of service. Reverse engineering basically consists in subjecting the device to all kinds of tests. This makes it possible to extract the components from the device's responses to the tests. We use the term 'test' as a general descriptor for all the methods that can be used in the investigation process. Reverse engineering can be either *invasive*, i.e. the device can no longer be used after it has been tested, or *non-invasive* in which case the device's integrity is maintained after the tests. Invasive reverse engineering can lead to two types of results. The attacker can recover the RTL schematic of the IC, or find weaknesses in the design in order to attack it. The first goal can be achieved using pattern recognition. The circuit is first taken apart, and then micro-photographed. The pictures are then processed by the pattern recognition algorithm. The final result is the RTL schematic of the IC. Before reaching this stage, many different processes can be used. For example, it is possible to extract the via from a silicon implementation [8]. This technique is also widely used to analyse weaknesses in the cipher design of an RFID tag. It allows the reverse-engineers to find weak random numbers in the device [9], and break the secret key. Non-invasive methods of reverse engineering are also described. Their main objective is to discover the behaviour of a design (i.e. the finite state machine diagram or the logical truth tables). The first naive method is to examine outputs after applying different word combinations to the inputs of the circuit [10]. This requires prior investigation of the type of device. It includes distinguishing between a Moore or Mealy finite state machine, and identifying an initial state. After that, states are simply determined by deducing correlations between input and output words. A generic investigation environment has also been described, which provides standard means to describe all kinds of combinational or sequential finite state machines [11].

Counterfeiting is a major problem for the microelectronics industry. Basically three types of counterfeiting exist [12]. The first is *relabeling*. Here the aim is to make the device look like it actually comes from another supplier (Fig.2-b), or is of a higher grade (Fig.2-c). This is simply done by erasing serial numbers or identifiers, which are then replaced by others. Such a method can be used, for instance, to pretend a chip fulfils military specifications when in fact it does not, or to advertise a very high computing frequency when the real one is lower [13, 14]. Counterfeiting can also take the form of *refurbishing* (Fig.2-d). Refurbishment is the act of reintroducing discarded products in the supply chain. It can also consist in reselling old parts as new ones. Refurbished parts are aesthetically improved to make them look new, and sold. The third type of counterfeiting we identified is *repackaging* (Fig.2-e). Repackaging consists in taking the original IC out of its package, and putting it into a different one. The aim may be to modify the number of pins, or to use smaller packaging. Obviously, the process of repackaging can alter the circuit. Because repackaging is usually undertaken in poor handling conditions, the final circuit may be altered and not work properly. For example, the top passivation layer could be scratched, or pins could be broken [15]. All these practices obviously deceive the end user as they have purchased products which are different from what they expected. Several detection techniques are possible, which we describe in Section 3.

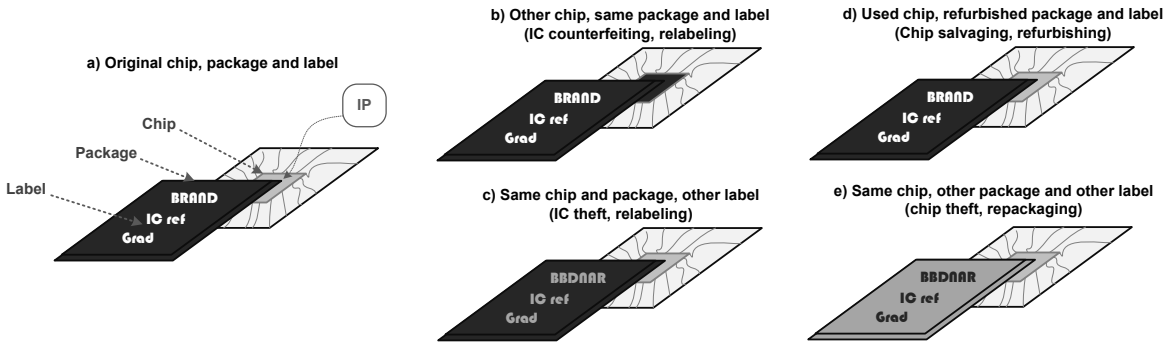

Figure 2 - Possible types of counterfeiting (b, c, d and e) from an original device (chip-package-label) (a)

Theft (hardware theft and cloning) has three types of consequences. First, the attacker can use *overbuilding*. That means the foundry or system integrator in charge of manufacturing the devices produces more of them than originally specified by the designer [16]. It can also result in *cloning* when the attacker makes an illegal copy of the design data. This happens when a copy of the design is obtained by an untrustworthy/unauthorised person. Finally, hardware parts may also be stolen, such as masks or ICs, which can then be used to build more devices than expected. Finally, the main consequence of theft is that extra circuits are built, unbeknownst to the rights owner. The attacker can then sell the design data or hardware to other parties,

and claim to be the owner.

A Hardware Trojan is a small piece of hardware, barely detectable, able to impair part of a device or allow information leakage without affecting system performance [17, 18]. Although hardware Trojans are a hot topic in the field of hardware security and applied cryptography, in this paper we only focus on the three first threats which are counterfeiting, reverse-engineering and theft. Indeed, hardware Trojans can be used to attack the design data, but are usually used to attack security critical data (i.e. the cipher secret key) or system security. For example, a hardware Trojan can steal a secret cypher key and send it outside the circuit, or trigger a denial of service [19]. But in this paper, we only consider threats which are mainly focused on design data.

Several of the parties listed in the previous sub-section are involved in the lifecycle of integrated circuits and IPs, from their manufacture to when they are discarded. Each could represent a threat to the security of the designer's intellectual properties. The rows in Table 1 represent the parties involved, while the columns represent the threats. A star at the intersection of a row and a column indicate that the threat represented by the column could come from the party represented by the row. The position of the stars in Table 1 is not the same throughout, but the manufacturer, system integrator, and broker appear to be the most threatening parties, and, in point of fact, these three parties do have easy access to design data and hardware (for the purpose of theft). As a consequence, protection schemes should be focused mainly on protecting the manufacturing process, the stock of ICs, and IP integration. In the following section (section 3), we present possible ways to protect design data from the abovementioned threats.

Table 1 – Threats to design data and the parties most likely responsible

	Reverse engineering		*Counterfeiting*			*Hardware theft*		*Cloning*	*Hardware Trojan Insertion*
	Invasive	*Non-invasive*	*Repackaging*	*Relabeling*	*Refurbishing*	*Overbuilding*	*Circuit theft*		
Designer									★
IP Provider									★
System integrator		★				★		★	★
Manufacturer		★	★	★		★	★	★	★
Tester							★		
Competitor	★	★							
User								★	
Broker			★	★	★				
Waste collector					★				

3. Protection schemes

3.1 Protecting design data

Many methods for the protection, detection and limitation of threats to design data have been already published. Before discussing the impact of each protection scheme on the threats described above, we first summarize the main ways of protecting design data based on hardware.

Hardware locking/unlocking is used to protect ICs and IPs from theft or cloning. This method consists in

making the circuit (or IP) unusable immediately after it has been built or sold. The unlocking step can be done before or after distribution/integration by the original designer, by the IP provider, or by a trustworthy party. Hardware locking/unlocking schemes can be integrated in the design control unit (FSM locking), memory unit, or even input-output blocks. The locking mechanism should be sufficiently secure to provide strong protection in case of hardware theft and cloning. Sometimes, locking schemes are called hardware encryption (when an unlocking key is required) or obfuscation. Obfuscation comes from the field of computer science in which developers wish to protect program codes against unauthorized reading. The following definition of code obfuscation is proposed by Hachez [20]: *Transform a program P into another program P' harder to reverse engineer with the same observable behaviour. If P fails to terminate or terminates with an error, then P' fails to terminate or terminates with an error. Otherwise, P' must terminate and produce the same output as P*. Hardware obfuscation consists in applying this definition in the hardware field, by changing the logic, FSM, or other part of a design.

Hardware camouflaging is used to protect design data against reverse engineering. Camouflaging is mainly used for military purposes. We propose the following definition for camouflaging: *set of means to physically hide details of a system from an optical inspection (which could use image processing techniques) without any modification of the system behaviour*. So, hardware camouflaging mainly consists in protecting the circuit layout against image processing based extraction of a gate-level netlist from an IC (reverse engineering).

Reconfigurable hardware is a flexible piece of hardware whose behaviour (functionality) can be changed (once only or in runtime) without modifying the hardware. FPGAs are a common example of reconfigurable hardware. Using reconfigurable hardware could protect the design data against non-invasive reverse engineering but the reconfiguration data (usually called bitstream) also needs to be protected. To do so, it is possible to use cryptographic services such as confidentiality and authentication.

Hardware identification can be implemented in different ways. The first is to embed a unique identifier (*ID*) inside the device, and to read it afterwards. It can be stored in non-volatile memory as a secret key (used for cipher/decipher), or be the output of a physical unclonable function (PUF). It can also be the result of an intended modification of the design, like in a watermarking scheme. Another way is to measure the physical properties of the device. This is called fingerprinting. Raw physical properties can simply be read or can be processed, for example using side channel (such as power consumption, electromagnetic emanation, thermal dissipation) analysis.

IP licensing (permanent or temporary) is specific protection because the IPs are delivered to the end user

in the form of a HDL code or a netlist (as shown in Fig.1). This type of protection scheme is inspired by software protection. For example, in microelectronics, it may be interesting for the designer to be able to offer IPs in demo mode. The performances of the IP would be lower, but the customer would be able to test it before purchasing. Moreover, the point here is also to sell IPs using a different business model. Instead of paying for a permanent license to use an IP and to be allowed to implement it in an unlimited number of devices, it may be attractive to only pay for the number of times the IP will be implemented.

3.2 Protections vs. threats

Table 2 lists the main hardware protections and their impacts on the threats described in Section 2 above (except for hardware Trojan). For each solution, Table 2 classifies the impact on each threat as *protection*, *limitation* or *detection*. *Protection* is the term used when the solution stops the threat; in this case the attacker has to perform an attack on the protection system in order to achieve a successful attack on the design data. *Limitation* is used when the solution reduces the threat, in this case the attacker has to invest more effort in making the design data attack succeed. *Detection* is used when the solution does not protect the design against attack but informs the user that an attack on design data has been successfully carried out. In some cases, it is also possible to trace the attack, and transmit it to the original designer.

Table 2 – Different protection schemes for different threats

	Reverse engineering		*Counterfeiting*			*Theft*		*Cloning*
	Invasive	*Non-invasive*	*Repackaging*	*Relabeling*	*Refurbishing*	*Overbuilding*	*Hardware theft*	
FSM locking / obfuscation						*Protection*	*Protection*	*Protection*
Memory locking						*Protection*	*Protection*	*Protection*
IOB locking						*Protection*	*Protection*	*Protection*
Logic encryption / obfuscation		*Limitation*				*Protection*	*Protection*	*Protection*
IC camouflaging	*Protection*	*Protection*						
Reconfigurable area		*Protection*				*Protection*	*Protection*	*Protection*
FPGA bitstream encryption		*Protection*				*Protection*	*Protection*	*Protection*
ID in non-volatile memory			*Detection*	*Detection*	*Detection*			*Detection*
PUF			*Detection*	*Detection*	*Detection*	*Detection*	*Detection*	*Detection*
Watermarking			*Detection*	*Detection*	*Detection*	*Detection*	*Detection*	*Detection*
Fingerprinting			*Detection*	*Detection*	*Detection*	*Detection*	*Detection*	*Detection*
IP licensing								*Protection*
Time-limited IP								*Protection*

In the following sections (4, 5 and 6), we give examples of protection schemes by explaining typical solutions, and provide the reader with references for further reading. We have used the following classification; hardware locking/unlocking (hardware encryption and obfuscation), hardware camouflaging, reconfigurable hardware and encryption bitstream of FPGA have been merged in the hardware protection section (Section 4) according to their impact on threats of reverse engineering, theft, and cloning. Identification techniques are presented in Section 5, and more details are given on IP dedicated schemes in Section 6.

4. Hardware protection

IPs and integrated circuits are becoming increasingly complex, and complete complex systems are now included on a single chip. System-on-chip offers the designer many ways to protect the design against the above-mentioned threats. To illustrate these possibilities, Fig. 3 presents a SoC architecture which contains processor cores (an ARM core and DSP), many hardware accelerators (for example an image co-processor, coder/decoder, cryptographic engine), a shared memory, a reconfigurable area with dedicated memories and IOB banks, shared communication based on buses, cross bar or network-on-chip, many controllers (interruption controller, memory controller, DMA, etc.) and peripherals such as CAD/DAC, UART, GPIO, etc. Other SoC architectures could have been shown and are available in the literature, but the one presented in Fig. 3 allowed us to include most of the published hardware protection schemes. The eight small circles in Fig. 3 identify the location of eight possible actions to protect design data based on hardware: (1) locking the control unit by modifying the FSM [21 - 23], (2) locking processing by logic encryption/obfuscation [16, 24, 25], (3) locking memory by address encryption, (4) locking processing using the reconfigurable area [26], (5) locking communication using encrypted barriers [27-28], (6) locking peripherals by changing parameters, and (7) locking input-output-blocks using an antifuse [30]. In addition, camouflaging techniques could be used at the layout level of the SoC (8) [31-33].

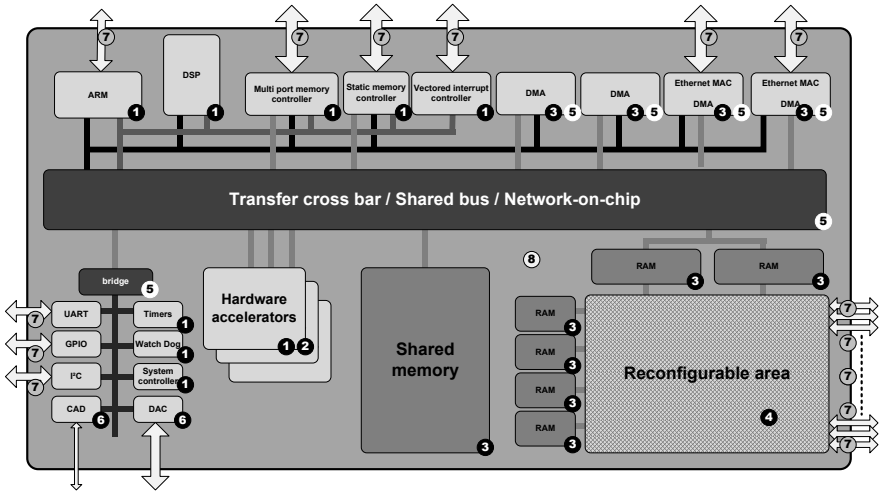

Figure 3 – A typical SoC architecture with eight possible locations for hardware protection.

4.1 Hardware locking/unlocking (hardware encryption and obfuscation)

To protect the digital synchronous circuit, the FSM can be modified, as shown in Fig. 4. The aim is to add extra blocking states to an FSM to prevent it from functioning if it remains in these states. The states concerned are usually start-up states.

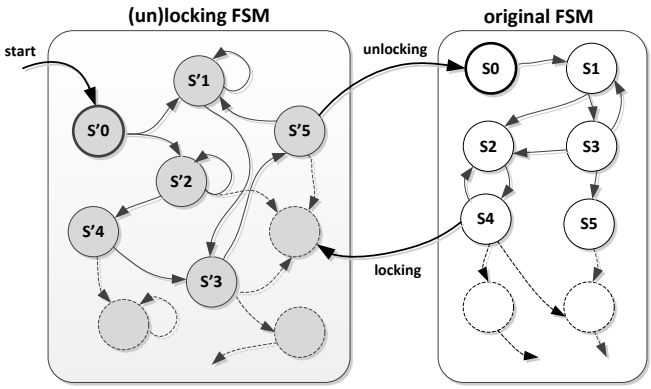

Figure 4 – Modification of a finite state machine

This was the first protection scheme based on a modification of the FSM, and was proposed in 2007 [21]. After a certain sequence is applied on the inputs, the FSM reaches its normal functional mode. The specific sequence applied to the inputs to unlock the circuit is called a key. The key is sometimes coupled with a random unique block [22] to strengthen the unlocking mechanism. This block has a single function, which is specific to each individual IC. It is composed of a bit selector circuit, which takes advantage of path delays, non-linearity, and the unique physical properties of each circuit.

Another way to use FSM properties as a protection is to exploit FSM synchronicity. This approach was recently described in [23] and has the advantage of not adding any extra states to the FSM. This property ensures that in any given state, after applying a predetermined sequence of inputs, the system will switch to a unique known state. In this way, it ensures the right key owner will be able to unlock the system, even if it is in an unknown state.

Different locking/unlocking schemes at the logical level have been proposed [25]. In [16] and [24], Roy et al. simply proposed adding XOR gates on non-critical paths. The key will then be applied to the inputs of these XOR gates, and will invert the signal (or not). Another way proposed in [29] is to block the buses instead, as they are particularly critical components of a numeric system. The lock is designed to use Benes networks, because they are cheap and relatively easy to use, and hard to recover. The key is transmitted using a Diffie-Hellman cyphering scheme. Key based locking schemes can be improved, the point being to add more security. The first way to do this is to replace the chosen secret identifier by a physical one. This is achieved by using a random unique block [22] or a physical unclonable function (PUF) [34]. Indeed, most locking/unlocking schemes require strong cryptographic services and secure storage of a secret key as illustrated in Fig. 5. One

example of such a secure locking scheme is described in [35]. The unlocking system presented in [35] is based on a unique chip identifier and a corresponding key, which is sent by the IP rights owner. The chip identifier consists in two identities, one which is secret and stored in non-volatile write-once registers, and a public identity, which could be a label on the package, for instance. The system integrator sends the public identity to the IP rights owner in order to obtain the secret key. He/she can then unlock the circuit by entering the key he received in it, which will be compared internally with the one stored in the permanent memory. If they match, the device will be unlocked. This basic scheme, named symmetric cryptography, is used in most key-based protection schemes.

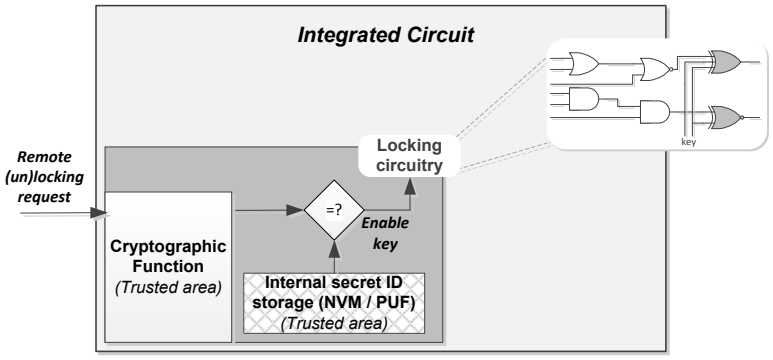

Figure 5 - Example of integrated circuit unlocking using a crypto-function and a secure storage of IC's ID

However, it is also possible to use public cryptography. In [36], each IC owns a unique signature, which is embedded in the memory. To unlock the IC, the user has to ask the designer for the proper key. The user does not see the key, as it is first encrypted by the public key, but sends it to the rights owner, who can send back the key to unlock the circuit, after deciphering it using his/her own private key. In this way, the user cannot unlock the IC unless he/she sends the right word to the designer. The designer can thus choose to unlock the ICs (or not), and can maintain a database to know how many circuits have been activated. He/she can also ensure that a circuit will not be activated twice.

As the reader can see, a wide range of key-based protection schemes is available. They can be adapted to most FPGA devices and ASIC, from low-cost to high-end systems.

4.2 Hardware camouflaging

Hardware camouflaging is a layout-level technique which consists in designing an ASIC with a non-standard gate library and dummy contacts [31]. The usual reverse engineering techniques, such as image recognition to

identify cell functions, are not efficient because it is impossible to distinguish the functionality at the transistor level using optical means. Such a camouflaging technique is proposed by a company called SypherMedia [32], who also propose EDA tools to help the designer. Another technique is to use split manufacturing when only one part of the IC manufacturing is outsourced, and a trustworthy manufacture (local manufacture) is used for the final manufacture of the IC (for example: wire connections, testing and packaging). Such a technique could be efficiently used in 3-D chips [33]. But the designer needs to have a foundry or to trust a local manufacturer to perform such a scheme. Perhaps split manufacturing could be a new way to manufacture ICs in the near future.

4.3 Reconfigurable area

Another way to protect an IC and to make design leaks useless is to add a reconfigurable area to the circuit, as shown in Fig. 6. The point here is to make a critical area, such as the instruction decoder unit of a processor [26] or a bus [27], reconfigurable. In the case of design theft, the attacker will not be able to use the circuit unless he/she has the bitstream required to make the reconfigurable area functional. This bitstream can be kept secret from the manufacturer, so that only the design rights owner and the system integrator share it. This protects it from theft, as the manufacturer cannot sell the design under his/her own name. An algorithm was proposed in [28] to find the best area to be made reconfigurable. The point is to choose the smallest possible area because integrating reconfigurable logic in a circuit is expensive in terms of silicon area overhead. But, at the same time, the impact of this area on the global design must be as high as possible. This is why data buses are good candidates, as they can easily lock the entire circuit [27].

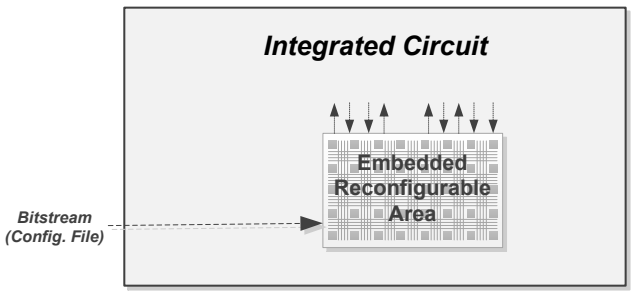

Figure 6 - Used of an embedded reconfigurable area to lock the device.

4.4 Bitstream encryption of FPGA configuration

When using FPGA, one way to protect the design data consist in bitstream encryption. In the last decade, many works focused on the security of the FPGA configuration process [37], [38]; the security of FPGA configuration remains an open topic. The first research work on this issue has proposed in 2003 [39] to increase security while maintaining flexibility by leaving the choice of the symmetric encryption algorithm to the user. In this case, a dynamic and partial reconfiguration of the FPGA is used. However, the system proposed in [39]

requires a complex control unit and configuring it also takes longer. Since this first work many propositions target static and partial secure reconfiguration [40-46]. Now, FPGA vendors propose bitstream encryption schemes for SRAM and FLASH technologies. Such schemes are suitable for many industrial applications. However, the security provided by current FPGA devices is not sufficient for many security applications. Recent works highlight security flaws in commercial products for Xilinx Virtex II [47], Virtex 4 and Virtex-5 [48], Altera Stratix II [49] and Actel/Microsemi ProASIC3 [50].

5. Hardware identification

Several methods of hardware identification are described in [12] and [13], including incoming inspection, which refers to shipping conditions. One can also use an external visual inspection to find labels, or use X-ray to identify the contents of a package [14]. Destructive methods can also be used like package stress exposure, die inspection, packaging evaluation, or characterization of the material, which are focused on clearly identifying the materials used to build the chip. It may be sufficient to determine if a product is counterfeit, but it is more interesting to have direct schemes to check whether or not the IC is a fake. Such schemes are called watermarking and fingerprinting. Fingerprinting is the measurement of a physical or behavioural characteristic of an IC (or an IP). This characteristic ensures one IC can identify another. Watermarking is a technique of steganography which proves the ownership of an IC (or an IP) by checking for the presence of hidden information called the watermark. The following two sub-sections give some examples of watermarking and fingerprinting schemes.

5.1 Watermarking

Watermarking can be used in two ways in the IC lifetime. First, the system integrator or the end user may want to check if all the items in a set of ICs he/she purchased are authentic. To do so, they check the watermark which is applied on all ICs, and authenticates each of them as illustrated at the top of Fig. 7. On the other hand, a designer can use watermarking to check that his IP has not been copied and fraudulently integrated in another system. To do so, he/she can compare the watermark he/she owns and the one which is on the IC, as illustrated at the bottom of Fig. 7. If they are identical, he/she can prove that the IC manufacturer has used his IP illegally.

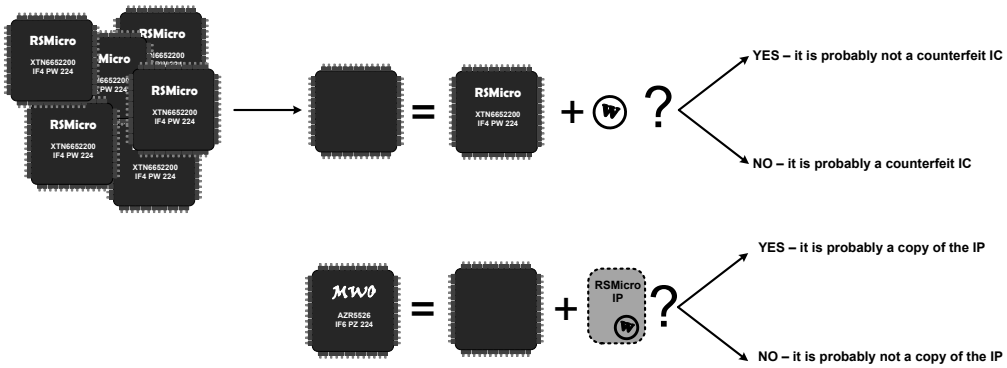

Figure 7 – Two scenarios for the use of watermarking, (*top*) detection of IC counterfeiting and (*bottom*) detection of IC or IP cloning.

Published watermarking schemes act on three different levels of abstraction from the application specifications to the circuit layout [51] and [52]. Firstly, pre-synthesis IP watermarking leads to algorithmic modifications of the application to hide the signature [53-56]. In [53] and [54], the authors target algorithm-level IP watermarking in the design flow. Both approaches [53] and [54] are based on slight changes to the digital filter parameters and do not affect system behaviour. [55] and [56] are based on adding new input/output sequences to the IP FSM. However, all the pre-synthesis IP watermarking schemes are too algorithm dependent; although these schemes may be suitable for some digital signal processing (DSP) applications, they are not practical for rapid marketing. Secondly, in-synthesis IP watermarking benefits from an automatic synthesis (behavioural or logic synthesis) tool to add a digital signature to the IP without significant overhead [57-59]. In [57], Kirovski presents IP watermarking based on combinational logic synthesis solutions. IP watermarking focused on behavioural synthesis techniques is described in [58] and [59]. It is also possible to modify the netlist before "place and route" to add a watermark to LUTs which are not entirely occupied [60]. By comparing the LUT content extracted from the bitstream and the original watermark, the designer can check the authenticity of the IP. However, this solution is not suitable if bitstream encryption is already used. Thirdly, post-processing IP watermarking uses the designers' knowledge of the circuit to manually change the IP hardware architecture and to add a digital signature [61, 62].

To conclude, pre-synthesis IP watermarking techniques are application dependent, and overhead costs are hard to measure. Post-synthesis IP watermarking techniques are time consuming, handmade, and design/device dependent. In-synthesis IP watermarking techniques introduce a power/area/timing overhead. Generalizing watermarking usage for IP identification is important. However, it requires sufficient generic IP watermarking techniques with very low area and timing overheads. Such techniques need to be implemented in automatic design flows for rapid component tagging in a designer friendly process.

Obviously, for the solutions listed above, extraction of the IP watermark cannot be described in detail here. Watermark extraction and checking can be difficult, especially when the IP is embedded in a larger system on chip (SoC). Other works propose dedicated schemes for checking watermarking: [63], [64] and [65] propose using side-channels (power consumption) to return the watermark data, which allows the designer to identify its design. Other examples are given in [66] and [67], in which heat is used as a contactless communication vector.

5.2 Fingerprinting

One of the common methods of IC fingerprinting is measuring specific path delays [68, 69]. With a theoretical delay for a specific path, and another after five years of use, it is possible to sort other circuits with more than 90% precision. This makes it possible to distinguish between new and old circuits, and refurbishing can be detected. Based on this principle, sorting functions have been proposed which could separate new and used circuits using a measurement vector [70]. The sorting function is built after a statistical study of trusted ICs provided by trustworthy manufacturers. Since it involves more parameters, it is more efficient in detecting old circuits. These methods can be applied on existing circuits, and do not require adding extra functions on the circuit.

The fingerprinting task can be much harder when it comes to IPs. Indeed, IPs are assumed to be implemented on different platforms or with different area constraints and timing requirements. Therefore, the sorting function used to distinguish between a tampered IP and a different form of the original IP will have some 'fuzzy edges'. For this task, *soft physical hash functions* are particularly useful. They can also be found in the literature using the following search terms: *robust* hash functions or *perceptual* hash functions. A hash function is defined as *soft* if it has the following property: *soft hash functions are such that similar objects should return highly correlated digests while different objects should produce uncorrelated ones* [71]. This property will allow the function to be immune to minor variations such as different place-and-route constraints, but larger modifications like security block tampering should be clearly highlighted. Both [72] and [73] suggest using power consumption as an input vector for the soft hash function.

Another fingerprinting technique involves physical unclonable functions (PUFs). This takes advantage of unpredictable physical variations in the silicon, which are caused by the fabrication process and can identify each IC in a unique way. Many silicon PUF structures have already been proposed, although only a few basic principles are known: one can use the race of delays between two symmetrical delay lines (arbiter PUF [61]), frequency mismatch in multiple ring oscillators (RO-PUF [74, 75]), metastability of a couple of cross-coupled elements (SRAM-PUF [76], TERO-PUF [77] and butterfly PUF [78]), and a mixture of a chain of configurable

delay lines and a ring oscillator (Loop-PUF [79]).

Finally, IC fingerprinting can be used to estimate the number of ICs built [80]. Based on animal counting methods, this method makes it possible to determine approximately how many chips were built, by measuring IC physical characteristics such as transistor threshold or power consumption. By randomly picking ICs and knowing the dispersion of the physical characteristics, it estimates with a precision of about 10% the number of chips which were produced. This is one way to fight overbuilding.

6. IP specific schemes

The IP market may need a different economic model than the IC market. This is due to the hybrid aspect of IPs between software and hardware. It is therefore necessary to develop a specific marketing scheme, which would allow designers to sell IPs under better suited conditions. In practice, IPs are not sold in the same way as circuits. Indeed, their target customers are system integrators, who may want to use only a small number of copies of the IP for their application. It is thus important to propose an alternative to the current system of marketing IPs. Basically, most people working in this field agree that a cryptographic approach is a good choice to build an IP protection scheme. This would make it possible to switch from a “pay once” to a “pay per use” licensing scheme [81], and would make it possible for small system integrators to purchase sophisticated IP cores, avoiding high one-off engineering charges. Cryptography can help by allowing the IP rights owner to unlock only a specified number of devices, and giving him/her control over the quantity of IPs he/she wants to sell. Examples of IP licensing schemes and IP temporary licensing schemes are presented in the following sub-sections.

6.1 IP licensing

The first IP licensing scheme was based on mutual identification between the IP rights owner and the manufacturer [82]. The IP rights owner has a smart card for each device, which is then used to individually authenticate it. The smart card is used in combination with a chip identifier to encrypt the design. The design is then deciphered inside the FPGA. It has also been proposed to include a trusted third party in the design exchange procedure. The third party would make the exchange more secure. The third party could be a middleman between other parties [34], or could contribute actively by ciphering and deciphering the design for secure exchanges between the parties [83]. Other more complex design encryption based schemes exist which take advantage of both symmetric and asymmetric cryptographic protocols [84].

An offline protocol was also defined to enable offline activation of IPs [85], with a trusted third party to

secure the exchange of information. An authentication module is embedded in each manufactured FPGA. Then the signature of each device is sent to the IP rights owner who can send back the encrypted IP. The IP will be deciphered inside the FPGA. In that way, the system integrator can use IPs which are proven to be original, and the IP rights owner only allows a specified number of FPGAs to be configured with his/her design.

When system integrators use IPs, they sometimes need to integrate multiple IPs in the same final design. In this case, when using cryptography to secure exchanges, it is important to consider the different cores of the design along with their associated unlocking mechanisms. In this context, a generic environment was proposed in [86], which would allow multiple cores to coexist in the same device, and solve activation problems. In particular, the authors provided details on the security requirements for the activation keys of the IP cores. One of the most important is that one key should not help recover the keys of other cores. Following these suggestions, efficient and appropriate IP protection schemes based on cryptography could be designed, and easily integrated.

6.1 Temporary unlocking

Other protection schemes take advantage of the characteristics of IPs, and only unlock the device temporarily. Here, temporary unlocking means that the device will only be unlocked for a limited period of time. The first time-limited unlocking scheme was developed in 2006 [87]. It is based on key exchange like in the case of permanent unlocking. In addition, a timer is embedded in the system. The condition for unlocking the device is that the system integrator owns the right key, but time must still remain in the timer. The time which is actually deducted can be computed from the use of the device. After this time has expired, the device is locked again.

Another interesting feature which can be achieved with temporary unlocking is distributing devices in demo mode [88]. Demo mode is usually characterized by lower performance. When the correct key is applied to the circuit, the full-functionality mode is activated. The unlocking mechanism takes advantage of unused instructions in the instruction set. These instructions are used to activate of the device.

An alternative is considering the circuit as initially unlocked, but with the possibility to lock it later. In this approach, some schemes are inspired by the techniques used by hardware Trojans. As described in [89], Trojans are activated after the circuit has gone through a set of "rare" nodes a certain number of times, at which point the circuit is considered to have been properly tested and quits the evaluation mode. At this point, the locking scheme is activated and causes the device to function incorrectly. The user can then request the unlocking key from the IP designer to disable the protection mechanism. This example shows us that investigating malicious

hardware design and behavior is also an opportunity to improve salutary hardware [90].

7. Summary

In this section, we compare the published protection schemes we referenced in Table 3 and Table 4 based on area and power overheads. Indeed, these are good indicators of the feasibility of a protection scheme. In Table 3 and Table 4, area overhead is sometimes given as a number of slices. Slices refer to the basic building blocks used in Xilinx FPGAs, usually four to six look-up tables and D flip-flops. When no other information than "low" was given in the article, we put "low", but the reader should be aware of the subjectivity of such an indication. "N/C". means no information was given on any particular point. We also give the name of the platform used to perform the protection. Finally, "-" in the column means the criterion was not relevant for the method of protection concerned.

Table 3 - Referenced works on hardware locking and IP licensing

Ref.	*Year*	*Category*	*Principle*	*Area overhead*	*Power overhead*	*Test platform*
[21]	*2007*	*Hardware locking*	*Modify the FSM*	*+0.5%*	*+0. 5%*	*ISCAS89 benchmark*
[22]	*2007*	*Hardware locking*	*Modify the FSM*	*+100%*	*+125%*	*MCNC91 benchmark*
[23]	*2013*	*Hardware locking*	*Exploit FSM synchronicity*	*low*	*low*	*N/C*
[16]	*2008*	*Obfuscation/camouflaging*	*Lock paths with XORs*	*<1%*	*N/C*	*c880 and c3540 benchmarks by ISCAS85*
[36]	*2008*	*Obfuscation/camouflaging*	*Public cryptography*	*18K slices*	*N/C*	*Xilinx Virtex II*
[24]	*2010*	*Obfuscation/camouflaging*	*Lock paths with XORs*	*0.036 mm² (130 nm techno) +10 000 gates for RSA*	*low*	*c880 and c3540 benchmarks by ISCAS85*
[29]	*2008*	*Obfuscation/camouflaging*	*Lock buses with XORs*	*1 500 slices for D-H +64 MUX gates*	*low*	*N/C*
[35]	*2003*	*Obfuscation/camouflaging*	*Key exchange*	*N/C*	*N/C*	*N/C*
[34]	*2008*	*Obfuscation/camouflaging*	*Key exchange + PUF and trusted 3rd party*	*+143%*	*+131%*	*MCNC91 benchmark*
[82]	*2005*	*Obfuscation/camouflaging*	*Mutual authentication*	*N/C*	*N/C*	*N/C*
[83]	*2012*	*Obfuscation/camouflaging*	*Design ciphering and trusted 3rd party*	*low*	*low*	*N/C*
[84]	*2007*	*Obfuscation/camouflaging*	*Symmetric and asymmetric ciphering*	*3 900 slices*	*N/C*	*Xilinx Virtex IV*
[67]	*2008*	*Obfuscation/camouflaging*	*Key exchange via heat*	*225 slices*	*0,5mW*	*Xilinx Spartan 3A*
[85]	*2006*	*Obfuscation/camouflaging*	*Offline protocol*	*4 400 slices*	*N/C*	*Xilinx Spartan 3*
[27]	*2010*	*Reconfigurable area*	*Reconfigurable bus*	*low*	*N/C*	*MCNC benchmark*
[28]	*2010*	*Reconfigurable area*	*Reconfigurable bus*	*low*	*N/C*	*MCNC benchmark*
[87]	*2006*	*Temporary un-locking*	*Key – timer*	*5%*	*N/C*	*Xilinx Virtex II*
[88]	*2009*	*Temporary un-locking*	*Demo-mode*	*250 slices for registers 1 500 slices for LUTs*	*N/C*	*Xilinx Virtex V*
[89]	*2012*	*Temporary un-locking*	*Hardware Trojan techniques*	*5%*	*5%*	*ISCAS89 benchmark*
[81]	*2002*	*IP licensing*	*"Pay-per-use" scheme*	-	-	-
[86]	*2009*	*IP licensing*	*Generic environment*	-	-	-

Table 4 – Referenced works on hardware identification.

Ref.	*Year*	*Category*	*Principle*	*Area overhead*	*Power overhead*	*Test platform*
[53]	*2000*	*Watermarking*	*Pre-synthesis watermarking*	*none*	*none*	*DSP algorithms*
[54]	*1999*	*Watermarking*	*Pre-synthesis watermarking*	*8 to 60%*	*N/C*	*31-tap FIR filter*
[55]	*2000*	*Watermarking*	*Pre-synthesis watermarking*	*0.2 to 143%*	*N/C*	*IWLS93 FSM benchmarks*
[56]	*2001*	*Watermarking*	*Pre-synthesis watermarking*	*10 to 300%*	*N/C*	*ISCAS89 benchmark*
[57]	*1998*	*Watermarking*	*In-synthesis watermarking*	*5%*	*N/C*	*MCNC suite*
[58]	*2005*	*Watermarking*	*In-synthesis watermarking*	*N/C*	*N/C*	*N/C*
[59]	*2012*	*Watermarking*	*In-synthesis watermarking*	*0.10%*	*low*	*Xilinx Virtex V*
[61]	*1999*	*Watermarking*	*Post-synthesis watermarking*	*0.1%*	*N/C*	*MIPS2000, ATR, DES…*
[62]	*2003*	*Watermarking*	*Post-synthesis watermarking*	*no*	*low*	*Data acquisition path, RISC core, video encoder and address generator*
[64]	*2012*	*Watermarking*	*Side-channel communication*	*88 slices*	*N/C*	*Xilinx Virtex II Pro*
[66]	*2009*	*Watermarking*	*Communication via heat*	*1 ring oscillator + 200 slices (sensor)*	*N/C*	*Xilinx Spartan 3 PIC 16F913*
[60]	*2008*	*Watermarking*	*LUT filling*	*5%*	*N/C*	*DES256 core*
[12]	*2011*	*Fingerprinting*	*Incoming inspection*	-	-	-
[13]	*2012*	*Fingerprinting*	*Incoming inspection*	-	-	-
[14]	*2013*	*Fingerprinting*	*X-ray inspection*	-	-	-
[68]	*2013*	*Fingerprinting*	*Measurement of path delays*	*N/C*	*N/C*	*HSPICE simulator MOSRA model*
[69]	*2013*	*Fingerprinting*	*Measurement of path delays*	*N/C*	*N/C*	*HSPICE simulator MOSRA model*
[70]	*2013*	*Fingerprinting*	*Sorting function*	*no*	*no*	*a DSP*
[73]	*2008*	*Fingerprinting*	*Arbiter PUF*	*N/C*	*N/C*	*180 nm RFID IC*
[74]	*2010*	*Fingerprinting*	*RO-PUF*	*N/C*	*N/C*	-
[75]	*2012*	*Fingerprinting*	*RO-PUF*	*N/C*	*N/C*	*ASIC*
[76]	*2007*	*Fingerprinting*	*SRAM-PUF*	*N/C*	*N/C*	*FPGA*
[77]	*2013*	*Fingerprinting*	*TERO-PUF*	*N/C*	*N/C*	*FPGA*
[78]	*2008*	*Fingerprinting*	*Butterfly PUF*	*no*	*N/C*	*FPGA*
[79]	*2012*	*Fingerprinting*	*Loop-PUF*	*low*	*low*	*Altera Cyclone II*
[80]	*2011*	*Fingerprinting*	*Number of ICs estimated*	*N/C*	*N/C*	*ISCAS benchmark*
[71]	*2012*	*Fingerprinting*	*Soft physical hash function*	*no*	*no*	*Atmel 644P*
[72]	*2013*	*Fingerprinting*	*Soft physical hash function*	*no*	*no*	*Xilinx Virtex II*
[63]	*2008*	*Fingerprinting*	*Power consumption as ID*	*N/C*	*N/C*	*Xilinx Spartan III Xilinx Virtex II*

The works listed in the two tables are very recent, more than 50% of published papers on the protection of design data were written in the last five years and 80% in the last seven years. This reflects the increased interest in this field. The observed overheads (area and power) vary considerably from one method to another. However, the passive methods (watermarking and fingerprinting), presented in Table 4 are the least expensive, but as the term “passive” suggests, they do not prevent the attack from taking place. On the other hand, the active methods (locking), listed in Table 3 can be very expensive, especially when they involve cryptographic functions. However, these provide the designer with the highest security levels.

Considering all the solutions, we are now able to list the characteristics of an efficient protection scheme. An efficient method for the protection of design data can be described using two types of characteristics. First, intrinsic specifications which define the system itself. An efficient protection method should be easy to unlock by the end user, but hard to analyse and break from a reverse-engineering point of view. According to these specifications, protection schemes based on cryptographic principles are clearly good candidates. We can also state that the solution should have a low overhead, in terms of both power and silicon area or logical elements. Indeed, once the device is finally unlocked, the locking system is useless. Consequently, it should have a low overhead and, if possible, no overhead at all. But at the same time, its locking scheme needs to be strong, so that circuits which are locked cannot be used. Specifically for IPs, it would be useful to be able to remotely control the locking level, in order to provide circuits in demo mode. A protection scheme should also be resistant to side-channel analysis. Given the possibility to use cryptographic methods, some specific characteristics are necessary. In particular, each instance of the circuit or IP should have its own key. A key that can be used for all the instances of the same design is not a sufficient criterion. But owning the key for one instance must not help recover other designs' keys.

We can also define extrinsic specifications for the protection scheme. This means how it should behave in the existing environment. First, it should be compatible with today's technologies and fabrication process in terms of physical requirements. In addition, design automation tools should be able to easily integrate it in the design flow.

It would be almost impossible to design a protection scheme which could be used for the protection of both ICs and IPs, as the distribution and sales constraints are not the same. In another connection, the protection scheme should obviously take economic problems into account, and these are not the same for ICs and IPs. Moreover, there is wide variety of threats, and a scheme to address them all would be impossible to develop. Considering all these criteria, the protection system must be strong and easy to implement.

Conclusion

Given all the protection schemes we analysed, we are now able to indicate some of the probable directions research will take. First of all, it is important to admit that none of the protection schemes we discussed here are effective against all the threats in our threat model. For that reason, a combination of several schemes should be used to ensure efficient protection of a circuit. Nevertheless, some significant gaps remain in the state of the art solutions. One is the fact that all the locking schemes are designed for synchronous digital logic circuits. To our knowledge, no protection scheme has been proposed to lock asynchronous digital logic circuits. This lack is even more apparent in the case of analogue circuits, which are mostly ignored.

In our opinion, locking schemes based on cryptographic protocols are the most reliable. Indeed, given a strong ciphering method and good resilience to side-channel attacks, it is possible for the designer to safely lock or unlock a circuit. Here, overhead could be increased for increased security, but it should be kept in mind that the locking mechanism is designed to be used only once. Theft can also be fought this way, by being able to unlock only a certain number of devices. This would prevent overbuilding. Cryptography can also be used to encrypt FPGA bitstreams, in order to protect the design against reverse engineering. However, when fighting against ASIC reverse engineering, the aim is to make design recovery harder than starting a new design from scratch.

Considering some of the examples we analysed, designing an efficient solution with no overhead at all appears to be feasible. It could be achieved by measuring the intrinsic characteristics of the device, which should be carefully chosen, based on their dependence on the integrity of the circuit. In this case, if the design is modified, it would be easily detected.

Our main aim now is to design a solution with low overheads and high security, with the ability to provide efficient protection against the widest possible range of threats.

Acknowledgement

The work presented in this paper was carried out in the framework of the SALWARE project number ANR-13-JS03-0003 supported by the French *Agence Nationale de la Recherche* and by the *Fondation de Recherche pour l'Aéronautique et l'Espace*.

References

[1] Hodges, D. A., 'Building the fabless/foundry business model [guest editorial]', IEEE Solid-State Circuits Magazine, 2011, 3, (4), pp. 7 – 44.

[2] Gorman, C., 'Counterfeit chips on the rise', IEEE Spectrum, 2012, 49, (6), pp. 16 – 17.

[3] AGMA, "AGMA, alliance for gray markets and counterfeit abatement." http://www.agmaglobal.org

[4] Pecht, M., Tiku, S., 'Bogus: electronic manufacturing and consumers confront a rising tide of counterfeit electronics', IEEE Spectrum, 2006, 43, (5), pp. 37 – 46.

[5] Laseter T., Ovchinnikov A., Raz G., 'Reduce, Reuse, Recycle or Rethink', *Strategy + Business*, vol. 61, 2010, pp. 1 – 5.

[6] Huang K., Guo J., Xu Z., 'Recycling of waste printed circuit boards: A review of current technologies and treatment status in China', Journal of Hazardous Materials, vol. 164, Elsevier, Paris, 2009, p. 399-408.

[7] McLoughlin, I., 'Reverse engineering of embedded consumer electronic systems', IEEE 15th International Symposium on Consumer Electronics (ISCE), Singapore, Singapore, June 2011, pp. 352 – 356.

[8] Masalskis, G., Navickas, R., 'Reverse engineering of CMOS integrated circuits', Elektronika ir Elektrotechnika - Electronics and Electrical Engineering, 2008, 8, (88), pp. 28 – 31.

[9] Nohl, K., Evans, D., Starbug, S., Plotz, H., 'Reverse-engineering a cryptographic RFID tag', 17th Conference on Security symposium, San Jose CA, USA, July 2008, pp. 185 – 194.

[10] Brutscheck, M., Franke, M., Schwarzbacher, A. T., Becker, S., 'Non-invasive reverse engineering of CMOS integrated circuits', IEEE 17th Telecommunications Forum TELFOR, Belgrade, Serbia, November 2009.

[11] Brutscheck, M., Franke, M., Schwarzbacher, A. T., Becker, S., 'Investigation of finite state machines in unknown CMOS integrated circuits', 14th Electronic Devices and Systems IMAPS CS International Conference, Brno, Czech Republic, September 2007, pp. 55 – 60.

[12] Sood, S., Das, D., Pecht, M., 'Screening for counterfeit electronic parts', Journal of Materials Science: Materials in Electronics, 2011, 22, (10), pp. 1511 – 1522.

[13] Koushanfar, F., Fazzari, S., McCants, C. *et al.*, 'Can EDA combat the rise of electronic counterfeiting?', 49th Design Automation Conference (DAC), San Francisco, USA, June 2012, pp. 133 – 138.

[14] Guin, U., Tehranipoor, M., DiMase, D., Megrdichian, M., 'Counterfeit IC detection and challenges ahead', ACM Special Interest Group on Design Automation, March 2013.

[15] Villasenor, J., Tehranipoor, M., 'The hidden dangers of chop-shop electronics', IEEE SPECTRUM, 2013, 50, (10), pp. 41 – 45.

[16] Roy, J. A., Koushanfar, F., Markov, I., 'EPIC: Ending piracy of integrated circuits', Design, Automation and Test in Europe, Munich, Germany, March 2008, pp. 1069 – 1074.

[17] Karri, R., Rajendran, J., Rosenfeld, K., Tehranipoor, M., 'Trustworthy hardware: Identifying and classifying hardware trojans', Computer, 2010, 43, (10), pp. 39 – 46.

[18] Tehranipoor, M., Koushanfar, F., 'A survey of hardware trojan taxonomy and detection', IEEE Design & Test of Computers, 2010, 27, (1), pp. 10 – 25.

[19] Jin, Y., Kupp, N., Makris, Y., 'Experiences in hardware Trojan design and implementation', IEEE International Workshop on Hardware-Oriented Security and Trust, San Francisco CA, USA, July 2009, pp. 50 – 57.

[20] Hachez, G., 'A comparative study of software protection tools suited for e-commerce with contributions to software watermarking and smart cards', PhD Thesis, Université Catholique de Louvain, March 2003.

[21] Alkabani, Y., Koushanfar, F., 'Active hardware metering for intellectual property protection and security', USENIX Security Symposium, Boston MA, USA, August 2007, pp. 291 – 306.

[22] Alkabani, Y., Koushanfar, F., Potkonjak, M., 'Remote activation of ICs for piracy prevention and digital right management', 10th IEEE/ACM international conference on Computer-aided design, Beijing, China, October 2007, pp. 674 – 677.

[23] Jung, E., Hung, C., Yang, M., Choi, S., 'An locking and unlocking primitive function of FSM-modeled sequential systems based on extracting logical property', International Journal of Information, 2013, 16, (8), pp. 6279 – 6290.

[24] Roy, J. A., Koushanfar, F., Markov, I., 'Ending piracy of integrated circuits', Computer, 2010, 43, (10), pp. 30 – 38.

[25] Rajendran, J., Pinot, Y., Sinanoglu, O., Karri, R., 'Security Analysis of Logic Obfuscation', 49th Annual Design Automation Conference, San Francisco CA, USA, June 2012, pp. 83 – 89.

[26] B. Liu, B. Wang, 'Embedded reconfigurable logic for ASIC design obfuscation against supply chain attacks, ' Design, Automation and Test in Europe Conference and Exhibition, pp. 1-6, March 2014.

[27] Baumgarten, A., Tyagi, A., Zambreno, J., 'Preventing IC piracy using reconfigurable logic barriers', IEEE Design & Test of Computers, 2010, 27, (1), pp. 66 – 75.

[28] Baumgarten, A., 'Preventing integrated circuit piracy using reconfigurable logic barriers', Master's thesis, Iowa State University, 2010.

[29] Roy, J. A., Koushanfar, F., Markov, I., 'Protecting bus-based hardware IP by secret sharing', 45th annual

Design Automation Conference, Anaheim CA, USA, June 2008, pp. 846 – 851.

[30] Basak, A., Zheng, Y., Bhunia, S., 'Active defense against counterfeiting attacks through robust antifuse-based on-chip locks', 32nd IEEE VLSI Test Symposium, Napa CA, USA, April 2014, pp. 1 – 6.

[31] Rajendran, J., Sam, M., Sinanoglu, O., Karri, R., 'Security analysis of integrated circuit camouflaging', ACM Conference on Computer & communications security, Berlin, Germany, pp. 709 – 720.

[32] SypherMedia, "Circuit Camouflage Technology. SMI IP Protection and Anti-Tamper Technologies," March 2012, http://www.smi.tv/SMI_SypherMedia_Library_Intro.pdf

[33] Imeson, R., Emtenan, A., Garg, S., Tripunitara, M. V., 'Securing computer hardware using 3D integrated circuit (IC) technology and split manufacturing for obfuscation', 22nd USENIX Security Symposium, Washington DC, USA, August 2013, pp. 495 – 510.

[34] Alkabani, Y., Koushanfar, F., 'Active control and digital rights management of integrated circuit IP cores', International conference on Compilers, architectures and synthesis for embedded systems, Atlanta GA, USA, October 2008, pp. 227 – 234.

[35] Adi, W., Soudan, B., Kassab, N., 'A protection mechanism for intellectual property rights (IPR) in FPGA design environment', IEEE 10th International Conference on Electronics, Circuits and Systems, Sharjah, December 2003, 1, pp. 92 – 95.

[36] Huang, J., Lach, J., 'IC activation and user authentication for security-sensitive systems', IEEE International workshop on Hardware-oriented security and trust, Anaheim CA, USA, June 2008, pp. 76 – 80.

[37] Badrignans B., Danger J.L., Fischer V., Gogniat G. and Torres L. (eds.), 'Security trends for FPGAs', Springer, 2011.

[38] Tehranipoor M., Wang C., 'Introduction to FPGA security and trust', Springer, 2011.

[39] Bossuet L., Gogniat G. and Burleson W., 'Dynamically Configurable Security for SRAM FPGA Bitstreams', Proc. IEEE Int'l Parallel and Distributed Processing Symposium, 2004, pp. 146-154.

[40] Zeineddini A.S. and Gaj K., 'Secure partial reconfiguration of FPGAs', Proc. IEEE Int. Conf. on Field-Programmable Technology, 2005, pp.155-162.

[41] Castillo J., Huerta P., Lopez V. and Martinez J.I., 'A secure self-reconfiguring architecture based on open-source hardware', Proc. Int. Conf. on Reconfigurable Computing and FPGAs, 2005, pp.7-10.

[42] Hori Y., Satoh A., Sakane H. and Toda K., 'Bitstream encryption and authentication with AES-GCM in dynamically reconfigurable systems', Proc. Int. Conf. on Field Programmable Logic and Applications 2008, pp.23-28.

[43] Drimer S. and Kuhn M. G., 'A Protocol for Secure Remote Updates of FPGA Configurations', Proc. Int'l Workshop on Reconfigurable Computing: Architectures, Tools and Applications, Springer, LNCS, vol. 5453, 2009, pp. 50-61.

[44] Castillo J., Huerta P. and Martinez J.I., 'Secure IP downloading for SRAM FPGAs', Microprocessors and Microsystems, Elsevier Science Publishers, vol. 31, issue 2, 2007, pp. 77-86.

[45] Braeken A., Genoe J., Kubera S., Mentens N., Touhafi A., Verbauwhede I., Verbelen Y., Vliegen J., and Wouters K., 'Secure remote reconfiguration of an FPGA-based embedded system', Proc. Int. Conf. on Reconfigurable Communication-centric Systems-on-Chip, 2011, pp.1-6, June.

[46] Vliegen J., Mentens N., and Verbauwhede I., 'A Single-chip Solution for the Secure Remote Configuration of FPGAs using Bitstream Compression', In International Conference on Reconfigurable Computing and FPGAs, IEEE Computer Society, 6 pages, 2013.

[47] Moradi A., Barenghi A., Kasper M., and Paar C., 'On the Vulnerability of FPGA Bitstream Encryption against Power Analysis Attacks: Extracting Keys from Xilinx Virtex-II FPGAs', Proc. ACM Conf. Computer and Communication Security, 2011, pp. 111-124.

[48] Moradi A., Kasper M., and Paar C., 'Black-Box Side-Channel Attacks Highlight the Importance of Countermeasures – An Analysis of the Xilinx Virtex-5 and Virtex-5 Bitstream Encryption Mechanism', Proc. Conf. on Topics in cryptology, Springer, LNCS, Vol. 7178, 2012, pp. 1-18.

[49] Moradi A., Oswald D., Paar C., and Swierczynski P., 'Side-Channel Attacks on the Bitstream Encryption Mechanism of Altera Stratix II', Proc. ACM/SIGDA Int. Symp. on Field-Programmable Gate Arrays , 2013, pp. 91-100, 2013.

[50] Skorobogatov S., and Woods C., 'In the blink of an eye: There goes your AES key', Cryptology ePrint Archive, Report 2012/296, 2012. http://eprint.iacr.org/

[51] Abdel-Hamid, A. T., Tahar, S., Aboulhamid, E. M., 'A survey on IP watermarking techniques', Design Automation for Embedded Systems, 2004, 9, (3), pp. 211 – 227.

[52] VSI Alliance, 'Intellectual property protection: schemes, alternatives and discussion', Intellectual Property Protection White Paper, 2001.

[53] Chapman, R., Durrani, T. S., 'Ip protection of DSP algorithms for system on chip implementation', IEEE

Transactions on Signal Processing, 2000, 48, (3), pp. 854 – 861.

[54] Rashid, A., Asher, J., Mangione-Smith, W. H., Potkonjak, M., 'Hierarchical watermarking for protection of DSP filter cores', IEEE Custom Integrated Circuits Conference, San Jose CA, USA, May 1999, pp. 39 – 42.

[55] Torunoglu, I., Charbon, E., 'Watermarking-based copyright protection of sequential functions', IEEE Journal of Solid-State Circuits, 2000, 35, (3), pp. 434 - 440.

[56] Oliveira, A. L., 'Techniques for the creation of digital watermarks in sequential circuit designs', IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2011, 20, (9), pp. 1101 – 1117.

[57] Kirovski, D., Hwang, Y.-Y., Potkonjak, M., Cong, J., 'Intellectual property protection by watermarking combinational logic synthesis solutions', IEEE/ACM International Conference on Computer-aided design, San Jose CA, USA, November 1998, pp. 194 – 198.

[58] Koushanfar, F., Hong, I., Potkonjak, M., 'Behavioral synthesis techniques for intellectual property protection', ACM Transactions on Design Automation of Electronic Systems, 2005, 10, (3), pp. 523 – 545.

[59] Le Gal, B., Bossuet, L., 'Automatic low-cost IP watermarking technique based on output mark insertions', Design Automation for Embedded Systems, 2012, 16, (2), pp. 71 – 92.

[60] Schmid, M., Ziener, D., Teich, J., 'Netlist-level IP protection by watermarking for LUT-based FPGAs', International Conference on Field-Programmable Technology, Taipei, Thailand, December 2008, pp. 209 – 216.

[61] Lach, J., Mangione-Smith, W. H., Potkonjak, M., 'Robust FPGA intellectual property protection through multiple small watermarks', 36th Design Automation Conference, New Orleans LA, USA, June 1999, pp. 831 – 836.

[62] Jain, A. K., Yuan, L., Pari, P. R., Qu, G., 'Zero overhead watermarking technique for FPGA designs', 13th Great Lakes symposium on VLSI, Washington DC, USA, April 2003, pp. 147 - 152.

[63] Ziener, D., Teich, J., 'Power signature watermarking of IP cores for FPGAs', Journal of Signal Processing Systems, 2008, 51, (1), pp. 123 – 136.

[64] Kasper, M., Moradi, A., Becker, G. T., *et al.*, 'Side channels as building blocks', Journal of Cryptographic Engineering, 2012, 2, (3), pp. 143 – 159.

[65] Marchand C., Bossuet L., 'IP Watermark Verification Based on Power Consumption Analysis', 27th IEEE International System-on-Chip Conference, Las Vegas, Nevada, USA, September 2014.

[66] Brouchier, J., Dabbous, N., Kean, T., Marsh, C., Naccache, D., 'Thermocommunication', IACR Cryptology ePrint Archive, 2009.

[67] Marsh, C., Kean, T., McLaren, D., 'Protecting designs with a passive thermal tag', 15th IEEE International Conference on Electronic Circuits and Systems, St. Julian's, Malta, August 2008, pp. 218 – 221.

[68] Moudgil, R., Ganta, D., Nazhandali, L., Hsiao, M., Wang, C., Hall, S., 'A novel statistical and circuit-based technique for counterfeit detection in existing ICs', 23rd Great Lakes symposium on VLSI, Paris, France, May 2013, pp. 1– 6.

[69] Moudgil, R., 'A statistical and circuit based technique for counterfeit detection in existing ICs', Master's thesis, Virginia Polytechnic Institute and State University, 2013.

[70] Huang, K., Carulli, J. M., Makris, Y., 'Counterfeit electronics: A rising threat in the semiconductor manufacturing industry', IEEE International Test Conference, Anaheim CA, USA, September 2013, pp. 1-4.

[71] Durvaux, F., Gerard, B., Kerckhof, S., Koeune, F., Standaert, F. X., 'Intellectual Property Protection for Integrated Systems Using Soft Physical Hash Functions', in Springer 'Information Security Applications', 2012, pp. 208 - 225.

[72] Kerckhof, S., Durvaux, F., Standaert, F. X., Gerard, B., 'Intellectual property protection for FPGA designs with soft physical hash functions: First experimental results', IEEE International Workshop on Hardware-Oriented Security and Trust, Austin TX, USA, June 2013, pp. 7 – 12.

[73] Devadas, S., Suh, E., Paral, S., Sowell, R., Ziola, T., Khandelwal, V., 'Design and implementation of PUF-based "unclonable" RFID ICs for anti-counterfeiting and security applications', IEEE International Conference on RFID, Las Vegas NV, USA, April 2008, pp. 58 – 64.

[74] Maiti, A., Casarona, J., McHale, L., Schaumont, P., 'A large scale characterization of RO-PUF', IEEE International Symposium on Hardware-Oriented Security and Trust, Anaheim CA, USA, June 2010, pp. 94 – 99.

[75] Katzenbeisser, S., Kocabas, U., Rozic, V., Sadeghi, A. R., Verbauwhede, I., Wachsmann, C., 'PUFs: Myth, fact or busted? A security evaluation of physically unclonable functions (PUFs) cast in silicon', Workshop on Cryptographic Hardware and Embedded Systems, Leuven, Belgium, September 2012, pp. 283 – 301.

[76] Guajardo, J., Kumar, S. S., Schrijen, G. J., Tuyls, P., 'Fpga intrinsic PUFs and their use for IP protection', Workshop on Cryptographic Hardware and Embedded Systems, Vienna, Austria, September 2007, pp. 63 – 80.

[77] Bossuet, L., Ngo, X., Cherif, Z., Fischer, V., 'A PUF based on transient effect ring oscillator and insensitive to locking phenomenon', IEEE Transaction on Emerging Topics in Computing, 2013, 2, (1), pp. 30 – 36.

[78] Kumar, S. S., Guajardo, J., Maes, R., Schrijen, G. J., Tuyls, P., 'The butterfly PUF protecting IP on every FPGA', IEEE International Workshop on Hardware-Oriented Security and Trust, Anaheim CA, USA, June 2008, pp. 67 – 70.

[79] Cherif, Z., Danger, J. L., Guilley, S., Bossuet, L., 'An easy-to-design PUF based on a single oscillator: the loop PUF', 15th Euromicro Conference on Digital System Design, Cesme, Turkey, September 2012, pp. 156 – 162.

[80] Wei, S., Koushanfar, F., Potkojnak, M., 'Integrated circuit digital rights management techniques using physical level characterization', 11th annual ACM workshop on Digital rights management, Chicago IL, USA, October 2011, pp. 3 – 14.

[81] Kean, T., 'Cryptographic rights management of FPGA intellectual property cores', ACM/SIGDA 10th international symposium on Field-programmable gate arrays, Monterey CA, USA, February 2002, pp. 113 – 118.

[82] Soudan, B., Adi, W., Hanoun, A., 'Novel secret-key IPR protection in FPGA environment', IEEE International SOC Conference, Herndon VA, USA, September 2005, pp. 267 – 270.

[83] Maes, R., Schellekens, D., Verbauwhede, I., 'A pay-per-use licensing scheme for hardware IP cores in recent SRAM-based FPGAs', IEEE Transactions on Information Forensics and Security, 2012, 7, (1), pp. 98 – 108.

[84] Guneysu, T., Moller, B., Paar, C., 'Dynamic intellectual property protection for reconfigurable devices', International Conference on Field-Programmable Technology, Kitakyushu, Japan, December 2007, pp. 169 – 176.

[85] Simpson, E., Schaumont, P., 'Offline hardware/software authentication for reconfigurable platforms', International Workshop on Cryptographic Hardware and Embedded Systems, Yokohama, Japan, October 2006, pp. 311 – 323.

[86] Guajardo, J., Guneysu, T., Kumar, S. S., Paar, C., 'Secure IP-block distribution for hardware devices', IEEE International Workshop on Hardware-Oriented Security and Trust, San Francisco CA, USA, July 2009, pp. 82 – 89.

[87] Couture, N., Kent, K. B., 'Periodic licensing of FPGA based intellectual property', IEEE International

Conference on Field Programmable Technology, Bangkok, Thailand, December 2006, pp. 357 – 360.

[88] Parrilla, L., Castillo, E., Garcia, A., Todorovich, E., Gonzalez, D., Lloris, A., 'Intellectual property protection of μP cores', Design of Circuits and Integrated Systems, Saragossa, Spain, November 2009.

[89] Narasimhan, S., Chakraborty, R. S., Bhunia, S., 'Hardware IP protection during evaluation using embedded sequential trojan', IEEE Design & Test of Computers, 2012, 29, (3), pp. 70 – 79.

[90] Bossuet L., Hely D., 'SALWARE: Salutary Hardware to design Trusted IC,' In Workshop on Trustworthy Manufacturing and Utilization of Secure Devices, TRUDEVICE 2013, Mai 2013.