Hindawi Publishing Corporation
VLSI Design
Volume 2011, Article ID 731957, 10 pages
doi:10.1155/2011/731957

Review Article

SoC: A Real Platform for IP Reuse, IP Infringement, and IP Protection

Debasri Saha and Susmita Sur-Kolay

Advanced Computing and Microelectronics Unit, Indian Statistical Institute, Kolkata 700108, India

Correspondence should be addressed to Debasri Saha, debasri_r@isical.ac.in

Received 12 October 2010; Revised 4 January 2011; Accepted 24 January 2011

Academic Editor: Shiyan Hu

Increased design complexity, shrinking design cycle, and low cost—this three-dimensional demand mandates advent of system-on-chip (SoC) methodology in semiconductor industry. The key concept of SoC is reuse of the intellectual property (IP) cores. Reuse of IPs on SoC increases the risk of misappropriation of IPs due to introduction of several new attacks and involvement of various parties as adversaries. Existing literature has huge number of proposals for IP protection (IPP) techniques to be incorporated in the IP design flow as well as in the SoC design methodology. However, these are quite scattered, limited in possibilities in multithreat environment, and sometimes mutually conflicting. Existing works need critical survey, proper categorization, and summarization to focus on the inherent tradeoff, existing security holes, and new research directions. This paper discusses the IP-based SoC design flow to highlight the exact locations and the nature of infringements in the flow, identifies the adversaries, categorizes these infringements, and applies strategic analysis on the effectiveness of the existing IPP techniques for these categories of infringements. It also clearly highlights recent challenges and new opportunities in this emerging field of research.

1. Introduction

In the recent era of automation, there are urgent needs of highly complex and application-specific multifunctional chips in every sphere of life. Customer's specification for complex chip causes explosion of gates on a single chip, advancement in process technology, and requirement for integrating heterogeneous technologies. The increased design complexity consequently needs more design effort. However, requirements for application-specific chips in every sphere mandate enhanced productivity and low cost. The only way to bridge the gap is to adopt hierarchical approach and reuse of already designed, optimized, and verified design components or fabricated and tested hardware cores to meet specification of a complex chip in time and at low cost. The way of designing an electronic system from the scratch has been replaced and system-on-chip (SoC) has emerged as an inevitable solution, where the major functional components of a complete end product are integrated into a single chip. Already-designed electronic components or fabricated hardware chips to be reused for these functional components constitute intellectual property (IP) cores. If an IP remains in electronic form, it is either a circuit description in hardware description language, that is, HDL (soft IP), may be any form of netlist, placed and routed design (firm IP), or design layout (hard IP); otherwise, it remains as hardware chip constituting a hardware IP core. A design tool is also treated as an IP. Soft IPs are more flexible but less optimized; on the other end, hardware IP cores are less flexible but more optimized. To be reused, an IP should have complete specification and proper documentation. The forms of the IPs suitable for reuse specifically on SoC will be discussed later.

An SoC usually contains reusable IPs, embedded processor(s) (a general-purpose processor and multiple special-purpose processors based on requirements) or controller(s), memory elements (SRAM, ROM, etc.), bus architecture (for interfacing IPs and other components on SoC), mixed signal blocks, programmable blocks (FPGA), voltage level shifter, clock circuits, test architecture, and so forth. An SoC may easily be enhanced by integrating more IP components with it. Furthermore, a number of SoCs can also be integrated to

realize a more complex system. The components of an SoC and the way of IP reuse in SoC environment are shown in Figure 1.

As reuse of IP is promoted in SoC environment, access control becomes essential for the IPs. In order to reuse an IP component on SoC, the SoC company should purchase the IP from its genuine vendor in legitimate way. Further, its reuse in SoC design house, in fabrication facility, and in SoC application environment should be protected. Unauthorized reuse of an IP by an SoC company and any other adversary renders loss of revenue to the genuine IP owner (IP vendor/IP creator), thus causes economic damage to the IP vendor.

An IP may be infringed during its design as well as during designing an SoC reusing that IP. So, in silicon industries, first, IP protection (IPP) techniques have been incorporated in VLSI design flow, and later on, security considerations have been extended for SoC design methodology. IPP techniques often rely on standard security mechanisms like cryptography, obfuscation, watermarking, fingerprinting, and so forth. Design concepts, system level knowledge and mechanism, design or chip level analysis, and characterization sometimes form the basis of the IPP techniques.

Section 2 discusses in detail the state of the art of IP reuse, IP infringement, and IP protection in SoC environment. Section 3 focuses on critical challenges, and Section 4 highlights the new opportunities in the field of SoC security. Conclusion appears in Section 5.

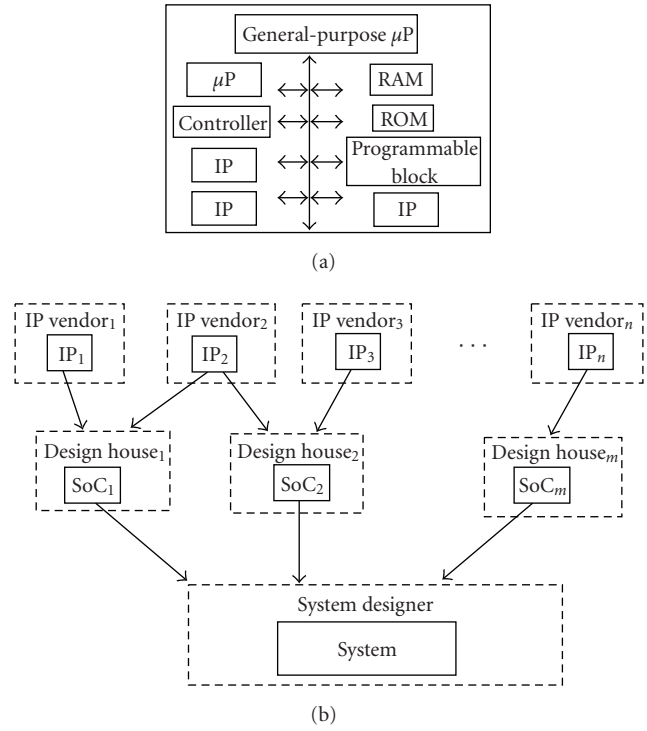

Figure 1: (a) Components on a system-on-chip, (b) IP reuse in SoC environment.

2. State of the Art

2.1. IP Reuse in SoC. Due to introduction of reuse techniques in designing an electronic system, design methodology undergoes transition from timing-driven ASIP (application-specific IP) design to block-based design of an SoC and then to platform-based design of a plug-and-play SoC. An ASIP, moderate in size and complexity, is optimized through synthesis, placement, and routing with great design effort to act as an IP component for reuse in the other two design techniques. In case of block-based design (BBD), an electronic system is partitioned into functional components, and these are mapped into available IP components (mostly firm IPs). For these firm IPs, their placements are retained, and routing, timing, and so forth are reoptimized contextwise for the overall optimization of these factors for the entire system. Finally, this software/hardware tradeoff in BBD is transformed into platform-based design (PBD), which provides an extensive and planned support to reuse of either hard IP or hardware IP on SoC. Here, based on functional requirements, IP blocks are chosen in a way so that each block can be well interfaced with its target blocks by designing suitable system architecture, their delay/power profiles satisfy constraints determined by the entire PBD, their test options are compatible with design-for-test mechanism on PBD, and those can function in one of the clock domains and voltage domains easily supported on SoC. In PBD, placement, routing, timing, delay calculation, physical verification, and test architecture construction all are performed hierarchically, with the only objective of properly designing the system (bus) architecture to realize the interface constraints [1, 2].

Reuse on SoC enhances productivity but not in a linear way; the reasons are the multiple design challenges faced in SoC design methodology and the overhead of integrating IPP techniques with the design flow. Design challenges include integration of heterogeneous device technologies and protocols, maintaining signal integrity, issues related to testing, clock timing, and voltage regulation [3, 4]. An IP is to be integrated with the other components on SoC, so it needs to satisfy several design constraints. Moreover, IPs are individually optimized, but the objective of SoC is to optimize the performance of the entire system obtained from integrating the IPs. Furthermore, it is to be noted that SoCs are mostly application specific, and, therefore, application of SoC defines the IPs. Different application needs different architecture, for example, buses, I/Os, processors, memories, and so forth. So, in order to meet constraints determined by the application of the SoC and the other components on the SoC, firm IPs are often partially redesigned. For hard IPs, one may use interface wrappers, which incur area overhead; otherwise, interface definitions of the available IPs are redesigned.

In IP-based SoC design flow, both software and hardware development are required (software/hardware codesign). Depending on the application on SoC, IPs close to requirement specifications are chosen, these are partially redesigned to be compatible for reuse on SoC, thereby transformed

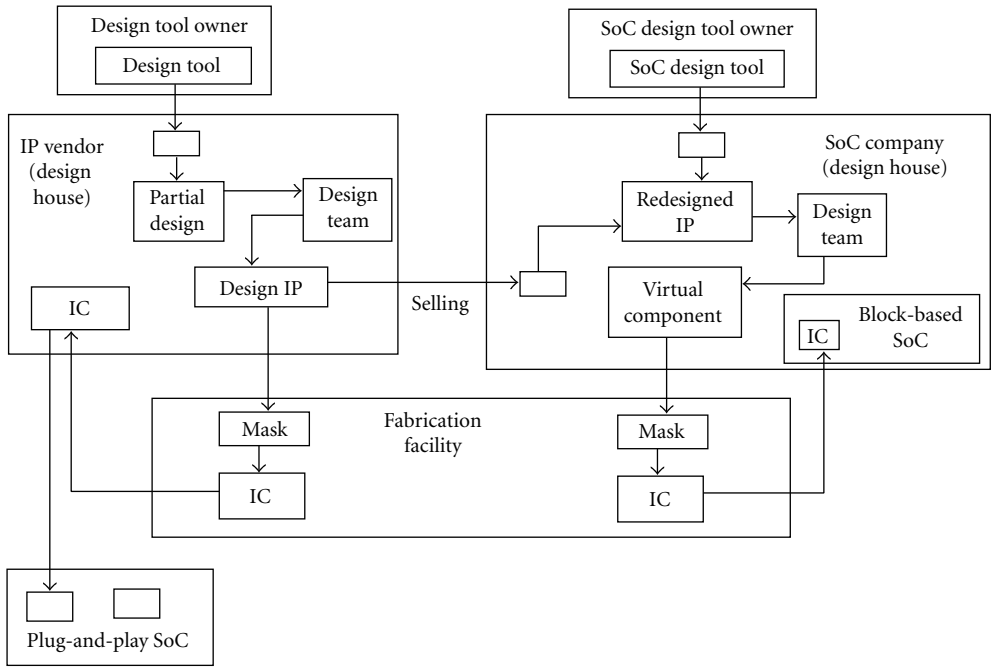

Figure 2: Movement of IP in integrated IP and SoC design flow.

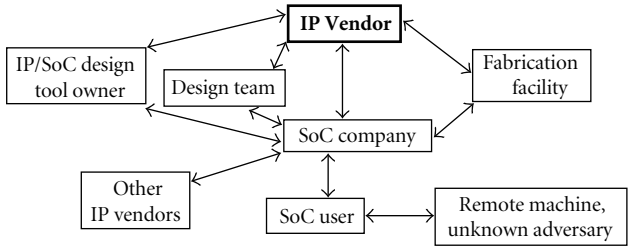

Figure 3: Interaction of IP vendor with various parties (may be adversaries).

into virtual components (VCs). VCs are emulated with programmable hardware on SoC and then fabricated to ICs. On the other hand, on SoC, bus architectures are designed to completely satisfy interface constraints, and other design works are completed, that is, placement of voltage level shifters, clock dividers, and so forth, and finally the SoC is fabricated and the firmware designers write the drivers for the components of SoCs [5].

The movement of an IP in IP-based SoC design flow is shown in Figure 2.

2.2. IP Infringement in SoC. Within an SoC company, significant design works are needed for IPs to transform those into virtual components (VCs) compatible on SoC. These VCs are then fabricated. Therefore, SoC company needs to have its own design team, design tool, and fabrication facility. However, in order to meet shrinking time to market, sometimes an SoC company hires designers and purchases SoC design tools from tool owner companies. An SoC company very often cannot afford a fabrication facility of its own. Consequently, it gets its VCs fabricated from separate fabrication foundry. So, an untrusted environment prevails between the SoC company, its hired design team, owner of the purchased design tools, and the fabrication facility used. A similar untrusted environment may occur for the IP vendor itself. Also, there are external adversaries. Following are the possible adversaries for misappropriation of an IP to be reused on SoC. Interactions of these parties with the IP vendor are shown in Figure 3.

(a) SoC company may intercept/hack an IP instead of legally purchase it from the IP vendor.

(b) An untrusted design team in IP/SoC design house may infringe an IP.

(c) Owner of an IP/SoC design tool may be malicious. Owner's Untrusted CAD tool, while used for designing an IP in IP design house or for partially redesigning the IP in SoC company, may infringe the IP.

(d) In case of a fabless IP company selling hardware IP or a fabless SoC company buying firm/hard IP, the IP is fabricated in external fabrication facility, which may be untrusted and misappropriates the IP.

(e) An IP may be misused by an SoC designer during realization of an SoC or by an SoC user, while the IP remains functional on SoC.

(f) Other IP vendors, providing IPs for the same SoC may be malicious. While an IP is exchanging data with the other IPs on SoC, those may extract valuable information from that IP.

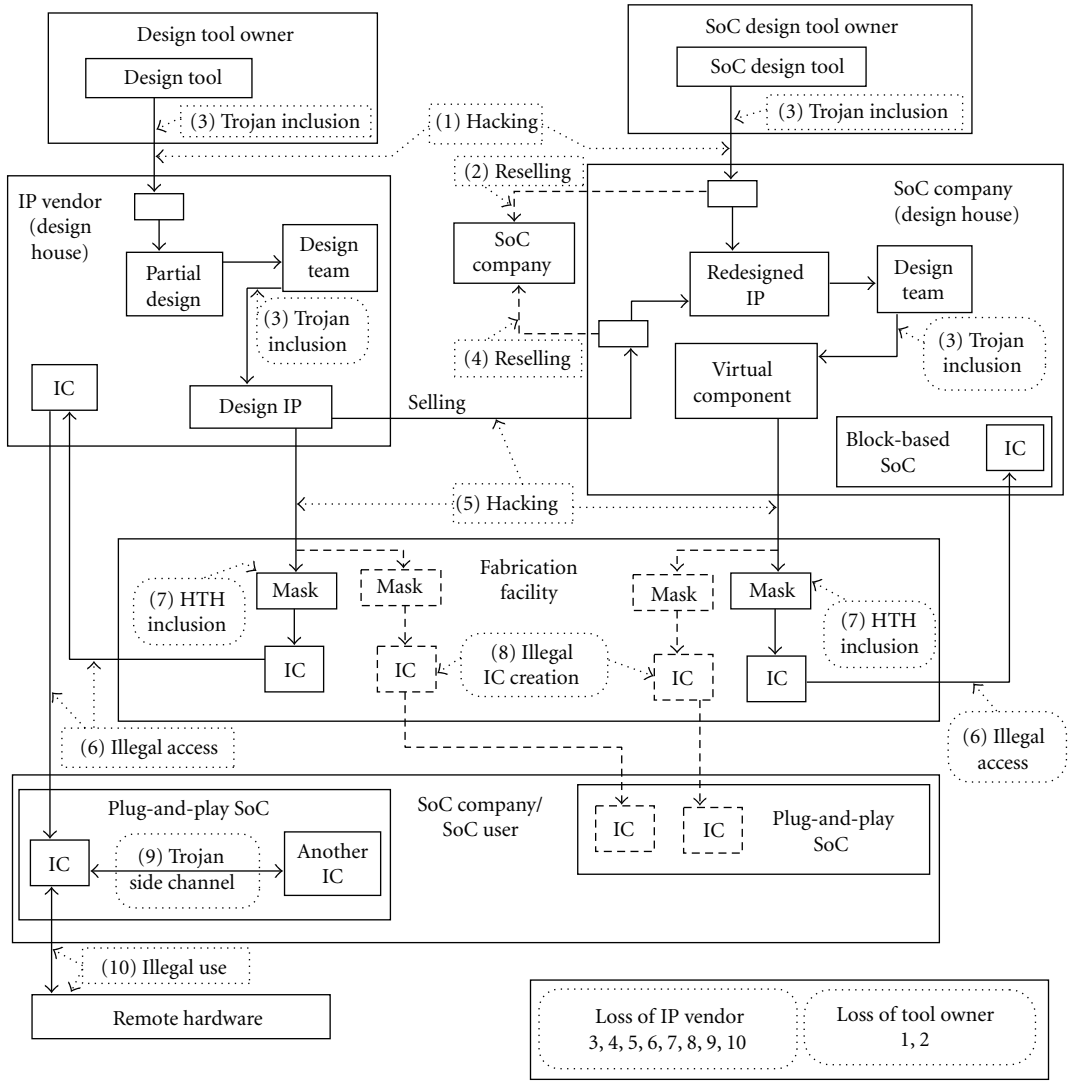

Figure 4: Locations and nature of IP infringements in IP-based SoC environment.

(g) Finally, an unknown adversary may misuse an IP, while on SoC, it is communicating with any remote hardware.

Misappropriation of IPs by any of these adversaries causes loss of revenue to the IP vendor. The threats for an IP can be divided into three categories, and these are discussed in the following subsections. The locations and nature of IP infringements in IP-based SoC environment are shown in Figure 4.

2.2.1. Unauthorized Access. An adversary may adopt the following ways to access an electronic or hardware IP or use it in unauthorized way [6].

(i) An SoC system design tool or an IP design tool may be hacked while transferred from tool owner to SoC/IP company. A firm IP may be hacked during its transfer from IP design house to SoC company. A hard IP can also be hacked during a similar transmission or in its way from fabless IP vendor/SoC company to the fabrication foundry.

(ii) An unauthorized person may intercept a hardware instance of an IP, that is, an ASIC, while it is distributed from the IP vendor to an SoC company or during its way from fabrication foundry to IP/SoC company.

(iii) In a security or networking application, an IP may communicate with a malicious remote hardware/application, or its communication may be intercepted by an adversary, thus an adversary may get benefited by the IP by misusing the IP.

2.2.2. Generation of Illegal Copies of IP. Illegal copies of an IP are generated due to intentional reselling of a firm/hard IP or fabrication of additional ICs in foundry [7, 8].

(i) An SoC design tool or a firm/hard IP is often resold by its legitimate buyer (an SoC design house) to another SoC design house in an illegal way.

(ii) From a mask of a hard IP or a virtual component, an untrusted fabrication facility may create illegal ICs or construct an additional mask, which is later on utilized to create any number of illegal ICs. Then the untrusted fabrication facility in unauthorized way sells these illegal ICs to SoC companies where those are reused on a plug-and-play SoC.

An hardware instance of an IP, that is, an IC, may be altered, replaced, or spoofed while is being used on a system. Therefore, each IC also needs separate authentication.

2.2.3. Information Retrieval through Trojan Horse. Trojan-based attack consists of inclusion of trojan horse (addition/modification of circuitry or design specification) into an IP by an adversary and later on, retrieval of circuit and design information through the already inserted trojan. This attack becomes relevant in SoC platform [9].

Trojan horse may be included in any of the following ways.

(i) Synthesis by an unreliable design team or usage of an untrusted design tool or an untrusted FPGA configuration tool may insert trojan horse into a soft/firm IP in IP design house or into a firm IP purchased in SoC design house.

(ii) In SoC company, unreliable firmware designer may maliciously modify interface definition of an IP, that is, its specification, to extract information from an IC after fabrication. Sometimes, untrusted SoC system design tool, used to redesign interface of an IP, facilitates unauthorized extraction of information from an IC fabricated from that IP.

(iii) Trojan Horses may be inserted into a mask by unreliable fabrication facility during fabrication so that the ASICs fabricated from that mask are trojan-infected.

Hardware trojan horse (HTH) acts as a side channel and leaks circuit and design information in the following scenarios.

(i) From an ASIC containing trojan horse, valuable information may be retrieved by the SoC designer during SoC realization or by an SoC user, while the IC is functional on SoC.

(ii) Information may also be extracted from an IP through any other untrusted hardware IP, while there is exchange of data between these two IPs.

(iii) While an hardware IP is communicating with a remote hardware (which may be malicious or their communication channel may be intercepted), trojan may leak information from the hardware IP.

In order to counterfeit these three major categories of IP misappropriation, several IP protection techniques have been adopted in various stages of IP-based SoC flow. Several critical attacks have also been designed to crack these security mechanisms or render these ineffective. The next section discusses on IPP techniques, attacks on those, and their countermeasures.

2.3. IP Protection in SoC

2.3.1. Locking-Based Security. This is a direct/active way to prevent unauthorized access of an electronic IP, to render illegally created/intercepted ICs useless and to protect communication of a hardware IP with remote devices. Locking-based security techniques include encryption, obfuscation, and remote activation.

(i) A hard IP, which is a design layout file in either GDSII (graphics data system II) or OASIS (open artwork system interchange standard) binary format, is locked by applying cryptographic encryption [10] on its binary content. It is encrypted while transferred from design house of IP/SoC company to fabrication foundry, where it is decrypted prior to fabrication. Symmetric (private) key cryptographic algorithms (e.g., DES, i.e., data encryption standard, AES, i.e., advanced encryption standard [11]) use same key for encryption and decryption, whereas, in public key cryptographic algorithms (e.g, RSA, ECC, i.e., elliptic key cryptography [11]) two separate keys are used for encryption and decryption. For FPGA design, corresponding bitfile core is kept encrypted [12] during its transmission to the SoC company, where it is decrypted using a decryption unit on FPGA hardware. Contrary to encryption which renders cipher text unreadable, another effective technique is obfuscation of an electronic IP, which renders the IP unusable to serve the purpose of security. The technique in [13] deterministically obfuscates a firm IP to a low-quality design using a secret key prior to transmission so that the high-quality IP can be regenerated only by the authorized person. It is directly applicable to firm IPs, thereby provides a faster way for its access control.

(ii) In the following techniques, IP is so designed that its each hardware instance needs a distinct secret key to be operational. If such an IC is illegally created by a malicious foundry or intercepted by an unauthorized person, its unauthorized user does not have the secret key, thus interception/illegal creation of ICs becomes useless.

An hardware instance of IP, that is, an IC, is locked by scrambling the control bus by controlled reversible bit permutations and substitutions [14]. Passive metering [15] registers each IC in a database uniquely based on its gate level characteristics. It can only detect illegal ICs by authenticating a chip against the database. In its active counterpart [16], design house keeps control of illegal ICs through monitoring of IC property and reuse, and by disabling functionalities of illegal ICs. The idea is further developed in remote activation technique [17], which replicates few states of underlying finite state machine (FSM) of an IP and then exploits inherent unclonable manufacturing variability to generate unique ID for each IC and adds a control, based on the unique ID, to the state transitions. Thus, it facilitates

piracy prevention and digital right management for each IC. An obfuscation technique discussed in [18], inserts a small FSM and constitutes a preinitialization state space, which is resilient against reverse engineering. It also uses a PUF-based IC-specific activation pattern to activate each IC with a distinct preinitialization state transition. Each preinitialization state is made observable as a particular output pattern for a predefined input sequence to authenticate each IC.

(iii) In the functional environment of SoC, in order to ensure security of hardware IPs communicating with an external device or remote hardware, their communications are encrypted using a cryptoprocessor embedded in hardware security module, thereby kept secret from the SoC user and other unknown adversaries. A cryptoprocessor is designed and implemented as a special-purpose embedded system optimized in terms of area, performance, and power for execution of a cryptographic algorithm in hardware. It is an SoC with several crypto-IP cores as coprocessors for high-speed execution of computation intensive operations, for example, SHA-1 hashing, wide operand modular arithmetic, and so forth, and for faster data compression required in the target encryption algorithm. Embedded cryptoprocessor has some area, performance, and power overhead on the chip quality. For example, an efficient FPGA implementation [19] of symmetric key encryption algorithm AES can achieve throughput of 24.922 Gb/s with the efficiency (throughput/area) of 6.97 Mb/s per slice of the FPGA used. An hardware implementation [20] of public key encryption technique ECC has an area requirement of 2.1 mm^2, delay 45 ns, and power 98.89 mW.

While a cryptoprocessor is tested on SoC, SoC designer may crack the encryption/decryption key exploiting information leaked from the cryptoprocessor. This class of attacks is known as side channel attacks, which include simple power analysis (SPA), dynamic power analysis (DPA), fault attack, cache timing attack, and attack using electromagnetic emanation. SPA and DPA are based on studying the device's power consumption while some key dependant sensitive variables are processed. Fault attack is based on the controlled induction of a fault and followed by analysis of a faulty cipher text produced from the cryptoprocessor. Cache timing attack uses the correlation of a sensitive variable with cache timing characteristics of an embedded implementation. Countermeasure to these attacks is to apply masking, permutation table, or random switching logic [21] to hide the nature of sensitive variables. A new class of side-channel attacks has been designed based on correlation power analysis (CPA) and mutual information analysis (MIA) [22]. This section of attacks awaits innovative solutions.

2.3.2. Authentication-Based Security. Security threat exists even after applying locking-based security due to the following reasons.

(a) An SoC design house may illegally resell the purchased firm/hard IP to another SoC company or resell the hardware IP and intentionally share its secret key (to make it operational) with the illegal buyer.

(b) The decryption key of the decryption unit on FPGA may be cracked applying side channel attacks, thus an attacker manages to get the unencrypted bitfile.

Watermarking and fingerprinting provide passive protection and authenticate an IP and an IC to establish digital rights of legal IP vendor and legitimate buyer of an IC, respectively, even if locking-based security is cracked. The process of embedding the signature of IP vendor in form of watermark is known as watermarking, whereas including that of IP buyer in form of fingerprint is termed as fingerprinting. Fingerprints may be constructed for a set of ICs fabricated from an IP by characterizing their manufacturing variability. If any misappropriation of an IP or IC is suspected, these signatures are verified from the IP or IC to identify the genuine IP vendor or legitimate IP buyer.

The following are the desiderata of a signature-embedding technique.

(a) A signature embedding technique should incur low overhead on the chip quality in terms of area, delay, and power.

(b) The signature embedding and signature verification techniques should be fast.

(c) The technique should be robust against typical attacks like tampering, finding ghost signatures, additive attack [8], as well as resilient in the design flow.

Signature embedded by applying constraints in place/route phase of physical design [8] or through incremental router [37] cannot be verified from an IC fabricated from that marked design. Hence, in order to provide effective security of an IP on SoC, the mark (i.e., watermarks and fingerprints) insertion technique should be resilient against fabrication. Furthermore, it is desirable that the embedded signatures should be resistant against process variation.

Marks embedded through any of the following techniques are resilient against fabrication.

(i) Signature of IP vendor, that is, watermark, may be hosted into nonused cells of memory structure described in HDL [24].

(ii) Underlying finite state machine is modified so that desired watermark can be detected at the chip's outputs for a particular key input sequence [25]. Such a way of generating watermark only for particular key inputs is known as dynamic watermarking.

(iii) Dynamic watermarking has been applied to reconfigurable scan architecture during physical synthesis so that desired watermark can be verified as scan outputs [26]. For both dynamic watermarking techniques, watermarks are easily verifiable.

(iv) Physically unclonable functions (PUFs) [27] authenticate each IC instance by leveraging manufacturing variability of each fabricated IC, based on a nonfunctional characteristics such as delay or power. So, the techniques based on PUFs can perform fingerprinting of ICs.

(v) A set of fingerprints is constructed for an IC family utilizing side channel informations such as power,

temperature, and electromagnetic profile. The technique is also capable of detecting trojan of 3-4 orders of magnitude smaller than the main circuit using signal processing [28]. The above two techniques in (iv), and (v) can be categorized as postfabrication IC fingerprinting techniques. In (iv), PUF structure is embedded in the IP design by the IP design house as a prerequisite.

(vi) Signature of IP vendor, that is, watermark, may be embedded into logic synthesis phase through incremental technology mapping of selective disjoint closed cones [29].

(vii) Signatures of both IP vendor and IP buyer are stored as configuration bitstream of unused configurable logic blocks (CLBs) of FPGA [30].

The techniques in (i) and (ii) are associated with behavioral phase and, therefore, are applicable to both ASIC and FPGA. The techniques in (iii), (iv) and (v) are for ASIC authentication and those in (vi), and (vii) are for FPGA bitfile core authentication.

While a hardware IP is operational on SoC and needs to communicate with an external device or remote hardware, hardware security module on the SoC authenticates the device prior to establishing communication. In the technique discussed in [23], a secure hardware/cryptoprocessor in an SoC authenticates the remote processor by challenging it to compute a check sum that depends on cycle-by-cycle activities of its internal microarchitectural mechanisms for a given code within a time limit. Thus, it controls unauthorized access of data from an IP operating on SoC.

2.3.3. Security against Trojan Horse. An IP, to be reused on SoC, is first handled by the SoC system designer, who is well equipped with circuit and system knowledge, and when it is operational on SoC, it exchanges data with other hardware IPs on SoC and other external devices or remote hardware. So, if the IP is already trojan-infected, trojan side channel attack is quite prominent. So, detection of existence of trojan becomes essential prior to dispatching the chip to SoC company. Trojan may be inserted in design level as well as during its fabrication. So, we need *efficient* trojan detection techniques effective for both design and hardware IP.

Information hiding strategy to design trusted system [31] is capable of detecting possible existence of trojan horse in design IP. In this work, an IP company (design team 1) creates a high-quality partial solution for a given problem specification and extracts a modified specification from that partial solution. The modified specification is then sent to another design team (team 2 which may be untrusted). From the complete design generated by the team 2, a partial solution is extracted to cross-check it with the high-quality partial solution created by team 1 to detect possible inclusion of trojan by team 2.

The technique in [32] precisely measures actual combinational delay of large number of paths. Thus, it can detect hardware trojan horse (HTH) due to increase in delay at certain paths. This method characterizes each fabricated IC instance based on manufacturing variability, therefore, it is effective for IC fingerprinting. However, it is exhaustive and not time efficient.

Among the trojan detection techniques based on gate-level characterization (GLC), [33] characterizes gates using physical properties specifically leakage current. Measurements on leakage current are processed with linear programming (LP). It imposes additional constraints on LP formulation to indicate nature and location of additional ghost circuitry. However, this technique cannot characterize all the gates due to collinearity and cannot detect collinear HTH. The technique in [34] breaks the correlations by applying thermal control on the process of GLC. Thermal conditioning imposes extra variations on gate level leakage power by characterizing switching power followed by heat dissipation due to it. Both of them are effective for process invariant trojan detection.

The authors of [35] proposed an IPP technique for detection of trojan horse inserted in FPGA design files, from bitfile core, or from FPGA hardware loaded with bitfile core. It is a parity-based method and uses two-level randomized ECC structures. Failing to detect desired parity relation signals possible existence of additional circuitry, that is, trojan in the FPGA design.

The technique in [36] resists an untrusted synthesis CAD tool to add/modify design specification. It employs the CAD tool under inspection to difficult scheduling and assignment synthesis tasks for a completely specified pertinent design so that there is no room for the tool to add malicious circuitry. The technique uses a trusted tool to fully account all resources at each step.

Effectiveness of several IPP techniques for various security aspects are summarized in Table 1. "Y" in a cell indicates the technique in the corresponding row is effective to achieve the security aspect specified in the corresponding column.

3. Challenges

(i) In the SoC environment, SoC designer has enough opportunity to misappropriate an IC during realization of the SoC. An IC exchanges data with other hardware IPs on SoC and external devices or remote hardware. SoC users, other IPs on SoC and external or remote device may be malicious, so an IC faces multiple security threats in SoC environment. Remote activation authenticates an IC as a legal instance of hardware IP, but if the IC belongs to some untrusted source, its remote access may pose threats to other trusted IP components on the same SoC.

(ii) In SoC environment, each instance of IP, that is, IC, needs to be protected. All the existing techniques to control access of ICs use PUF-based IC fingerprinting. However, signature of an IC is limited in length, and an attacker may guess a signature by developing a timing model for PUF structure.

(iii) Emphasis has been given to IC fingerprinting and design IP watermarking. However, fingerprinting of design IPs is quite relevant as transaction of IP often takes place in form of firm/hard IP between IP vendor and SoC company. A fingerprinting technique

Table 1: Effectiveness of IPP techniques for the following security aspects.

IPP techniques	Access control			Authentication		Trojan detection	
	IP	IC	Data from IC	IP	IC	From design	From hardware
Charbon and Torunoglu 2000 [10]	Y						
Adi et al. 2006 [12]	Y						
Saha and Sur-Kolay 2009 [13]	Y						
Roy et al. 2008 [14]		Y					
Alkabani et al. 2008 [15]		Y					
Alkabani and Koushanfar 2007 [16]		Y					
Alkabani et al. 2007 [17]		Y			Y		
Chakraborty and Bhunia 2009 [18]		Y			Y		
Granado-Criado et al. 2010 [19]			Y				
Dyka and Langendoerfer 2005 [20]			Y				
Suzuki et al. 2004 [21]			Y				
Deng et al. 2009 [23]			Y	Y			
Castillo et al. 2007 [24]				Y			
Abdel-Hamid et al. 2005 [25]				Y			
Saha and Sur-Kolay 2010 [26]				Y			
Majzoobi and Koushanfar 2009 [27]					Y		
Agrawal et al. 2007 [28]					Y		Y
Cui et al. 2008 [29]				Y			
Lach et al. 2001 [30]				Y	Y		
Gu et al. 2009 [31]						Y	
Li and Lach 2008 [32]					Y		Y
Potkonjak et al. 2009 [33]							Y
Wei et al. 2010 [34]							Y
Dutt and Li 2009 [35]						Y	Y
Potkonjak 2010 [36]						Y	

needs wider space for mark insertion compared to watermarking a design IP.

(iv) CAD tools are too complex to be traced. Therefore, synthesis tools and libraries, testing and verification tools and configuration scripts act as significant sources of hardware trojan. Designing trustable ICs and systems using untrusted CAD tools and untrusted reusable IP cores has been emerged as a true challenge in IP-based SoC security.

(v) With very few techniques, for example, gate-level characterization, process invariant trojan detection is possible. However, these techniques are exhaustive and consequently inefficient. There is a tradeoff between efficiency and strength of a protection technique for certain security aspects.

(vi) Trojan can be inserted in design level as well as during its fabrication. It is difficult to design a technique for both pre and postfabrication trojan detection. Furthermore, the postfab HTH detection should be process invariant.

4. New Opportunities

(i) There is no sole technique to ensure protection in all the three major security aspects shown in Table 1. Furthermore, any technique, which serves two security purposes, is not equally efficient and robust in both the tasks.

(ii) We are in need of efficient techniques for certain detection of presence of trojan horse both at design level as well as at hardware level.

(iii) The existing countermeasures to side channel attacks cannot prevent the new class of side channel attacks based on CPA and MIA. SoC-based industry, therefore, awaits for innovative solutions to resist these attacks.

5. Conclusion

Rapid growth of technology in semiconductor industry continually creates new security holes specially in SoC platform. As security threats have direct impact on Psilicon-based

economy, these threats demand immediate solutions to check misuse of technology and consequently loss of revenue for the IP companies. In recent trends, design-for-security for IP-based SoC design methodology forms an open research area, where constant effort and domain expertise are needed to check misappropriation of IPs.

References

[1] R. Rajsuman, *System-On-a-Chip*, Artech House, London, UK, 2009.

[2] W. Wolf, *Modern VLSI Design: IP-Based Design*, Prentice Hall, New York, NY, USA, 2008.

[3] S. Raif and K. Arno, *Reuse Techniques for VLSI Design*, Kluwer Academic Press, Boston, Mass, USA, 1999.

[4] R. Seepold and N. M. Madrid, *Virtual Components Design and Reuse*, Kluwer Academic Press, Boston, Mass, USA, 2000.

[5] D. Mathaikutty and S. Shukla, *Metamodelling-Driven IP Reuse for SoC Integration and Microprocessor Design*, Artech House, London, UK, 2009.

[6] G. Qu and M. Potkonjak, *Intellectual Property Protection in VLSI Designs: Theory and Practice*, Kluwer Academic Press, Boston, Mass, USA, 2003.

[7] D. Saha, P. Dasgupta, S. Sur-Kolay, and S. Sen-Sarma, "A novel scheme for encoding and watermark embedding in VLSI physical design for IP protection," in *Proceedings of the International Conference on Computing: Theory and Applications (ICCTA '07)*, pp. 111–116, March 2007.

[8] A. B. Kahng, J. Lach, W. H. Mangione-Smith et al., "Constraint-based watermarking techniques for design IP protection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 20, no. 10, pp. 1236–1252, 2001.

[9] A.-R. Sadeghi and D. Naccache, *Towards Hardware-Intrinsic Security: Foundations and Practice*, Springer, New York, NY, USA, 2010.

[10] E. Charbon and I. H. Torunoglu, "On intellectual property protection invited paper," in *Proceedings of the 22nd Annual Custom Integrated Circuits Conference (CICC '00)*, pp. 517–522, May 2000.

[11] A. Menezes, P. V. Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, New York, NY, USA, 1996.

[12] W. Adi, R. Ernst, B. Soudan, and A. Hanoun, "VLSI design exchange with intellectual property protection in FPGA environment using both secret and public-key cryptography," in *Proceedings of the IEEE Annual Symphosium on VLSI (ISVLSI '06)*, pp. 24–32, 2006.

[13] D. Saha and S. Sur-Kolay, "Encoding of floorplans through deterministic perturbation," in *Proceedings of the 22nd International Conference on VLSI Design*, pp. 315–320, January 2009.

[14] J. A. Roy, F. Koushanfar, and I. L. Markov, "Protecting bus-based hardware IP by secret sharing," in *Proceedings of the 45th Design Automation Conference (DAC '08)*, pp. 846–851, June 2008.

[15] Y. Alkabani, F. Koushanfar, N. Kiyavash, and M. Potkonjak, "Trusted integrated circuits: a nondestructive hidden characteristics extraction approach," *Proceedings of the Information Hiding*, pp. 112–117, 2008.

[16] Y. Alkabani and F. Koushanfar, "Active hardware metering for intellectual property protection and security," in *Proceedings of the USENIX Security Symphosium*, pp. 291–306, 2007.

[17] Y. Alkabani, F. Koushanfar, and M. Potkonjak, "Remote activation of ICs for privacy prevention and digital right management," in *Proceedings of the International Conference on CAD*, pp. 674–677, 2007.

[18] R. S. Chakraborty and S. Bhunia, "HARPOON: an obfuscation-based SoC design methodology for hardware protection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 10, pp. 1493–1502, 2009.

[19] J. M. Granado-Criado, M. A. Vega-Rodríguez, J. M. Sánchez-Pérez, and J. A. Gómez-Pulido, "A new methodology to implement the AES algorithm using partial and dynamic reconfiguration," *Integration, the VLSI Journal*, vol. 43, no. 1, pp. 72–80, 2010.

[20] Z. Dyka and P. Langendoerfer, "Area efficient hardware implementation of elliptic curve cryptography by iteratively applying karatsubas method," in *Proceedings of the Design Automation and Test in Europe*, vol. 3, pp. 70–75, 2005.

[21] D. Suzuki, M. Saeki, and T. Ichikawa, "Random Switching Logic: A Counter-Measure against DPA based on Transition Probability," ePrint Archive, 2004.

[22] E. Prouff and R. McEvoy, "First-order side-channel attack on the permutation table countermeasures," in *Proceedings of the Cryptographic Hardware and Embedded Systems*, pp. 81–96, 2009.

[23] D. Deng, A. H. Chan, and G. Edward Suh, "Hardware authentication leveraging performance limits in detailed simulations and emulations," in *Proceedings of the 46th Design Automation Conference (DAC '09)*, pp. 682–687, 2009.

[24] E. Castillo, U. Meyer-Baese, A. García, L. Parrilla, and A. Lloris, "IPP@HDL: efficient intellectual property protection scheme for IP cores," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 15, no. 5, pp. 578–591, 2007.

[25] A. T. Abdel-Hamid, S. Tahar, and EL. M. Aboulhamid, "A public-key watermarking technique for IP designs," in *Proceedings of the Design, Automation and Test in Europe (DATE '05)*, pp. 330–335, March 2005.

[26] D. Saha and S. Sur-Kolay, "A unified approach for IP protection across design phases in a packaged chip," in *Proceedings of the 23rd International Conference on VLSI Design*, pp. 105–110, January 2010.

[27] M. Majzoobi and F. Koushanfar, "Techniques for design and Implementation of secure reconfigurable PUFs," *ACM Transactions on Reconfigurable Technology and Systems*, vol. 2, no. 1, article 5, 2009.

[28] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar, "Trojan detection using IC fingerprinting," in *Proceedings of the IEEE Symposium on Security and Privacy (SP '07)*, pp. 296–310, May 2007.

[29] A. Cui, C. H. Chang, and S. Tahar, "IP watermarking using incremental technology mapping at logic synthesis level," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 9, pp. 1565–1570, 2008.

[30] J. Lach, W. H. Mangione-Smith, and M. Potkonjak, "Fingerprinting techniques for field-programmable gate array intellectual property protection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 20, no. 10, pp. 1253–1261, 2001.

[31] J. Gu, G. Qu, and Q. Zhou, "Information hiding for trusted system design," in *Proceedings of the 46th Design Automation Conference (DAC '09)*, pp. 698–701, July 2009.

[32] J. Li and J. Lach, "At-speed delay characterization for IC authentication and Trojan horse detection," in *Proceedings of the IEEE International Workshop on Hardware-Oriented Security and Trust (HOST '08)*, pp. 8–14, June 2008.

[33] M. Potkonjak, A. Nahapetian, M. Nelson, and T. Massey, "Hardware trojan horse detection using gate-level characterization," in *Proceedings of the 46th Design Automation Conference (DAC '09)*, pp. 688–693, July 2009.

[34] S. Wei, S. Meguerdichian, and M. Potkonjak, "Gate-level characterization: foundations and hardware security applications," in *Proceedings of the Design Automation Conference (DAC '10)*, pp. 222–227, 2010.

[35] S. Dutt and L. Li, "Trust-based design and check of FPGA circuits using two-level randomize ECC structure," *ACM Transactions on Reconfigurable Technology and Systems*, vol. 2, no. 1, 2009.

[36] M. Potkonjak, "Synthesis of trustable ICs using untrusted CAD tools," in *Proceedings of the Design Automation Conference*, pp. 633–634, 2010.

[37] T. Nie and M. Toyonaga, "An efficient and reliable watermarking system for IP Protection," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E90-A, no. 9, pp. 1932–1939, 2007.