NOVEL SECRET-KEY IPR PROTECTION IN FPGA ENVIRONMENT

Bassel Soudan[(1)], Wael Adi[(2)], and Abdurahman Hanoun[(3)]

(1) bsoudan@sharjah.ac.ae, University of Sharjah, Sharjah, United Arab Emirates
(2) wadi@ieee.org, Technical University of Braunschweig, Braunschweig, Germany
(3) a.hanoun@tu-harburg.de, Technical University Hamburg, Harburg, Germany

ABSTRACT

Some VLSI IP owners prefer to leave programming their IP into a Field Programmable Gate Array (FPGA) to the end customer. A major concern is the possible over-deployment of the IP into more devices than originally licensed. In this paper, we propose a system based on secured handshaking with encrypted device and design authentication information ensuring that the IP can only be deployed into agreed upon devices. The system consists of hardware-supported design encryption and secured authentication protocols.

I. INTRODUCTION

There exist today a large number of fab-less design houses that offer their designs exclusively in the form of FPGA-destined IP. Their rights to the IP must be protected against three possible forms of attack. Interception during IP transfer to the customer (or device), duplication by cloners after the IP has been deployed into the market, and possible over-deployment by the customer or the customer's out-sourced device programmer.

Device manufacturers have implemented different mechanisms to ensure that the details of the design cannot be meaningfully intercepted en-route to the device or accessed once on it. This helps protect against interception and cloning. However, it does not address over-deployment.

With current technology, there is no way to limit the number of deployments once the IP arrives at the programming site. This paper discusses a secure mechanism for protecting against IP over-deployment. Our proposed solution allows the IP to be licensed for specific uniquely identifiable devices. It is impossible to deploy the IP into any additional device without the involvement of the IP owner (or provable illegal collaboration of the device manufacturer).

The rest of this paper is organized as follows: section 2 discusses current FPGA-based IPR protection mechanisms, section 3 describes the proposed IPR protection technology and protocols, section 4 discusses security threats and possible attacks and section 5 presents a summary and conclusion.

II. CURRENT FPGA-BASED IPR PROTECTION MECHANISMS

Several mechanisms for protecting the IP programmed into FPGA devices have been implemented by manufacturers or proposed by researchers. These can be categorized into methods for preventing the IPR violation and methods for detecting the violation.

A. Methods for Preventing IPR Violation

FPGA manufacturers have employed design stream encryption and other security measures to guard against interception and cloning.

In its ProASIC^{PLUS} and ProASIC3 lines of FPGAs, Actel replaced volatile SRAM on-chip configuration cells with non-volatile floating gate Flash ROM elements (FROM) [3]. Since FROM elements are non-volatile, the on-board bit stream to allow autonomous re-configuration after power interruption is eliminated. Therefore, there is no bit stream to intercept or clone in the deployed product.

In addition, Actel supported the use of encrypted design bit streams by integrating an AES block cipher core into the device, as shown in Figure 1 [2]. The AES block is used to decrypt the design bit stream as it is being loaded into the device. The decryption key is programmed into the target device typically at the IPR owner's own facility. The decryption key is stored in secure on-chip FROM. After the key has been programmed into the device, it only accepts a design bit stream that has been encrypted with the same key.

Xilinx on the other hand, integrated DES and Triple DES decryptors into its latest Virtex-II devices [4]. Device programming is done in two phases. In phase one the decryption keys are programmed into the device in a secure environment (typically the IPR owner's own

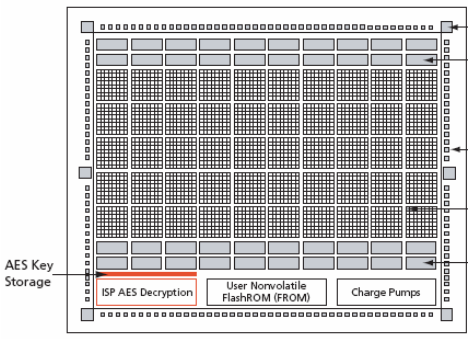

Figure 1. AES Decryption Core inside the ProASIC3/E FPGA[1]

facility). In phase two the encrypted design stream is downloaded into the device. The keys are stored in a small amount of on-chip RAM that should be backed up with a battery. Xilinx also made partial reconfiguration and readback impossible ensuring that a device cannot be altered once it is programmed with a secure bitstream.

In both solutions from Actel and Xilinx intercepting the design bit stream is useless and cloning is impossible without knowing the exact key. However, in both solutions the IP owner must pre-program the devices with the decryption key before sending them to the out-sourced or end user's programming facility. If the IP owner is going to bring the devices in-house for key pre-programming, they might as well download the IP into the devices themselves. Most IP owners don't want (or are not equipped) to be directly involved in the programming of the devices, even for key insertion. They are simply design houses that would prefer to leave dealing with the devices to the end user or the contract programming facility.

B. Methods for Detecting IPR Violation

There have been several proposed methods for detecting when an IPR violation occurs. Most are based on embedding hidden watermarks in the design [5] and [6]. While these methods may be successful in determining when an IPR violation has occurred, they are not useful in addressing our concern of preventing the violation in the first place. One needs to suspect that a violation has occurred and have a clue of where it has occurred before being able to detect it. The sheer number of FPGA based designs on the market makes it prohibitive to even contemplate checking every single design for possible IPR violations.

[1] Figure courtesy of Actel.

III. SECRET KEY IPR PROTECTION MECHANISM

To simplify the process of IP deployment, the number of parties involved should be reduced to the absolute minimum. The aim of this proposal is to eliminate the involvement of the device manufacturer once the blank device has been delivered to the end-customer. It also aims to eliminate the need for the IP owner to have first hand possession – even temporarily – of the devices for key programming. The communication of the IP is carried out electronically and dealing with the physical devices is left to the end-customer or contract programmer.

A. Requirements for Proposed Mechanism

The proposed IPR protection mechanism is designed to satisfy the following requirements:

- The IP should be securely distributable over an open channel like the Internet.
- The system security should be based on known unbroken ciphering technology.
- Secret–key cryptography should be employed to keep the system time and hardware complexity as low as possible.
- IPR owner should not need to pre-program the devices on-site with a decryption key.
- IPR owner must be able to limit the number of system deployments as part of a business agreement.
- The design bit stream must not be extractable from the device after programming to prevent illicit replication.

B. Hardware Components

The proposed system implementation includes the following two hardware components:

Device Identity Module **DIM.** This module guarantees the essential physical uniqueness of each device. It should reside in the FPGA in a tamper-proof area where no attack would be possible in any operating mode. Figure 2 represents a simplified functional block diagram for the proposed DIM.

Verification Smart Card **VSC.** A smart card published by the FPGA manufacturer to allow the IPR owners to securely authenticate each FPGA device based on its open identity.

C. *Manufacturer System Initiation:*

The manufacturer needs to initiate the mechanism at the time the device is manufactured. For each device, the manufacturer establishes a unique public *device identity* DI, which can be branded on the device itself and/or stored in a readable area in the device. Then the manufacturer implements in each device a DIM like the one shown in Figure 2.

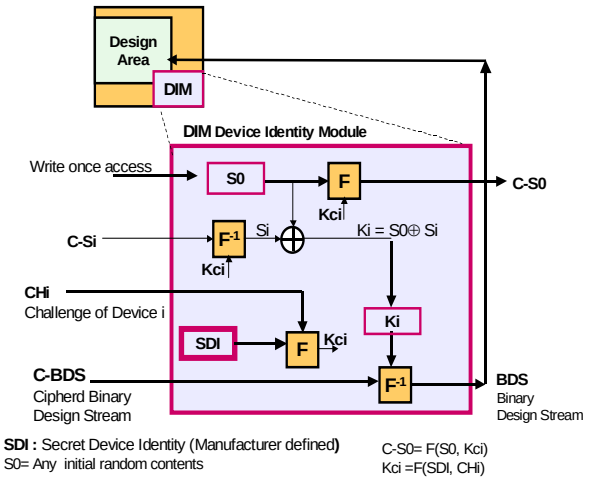

Figure 2. Architecture of a Possible FPGA Device Identity Module (DIM)

The DIM includes the following elements:

- A non-volatile *write-once memory* register to hold the *secret device identity* SDI. The SDI register must not be modifiable without knowledge of its current contents. SDI is mapped from DI such that no key collision is possible:

$$SDI = F(DI, SMK) \quad (1)$$

  where SMK is the manufacturer's Secret Master Key for that particular FPGA type.
- A second write-once-register S0 whose contents should be fully random and not necessarily known to the FPGA manufacturer.
- An hardware decipher block F^{-1}.

The function F (and its inverse F^{-1}) can be a strong cipher with a size of 128-bits such as AES, or any secure hash function [7] – [9]. The size of all registers, secret keys and other vectors should be 128 bits the same as the cipher block size.

The device manufacturer creates a VSC for each particular FPGA type. VSCs can be openly distributed without loss of system security. The IPR-owner obtains a VSC from the manufacturer to use for authenticating DIs provided by the customer. The VSC will also assist the IPR-owner in generating an encryption key for a device with a particular DI as shown in Figure 3.

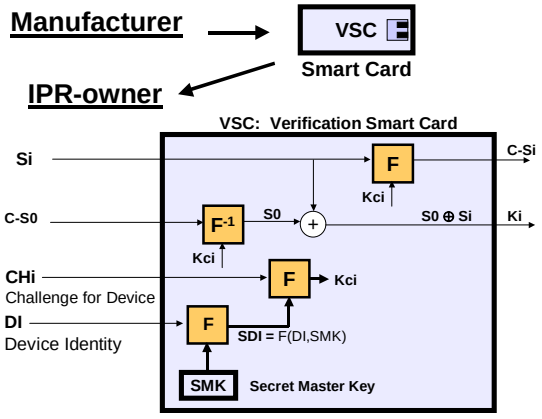

Figure 3. IPR-Owner gets a Verification Smart Card (VSC) from the manufacturer

D. *Secured Design Transfer Protocol:*

The secured design transfer can be expressed as follows (refer to the 4-step protocol in Figure 4):

1. The customer sends a purchase order to the IPR-owner including a list of all DI's on which the design will eventually be downloaded. The customer must already own the devices in order to obtain their DI's.
2. For each device (i) in the customer's order, the IPR-owner generates a random challenge CH_i together with a secret stream S_i. The IPR-owner encrypts the secret stream S_i in the VSC to get its ciphered version $C\text{-}S_i$:

$$C\text{-}S_i = F(S_i, Kc_i) \quad (2)$$

   where Kc_i is a secret key generated within the VSC that the IPR-owner cannot access. Kc_i is based on DI_i and CH_i as follows:

$$Kc_i = F(CH_i, SDI_i) \quad (3)$$

   SDI_i is generated automatically in the smart card as shown in Figure 3. The group of CH_i's and $C\text{-}S_i$'s are sent to the customer.
3. The customer collects the CH_i and $C\text{-}S_i$ for the particular device and applies them to the two dedicated inputs of the corresponding FPGA.

Internally, an identical Kc_i is generated and used to decipher $C\text{-}S_i$ and obtain the original S_i. The device uses S_i and its internal S0 to generate a design decryption key K_i:

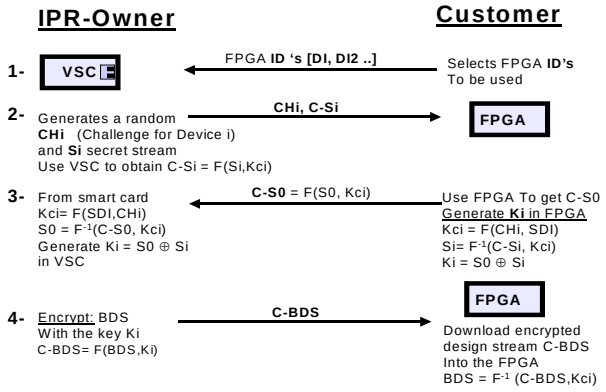

Figure 4. Secured design transfer protocol

$$K_i = S0 \oplus S_i \quad (4)$$

The FPGA is now ready to download an encrypted design. The device also generates a new response C-S0 as the ciphered version of S0 of that FPGA device.

$$C\text{-}S0 = F(S0, Kc_i) \quad (5)$$

The end customer collects all C-S0's for the devices and sends them to the IPR-owner.

The IPR-owner applies each device's C-S0 to the VSC. Internally, the VSC uses the key Kc_i to decipher C-S0 and obtain the original S0 of the device. Then the VSC uses the IPR-owner's S_i and the device's S0 to generate a design encryption key K_i for that particular device as shown in Figure 3.

4. For each device, the IPR-owner encrypts the binary design stream BDS using the device's K_i to get its encrypted version $C\text{-}BDS_i$ as

$$C\text{-}BDS_i = F(BDS, K_i) \quad (6)$$

The set of C-BDS's is sent over the Internet to the customer ready to be downloaded to the individual devices.

The above protocol can run in parallel m times for all required device identities.

IV. SECURITY THREATS AND POSSIBLE ATTACKS

The system is breakable in only two possible scenarios, both of which require the collaboration of the manufacturer and end customer:

1. The system depends on a supposedly unique DI/DIM pair for every device. The device manufacturer can break the system by generating devices with duplicate DI/DIM pairs.
2. The device manufacturer may build a backdoor into the device where the customer can access the decrypted BDS directly.

Both cases can be easily traced if the transfer protocol is authenticated and legally prove the IPR violation and its originator.

V. SUMMARY AND CONCLUSION

The proposed system offers a mutually-authenticated system for design transfer that prevents IPR violations in an FPGA design environment. The design distribution can be done over public Internet media without loss of security. The mechanisms employed are based on trustable secret-key low complexity functions such as those used in mobile systems. The system is breakable only if the device manufacturer collaborates with the end customer, which can be easily proven as the device uniqueness incorporates global authentication.

REFERENCES

[1] Adi, W., "Secured Mobile Device Identification with Multi-Verifier", Proceedings of the International Conference on Telecommunications (ICT2001), pp. 289 – 292, 2001

[2] Actel, "ProASIC3/E Security," Application Note available at http://www.actel.com, cited on 14/4/2005.

[3] Actel, "Implementation of Security in Actel's ProASIC and ProASIC^{PLUS} Flash-Based FPGAs," Application Note available at http://www.actel.com, cited on 14/4/2005.

[4] Peattie, M., "Use Triple DES for Ultimate Virtex-II Design Protection," XCell Journal, Summer 2001, pp. 29 – 29, available at http://www.xilinx.com, cited on 14/4/2005.

[5] Kahng , A. B., Kirovski , D., Mantik, S., Potkonjak, M., and Wong, J. L., "Copy Detection for Intellectual Property Protection of VLSI Design." Proc. IEEE/ACM Intl. Conference on Computer-Aided Design, November 1999, pp. 600-604.

[6] Newbould, R. D., Carothers, J. D., Rodriguez, J. J., and HolmanW. T., "A Hierarchy Of Physical Design Watermarking Schemes For Intellectual Property Protection Of IC Designs," Proceedings of the International Symposium on Circuits and Systems, 2002, Vol. IV, pp. 862 – 865

[7] Technical Specification 3G Security, Security Architecture 3G TS 33.102 V. 3.2.0 from 10.1999

[8] Specification of the MILENAGE Algorithm Set. 3GPP TS 35.206 V5.0.0 ETSI, http://www.3gpp.org

[9] AES, Advanced Encryption Standard, Federal Information Processing Standards Publication, FIPS 197, 2001. Or http://csrc.nist.gov/publications/.