A Method for Hardware Metering

M. MOIZ KHAN, SPYROS TRAGOUDAS
Dept. of Electrical & Computer Engineering
Southern Illinois University at Carbondale
1230 Lincoln Dr. MC 6603 Carbondale IL 62901-6603
USA
http://mypage.siu.edu/moiz

Abstract: - In this paper we present a hardware metering method to meter or monitor legitimate use of integrated circuits embedded in a computer system. The motivation for this work comes from fabless semiconductor design business/operation model which has become very effective and popular in last few years. A fabless design house produces a design and gives it to a foundry to manufacture the final chip. Once design masks are with the foundry, the original designer has no effective control over the number of copies of the chip that are produced by foundry. We present a finite state machine based automated hardware metering technique to control the use of chip regardless of the number of copies being manufactured.

Key-Words: - Hardware Metering, Intellectual Property Protection

1 Introduction

Intellectual Property Protection (IPP) issues have gained significant attention in last few years as a result of IP reuse and system-on-chip design methodology. Many IP protection techniques like watermarking and fingerprinting have been proposed in the literature. These techniques aim at protecting design from theft and manipulation by a user and in case of theft or manipulation the authorship of the real owner/designer can be proved with a certain degree of confidence. Watermarking [3,4,5,6,7,8,10] is a mechanism for identification that is designed to identify the original designer or author of the design. A watermark can be inserted by imposing certain characteristics in the solutions of various optimization problems involved in the design phase. For example, [5] gives algorithms to watermark solutions of graph coloring problem and to final layout [4]. Fingerprinting [11,12,13] is a special case of watermarking targeting primarily FPGA designs. Watermarking combinational logic synthesis solutions have been proposed by Kirovski *et al*. in [8]. Design rewiring based watermarking [6] schemes have also been proposed.

Although, IP protection and proof of authorship are important issues, it does not provide control on IP usage since the designer cannot stop illegal manufacture and use of the IP. Particularly, fabless design houses produce their design and give it to a silicon manufacturing foundry to fabricate and then to a vendor to sell the final manufactured silicon chip. Once design is with foundry design house has no control over the number of copies of the chip being produced. Illegal manufacturing of chip results in loss of revenue for the company in addition to copyright and patent infringements. The Virtual Socket Interface Alliance's (VSIA) development Working Group on IP Protection has also recognized hardware metering as one of the key requirements for IP protection [9,10].

With this is mind Koushanfar e*t al.* [1,2] proposed first known solution in literature to IP metering problem where each copy of chip is reconfigured such that each chip is functionally equivalent to original design but with a different implementation.. According to [1, 2] hardware metering problem is based on four basic principles: **P1:** Create many distinct copies with the same functionality. **P2:** Once two identical copies are found, prove ownership. **P3:** Determine number of tests to be conducted to gain a certain level of confidence that there are no unauthorized copies in the market. **P4:** If there are unauthorized copies, one must estimate the number of unauthorized copies made.

Principle P2 can be tackled by IP protection techniques like watermarking and fingerprinting as mentioned above. P3 seeks to provide an estimate of the designer's effort to prove foundry's dishonesty. P4 estimate number of unauthorized copies if they exist. [1, 2] propose statistical models to evaluate these samples and the number of tests needed to attain a required confidence level. Although, statistical models are effective tools and could be used to evaluate chip samples from market, the basic problem of detecting existence of unauthorized chips still remains with the designer. As an alternative, we propose a hardware metering scheme in which the end user must communicate with the designer and activate the chip before it can be used.

In this paper a new solution to P1 and fast methods to incur minimum effort by designer to address P3 and P4 as redefined below. **P3** How can one devise an automated mechanism so that the probability that illegal copies of chip can be made is negligible? **P4:** If an illegal copy exist then how fast can it be detected with minimum effort and an *automated* mechanism ?

The rest of the paper is organized as follows. In Section 2 we analyze the drawbacks of existing methods and outline our approach. In Section 3 we present details of our proposed metering scheme, and conclude in Section 4.

2 Problem Formulation & Rationale

In this section we formulate our hardware metering problem, redefine it as seen by Koushanfar *et al* [1, 2] and provide a rationale for the proposed scheme. Let *A* be the original IP designer or creator of the design. *A* is generally a fabless design house which does not have its own silicon fabrication facility. *B* is silicon foundry and *C* is the end-use/customer of the chip. Designer *A* gives his design (IP) to foundry *B* to fabricate (say) *N* copies of chip. After fabrication chips are sold to end-user *C* via a vendor. Once *A* gives design to *B* it has no control over the number of copies of design made and sold. Although, designs are given to foundry under strict legal non-disclosure agreements, the hardware metering problem arises on the pretext that foundry can make more than *N* chips and sell illegal copies which results is a loss of revenue to designer.

Solution to hardware metering problem as presented by Koushanfar *et al.* [1,2] has the following disadvantages: (1) The responsibility of detecting existence of illegal copies of chip and then proving ownership rests solely on *A.* (2) *A* must take sample from the market and do statistical analysis to detect illegitimate chips, since size of sample determines confidence level. (3) Cost of configuring each chip differently and subsequent testing of each chip for correct functionality is prohibitive. We now redefine

Hardware Metering Problem: *Given that A gives a design to foundry B for fabrication and then chips are sold to end user C, devise an automated mechanism for metering the use of an illegally produced chip and detect an illegal chip with minimum detection effort by A*

In our approach the end user of the chip must contact the IP designer to get an *activation key* or *password*. The activation key or password must be unique to each chip to prevent the user of a particular chip to illegally pass key to other buyers of chip. Such a mechanism enables *A* to control IP usage when *C* wants to use it, and thereby prevent illegal use. *A* will then provide *C* with a key/password which will enable him to use IP.

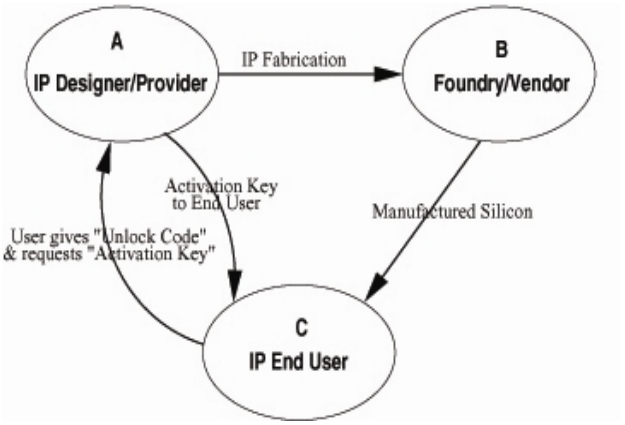

Fig 1.Activation key based Hardware IP Metering

The advantages of this approach are: (1) An automated self initiating approach to hardware metering problem. (2) Enables *A* to take control (3) *A* does not need to rigorously monitor the market for illegal copies. (4) Foundry/Vendor *B*, is bypassed by end-user *C*. (5) The IP designer does not have to put in efforts configure each IC to make functionally equivalent ICs.

3 Proposed Hardware Metering Method

In this section we present our hardware metering method and describe the required components.

3.1 Protocols & Required Components

In principle, the protocol requires that the chip on first start up will generate a *Locking code* which the user will take to original designer to get a corresponding *Unlock Code*. The IP can be operated only after unlocking it with an *Activation Key* or *Unlock code*. A small finite state machine with many paths from initial state to final state is augmented to the FSM of the IP. To achieve this, the state transition diagram (STG) of the FSM is augmented to the STG of chip as shown in Fig.2. Following steps describe the protocol:

1. IP Designer *A* gives design to foundry *B* to fabricate the chip.

2 The design has *k* counters of *m* bits each. On first start up (or power on) each counter gets a random initial state.

3.*Locking Code*: When the user receives the Chip, he read the contents of the *k* counters and reports it

to the designer to get an Activation Key. The initial content of counter is the Locking code.

4. *The Activation Key* is a sequence of input patterns which when applied to the chip, traverses a particular sequence of states (path) in the STG of augmented FSM to reach initial state of FSM of IP. This path must contain the states corresponding to the *k m*-bit counters.

5. The designer gets Locking code from user and gives him *Activation key* so that the augmented FSM traverses states corresponding to *k m*-bit counters and reaches the initial state of FSM of IP.

6.*Checker Circuit:* A built in comparator mechanism checks that when input patterns are applied the states traversed by the augmented FSM corresponds exactly to the states represented by the *k* counters.

7. Once the *Activation Key* is applied, the Augmented FSM traverses a unique path from SN to SF and the original FSM is activated to operate in "Normal Mode".

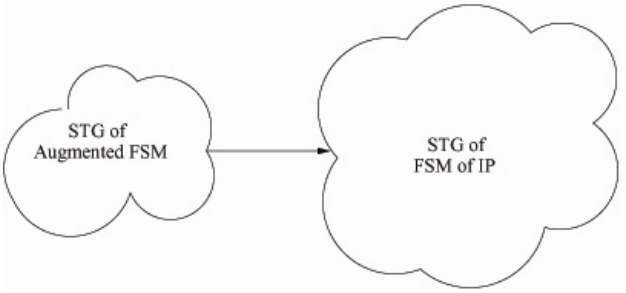

Fig. 2 Augmented Finite State machine

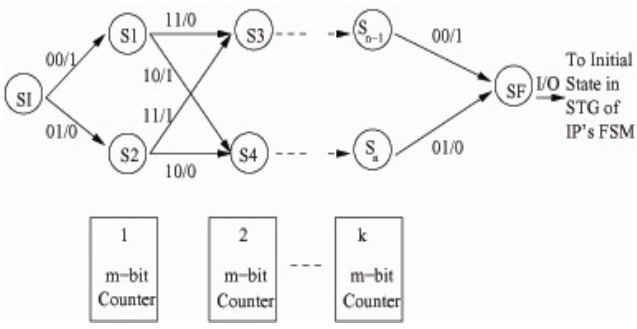

Fig. 3 Augmented FSM with *n*-states and *k m*-bit counters

Each path in the state transition graph of augmented FSM is a valid sequence of input patterns which when applied to the FSM will make the machine traverse the corresponding paths in STG. A path in STG together with counter values uniquely identify a copy of the IC. For a *k* level STG with *s* states at each level the probability *P(s,k)* that a key is requested *n* times is $P(s,k)= (1/s^k)^n$. P(s,k) is probability that two or more (*n*) end-users request the same activation key.

The end-user provides the contents of the counters and the IP provider gives an activation key based on the counter contents. This communication in can be done in various ways for example, by telephone, internet, a wireless port on chip, by software (firmware or operating system) etc. The user can provide the counter contents on the internet based on which the IP provider will provide activation key from a key database. Once an activation key is issued, the end-user is noted, the particular key is marked and database is updated. This process assures that when a malicious activity occurs when someone tries to provide identical counter contents to get an activation key the fraud is detected immediately.

3.2 Design of Augmented FSM

Procedure 1 describes steps to design a FSM of *l-levels* with 2^m *states per level* with many paths in the STG. The *l-levels* in STG of the FSM is the number of *m-bit* counters needed per level with at most 2^m states in each level. However, tt is not necessary to have all the 2^m states at each level.

Procedure 1 Design of STG for Augmented FSM
(1) Create initial state *SI*
(2) Create a final state *SF*
(3) Let *i =1*
(4) **For** *j = 0 to l*
(4.1) **For** $k = 0$ *to* 2^m
(4.1.1) Create state S_i at level ; *i++*
(4.2) **End For**
(4.3) **End For**
(5) From State *SI* assign an outgoing edge to all states at next level.
(6) **For** $i = 0$ *to* $l*2^m$
(6.1) Assign an outgoing edge to all states at next level of S_i
(6.2) Set I/O transition for each edge.
(6.3) **End For**
(7) From State *SF* assign an outgoing edge to initial state of STG of FSM of IP.

Fig. 3 shows an example of an augmented finite state machine with *n* total states and *k m*-bit counters. Once such a finite state machine is designed, we can augment the STG of FSM of the original IP block with this STG and produce the resulting FSM from the STG. For illustration purposes in our experiments we created an STG for augmented FSM and synthesized it to get an estimate on hardware overheard due to augmented FSM. We synthesized two augmented FSM with 16 states (four levels and four states per level) and 80

states (10 levels with 8 states per level). The number of paths and hardware required to implement FSMs was computed and is as shown in table 1. Large number of individual ICs each with a unique key can be produced easily by increasing number of states and levels in the augmented STG. For

States	Paths	Latches	Gates
16	256	5	6
80	2^{30}	7	6

Table 1. Hardware Required for Augmented STG

example consider an STG with 10 levels with 4 states in each level. With our procedure above the total number of paths in this STG is 4^{10} = 1,048,576, and 8^{10} = 2^{30} =1,073,741,824 for a STG with 10 level and 8 state in each level. Corresponding probabilities are P(4,10) = 0.90×10^{-12} and P(8,10) = 8.66×10^{-19}.

Once the Augmented STG is produced the IP provider must provide an efficient database to store all activation keys and a fast retrieval of a particular key when requested by an end user. Since here the activation key to be stored is a sequence of input patterns which are nothing by minterms. This can be easily done by Binary Decision Diagrams (BDD). BDDs are efficient data structures to store and manipulate Boolean functions in canonical form. Each path in a BDD is a mintern and can be extracted efficiently.

4 Conclusion

We presented another alternative to only known hardware metering work [1,2]. The scheme provides a mechanism for designers to secure and control usage of their integrated circuit chips (Intellectual Property). Large number of activation keys can be created by our method and can be efficiently stored for delivery to user. We proposed a activation key based model for metering which is analogous of prevalent software licensing schemes. Activation key enables the identification of individual user at the time when key request is generated and therefore also identifies each individual user of the chip.

References:

[1] F. Koushanfar, G. Qu, M. Potkonjak, "Intellectual Property Metering", *Information Hiding Workshop*, Pittsburgh, PA, April 2001.

[2] F. Koushanfar F.,G. Qu, "Hardware Metering", *Proc. Design Automation Conference*, June 2001, pp. 490-493

[3]A.B.Kahng, J.Lach, W.H. Mangione Smith, S.Mantik, I.L.Markov, M.Potkonjak, P.Tucker, H.Wang, G.Wolfe, "Watermarking Techniques for Intellectual Property Protection", *Proc. of ACM/IEEE 35th Design Automation Conference, 1998*, pp. 776-781, June 1998.

[4]A.B.Kahng, S. Mantik, I.L .Markov, M. Potkonjak, P.Tucker, H.Wang, G.Wolfe, "Robust IP Watermarking Methodologies for Physical Design", *Proc. ACM/IEEE 35th Design Automation Conference*, pp. 782-787, June 1998.

[5] G.Qu, M.Potkonjak, "Analysis of Watermarking Techniques for Graph Coloring Problem", *Proc. Int.Conf. Computer Aided Design*, pp. 190-193, Nov. 1998.

[6]M.Moiz Khan, S. Tragoudas, "Rewiring for Watermarking Digital Circuit Netlists", *IEEE Trans. on Computer Aided Design of Integrated Circuits and Systems*, July 2005.

[7] A.Oliveira, "Techniques for the Creation of Digital Watermarks in Sequential Circuit Designs", in *IEEE Trans. on Computer Aided Design of Integrated Circuits and Systems*, pp. 1101-1117, Sept. 2001.

[8] D.Kirovski, Y.Hwang, M.Potkonjak and J.Cong, "Intellectual Property Protection by Watermarking Combinational Logic Synthesis Solutions", in *Proc. Int. Conf. Computer Aided Design*, pp.194-198, Nov 1998.

[9]Virtual Socket Interface Alliance (http://www.vsia.org).

[10] VSI Alliance, "Intellectual Property Protection White Paper: Schemes,Alternatives and Discussion Version 1", Issued by *Intellectual Property Protection Development Working Group*, Ver.1.1 Released August 2000, Revision 08 Jan 2001.

[11] J.Lach, W.Mangione-Smith, M.Potkonjak, "Signature Hiding Techniques for FPGA Intellectual property Protection", *Proc. Int. Conf. Computer Aided Design*, pp. 186--189, Nov. 1998.

[12] J.Lach, W.H.Mangione-Smith, M.Potkonjak, "Fingerprinting Techniques for Field Programmable Gate Array Intellectual Property Protection", *IEEE Trans. on Computer Aided Design of Integrated Circuits and Systems*, pp. 1253 -1261, Oct. 2001.

[13] J.Lach, W.Mangione Smith, M.Potkonjak, "Fingerprinting Digital Circuits on Programmable Hardware", *Information Hiding Workshop*, pp. 16--31, April 1998.

[14] S.B. Akers, "Binary Decision Diagrams", *IEEE Transactions on Computer*, Vol C-27, pp. 509-516, June 1978.