FSM Watermarks Based on Ordering of Flip Flops

Edward Jung
Southern Polytechnic State Univ.
1100 S. Marietta Pkwy
Marietta, GA 30067
+1-678-915-5546
ejung@spsu.edu

Chie-Cheng Hung
Southern Polytechnic State Univ.
1100 S. Marietta Pkwy
Marietta, GA 30067
+1-678-915-3574
chung@spsu.edu

Seonho Choi
Bowie State University
14000 Jericho Park Road
Bowie, MD 20715
+1-301-860-3967
schoi@cs.bowiestate.edu

ABSTRACT
In this paper, we propose a method for constructing and verifying a watermark of finite state machines based on ordering of flip flops. The underlying idea is to use the flip-flop arrangement information as part of the IP owner's secret. We present simple watermark construction and verification algorithms. We demonstrate the feasibility of the proposed method for a class of FSMs which satisfies certain conditions.

Categories and Subject Descriptors
B.5.1 [Register-Transfer-Level Implementation]: Design---Styles; **K.5.1 [Legal Aspects of Computing]**: Hardware and Software Protection---Proprietary rights

General Terms
Security, Design, Algorithms

Keywords
FSM watermarking, flip-flop arrangement, cyclic property

1. INTRODUCTION
Watermarking is a technique that can be used to securely identify the ownership of the origin of design intellectual properties (IPs). A variety of techniques have been proposed for watermarking different steps of the design process [1][7][9][10][14][15][16][17][19][20]. There are two main classes of approaches. One approach is hardware metering [8], which allows design houses to have post-fabrication control on the produced ICs, and monitor their usage. Another popular approach to IP protection is hardware watermarking [12], in which certain identity information is inserted into behavioral specification or sequential structure of the design. Finite state machines (FSMs) are the backbone of a sequential system design. In this paper, we focus on FSM hardware watermarking.

The central idea of proposed method is based on a decomposition of FSMs. Consequently, this scheme is related to the classical problem of state assignment which had been studied extensively during the 1970s – 1980s. These studied had been conducted for designing and synthesizing sequential circuits with focused goals of reducing circuit delay, areas, power consumptions, and/or of improving testability. However, in this paper, we investigate this problem to see if it can be applicable to protect the design IPs.

This paper introduces a new method of FSM watermarking. Specifically, we focus on proposing a watermarking method which utilizes the flip-flop ordering information as part of IP owner's secret. We present simple watermarking algorithms for constructing and for verifying the watermark. The main contributions are: (1) to the best of our knowledge, it is the first attempt to utilize the flip-flop ordering information as part of IP owner's secret without adding new states or state transitions in order to construct and to verify the watermark, (2) we analyze the proposed scheme for estimating the chance of guessing the watermark without knowing the designer's secret.

Related work and preliminary background are presented in Section 2 and Section 3, respectively. The watermarking algorithms are described in Section 4. Analysis is performed in Section 5. We show the feasibility in Section 6. We conclude in Section 7.

2. RELATED WORK
Most popular traditional approaches include: (a) *FSM watermarking based on Unused Transitions*: the authors in [18] introduced the first IP protection using FSM watermarking. The algorithm is based on extracting the unused transitions in a state transition graph (STG) of the behavioral model. In their solution, extra transitions are added to satisfy the design goals. (b) *FSM watermarking by Property Implanting*: the author in [13] tried to manipulate the STG of the finite state machine to implant the watermark as a property. The property was topological in nature and was defined in terms of visited states ($s_1 \rightarrow s_2 \rightarrow \cdots \rightarrow s_i$). In order to define the topological property, the author added extra states and state transitions in a systematic way to satisfy a specific topological requirement. (c) *FSM watermarking by Integration of Two Distinct FSMs*: the authors in [6] designed a completely new FSM as a watermark and then the watermark FSM was combined with the original FSM to create an integrated composite FSM. Constructing a new watermark FSM was done by adding new states and transitions.

More recently, a FSM watermarking scheme by making the authorship information a non-redundant property of the FSM was proposed in [3]. In this work, the watermark bits were added into the outputs of the existing and free transitions of STG. Another method was proposed in [11]. In this work, a set of edges were added as a dummy entity. This was done by assigning state

RACS'14, October 5–8, 2014, Baltimore, MD, USA.

encoding values. The new edges created by this method were paired with an unused state input combination, and the output was specified as a don't-care condition.

Despite these popular methods which can be effective in protecting IPs of FSMs as demonstrated in these works, these approaches are fundamentally based on expanding the original FSM to an enlarged FSM with new states and/or state transitions.

In this paper, we investigate a new method which does not depend on adding new states and/or state transitions.

3. PRELIMINARY

In this section we provide definitions and assumptions, followed by an illustrative example.

3.1 Definitions and Assumptions

Basic definitions which will be needed at a minimum level in this paper are presented below [4].

Definition 1: FSM = (I, O, S, δ, λ) where I, O, and S are finite, nonempty sets of inputs, outputs, and states, respectively. δ: I × S → S is the state transition function. λ: I × S → O is the output function.

Definition 2: A closed partition π on S of a FSM = (I, O, S, δ, λ) is defined as: if, for every two states which are in the same block of π and any input in I, the next states are in a common block of π.

Definition 3: A partition τ_i on S of a FSM = (I, O, S, δ, λ) is *input-independent*, if for every state and all inputs, the next states are in the same block of τ_i.

Definition 4: A partition τ_i on S of a FSM = (I, O, S, δ, λ) is the *smallest* input-independent, if τ_i contains the maximum number of blocks in it.

Note that the states in *S* are encoded and then realized using a register (i.e., a set of flip flops). The binary values stored in a register can be observed by the user to check the current states of system.

Assumption 1: The internal values of flip flops can be checked by the user, if needed.

Checking the internal values of flip flops can be done using either a partial scan or a full scan.

3.2 Illustrative Example

Table 1 shows an example of a sequential design that is represented in a state table [5]. In D_1, there are six states. When a stimulating external input value is applied on the present state, it is deterministically moving to the next state while generating an external output.

One possible complete flip-flop arrangement is: {(A, 000), (B, 001), (C, 010), (D, 011), (E, 100), (F, 101)}. The corresponding logical equations *using* this particular assignment are: $Y_1 = y_2$, $Y_2 = y_1'y_2'$, $Y_3 = xy_3 + xy_2 + x'y_2'y_3' + y_2y_3$, and $z = xy_3'$, where y_i and Y_i represent the present state and the next state, respectively.

Note that D_1has the following interesting properties. It contains a component that is both (1) independent of the external input, and (2) three states forming a cycle. Consider three combined states {α; β; γ} = {A+B; C+D; E+F}, where "+" is used to denote an operator to combine two states. Then, the transition function of this component is defined as $(\alpha, -) = \beta$, $\delta(\beta, -) = \gamma$, and $\delta(\gamma, -) = \alpha$, where "−" denotes a *don't-care* condition.

Table 1. Sequential Design (D_1)

Present State Q(t)	Next State Q(t+1), Output z	
	Input x = 0	Input x = 1
A	D, 0	C, 1
B	C, 0	D, 0
C	E, 0	F, 1
D	F, 0	F, 0
E	B, 0	A, 1
F	A, 0	B, 0

Putting it all together, the actual flip-flop arrangement can be made as follows: {(A, 000), (B, 001), (C, 010), (D, 011), (E, 100), (F, 101)} with {(α, 00), (β, 01), (γ, 10)} and {(a, 0), (b,1)}. Note that the original states are realized by two internal states: *A* = (α, a), *B* = (α, b), *C* = (β, a), *D* = (β, b), *E* = (γ, a), *F* = (γ, b). For instance, the state "*A*" is realized by two internal states "*α*" and "*a*" using three flip flops.

Figure 1 shows an example of the three flip flops that can store the binary values in three flip flops. For instance, the state C can be realized with the binary values of flip flop = "010" as shown.

y_1	y_2	y_3
0	1	0

Figure 1. Ordering of flip-flop arrangement.

During the verification, the IP owner can verify his/her ownership by checking the three states (α, β, γ) in sequence, irrespective of input signals. During the (p + 1) = 4 time units, a cycle of states should be verifiable, as shown in Figure 2. The verification is done by checking the internal flip flop values.

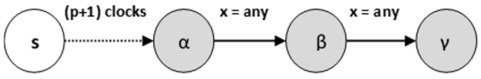

Figure 2. Verification of a cycle of states (with periodicity = 3)

4. WATERMARKING ALGORITHM

In this section, we present the algorithms for creating and verifying a watermarked FSM.

4.1 Watermark Creation (δ-WFSM)

The watermarked FSM is created using a specific property. The specific property chosen as an example is maximal input-independent periodicity representing a cyclic behavior of a sequential FSM. This specific property is denoted by P*. The watermarked FSM, denoted by δ-WFSM**,** is defined with the four parameters.

Procedure Creation ( )
Input: a "n"-state FSM (FSM = (I, O, S, δ, λ), $|S| = n$)
Output: a watermarked FSM (δ -WFSM = (I_w, O_w, S_w, δ_w))
1) Find the *maximal* input-independent periodicity p_{max} from FSM
2) Construct δ -WFSM = (I_w, O_w, S_w, δ_w) as follow:
   a. the input: I_w = I
   b. the output: O_w = O
   c. the set of states: $S_w = \{s_1^*, s_2^*, \dots, s_{p_{max}}^*\}$
   d. the state transition: $\delta(s_1^*, a) = s_2^*$, $\delta(s_2^*, a) = s_3^*$, $\delta(s_3^*, a) = s_4^*$,....., $\delta(s_{p_{max}}^*, a) = s_1^*$ for $\forall a \in I$

Finding p_{max} is important since it will increase the security level of the hidden watermark. Note that δ –WFSM can usually contain a unique characteristic of a cyclic behavior. Step d above is to extract such a cyclic behavior.

Example 4-1: Suppose p_{max} = 3 with three states {1, 2, 3}. Then, the state transition in δ –WFSM can be represented as an *ordered* set of states: {(1, 2, 3), (1, 3, 2), (2, 1, 3), (2, 1, 3), (3, 1, 2), (3, 2, 1)}.

The algorithm of finding p_{max} can be developed as follow. The basic idea is given in [5].

Procedure Max Periodicity ( ):
Input: a "n"-state FSM (FSM = (I, O, S, δ, λ), $|S| = n$)
Output: p_{max}
1) Find a set of a closed partition $\{\pi_1, \pi_2, \dots \dots \pi_t\}$ and a nontrivial input-independent partition τ on S, where $\pi_i \geq \tau$, for $i = 1, 2, \dots t$.
2) Choose the *smallest* closed partition π_{min}
3) The number of blocks in π_{min} is the maximal periodicity p_{max}

The complexity of finding the maximal periodicity is $O(n^2)$ since a pairwise comparisons of each state is needed in Step 1. The overall complexity of creating the watermarked FSM is $O(n^2)$.

Given a set of states $\{1, 2, 3, \dots, p_{max}\}$, the assignment of flip-flop values in a specific order can be used to further reduce attack possibility. Suppose these states are implemented by a register R with m flip flops $R = [R_1, R_2, \dots, R_m]$, where R_i denotes the i-th flip flop and $m = \lceil \log_2 n \rceil$. Note that we need only $r = \lceil \log_2 p_{max} \rceil$ number of flip flops and $m \geq r$. Selection of an *ordered* subset of r flip flops out of m flip flops will suffice. Using the minimum number of flip flops is to ensure that there will be no additional states to be added.

4.2 Watermark Verification

The verification can be done using both the ordered set of states being visited, according to the maximal periodicity (Step 1 in Section 4.1), and the exact location of flip flops.

Procedure Verification ()
Input: $R = [R_1, R_2, \dots, R_m] = [FF_1, FF_2, \dots, FF_m]$
Output:
1) Apply a random input to FSM /* input-independence */
2) Given $R = [R_1, R_2, \dots, R_m] = [FF_1, FF_2, \dots, FF_m]$
   a. select the ordered subset of flip flops $< FF_{a1}, FF_{a2}, \dots, FF_{ak} >$
   b. check the expected ordered set of the flip-flop values
3) If successful, the ownership is considered to be verified.

By performing this verification, the IP owner has verified the following: (1) the correct period of δ-WFSM (e.g., period = 3), (2a) the exact order of cyclic states, and (2b) the specific placement of flip flops and its value. Note that these information are only known to the IP owner.

5. ANAYLSIS

Analyzing any watermarking schemes can be broad since many different types of attacks are possible. Also, there are many well-known requirements for any watermarking solutions [1]. In this section, we focus on the most basic analysis for the proposed scheme.

5.1 Existence of the Property

Based on [5], we provide the results without a formal proof.

Theorem 1: The existence of a closed partition π and a nontrivial input-independent partition τ_i on S in the original FSM = (I, O, S, δ, λ), where $\pi \geq \tau_i$, is a necessary and sufficient condition for the existence of the watermark FSM δ -WFSM = (I_w, O_w, S_w, δ_w). □

Corollary 1: If the number of blocks in an input independent partition τ_i is equal or greater than 2, it is guaranteed to have a nontrivial periodicity of 2 or higher. □

Corollary 2: A periodicity is *maximal* if the number of blocks (or elements) in τ_i is the largest. □

5.2 Guessing the Watermark

In guessing the watermarked hidden information, analysis is performed in two different cases: (*C1*) the known periodicity, (*C2*) both the known periodicity and the known assignment of flip-flops. Table 2 summarizes the parameters used in the analysis.

Table 2. Parameter List

Parameters	Description	Conditions
n	$\|S\|$, the number of states in FSM = (I, O, S, δ, λ)	
m	$\|FF\|$, the minimum number of flip flops	$m = \lceil \log_2 n \rceil$
$C; c$	C, an ordered set of states; $c = \|C\|$ = the cardinality of C	$1 \leq c \leq n$
p^*	The maximal periodicity of δ -WFSM	$p^* = \max_{all\ cycles\ of\ C} \|C\|$ $1 \leq p^* \leq n$
m'	$\|FF'\|$, the minimum number of flip flops	$m' = \lceil \log_2 p^* \rceil$ $m' \leq n$
$\pi(c)$	The number of the ways of arranging $c = \|C\|$ states in order	$\pi(c) = \|C\|!$ $1 \leq \pi(c) \leq n!$
$\Gamma(n, c)$	The number of the ways of arranging c states out of n states	(1)
$G(n, c)$	The number of the ways of arranging both cyclic states and flip flops in order	(2A) and (2B)

Formula (1), (2A) and (2B), respectively, are derived as below.

$$\Gamma(n, c) = (\lceil \log_2 c \rceil)! \times \binom{\lceil \log_2 n \rceil}{\lceil \log_2 c \rceil} \qquad (1)$$

Note that $\Gamma(n, c)$ is determined by encoding p^* states using m' (=$|FF'|$) flip flops out of $m = |FF|$ flip flops as an ordered set.

$G(n,c)$ is determined in terms of $\pi(c)$ and $\Gamma(n,c)$. Thus, $G(n,c) = \pi(c) \times \Gamma(n,c)$.

$$G(n,c) = \pi(c) \times (\lceil \log_2 c \rceil)! \times \binom{\lceil \log_2 n \rceil}{\lceil \log_2 c \rceil} \qquad (2A)$$
$$1 \leq \pi(c) \leq n!$$

The implication of the formula (2A) is that if both the cyclic ordering of states and the ordering of flip-flop arrangement are unknown, the adversary would try this many possible ways to guess the watermark (in the worst case), *provided that* the adversary knows the periodicity.

General Case: In general, however, the maximal periodicity can be kept in secret (by the owner). In this case, the security level increases by a factor of "n" as shown in (2B):

$$G(n,c) = n \times \pi(c) \times (\lceil \log_2 c \rceil)! \times \binom{\lceil \log_2 n \rceil}{\lceil \log_2 c \rceil} \qquad (2B)$$
$$1 \leq \pi(c) \leq n!$$

Lower and Upper Bound: The lower and upper bound of $G(n,c)$ occur when $\pi(c) = 1$ and $\pi(c) = n!$, respectively.

$$G(n,1) = n \qquad (3A)$$

$$G(n,n) = n \times n! \times (\lceil \log_2 n \rceil)! \qquad (3B)$$

Other analysis such as coincidence (i.e., false positive collision) can be done, but we focus on analyzing the degree of difficulty in guessing the watermark without knowing the designer's secret in this paper.

5.3 Limitation

Some FSMs may not satisfy *Theorem 1*. In this case, we should consider a *weaker* condition (e.g., relaxing maximal periodicity) at the cost of lower security (e.g., a higher probability of guessing the watermark).

6. FEASIBILITY

In this section we investigate the feasibility of the proposed method. Note that the main goal is to evaluate the degree of difficulty in guessing the watermark in the *best* scenario (i.e., the attackers should try these many attempts to break the secret watermark information, provided that the given FSM satisfies *Theorem 1*.) In practice, however, some FSMs may not satisfy the conditions.

Table 3 shows the lower and upper bound of $G(n,c)$ derived in (2B) in the previous section. The number of states n is from the FSM benchmarks [2]. Table 4 shows the value of $G(n,c)$ for the more common cases. We considered the relatively low value of c as a function of n. That is, approximately, $c = \left\lceil \frac{n}{k} \right\rceil$ where $k = 2, \ldots, \left\lceil \frac{n}{2} \right\rceil$. Note that we take a more conservative assumption (i.e., not the best scenario) in a sense that (1) the greater the value of periodicity (or the value of c), the more difficult the prediction is, and (2) we assumed the maximal periodicity does not exist beyond $c = \left\lceil \frac{n}{2} \right\rceil$. Also, the subset of the benchmark circuits are selected since the system with small number of states are likely being used in a sequential design of systems (e.g., controller of embedded system).

Table 3. Lower and upper bound of $G(n,c)$

Circuits	FFs	States (n)	Lower Bound ($c = 1$; $\pi(c) = 1$)	Upper Bound ($c = n$; $\pi(c) = n!$)
s27	3	8	8	1935360
s820 s832	5	32	32	1.010422E+39
s1488 s1492 s386 s510	6	64	64	5.8469498E+93
s208	8	256	256	8.854326E+513
s27-n3	9	512	512	6.460615E+1174
S1196 s1238	18	262144	262144	4.45277E+1306615
s991	19	524288	524288	7.115547E+2771033
s382 s400 s444 s526 s526n	21	2097152	2097152	1.576848E+12346668
s15850	597	5.1E+179	5.1E+179	Non-computable
s35932	1728	2^{1728}	2^{1728}	Non-computable

Table 4. $G(n,c)$ for the various values of c

Circuits	States (n)	Periodicity (c)	G(n, c)
s27	8	2	48
		4	1152
s820 s832	32	2	320
		4	15360
		8	7.74144E+7
		16	8.0343513E+16
s1488 s1492 s386 s510	64	2	768
		4	46080
		8	3.096576E+8
		16	4.8206108E+17
		32	1.21250689E+40
s208	256	2	4096
		4	344064
		8	3.46816512E+9
		16	8.99847347E+18
		32	4.5266924E+41
		64	6.5485838E+95
		128	3.980343769E+222

Despite the limited use of the benchmark FSMs, we can make several observations in Table 3. First, for the most of the sequential circuits even with the small number of states (e.g., *s27*), the upper bound $G(n,c)$ is very high, which indicates that the brute-force type guessing work can be realistically infeasible. Second, however, there are some cases that the lower bound $G(n,c)$ is quite low. For instance, the circuits "*s27*" through "*s27-n3*" have the values of lower bound, 8 through 512. Third, as shown in Table 5, the more common cases show that $G(n,c)$ is reasonably high for most of the FSMs.

7. CONCLUSION

We presented a FSM watermarking scheme which can be created by the IP owner utilizing the arrangement of flip flops. The underlying idea was to use the ordering of flip flops as part of designer's secret. Despite the proposed method is not universal, it was illustrated that the FSM watermarking can be done using a simple flip-flop arrangement (similar to state assignments) for a class of FSMs.

8. ACKNOWLEDGMENTS

Our thanks go to the reviewers for the constructive suggestions.

9. REFERENCES

[1] Amr T. Abdel-Hamid, Sofiène Tahar and El Mostapha Aboulhamid. 2004. A survey on IP watermarking techniques. In *Design Automation for Embedded Systems*, Vol. 9, Springer Science+Business Media, Berlin, 211-227. DOI:http://dx.doi.org/10.1007/s10617-005-1395-x

[2] Franc Brglez, David Bryan and Krzysztof Kozminski. 1989. Combinational profiles of sequential benchmark circuits. In *Proceedings of the International Symposium on Circuits and Systems*. IEEE Computer Society, Portland, OR, 1929-1934. DOI:http://dx.doi.org/10.1109/ISCAS.1989.100747

[3] Aijiao Cui, Chip-Hong Chang, S. Tahar, and A. T. Abdel-Hamid. 2011. A Robust FSM Watermarking Scheme for IP Protection of Sequential Circuit Design. *IEEE Trans. Comp.-Aided Des. Integ. Cir. Sys.* 30, 5 (May2011), 678-690.DOI:http://dx.doi.org/10.1109/TCAD.2010.2098131

[4] Juris Hartmanis and R. E Stearns. 1966. *Algebraic Structure Theory of Sequential Machines* (Prentice-Hall International Series in Applied Mathematics). Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

[5] Zvi Kohavi. 1978. *Switching and Finite Automata Theory* (2nd ed.). McGraw-Hill.

[6] Farinaz Koushanfar and Yousra Alkabani. 2010. Provably secure obfuscation of diverse watermarks for sequential circuits. In *Proceedings of the Inter. Symp. on Hardware-Oriented Security and Trust* (HOST '10), 42-47. DOI:http://dx.doi.org/10.1109/HST.2010.5513115

[7] Farinaz Koushanfar, Inki Hong, and Miodrag Potkonjak. 2005. Behavioral synthesis techniques for intellectual property protection. *ACM Trans. Des. Autom. Electron. Syst.* 10, 3 (July 2005), 523-545. DOI:http://doi.acm.org/10.1145/1080334.1080338

[8] Farinaz Koushanfar and Gang Qu. 2001. *Hardware metering*. In *Proceedings of the 38th annual Design Automation Conference* (DAC '01). ACM, New York, NY, USA, 490-493. DOI:http://doi.acm.org/10.1145/378239.378568

[9] Farinaz Koushanfar. 2012. Provably Secure Active IC Metering Techniques for Piracy Avoidance and Digital Rights Management. *IEEE Trans. Info. For. Sec.* Vol. 7, Issue 1 (February 2012), 51-63. DOI:http://dx.doi.org/10.1109/TIFS.2011.2163307

[10] John Lach, William H. Mangione-Smith, and Miodrag Potkonjak. 1999. Robust FPGA intellectual property protection through multiple small watermarks. In *Proceedings of the 36th annual ACM/IEEE Design Automation Conference* (DAC '99), Mary Jane Irwin (Ed.). ACM, New York, NY, USA, 831-836. DOI:http://doi.acm.org/10.1145/309847.310080

[11] Matthew Lewandowski, Richard Meana, Matthew Morrison and Srinivas Katkoori. 2012. A Novel Method for Watermarking Sequential Circuits. In *Proceedings of the IEEE International Symposium on Hardware-Oriented Security and Trust* (HOST'12), San Francisco, CA. 21-24. DOI: http://dx.doi.org/10.1109/HST.2012.6224313

[12] Arlindo L. Oliveira. 1999. Robust techniques for watermarking sequential circuit designs. In *Proceedings of the 36th annual ACM/IEEE Design Automation Conference* (DAC '99), Mary Jane Irwin (Ed.). ACM, New York, NY, USA, 837-842. DOI:http://doi.acm.org/10.1145/309847.310082

[13] Arlindo L Oliveira. 2001. Techniques for the creation of digital watermarks in sequential circuit design. *IEEE Trans. Comp.-Aided Des. Integ. Cir. Sys.* Vol. 20, Issue 9, (September 2001), 1101-1117. DOI: http://dx.doi.org/10.1109/43.945306

[14] Fabien AP Petitcolas, Ross J. Anderson and Markus G. Kuhn. 1999. Information hiding – a survey. In *Proceedings of the IEEE*, Vol. 87, Issue 7 (July 1999), 1062-1078. DOI:http://dx.doi.org/10.1109/5.771065

[15] Gang Qu and Miodrag Potkonjak. 1998. Analysis of watermarking techniques for graph coloring problem. In *Proceedings of the 1998 IEEE/ACM international conference on Computer-aided design* (ICCAD '98). ACM, New York, NY, USA, 190-193. DOI:http://doi.acm.org/10.1145/288548.288607

[16] Pramod Subramanyan, Nestan Tsiskaridze, Wenchao Li, Adrià Gascón, Wei Yang Tan, Ashish Tiwari, Natarajan Shankar, Sanjit A. Seshia and Sharad Malik. 2014. Reverse Engineering Digital Circuits Using Structural and Functional Analyses. To appear in *IEEE Transactions on Emerging Topics in Computing*.

[17] Mohammad Tehranipoor and Cliff Wang (Ed.). 2012. *Introduction to Hardware Security and Trust*. Springer Science+Business Media, LLC. Chapter 17.

[18] Ilhami Torunoglu and Edoardo Charbon. 2000. Watermarking-based copyright protection of sequential functions. *IEEE Journal of Solid-State Circuits.* Vol. 35, Issue 3 (2000), 434-440. DOI:http://dx.doi.org/10.1109/4.826826

[19] J. L. Wong, Gang Qu, and M. Potkonjak. 2006. Optimization-intensive watermarking techniques for decision problems. *IEEE Trans. Comp.-Aided Des. Integ. Cir. Sys.* 23, 1 (November 2006), 119-127. DOI:http://dx.doi.org/10.1109/TCAD.2003.819900

[20] Lin Yuan and Gang Qu. 2004. Information hiding in finite state machine. *In Proceedings of the 6th international conference on Information Hiding* (IH'04), Jessica Fridrich (Ed.). Springer-Verlag, Berlin, Heidelberg, 340-354. DOI:http://dx.doi.org/10.1007/978-3-540-30114-1_24