Computer Science and Information Systems 13(2):609–622 DOI: 10.2298/CSIS160210018L

A Robust Low-overhead Watermarking for Field Authentication of Intellectual Property Cores

Jing Long[1], Dafang Zhang[1], Chen Zuo[1], Jiajun Duan[2], and Weihong Huang[1]

[1] College of Computer Science and Electronic Engineering, Hunan University, Changsha, Hunan, 410082, China
{jlong, dfzhang, chenzuo, whhuang}@hnu.edu.cn
[2] Department of Electrical and Computer Engineering, Lehigh University, 18015, Bethlehem, USA
jid213@lehigh.edu

Abstract. Most of existing field programmable gate array (FPGA) based watermarking algorithms have two primary weaknesses, large overhead and robustness. In this work, a robust low-overhead watermarking algorithm is proposed for intellectual property (IP) protection. The ownership is split into orderly small watermarks. The watermark positions are generated by the watermarks. Location mapping is performed to each position to make it not leak in verification. The real content of embedded watermarks is compressed to be one third of the original number. The configuration of small watermark has left lots of space for correcting. So, it can locate the attackers by checking each watermark. The experimental results illustrate a low-overhead on resource and delay. The efficiency and robustness of the proposed scheme are encouraging.

Keywords: FPGA, IP protection, low-overhead, location mapping, correcting.

1. Introduction

With the development of semiconductor technology, it is possible to integrate multiple predesigned components in a chip. The use of these components, known as intellectual property cores (IPs), has enormous advantages. It offers fast development of a complex system and reduces time to market [1][7]. Meanwhile, the design risk is much lower. Due to these advantages, this technology is prevalent in implementation of complex integrated circuits. But, these IP cores face the issue of being stolen or sold illegally. It may cause great economic damage to the owner of IPs. Therefore, researchers attempt to propose methods to protect these reused IP cores.

By analyzing the reported technologies, there are three kinds of IP protection approaches, i.e., patenting, encryption and watermarking [12]. The use of patenting is a deterrent method to stop attempts for illegal distribution. Encryption has ability to prevent unauthorized use of IP, which belongs to protective method. However, these two technologies have no substantial physical protection to IP itself. Comparatively, digital watermarking is more effective in IP protection. It usually hosts a signature into IP design to declare the copyright or trace infringement [22][34]. Illegal use of IPs could be detected and traced by the embedded signature. IP watermarking based protection is different from privacy protection of multimedia or sensitive data in network [20] , because

the functional correctness must be preserved. Once piracy is suspected, legal IP owner can apply to verify the suspected IP design, the overview of IP watermarking is shown in Fig.1. Successful detection of a valid signature proves the ownership of IP core.

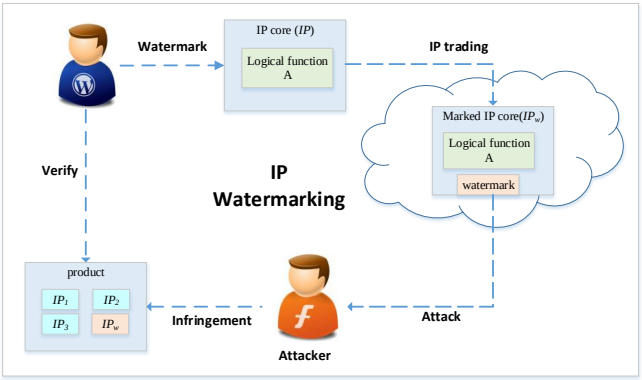


Fig. 1. Overview of IP watermarking

This work is organized as follows. Section 2 classifies existing IP watermarking methods and analyzes their performances. Section 3 gives an overview of the robust low overhead watermarking algorithm. The detailed watermarking algorithm, including watermark preparation, configuration, verification, is introduced in section 4. The experimental results are illustrated and compared in section 5. Section 6 summarizes this work.

2. Related Work

The watermarked IP propagates in semiconductor market instead of the original one. The case is the same to IPs in application specific integrated circuit (ASIC), field programmable gate array (FPGA), etc. But the implementations of watermarks are various in each platform. In recent years, FPGA becomes a prevailing platform due to the reprogrammable feature. The watermarking algorithms for protection of FPGA-based IPs are widely studied. These algorithms are classified into two groups: constraints-based watermarking and additive watermarking [33]. Constraints-based watermarking methods in FPGA mainly place extra constraints for configurable logic blocks (CLBs) in odd/even rows [11], restrict the timing of uncritical path [10], exploit the scan chains [9][8][4] or preserve nets during the procedure of logic synthesis [13]. Other additive methods usually add a signature into the functional IP core. The use of lookup tables is typical in this type of watermarking. For FPGA design, the Xilinx ISE is a common tool with lots of small tools integrated. The synthesis tool usually maps the combinational logic of a design into lookup tables of FPGA and writes the values into netlist of IP design. The content of lookup tables is known after synthesis. It can be parsed and read from the bitfile with Jbits [27]. By considering this, the watermarks can be also realized in this way. If a company is accused to use an unlicensed IP core in a product, the bitfile will be extracted. According

to each watermark positions, the ownership verifier can successfully read out the watermark content and construct a signature. If it is consistent with the declared signature, the ownership of IP core can be proven.

These are effective methods and easy to implement. But large overhead is caused by watermark insertion, especially for the additive methods. Furthermore, the major weakness is lack of robust verification of watermarks from an accused product. For this type of methods, the watermarks are always detected in configuration bitstream of FPGA, which is the most vulnerable to attacks [32]. There are several types of attacks. (1) Cloning. If the bitfile is in plaintext, it can be directly cloned and implemented on other FPGA chips. (2) Tampering. The bitfile can be altered to leak confidential information. The watermarks are also under this type of threat. (3) Physical attacks. The FPGA device may be tampered to realize a desired fault. The sensitive information is extracted by analyzing the correct and fault outputs. Or vicious user uses some sophisticated tools to probe inner structure of the chip and to learn information of the design. In addition, the side-channel information of a design, such as power, computation time, may be illegally used to infer the design secrets. As the watermarks are inserted into design like other configuration data, these attacks should all be considered and addressed to make the watermarks robust. No matter where the watermarks are implemented (HDL level [3][14], behavioral level [23] [5][2][29], netlist level [25] or physical level [31][21][17]), the robust verification is the key issue for authentication of IP ownership. Error correcting code (ECC) is introduced in watermarking for FPGA-based IPs [19][15][16]. There are several ECCs in information security, e.g., hamming code, cyclic code, parity-check code and reed-solomon code. It can make the watermarks resilient to tampering. But the redundancy caused by check codes has limited the ability of correcting since the author should make a compromise between overhead and robustness.

We consider this issue and propose robust low-overhead watermarking technique for protection of FPGA-based IP design. The contributions are stated as follows.

(1) The ownership information is split into a set of orderly watermarks. So, it is unnecessary to consider the order of extract watermarks.

(2) The watermarks are effectively compressed and inserted into design. The watermarks are indicated with information of locations and contents of lookup tables. So, it frees more space for enhancing the error correcting ability. The experiments show the overhead is much lower than that of previous techniques.

(3) The tampering can be located and corrected. It achieves fast location of tampering attacks. The compression based insertion offers great improvements on correcting ability. The experimental results illustrate the effectiveness.

3. An Overview of Robust Low-overhead Watermarking

The mathematical definition of the proposed technique can be described as follows. Let ip and ip_w be original marked IPs. G is the function to generate watermarks. s represents the ownership information of IP owner. κ_1 and κ_2 denote the keys for watermark generation and embedding. w and $E(ip, w, \kappa_2)$ represent watermark and function of watermark insertion. s, κ_1, κ_2 are kept as privacy. w is encoded as $\zeta(w)$.

With these notations, we have $G(s, \kappa_1) \rightarrow w$, $E_w(ip, w, \kappa_2) \rightarrow ip_w$. G maps s into w under the control of κ_1. The function $E_w(ip, w, \kappa_2)$ inserts w into ip. So, ip_w is generated.

The set I_{ip} includes all acceptable sequences. The output of any input $i \in I_{ip}$ is denoted by $O_{ip}(i)$. E_w should make $\forall i \in I_{ip}$, $I_{ip} = I_{ip_w}$ and $O_{ip} = O_{ip_w}$. IP circuit has high requirement on functional correctness, which cannot be altered. ip_w will propagate in semiconductor market instead of ip. If user A integrates and unauthorized in his product p, IP owner can apply to check s from p. He needs to detect $\zeta(w')$ in p with the reserved κ_2. A check function $T(\zeta(w'))$ is designed to check correctness of w'. If $T(\zeta(w'))$ = 1, w' is equal to w. Otherwise, it is tampered. The self-correcting procedure is activated. Under the control of κ_1 , we have $D_w(w', \kappa_1) \rightarrow s'$. Let S_i and F_i respectively denote "success" and "failure" of checking a watermark at location i. $\cap_{i=1}^{n} S_i = 1$ indicates no tampering and $\cup_{i=1}^{n} F_i = 1$ represents there are some watermark locations being tampered. The index i infers the tampered position.

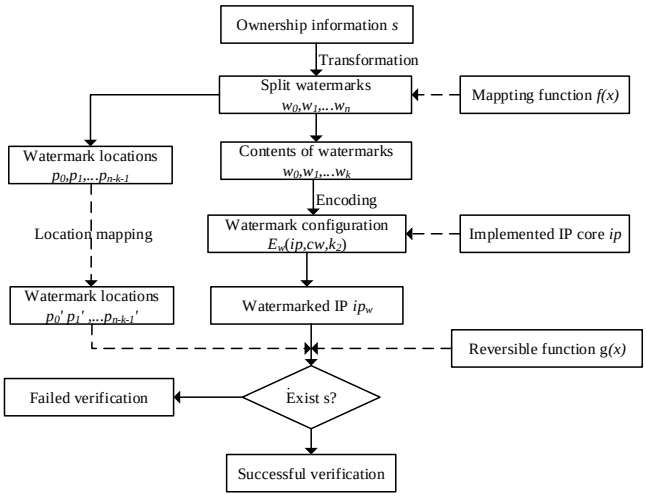


Fig. 2. Overview of the proposed technique

Fig.2 shows an overview of the proposed solution. The ownership information is transformed into a watermark value. It will be split into multiple small watermark values with a mapping function. The encoded split values are then used to generate the watermark positions and real contents of watermarks. The location mapping is performed to cover up real positions of watermarks. In verification, if the ownership information is successfully detected, the ownership will be proven.

4. Detailed Watermarking Scheme

In this section, we divide the proposed technique into three parts by considering the flow of digital watermarking, i.e., preparation, configuration and verification.

4.1. Watermark Preparation and Configuration

For watermark preparation in actual application, the IP owner hopes to insert some meaningful information into IP design, such as his own signature. However, the circuit can only accept the digits of "0" and "1". So, the first thing is to transform the information into digits. A large value can be decomposed into multiple small values by using a mapping function. Let W be the large number. n is the number of split values, i.e., granularity of decomposition. $\{W_1, W_2, ..., W_n\}$ is a set of split values. $f(x)$ represents the mapping function. It is used to split W into $\{W_1, W_2, ..., W_n\}$. So, it is a one-to-many function. However, the reversible function $g(x)$ is many-to-one and can be used to recreate W. In this section, we divide this procedure into the following steps.

Orderly Watermark Splitting Let s be a string representing public ownership information (such as company name, brand) of IP owner and H_s be a function to transform s into a value v_w . The goal of this mechanism is to split w into a set of n orderly watermark values, $n \geq 2$. Empirically, the value of n is determined between 5 and 15 [6]. Concretely, there are three steps to get the final watermarks.

The transformation function H_s first transforms s into a bitstring s_{bs}. It can be represented by ASCII codes and then transformed into a value w. No encryption technique is used in this step. Once adversary finds the correct locations of watermarks, he will possibly remove the watermarks other than infer the ownership information. The watermark value is split into multiple small values as follows.

1. Compute the minimum exponent λ such that the value v_w can be denoted using $n-1$ digits of base 2^λ. The value of 2^λ cannot exceed the value of $n-1$. So, we have (1).

$$\lambda = \lceil N/(n-1) \rceil \tag{1}$$

Here, N is the length of bitstring of v_w.

2. Split the value v_w into digits $v_{w_0}, v_{w_1}, ..., v_{w_{n-2}}$ such that $v_{w_i} \in [0, 2^\lambda)$ and the following formula (2) is satisfied.

$$v_w = \sum_{i=0}^{n-2} 2^{i\lambda} v_{w_i} \tag{2}$$

3. Encode the digits in the multiset $\{w_0, w_1, ..., w_{n-1}\}$ where $w_0 = \lambda - 1$ and $w_i = w_{i-1} + v_{w_{i-1}}$.

For example, we split a watermark value of 3749523692 with =12. The bitstring is "11011111011111010011000011110110" with length of 32. So, we calculate the minimum exponent using $\lambda = \lceil N/(n-1) \rceil = 3$. This produces a list of 3,3,7,3,7,2,3,0,3,5,4 and finally the multiset 2,5,8,15,18,25,27,30,30,33,38,42 on basis of step 3. The base number of 2 can be extended to m.

The elements of multiset $w_0, w_1, ..., w_{n-1}$ will be classified into two subsets: P_w and C_w. Assume k is the number of elements in C_w. Elements $p_{w_0}, p_{w_1}, ..., p_{w_{n-k-1}}$ in P_w are not real embedded watermarks, just watermark indications, or namely, watermark locations. Differently, the proposed technique generates watermark locations from watermarks themselves. And C_w contains the physically contents of configured watermarks $c_{w_0}, c_{w_1}, ..., c_{w_k}$. The elements of P_w and C_w are sensitive information. So, we map P_w

to another $P_w{}'$ to cover up the real locations. Elements in C_w as real contents of configuration watermarks, will be further encoded by encoding function $\zeta(c_{w_i})$, $i \in [0, k-1]$.

Content Encoding Due to the selection range of n , we derive the largest value of configuration bitstring $\max\{L(c_{w_i})|i \in [0, k-1]\}$ to be small. It leaves lots of space to enhance the error correcting bits since the least bits for configuration depends on t inputs of LUTs, denoted by M_{ec}=$pow(2,t) - \max\{B(c_{w_i})|i \in [0, k-1]\}$. $pow(2,t)$ represents 2 to the power of t. The correcting ability is related with the number of check codes N_{ck}, and for each c_{w_i}, $L(c_{w_i})$ may be various. So, $N_{ck} \in [Mec, 2^t - L(c_w)]$. The encoding for c_{w_i}, $i \in [0, k-1]$, can be dynamically adjusted on basis of $L(c_{w_i})$.

Location mapping The set P_w contains real watermark positions and C_w includes contents of configuration watermarks. The elements in P_w are orderly divided by pairs and each pair $(p_{w_i}, p_{w_{i+1}})$ denotes a coordinate of watermark location, $i \in [0, n-k-1]$. k is the number of watermark positions. We represent the coordinate with (x_i, y_i) to make it clear. To cover up the sensitive positions, we have a location mapping M_p : $(p_{w_i}, p_{w_{i+1}}) \rightarrow (p_{w_i}{}', p_{w_{i+1}}{}')$. Actually, the contents of watermarks are still in $(p_{w_i}, p_{w_{i+1}})$. We define M_p as $p_{w_i}{}' = p_{w_i} + \zeta(c_{w_i})$, $p_{w_{i+1}}{}' = \zeta(c_{w_i}) - p_{w_{i+1}}$. So, the mapped positions are kept as privacy. In verification, the IP verifier can calculate real positions $(p_{w_i}, p_{w_{i+1}})$ with $(p_{w_i}{}', p_{w_{i+1}}{}')$ and contents of $\zeta(c_{w_i})$. In this case, $(p_{w_i}, p_{w_{i+1}})$ will not be leaked in verification.

Configuration The encoded $\zeta(c_{w_i})$, $i \in [0, k-1]$, is placed into $(p_{w_i}, p_{w_{i+1}})$ by configuring as a logic function Fc . In bitfile, Fc can be represented by a configuration data Dc , which has no differences with that of the functional one. Furthermore, the mapped $(p_{w_i}{}', p_{w_{i+1}}{}')$, $i \in [0, n-k-1]$, is kept by IP owner as an evidence in verification.

We summarize this procedure in the following pseudo-codes.

```
Watermark_Embedding(s, ip){
   Compute v_w = H_s(s); // H_s(s) is the transforming function
   Set the default value n; // the number of split values
   Compute the minimum exponent of base 2^λ;
   Split v_w into v_{w_i}, i ∈ [0, n-2];
   Set w_0 = λ - 1;
   For each i in [0, n-1]
      Compute w_i = w_i - 1 + v_{w_i};
   End
   Divide into P_w and C_w;
   For each pair in P_w
      Map (p_{w_i}, p_{w_{i+1}}) to (p_{w_i}', p_{w_{i+1}}') ;//location mapping
   End
   For each element in C_w
      Perform encoding ζ(c_{w_i}) ;
      Configure ζ(c_{w_i}) into (p_{w_i}, p_{w_{i+1}}) ;
   End
   Return ip_w;
}
```

4.2. Watermark Verification

In the market of IP trading, vicious users may intentionally crack, copy or forge an IP for his illegal purpose. All these behaviors will bring losses to IP owner, both in economics and efforts. If the unauthorized IP is used in a product, IP owner could buy such a product and verify whether containing his ownership information. The verification procedure requires nothing else from the company of the suspected product, i.e., it is blind detection. IP owner is almost impossible to detect the ownership information s in the accused product if ip_w not exists actually. Otherwise, IP owner could verify s as follows.

In fact, IP owner knows the information about $(p_{w_i}, p_{w_{i+1}})$,$i \in [0, k-1]$. However, he will use $({p_{w_i}}', {p_{w_{i+1}}}')$ to verify the watermarks since it will not reveal the actual positions with watermarks. The verification algorithm will locate the watermarks with $({p_{w_i}}', {p_{w_{i+1}}}')$ and the reserved information. The configuration data Dc is read out, which relates to the encoded blocks $\zeta(c_{w_i})$, $\in [0, k-1]$. During this procedure, the derived $(p_{w_i}, p_{w_{i+1}})$ should be recorded for reconstructing a final ownership information s'. The function $T(\zeta(c_{w_i}))$ is used to check if the i^{th} $\zeta(c_{w_i})$ is tampered. $\cap_{i=0}^{k-1} T(\zeta(c_{w_i}))$ =1 represents no tampering. Otherwise, it checks the error bits and performs correcting. After decoding, a list of c_{w_i} ,$i \in [0, k-1]$, is produced. An orderly permutation A is generated according to the values of p_{w_i} and c_{w_i}. Actually, the elements in A are the extracted split values. Since all the values are orderly, it is unnecessary to record the order of permutation of each value. Here, we come to the reversible splitting. In permutation A, $< p_{w_i}$, $p_{w_{i+1}}$, $c_{w_i} >$, $i \in [0, k-1]$. The index i for each c_{w_i} is mapped to $i+2$. So, A can be represented as $< w_j, w_{j+1}, w_{j+2} >$, $j \in [0, k-3]$. On basis of the splitting, the split values are ${v_{w_i}}' = w_{j+1}$, $i \in [0, n-2]$, $j \in [0, n-3]$. The value ${v_w}'$ can be calculated using equation (1) in above section. Finally, we compute ${H_s}'({v_w}')$ for s'.

The verification function $V(s, s')$ is used to verify the consistence of s and s'. If $V(s, s') = 0$, no infringement occurs. If $V(s, s') = 1$, the ownership information is successfully detected. It proves an unauthorized use of ip_w in accused product. The verification procedure is summarized in the following pseudo-codes.

```
Signature_Verification(ip_w, P_w') {
   Map P_w' to the real position P_w;
   //watermark extraction
   For each element in P_w
      Readout configuration data ζ(c_{w_i}) ;
      Record information of p_{w_i};
   End
   //check and decode
   For each i in [0,k]
      Check the value of T(ζ(c_{w_i})) ;
      If T(ζ(c_{w_i})) =0 then
         Correcting;
      Else
         Decode and compute c_{w_i};
      End
   End
   //watermark reconstruction
   Combine P_w and C_w;
```

```
    Compute v_{w_i}' and s' = H_s'(v_w') ;
    //ownership verification
    If s = s' then
       Infringement;
    Else
       No infringement;
    End
}
```

5. Experimental Results

The experiments implement the proposed scheme in circuits from IWLS 2005 and ISCAS89. The implementation uses Xilinx Virtex II Pro FPGA [26]. In this section, we aim to verify overhead, efficiency of watermark embedding, and robustness. All the experiments are performed on a machine with 3.4GHz Intel(R) core(TM) i3 CPU and 4G memories.

5.1. Overhead Evaluation

We evaluate the overhead in resource and delay using the metrics in [30]. The resource is measured by the usage of LUTs and delay is evaluated by the minimum clock cycle. The evaluation is implemented on circuit *ethernet* . The bistring with length of 192 is inserted into *ethernet* after preparation. The evaluation data is summarized in Table 1.

Table 1. Evaluation data of in original and marked *ethernet* designs

Evaluation	#number of used LUTs	#minimum clock cycle (ns)
Original *ethernet* design	3019	8.663
Marked *ethernet* design	3024	8.663
Overall watermark overhead	0.166%	0%
Average overhead	0.011%	0%

The results show the resource and delay overhead is 0.166% and 0%. The average resource overhead caused by each watermark is 0.011%, which is much lower. We compare the resource overhead with methods in [24] and [18]. Fig.3 describes the comparing results. The usage of LUT in [24] increases linearly since the watermarks are generated by directly dividing. The method in [18] has large overhead increase when the bitstring is less, more than that of [24]. But with the increase of bistring, it causes less increase than [24]. Our scheme obviously causes lower resource of LUT than the comparative methods.

5.2. Efficiency of Watermark Embedding

We use the rate of watermark embedding in [28] in evaluation. This metric indicates the ratio of real embedded watermarks which cause impact on overhead in all watermarks.

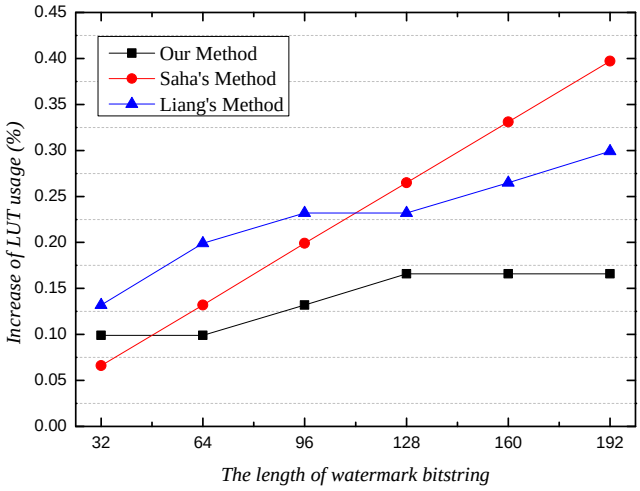


Fig. 3. Compare the usage of LUTs to the methods in [24] and [18]

Let L_s be the number of bitstring s_{bs}. The minimum groups of s_{bs} should be $N_g = L_s/L(\max\{Dc\})$ in order to successfully embed s_{bs}. $L(\max\{Dc\})$ is the length of the maximum value of Dc. n is the number of splitting values.

In the proposed technique, we split a large watermark value into n multiple small values. The number of watermarks is extended to n times of the original one. But the compressed watermark embedding divides the n watermarks into groups by three units. In each group, there is only one real embedded watermark. So, the number of real watermarks N_w in ip_w is $n/3$. Due to the constraint on n , it will not increase with L_s. Fatherly, we define the rate of watermark compression r_w to evaluate the embedded bitstring, denoted as equation (3).

$$r_w = N_w/N_g = nL(\max\{Dc\})/3L_s \tag{3}$$

Generally, r_w is expected to be less than 1 since it indicates less extra resource overhead. But it cannot be overly small and the embedded watermarks should provide enough credibility as court evidence. If L_s is a constant, the smaller value of n achieves a lower rate of watermark embedding.

Table 2. Efficiency of watermark embedding

IP Circuits	FPGA device	L_s	N_w	r_w
s9234_1	xc2vp2-6fg256	32	3	1.500
s5378	xc2vp2-6fg256	64	3	0.750
mem_ctrl	xc2vp20-6ff896	96	4	0.667
ethernet	xc2vp7-6fg456	128	5	0.625
b22-1	xc2vp7-6fg456	160	5	0.500
pref_des	xc2vp20-6ff896	192	5	0.417

We choose circuits from IWLS 2005 and ISCAS'89 to evaluate the efficiency of watermark embedding. The circuits are implemented on the most suitable FPGA device. We list the experimental results in Table 2. We verify the efficiency when "L_s" has the length of "32", "64", "96", "128", "160" and "192". N_w is the number of actual inserted contents. r_w is the rate of watermark embedding. The next column shows the increase of lookup tables in marked circuits. We insert bitstring with various length on basis of the circuit scale. The value of r_w is almost less than "1" except the result of "s9234_1". It demonstrates that the proposed scheme performs well on embedding efficiency. Namely, more ownership information can be indicated using the proposed scheme if the extra overhead is constant. The value of "1.5" illustrates an expansion of watermarks. Actually, it causes a redundancy. But the results show r_w decreases with the increase of watermark bitstring.

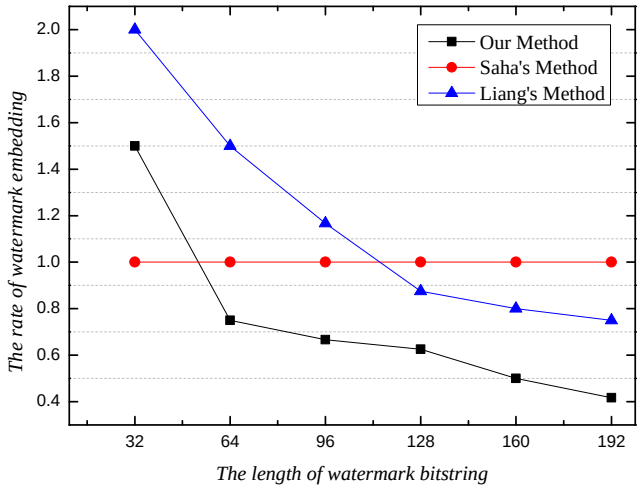


Fig. 4. Comparison of rate of watermark embedding

We compare the efficiency of watermark embedding of our method with that of methods in [24] and [18]. Each lookup table in the given devices can store 16 bits at most [33]. The method in [24] divides the bistring by the maximum stored number. The method of [18] shares the watermark into multiple values by using threshold theory. The real embedded bitstring is compressed but the number of watermark blocks is not reduced. The results are shown in Fig.4. The rate of watermark embedding is unchanged and keeps a value of "1" in [24]. Because all the watermarks groups are inserted into design. On basis of equation (3), the value of is always "1". While in [18], the sharing method actually expands the inserted contents when the length of watermark bitstring is almost less than 112 bits. When the bistring exceeds the value, the rate of watermark embedding decreases below "1". The reason is caused by limitation of threshold value. In our method, the rate of watermark embedding shows a downtrend and the watermark expansion occurs when the watermark bistring is less than nearly 48. With the growth of bistring, it is much less

than that of method in [18]. In this view, we could indicate more contents of ownership information when the caused resource overhead is the same to that in comparative methods.

5.3. Watermark Strength and Robustness

In this section, we aim to evaluate watermark strength and robustness against tampering attacks. r_f is the key metric to verify the strength of a watermarking method. It can be evaluated by using equation (4).

$$r_f = \frac{1}{p^k C_m^k} \tag{4}$$

k and m denote the number of inserted watermarks and the usage of LUT, respectively. C_m^k represents the number of possible permutations to get the correct watermark positions. The evaluation of watermark strength is shown in Table 3.

Table 3. Evaluation of watermark strength

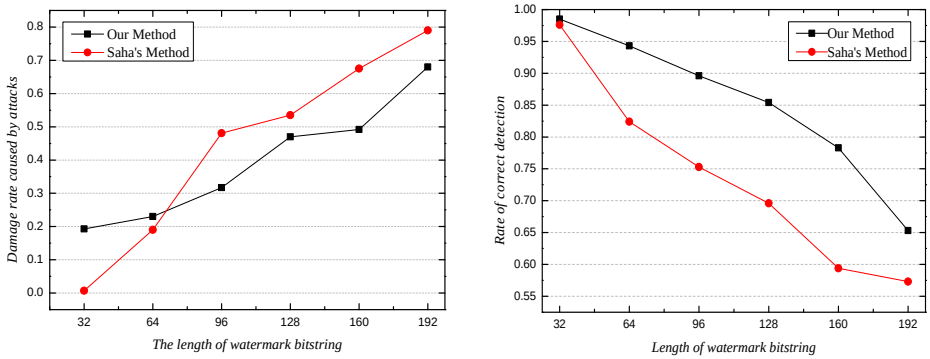
IP Circuits	The usage of LUTs m	Length of watermark L	False alarm rate r_f
s9234_1	336	32	5.670E-22
s5378	412	64	3.071E-22
mem_ctrl	1604	96	1.973E-31
ethernet	3019	128	3.971E-40
b22-1	7469	160	4.276E-42
pref_des	10492	192	7.815E-43

The robustness against attacks is measured by r_c .The experiment is conducted on circuit *ethernet* with Xilinx xc2vp7-6fg456 FPGA. The damage rate r_d caused by attacks and the rate of correct detection r_c are evaluated and the result is shown in Fig.5.

Fig.5 (a) shows the curves in both methods are uptrend. The proposed method has suffered less damage from the attacks. The rate of correct detection is evaluated under the damage rate in (a). When the damage rate is less, both the methods show a high rate of correct detection, as shown in Fig.5 (b). With the growth of watermark bitstring, the curve in method [24] shows a sharp downtrend. But the curve in our method is gentler. The reason is analyzed that the compression frees more space for correcting, which brings better performance on rate of correct detection. By comparing, it is superior to that of the method in [24].

6. Conclusions

In this work, we present a low-overhead watermarking technique by compressing real inserted contents of watermarks. The ownership information is split into multiple orderly small values. One part of watermarks is regarded as watermark locations and the remaining is real embedded watermarks. To enhance the ability against attacks, the real embedded contents will be further encoded to enhance the correcting ability. Only the



(a) Damage rate caused by attacks (b) Rate of correct detection

Fig. 5. Robustness evaluation on circuit *ethernet*

later watermarks may affect the performance of IP design. The compressed mechanism frees more space to enhance the correcting ability. Orderly watermarks are helpful to watermark reconstruction, not considering the out-of-order problem. The experiments show the physical resource overhead decreases by 2/3 and the ability against tampering is encouraging.

Acknowledgement. This work is supported by the National Science Foundation of China (No. 61472130, 61572188), National Basic Research Program of China (973) (No. 2012CB315805), the Scientific Research Fund of Hunan Provincial Education Department (No. 14A047), the China Postdoctoral Science Foundation funded project (No. 140778) and Scientific Research Fund of Hunan Provincial Education Department (No.10C0689).

References

1. Abdel-Hamid, A.T., Tahar, S., Abboulhamid, E.M.: A survey on ip watermarking techniques. International Journal on Design Automation for Embedded Systems 9(3), 211–227 (2005)
2. Abdel-Hamid, A.T., Tahar, S., Aboulhamid, E.M.: A public-key watermarking technique for ip designs. In: Design, Automation and Test in Europe. pp. 330–335. IEEE, Munich, Germany (2005)
3. Castillo, E., Meyer-Baese, U.and Garcia, A.: Ipp @hdl: Efficient intellectual property protection scheme for ip cores. IEEE Transactions on Very Large Scale Integration Systems 15(15), 578–591 (2007)
4. Chang, C.H., Cui, A.: Synthesis-for-testability watermarking for field authentication of vlsi intellectual property. IEEE Transactions on Circuits and Systems I: Regular Papers 57(7), 1618–1630 (2010)
5. Chang, C., Zhang, L.: A blind dynamic fingerprinting technique for sequential circuit intellectual property protection. IEEE transactions on computer-aided design and integrated circuits and systems 33(1), 76–89 (2014)
6. Collberg, C., Huntwork, A., Carter, E., Townsend, G.: Graph theoretic software watermarks: Implementation, analysis, and attacks. In: Information Hiding. pp. 192–207. Springer Verlag, Berlin (2004)

7. Cui, A., Chang, C.H., Tahar, S.: Ip watermarking using incremental technology mapping at logic synthesis level. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 27(9), 1565–1570 (2008)
8. Cui, A., Chang, C.H., Tahar, S.: A robust fsm watermarking scheme for ip protection of sequential circuit design. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 30(5), 678–690 (2011)
9. Cui, A., Qu, G., Zhang, Y.: Ultra-low overhead dynamic watermarking on scan design for hard ip protection. IEEE Transactions on Information Forensics and Security 10(11), 2298–2313 (2015)
10. Jain, A.K., Yuan, L., Pari, P.R., Qu, G.: Zero overhead watermarking technique for fpga designs. In: In Proceedings of Great Lakes Symposium on VLSI. pp. 147–152. ACM, New York, NY, USA (2003)
11. Kahng, A.B., Mantik, S., Markov, I.L., Potkonjak, M., Tucker, P., Wang, H., Wolfe, G.: Robust ip watermarking methodologies for physical design. In: Design Automation Conference. pp. 782–787. IEEE, San Francisco, CA, USA (1998)
12. Kahng, A., Lach, J., Mangione-Smith, W.: Constraints-based watermarking techniques for design ip protection. IEEE transactions on computer-aided design of integrated circuits and systems 20(10), 1236–1252 (2001)
13. Kirovski, D., Hwang, Y.Y., Potkonjak, M., Cong, J.: Intellectual property protection by watermarking combinational logic synthesis solutions. In: International Conference on Computer Aided Design. pp. 194–198. IEEE, San Jose, CA, USA (1998)
14. Kufel, J., Wilson, P.R.: Sequence-aware watermark design for soft ip embedded processor. IEEE Transactions on Very Large Scale Integration Systems 24(1), 276–289 (2015)
15. Lach, J., Mangione-Smith, W.H., Potkonjak, M.: Signature hiding techniques for fpga intellectual property protection. In: IEEE/ACM International Conference on Computer-Aided Design. pp. 186–189. ACM, New York (1998)
16. Lach, J., Mangione-Smith, W.H., Potkonjak, M.: Fingerprinting techniques for field-programmable gate array intellectual property protection. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 20(10), 1253–1261 (2001)
17. Liang, W., Wu, K., Xie, Y., Duan, J.: Tdcm: An ip watermarking algorithm based on two dimensional chaotic mapping. Computer Science and Information System 12(2), 823–841 (2015)
18. Liang, W., Zhang, D., You, Z.: A digital ip watermarking scheme based on self-recovery secret information. Journal of Computational and Theoretical Nanoscience 11(11), 1–5 (2014)
19. Long, J., Zhang, D., Liang, W., Bi, X.: Signature restoration for enhancing robustness of fpga ip designs. International Journal of Information Security and Privacy 9(3), 41–56 (2015)
20. Ma, T., Zhou, J., Tang, M., Tian, Y., Al-Dhelaan, A., Al-Rodhaan, M., Lee, S.: Social network and tag sources based augmenting collaborative recommender system. IEICE transactions on Information and Systems E98-D(4), 902–910 (2015)
21. Nie, T., Li, Y., Zhou, L., Toyonaga, M.: A multilevel fingerprinting method for fpga ip protection. In: IEEE International Symposium on Circuits and Systems. pp. 1789–1792. IEEE, Beijing (2013)
22. Qu, G.: Publicly detectable watermarking for intellectual property authentication in vlsi design. IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems 21(11), 1363–1368 (2002)
23. Saha, D., Sur-Kolay, S.: A unied approach for ip protection across design phases in a packaged chip. In: Proceedings of 2010 23rd International Conference on VLSI Design. pp. 105–110. IEEE, Bangalore (2010)
24. Saha, D., Sur-Kolay, S.: Secure public verification of ip marks in fpga design through a zero-knowledge protocol. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 20(10), 1749–1757 (2012)

25. Schmid, M., Ziener, D., Teich, J.: Netlist-level ip protection by watermarking for lut-based fpgas. In: Proceedings of the 2008 International Conference on Field-Programmable Technology. pp. 209–216. IEEE, Tapei (2008)
26. Xilinx: Virtex-ii pro and virtex-ii pro x platform fpgas: Complete data sheet. Xilinx (2011), [Online].Available:http://www.xilinx.com/support/documentation/data_sheets/ds083.pdf. (current October 2015)
27. Xilinx, I.: Jbits sdk, [Online]. Available: www.xilinx.com/labs/projects/jbits/
28. Xu, J., Long, J., Peng, L.: A high-capability scattered ip watermarking algorithm. Journal of Computer Research and Development 50(11), 2389–2396 (2013)
29. Zhang, L., C.C.H.: State encoding watermarking for field authentication of sequential circuit intellectual property. In: IEEE International Symposium on Circuits and Systems. pp. 3013–3016. IEEE, Seoul, Korea (2012)
30. Zhang, J., Lin, Y.: A chaotic-based public verifiable fpga ip watermark detection. Scientia Sinica 43(9), 1096–1110 (2013)
31. Zhang, J., Lin, Y., Che, W.: Watermarking fpga bitfile for intellectual property protection. Radioengineering 21(2), 764–771 (2012)
32. Zhang, L., Chang, C.: A pragmatic per-device licensing scheme for hardware ip cores on sram-based fpgas. IEEE Transactions on Information Forensics and Security 9(11), 1893–1905 (2014)
33. Ziener, D., Amus, S., Teich, J.: Identifying fpga ip-cores based on lookup table content analysis. In: In Proceedings of 16th International Conference on Field Programmable Logic and Applications. pp. 481–486. IEEE, Madrid, Spain (2006)
34. Ziener, D., Teich, J.: Power signature watermarking of ip cores for fpgas. Journal of Signal Processing Systems 51(1), 123–136 (2008)

Jing Long received her MS degree in College of Computer Science and Engineering, from Hunan University of Science and Technology, China, in 2012. Currently, she is a PhD student in College of Computer Science and Electronic Engineering, Hunan University, China. Her current research interests include hardware security and IP protection.

Dafang Zhang, corresponding author of this work, received his PHD degree in applied mathematics from Hunan University in 1997. He is a professor in college of computer science and Electronic Engineering, Hunan University, China. His research interests include dependable systems/networks, network security, network measurement, hardware security and IP protection.

Zuo Chen received his PHD degree in College of Computer Science and Engineering, Hunan University, China, in 2007. His current research interests include computer network, secure embedded system, parallel computing and data mining.

Jiajun Duan is a research assistant and a PhD in Electrical Engineering the Lehigh University, USA, in 2014. His current research interests include real-time embedded systems, intellectual property protection, and field programmable gate arrays.

Weihong Huang is a lecturer in Hunan University of Science and Technology. His is a PhD student in College of Computer Science and Electronic Engineering, Hunan University, China. His current research interests include embedded systems, intellectual property protection, and field programmable gate arrays.

Received: February 10, 2016; Accepted: May 22, 2016.