Pranjali Akkawar et al., Int.J.Computer Technology & Applications, Vol 5 (5), 1614-1619

Netlist Level Hardware IP Protection by Obfuscation Technique using Key-FSM

Pranjali Akkawar
Electronics & Telecomm. Dept.
Cummins College of Engineering for Women
Pune, India
pva.pranjali@gmail.com

P. V. Srinivas Shastry
Electronics & Telecomm. Dept.
Cummins College of Engineering for Women
Pune, India
pvs.shastry@cumminscollege.in

Abstract

Now-a-days, Complexity & size of electronic devices is increasing which leads to design reuse. Re-use of design is beneficial but it increases the risk of security threats like IP cloning, reverse engineering, tampering, etc. In this paper one obfuscation method is proposed to overcome such security threats at netlist level. The steps to implement this method are simple & common but each design is uniquely obfuscated such that it proves to be efficient in securing IP core. This method is implemented on five ISCAS-89 benchmark circuits & efficiency of it is also calculated. Here IP security is achieved at cost of approximately 10% area & power overhead for circuit. For small circuit whose gate count is less than 100, overhead is more.

1. Introduction

The Modern System on Chip (SoC) design is largely based on effective re-use of existing design components. By re-using components of existing designs, it will be possible to build extremely large systems in short time. Re-usable blocks can either be taken from internal libraries or purchased from third parties. In the latter case, re-usable blocks are often called intellectual property components (IPs) [1]. All parties involved in the IC design flow are vulnerable to different forms of IP infringement which can result in loss of revenue and market share. Hence, there is a critical need of a piracy-proof design flow that equally benefits the IP vendor, the chip designer, as well as the system designer. At different stages of IC processing, IP designs are shared in different forms. At RTL level, IP is shared in soft form. Whereas the IP core at netlist level form is known as Firm IP. Firm IP is distributed to vendors/fabricators when all modules of designs are integrated into one design & process is ready to implement that design. The adversary can reverse-engineer or can make cloning of this design. GDS-II form of IP core is known as Hard IP. While sharing IP cores, it becomes venerable to various forms of security threats like IP cloning, reverse-engineering, and tampering, etc. In this paper we are trying to solve issues related to firm IP core i.e IP core at netlist level by inserting FSM in existing design which controls the operating conditions of circuit.

In this paper, we propose key based obfuscation method at netlist level. One FSM is used as key to unlock normal operation of design. We have implemented it for ISCAS-89 benchmark circuits and its efficiency is calculated. The obfuscation efficiency & results of this method is presented in further sections.

The rest of this paper is arranged as follows. Section II is a summary of work which has been published by previous authors in IP protection domain. Section III contains type of obfuscation technique. Section IV gives our approach of netlist level obfuscation with steps to implement our method of key based obfuscation. In section VI, the derivation for obfuscation efficiency is proposed. Section VII & VIII are results & conclusion respectively.

2. Related Work

The impact of IP threats at different stages of SoC design flow is tried to reduce by approach like encryption, Obfuscation, watermarking, etc. Encryption protects design by making it accessible to only valid users who possess the key to decrypt the design. This can prevent IP core from probable cloning attacks but the reverse engineering attacks persists. Watermarking approach is embedding a signature or fingerprint in design. It protects design from being stolen as one can claim the design to be his own by giving proof of signature or fingerprint. But it cannot protect design from reverse engineering or cloning. Obfuscation is a technique that transforms an application or a design into one that is functionally equivalent to the original but is significantly more difficult to reverse-engineer [2]. So it protects design from obvious simulation & structural attacks. In recent time, certain work has been published related to IP core protection by encryption & watermarking method. But very less work is published for obfuscation of hardware IP protection. The work which has been published by previous authors is summarized below:

At net-list level obfuscation is achieved by systematically modifying state transition function & internal-circuit structure in such a way that circuit operates in normal mode only upon
application of a predefined enabling sequence of patterns at primary inputs [3]. At RTL level, obfuscation is achieved by transforming RTL core into control & data flow graph & then integrating a mode-control FSM in it, such that design will function normally only after applying specific input sequence [4]. In passive obfuscation technique, the design description is modified in soft form such that it becomes difficult for human reader to understand the functionality of design. Here code is obfuscated by removing comments & modifying internal net names, so that reader is unable to understand its functionality [5]. Also the source code is obfuscated by removing comments & modifying internal net names. Additionally the modified net names are substituted by simple string such that the functionality remains unchanged but description is not intelligible to reader [6]. In passive obfuscation technique, one can easily use available design code as black box hence it is not that much effective to stop intruders as functionality of design is not affected in this technique.

The encryption approaches are as follows. One of the encryption techniques embeds encryption keys into circuit & use a new protocol for chip activation [7]. In other approach, encryption is achieved by locking/unlocking the IC or IP core remotely [8]. Also encryption is done by using Physical Unclonable Functions (PUF) to store keys in secure way [9]. Another approach describes the combination of bit-stream public key encryption & a key extracted from PUF works for FPGA IPP [10]. The dynamic IPP technique propose use of both public-key & symmetric cryptography without burden of the FPGA overhead for public key cryptography [11] [12].

For watermarking approach, the signature is added to functional core, by using empty look-up tables in FPGA [13]. IPP@HDL[14] proposes signature spreading through memory structures or combinational logic included in HDL design. Watermarking can also be done by making each design slightly different from other [15].

3. Netlist Level Obfuscation Technique

Obfuscation can be done in enormous way, at different stages in IC design with varied techniques. One way of obfuscating a design is modifying the circuit description in a soft form [5] (e.g. syntactic changes), such that it becomes difficult for a human reader to understand the functionality of the circuit. A major shortcoming of this approach is that it does not modify the functionality of a circuit, and hence cannot prevent possible usage of an IP as black-box in a design.

The other type of obfuscation is key based approach which alters functionality such that design operates in normal mode only after satisfying given constraints. In key based obfuscation technique, adversary will be unable to get correct output from give design unless & until he applies required input pattern in defined sequence i.e. specific sequence acts as key to normally function the circuit.

The previous attempt of obfuscation at netlist level [3] is done by using existing FSM of original circuit. The FSM in original circuit is first derived and then obfuscated by adding gates to it. To do obfuscation by this approach, designer has to reveal his FSM in design. While doing this, the details of FSM can be leak out to adversary. Hence adversary can easily find out desired response of circuit from available information. In contrast to this approach, our method of obfuscation does not require FSM information. We insert a general FSM directly in netlist form of original design. For this FSM, input bits & sequence steps are decided based on circuit size & structure & required efficiency of obfuscation. Also the steps to implement our method are general for any circuit. Even though the steps are general, but each design is uniquely obfuscated such that it proves to be efficient in securing IP core. The obvious attack by hackers to understand the functionality of design is by simulation based reversed engineering or by structural analysis of net-list to identify original design from obfuscated design. Hence, our approach tries to achieve a phenomenon where simulation will never match the desired response of circuit for the maximum possible input vectors, as well as original structural is not recognized for maximum possible circuit nodes. Our key based approach of obfuscation at net-list level is implemented in further section.

4. Our Approach of Key Based Obfuscation

In our approach, we select inputs for FSM partially from available primary inputs and internal gates of original design. The number of inputs for FSM from primary inputs and total states of FSM can be decided according to required efficiency of obfuscation & according to size & structure of circuit. As one of the inputs is hosted on existing circuit, it becomes difficult to recognize & remove FSM circuit separately. After proper execution of FSM, the enable signal becomes active and circuit operates in normal mode of operation.

The state transition graph for key FSM is given in figure 1. It can be observed from figure 1 that only after application of desired input in given sequence will make transition from one state to another. As combination of this FSM acts as secret key to open door for normal functioning, hence this FSM is named as key FSM. The input other then desired will return to first state. In our design, we have considered first three bits of FSM input from primary inputs and last bit is hosted from one of the node from original design. And enable signal is obtained only after achievement of final state When enable signal becomes active, circuit operates in normal mode. When enable signal is not active then circuit operates in obfuscated mode. Hence enable signal generated from FSM controls the functionality of circuit.

We have proposed & implemented obfuscation at net-list level by inserting additional gates along with existing schematic. The node from which input for FSM is hosted should have high fan-out cone. So any attempt to remove FSM
& hence this node will adversely affect the circuit functionality. The additional gates are required to combine enable signal with circuit nodes such that it will control output depending on enable signal. Existing node can also be use as host and in that case area overhead is also very less.

Obvious way of hackers to understand the functionality of design by is simulation based reversed engineering or by structural analysis of net-list to identify original design from obfuscated design. Hence, our approach tries to achieve a phenomenon where simulation will never match the desired response of circuit for the maximum possible input vectors, as well as original structural is not recognized for maximum possible circuit nodes. Also to increase the effectiveness of our method, we can embed watermark in this circuit. The input pattern sequence as well as number of bits can be decided by designer. Hence it can be used as signature of designer. Also output is obfuscated until enable signal is active. One can modify circuit such that output will be our signature bit until enable signal is inactive. As soon as enable signal is active circuit will operate in normal mode and gives output accordingly.

5. IMPLEMENTATION STEPS OF OBFUSCATION AT NETLIST LEVEL

The steps to implement above mentioned approach are as follows:

- Designer will decide the pattern of input sequence as well as number of bits of key FSM depending on analysis of circuit and required efficiency of obfuscation.
- Out of total number of input bits for FSM, designer will decide which bits will be hosted in circuit and which will take input directly from primary inputs.
- Apply desired sequence at primary inputs and check output of number of nodes in circuit.
- Select appropriate node which gives desired output that can act as input bit for FSM operation. Figure 3 & 4 shows one example circuit s526 where node NOR2_5 in shown before and after obfuscation.

**Fig. 1.** Key FSM diagram

The representation of process flow to implement our approach is as shown in figure 2 given below. The original net-list is combined with a key FSM. For normal operation, FSM should give enable signal then only circuit will operate in normal mode. This is achieved by combining enable signal with selected nodes which directly affects the state of output. This combine circuit of FSM & original circuit are then again synthesized & optimized. Thus the synthesized circuit will work as obfuscated circuit to compete any attacks from adversary.
Fig. 4. Obfuscated s526 circuit nodes
- Merge FSM netlist with original netlist taking care of host bits & nodes in design.
- Simulate and check that enable signal is working as desired
- Find critical nodes which can effect directly to output & combine it with enable signal using existing node or adding AND gates. Figure-5 & 6 shows example original s526 circuit & additional gates inserted in it.

Fig. 5. Selected nodes of original s526 circuit

Fig. 6. Obfuscated s526 circuit with additional gates
- Simulate and check for obfuscated & normal functionality of circuit. Fig.-7 & 8 shows simulation of original and obfuscated s526 circuit.

Fig. 7. Original s526 circuit simulation
- Synthesis & check the overheads are within limits or not. If not then repeat above procedure with optimize overheads.

Fig. 8. Obfuscated s526 circuit simulation

6. Obfuscation Efficiency

We now derive an equation to find the efficiency of proposed obfuscation method. Adversary will try number of input combinations to get desired response. Our obfuscation technique tries to give maximum number of response which does not match our sequence for these random trials. Design will function only when key combination in proper sequence is applied at primary input. Thus more is the failure response for random trials more will be the efficiency. Let $p$ be primary inputs to original net-list. Let $p_1$ be subset of primary input which is input to key FSM & $p_2$ is input hosted from internal nodes. Let the number of states of key FSM be $n$ & total number of nodes is $z$ then:

$P_1$ inputs are selected randomly from available $p$ inputs. Hence the probability of selecting $p_1$ inputs from available $p$ inputs is given as $2^p$.

As $p_2$ is number of FSM input taken from internal nodes. Let $z$ be total nodes then chances of selecting 1 nodes form to total $z$ nodes minus 1 correct combination which is given in equation-2.

$$F_1 = 2^p - 2^p - p_1$$

We take 3 inputs of FSM from primary inputs. As correct combination is only 1 in $2^3$ chances for 3 inputs, hence number of falling pattern is given in equation-1. 

$$F_1 = 2^p - 2^p - p_1$$

As $p_2$ is number of FSM input taken from internal nodes. Let $z$ be total nodes then chances of selecting 1 nodes form to total $z$ nodes minus 1 correct combination which is given in equation-2.
$F_2 = \frac{z^l}{p^2!} - 1$  \hspace{1cm} (2)

Next, the input $p$ is applied to key FSM. Number of states of this FSM is $n$ & input combination is of $p$ bits. The total combination for this key FSM is given by: $[n \times (2^p)]$

Out of these combinations, only one is correct which completes state transition flow and generates enable signal. Hence probability of obtaining correct combination is given in equation-3.

$F_3 = [n \times (2^p)] - 1$  \hspace{1cm} (3)

Probability of obfuscation can also be estimated from falling pattern. Hence the falling pattern is given by equation-4 below:

$F_{ob} = (2^n - 2^n - p^1) \times \frac{z^l}{p^2!} - 1) \times (n \times 2^p - 1)$  \hspace{1cm} (4)

The falling pattern for ISCAS-89 benchmark circuit is calculated and given in table-1.

**TABLE I. COMPARISON OF FAILURE PATTERN FOR ISCAS-89 BENCHMARK CIRCUITS**

<table>
<thead>
<tr>
<th>ISCAS-89 BENCHMARK CIRCUIT</th>
<th>Our Approach</th>
<th>HARPOON [3l]</th>
</tr>
</thead>
<tbody>
<tr>
<td>S526</td>
<td>21952</td>
<td>1842</td>
</tr>
<tr>
<td>S838</td>
<td>6.6x10^13</td>
<td>8096</td>
</tr>
<tr>
<td>S1196</td>
<td>124321792</td>
<td>9408</td>
</tr>
<tr>
<td>S1488</td>
<td>229248</td>
<td>1422</td>
</tr>
<tr>
<td>S5378</td>
<td>6.9x10^14</td>
<td>56638</td>
</tr>
</tbody>
</table>

Table-1 shows obfuscation efficiency with area overhead adjusted for 15-20% target. The functional obfuscation efficiency was estimated by number of Falling patterns. Here our approach is more efficient compared with existing work in this field.

7. RESULT

In this section simulation results are presented to demonstrate the effectiveness of proposed hardware obfuscation methodology for a set of ISCAS-89 benchmark circuit [16]. All circuits were synthesized using Cadence Encounter(R) RTL Compiler. The standard cell library of 180nm is used.

Table-2 & 3 gives comparison of area and power of original and obfuscated circuit. For circuit s526 whose gate count is less than 100 gates, the percent increase in area & power is between 15-20% percent approximately. Whereas, for circuit whose gate count is above 100 gates, percent increase is below 10% for the ISCAS-89 benchmark circuit.

**TABLE II. COMPARISON OF AREA FOR ISCAS-89 BENCHMARK CIRCUITS**

<table>
<thead>
<tr>
<th>ISCAS-89 BENCHMARK CIRCUIT</th>
<th>Original (µm sq.)</th>
<th>Obfuscated (µm sq.)</th>
<th>% increase in Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>S526</td>
<td>3169.782</td>
<td>3716.42</td>
<td>17.25%</td>
</tr>
<tr>
<td>S838</td>
<td>5697.72</td>
<td>6125.49</td>
<td>7.51%</td>
</tr>
<tr>
<td>S1196</td>
<td>6458.13</td>
<td>7056.806</td>
<td>9.27%</td>
</tr>
<tr>
<td>S1488</td>
<td>6391.602</td>
<td>6828.57</td>
<td>6.84%</td>
</tr>
<tr>
<td>S5378</td>
<td>2565.836</td>
<td>26250.664</td>
<td>2.33%</td>
</tr>
</tbody>
</table>

Table-4 gives the comparison of timing of original and obfuscated circuit. It can be observed that timing overhead is negligible and timing constraint meet with positive slack in most cases. The negative overhead is due to lot of positive slack is kept at the time of synthesis for layouting & optimization done for obfuscated circuit while combining original netlist & FSM netlist.

**TABLE III. COMPARISON OF POWER FOR ISCAS-89 BENCHMARK CIRCUITS**

<table>
<thead>
<tr>
<th>ISCAS-89 BENCHMARK CIRCUIT</th>
<th>Original (µW)</th>
<th>Obfuscated (µW)</th>
<th>% increase in Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>S526</td>
<td>565.891</td>
<td>664.675</td>
<td>17.46%</td>
</tr>
<tr>
<td>S838</td>
<td>917.28</td>
<td>998.424</td>
<td>8.85%</td>
</tr>
<tr>
<td>S1196</td>
<td>1168.474</td>
<td>1291.954</td>
<td>10.57%</td>
</tr>
<tr>
<td>S1488</td>
<td>1172.002</td>
<td>1271.491</td>
<td>8.49%</td>
</tr>
<tr>
<td>S5378</td>
<td>4570.171</td>
<td>4636.498</td>
<td>1.45%</td>
</tr>
</tbody>
</table>

**TABLE IV. COMPARISON OF TIMING FOR ISCAS-89 BENCHMARK CIRCUITS**

<table>
<thead>
<tr>
<th>ISCAS-89 BENCHMARK CIRCUIT</th>
<th>Original (ps)</th>
<th>Obfuscated (ps)</th>
<th>% increase in Timing</th>
</tr>
</thead>
<tbody>
<tr>
<td>S526</td>
<td>1889</td>
<td>1886</td>
<td>-0.16%</td>
</tr>
<tr>
<td>S838</td>
<td>4956</td>
<td>4915</td>
<td>-0.83%</td>
</tr>
<tr>
<td>S1196</td>
<td>3776</td>
<td>3761</td>
<td>-0.40%</td>
</tr>
<tr>
<td>S1488</td>
<td>3026</td>
<td>3020</td>
<td>-0.20%</td>
</tr>
<tr>
<td>S5378</td>
<td>3519</td>
<td>3551</td>
<td>0.91%</td>
</tr>
</tbody>
</table>

8. CONCLUSION

Obfuscation can provide active defense mechanism that can prevent IP stealing at different stages of IC design &
fabrication flow, thus protecting IP vendor’s interest, while incurring low design overhead.

Our approach of obfuscation techniques involves additional design overhead and requires application of enabling key for normal operation. Here, IP security is achieved at cost of approximately 10% area & power overhead for circuit. For small circuit whose gate count is less than 100 gates, overhead is more. The measure of obfuscation level is also given for our approach. Our approach is proven more effective than previous work in this field. As circuit functions normally only after proper execution of FSM, this increases initial time required and complexity while interfacing with other circuit. Our obfuscation technique is useful to realize a low-cost and robust authentication feature by introducing appropriate modification in the state transition function. The steps to implement our approach can easily be generalized to obfuscate numerous types of circuit.

Further work will target to reduce the overheads by using unused gates of original net-list design for additional overhead of key FSM.

9. REFERENCES