Anti-Counterfeit and Anti-Tamper Implementation using Hardware Obfuscation

Avinash Desai

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Engineering

8 August, 2013
Blacksburg, Virginia

Keywords: Anti-Counterfeit, Anti-Tamper, Trojan Detection
Integrated Circuits, Seal, Time-Stamp

Anti-Counterfeit and Anti-Tamper Implementation
using Hardware Obfuscation

Avinash Desai

(ABSTRACT)

Tampering and Reverse Engineering of a chip to extract the hardware Intellectual Property (IP) core or to inject malicious alterations is a major concern. First, offshore chip manufacturing allows the design secrets of the IP cores to be transparent to the foundry and other entities along the production chain. Second, small malicious modifications to the design may not be detectable after fabrication without anti-tamper mechanisms. Counterfeit Integrated Circuits (ICs) also have become an important security issue in recent years, in which counterfeit ICs that perform incorrectly or sub-par to the expected can lead to catastrophic consequences in safety and/or mission-critical applications, in addition to the tremendous economic toll they incur to the semiconductor industry. Some techniques have been developed in the past to improve the defense against such attacks but they tend to fall prey to the increasing power of the attacker. We present a new way to protect against tampering by a clever obfuscation of the design, which can be unlocked with a specific, dynamic path traversal. Hence, the functional mode of the controller is hidden with the help of obfuscated states, and the functional mode is made operational only on the formation of a specific interlocked Code-Word during state transition. A novel time-stamp is proposed that can provide the date at which the IC was manufactured for counterfeit detection. Furthermore, we propose a second layer of tamper resistance to the time-stamp circuit to make it even more difficult to modify. Results show that methods proposed offer higher levels of security with small area overhead. A side benefit is that any small alteration will be magnified via the obfuscated design proposed in these methods.

$\sim$ *To my Grandparents* $\sim$

Acknowledgements

Firstly, I would like to extend my sincere thanks to my graduate research advisor **Dr. Michael Hsiao** for giving me an opportunity to work in PROACTIVE group. His invaluable guidance and support was the driving force throughout my research. I have learned many valuable lessons of being a good student and a researcher through countless number of interactions with him. This work was possible due to his constant feedback and suggestions on ideas throughout this project.

I express my sincerest gratitude to Dr. Leyla Nazhandali and Dr. Chao Wang for serving on my thesis committee. Brain storming sessions during weekly meetings with Dr. Hsiao, Dr. Leyla and Dr. Wang and Dr. Simin Hall have really helped to carve out different techniques proposed in this thesis. I would also like to thank the student members of the project group Rashmi Moudgil, Dinesh Ganta and Bing Liu for working with me as a team and constantly encouraging to try out different things during the project.

I sincerely that my labmate and best friend Sarvesh Prabhu for all the fun we had inside and outside lab, for pushing me to achieve higher goals even during difficult phases of my life and for direct and indirect support.I thank my sister Indira for being there with me throughout my masters degree and encouraging me to do my best. I also thank Dilip, Sharad and Vineet for sharing many wonderful moments in the lab. I thank my roommates Hemanth, Rajat, Tanmay and Sushrut for maintaining an awesome atmosphere at home.

I take this opportunity to thank my beloved parents **Mrs. Vasanti Desai** and **Mr. Rajendra Desai** for keeping faith in me, multiplying my happiness and nullifying my sorrows throughout my life. I also thank my angel sister Asmita for constant encouragement and trusting me.

Finally, I express my gratitude to my best friend Ashwini, for standing besides me with whatever I do, giving me encouragement and unconditional emotional support when things do not go well and reinforcing my confidence whenever I need it.

Avinash Desai

Blacksburg

August 08, 2012

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In recent years, the horizontal semiconductor business model has proven to be more beneficial and profitable to the semiconductor industry as it improved both the cost and time-to-market. With globalization, a device can be manufactured from practically any part of the world. This type of business model gives rise to two major problems as listed below:

1. Intellectual Property (IP) is revealed to third party untrusted entities making them vulnerable to attacks.
2. The authenticity of the manufactured Integrated Circuits(ICs) gets a setback as counterfeit products are supplied in the market by adversaries.

The protection of sensitive information in a device is generally considered as a responsibility of the user. This information can be of critical importance for defense organizations, especially if the device falls in the hands of an adversary. The adversary seeks to extract as much sensitive information he/she can have from the device with the help of sophisticated techniques. An adversary may also be interested in learning about an enemy's (or competitor's) latest design by stealing or capturing one or more prototypes/functional devices and dismantling it.

To make things worse, in recent years, outsourcing of manufacturing and chip-fabrication requires revealing the design IP to external entities, creating many opportunities for IP infringements, counterfeiting, piracy, and/or insertions of malicious alterations. The problem is exacerbated by contracting the offshore foundries to lower the labor and manufacturing costs. Attacks are thus possible at major entities in the production and supply chains during third party manufacturing. Without proper anti-tamper mechanisms, chips can be reverse-engineered to extract the important IP within the chips. Pirated chips can then be sold at a very low cost. In the same way, insertion of malicious hardware (e.g., Trojans) by the untrusted manufacturer may be easy without anti-tamper features. Once inserted, the Trojan may be extremely difficult to detect, thereby compromising security. Additional threats such as reverse-engineering, or re-marking of Integrated Circuits (IC) are possible when there is lack of protection of the design. The estimated U.S. sales losses due to copyright piracy in 2004 is approximately $12.54 billion in total [2] with a significant contribution coming from hardware IPs.

Many techniques have been proposed to protect the circuit at different levels, including both active and passive methods. But with the increase in both strength and sophistication of the techniques used by the adversary, existing methods may not be strong enough. Our goal is to make anti-tamper easy to implement, yet offer a strong protection, of the design. In the proposed method for anti-tamper circuit we use obfuscated hardware to solve the problem.

A chip that resembles another visually may be used by an adversary as a counterfeit device are originate from discarded parts. Not only will this incur tremendous financial loss to the leading-edge design houses, but they also pose severe security concerns. If the device is not genuine, it may have incorrect or sub-par performance such that it may fail a system

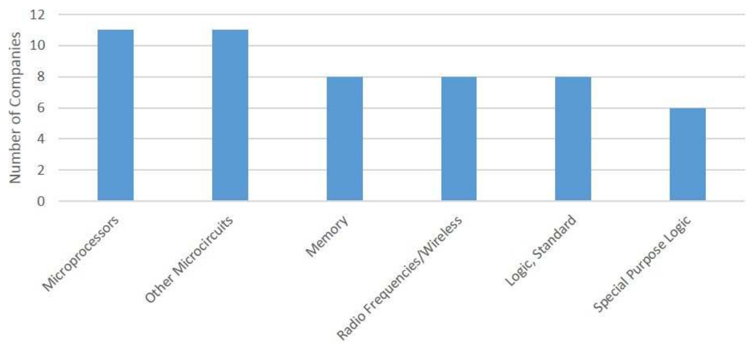

Figure 1.1: Types of manufactured parts suspected/confirmed to be counterfeit

in the most critical moment of the operation. For critical applications used in medical, power-grid, and defense, trust remains to be of utmost priority. In fact, one major concern raised in a White House report on National Strategies for Smart Grid, Cybersecurity and Supply Chain 2009 [3] was the lack of broadly applicable tools, techniques, and processes to detect or defeat counterfeiting and tampering in the supply chain or deployed systems. The report emphasizes on research needed in technology solutions to detect and prevent counterfeiting and overproduction. In another report, "we do not want a $12 million missile defense interceptor's reliability compromised by a $2 counterfeit part," indicated by General O'Reilly Director, Missile Defense Agency [4].

Most of the high-end defense products have a huge number of ICs in the system. For example, the U.S.'s next generation multi-role fighter contains more than 3,500 ICs [4]. Compromising the trustworthiness of a single chip could result in a catastrophe. On the other hand, instances of counterfeits have been rising alarmingly. Another report by U.S. Department of Commerce [5] brings to light that the number of counterfeit incidents in the defense supply chain has increased drastically from 3,868 in 2005 to 9,356 in 2008. Figure 1.1 shows the number of companies suspected/confirmed to give counterfeit microelectronic parts as per U.S. Department of Commerce Survey [5]. Financially, counterfeit ICs have a serious impact on revenue generated by a particular design house. The Semiconductor

Industry Association (SIA) estimates that counterfeits cost U.S semiconductor companies more than \$7.5 billion annually in lost revenue, a figure SIA says results in loss of nearly 11,000 American jobs [4].

Considering all the above, it is extremely important to be able to determine if an IC is authentic and filter out all the counterfeit parts. To determine this, one needs to answer two questions as illustrated in Figure 1.2: first, whether the IC is new or used; second, the batch information when the IC was manufactured, for example, the date of manufacturing.

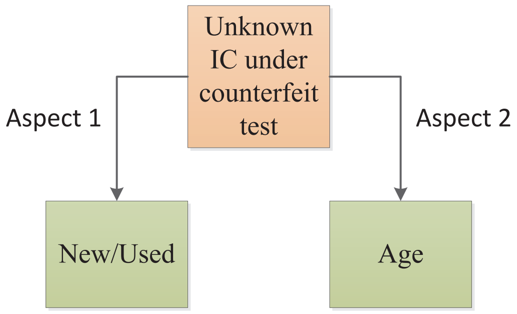

Figure 1.2: Two Aspects for Counterfeit detection

In this work we propose a novel time-stamp method which gives out the batch information by extending the obfuscated hardware used for anti-tamper. A tamper-proof time-stamp circuit is also proposed which helps the time-stamp circuit to resist attacks by adversary.

1.1 Contributions

The major contribution of this thesis are as follows:

1. We propose a new obfuscation and key based anti-tamper circuit design, making clever use of "Code-Word" logic to secure the circuit even in the functional mode. It is shown that it is extremely difficult for an adversary to successfully tamper the circuit in realistic time.

2. A novel time-stamp circuit is proposed which gives out the batch information as a response to a query i.e. sequence if inputs to the circuit.

3. The time-stamp circuit is made tamper-proof by extending the obfuscation logic used for anti-tamper design.

1.2 Thesis Organization

Chapter 2 presents the background related to attack possibilities in production chain, reverse engineering techniques, previous anti-tamper techniques and previous work for counterfeit detection.

Chapter 3 introduces to anti-tamper hardware design making clever use of obfuscated "Code-word." Design properties and steps followed are discussed in this chapter.

Chapter 4 provides a detail discussion on the novel time-stamp methodology and the design steps. Chapter extends the idea to tamper-proof time-stamp circuit. The efficiency in terms of area overhead is discussed.

Chapter 5 concludes the thesis and provides ideas for future work.

Chapter 2

Background

Tampering and counterfeiting of Integrated Circuits (ICs) has affected the global production chain. This chapter gives a background about these attacks and the techniques currently present to counter these attacks.

2.1 Production Chain and Attack Possibilities

Figure 2.1 shows the global production chain for the manufacturing of Integrated Circuits (ICs). The design house completes the design according to specifications and gives it to the manufacturing house. Here the chip is fabricated and returned to design house for testing and distribution purposes.

There are two possibilities where tampering of devices take place:

1. Tampering can take place in the fabrication house. The manufacturer produces steals the design and sells it to some other company to make profit. The manufacturer may also insert a Trojan into the chip during fabrication for malicious purposes.

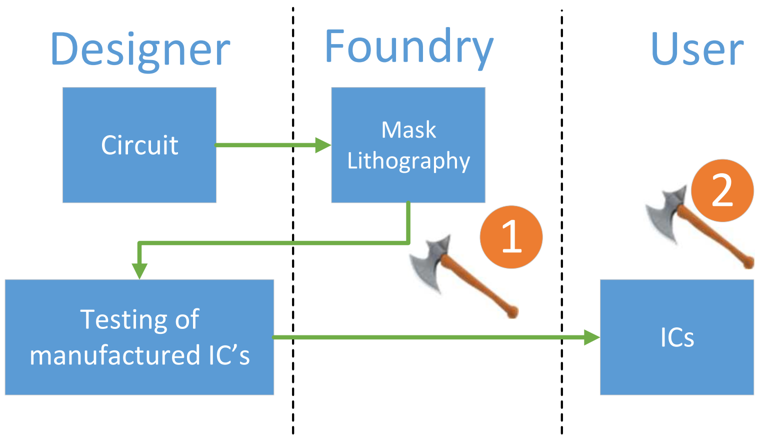

Figure 2.1: Production Chain and Attack Possibilities.

2. This attack is at the distribution and user end. Users can misplace their device or due to some other circumstances the IC falls in the hands of adversary. The adversary then reverse-engineers the chip, makes clone chips and sells it to make profit.

In addition to tampering, counterfeiting problem is also a major problem in the supply chain.

1. Counterfeit chips can be supplied by the manufacturing house to the design house. A fresh batch of manufactured ICs can be mixed with chips manufactured in previous batches or scavenged chips from used circuits.
2. Counterfeit chips can be flooded in the market and sold at lower prices directly to the consumers.

2.2 Reverse Engineering Tools and Methodology Used

Reverse engineering is the process through which one can obtain the details of the circuit given an IC. There are positive applications of reverse engineering like bug detection in equipment. Thus a lot of research has been done in this field and there are specialized labs and sophisticated tools for reverse engineering. Unfortunately, an adversary can also use these tools to get a complete knowledge about the circuit. An adversary can use a range of techniques to identify circuit intent and structure using a small set of known characteristics. This small set of known characteristics can be collected through combotronics and using an old version of similar kind of chip. The techniques used in reverse engineering can be generally classified as:

2.2.1 Black Box Testing [28]

This approach involves the exercising of input combinations and mapping the associated outputs. A common misconception regarding black-box RE on modern circuits is that it is a futile effort, owing to the increasing I/O space and circuit complexity. However, this technique can be an efficient method even with a large input space. The main idea established in [28] is the ability to recognize patterns in a partial truth table. They then relate to the characteristic of common functions.

2.2.2 White Box Testing

In white box analysis, the complete circuit is available to the user/attacker. By analyzing the circuit structure it is possible to find an input for particular output. The circuit structure is available in the form of a design netlist, circuit imagery etc. [28]. Techniques used to construct the logic diagrams include electron beam microscopy, pattern recognition and netlist extraction techniques. Creating gate library and targeting on areas dense in XOR is one of the methods proven helpful for reverse engineering for cryptographic algorithms [29]. The key to the success of such an approach is the fact that cryptographic algorithms rely heavily on bit-wise operations.

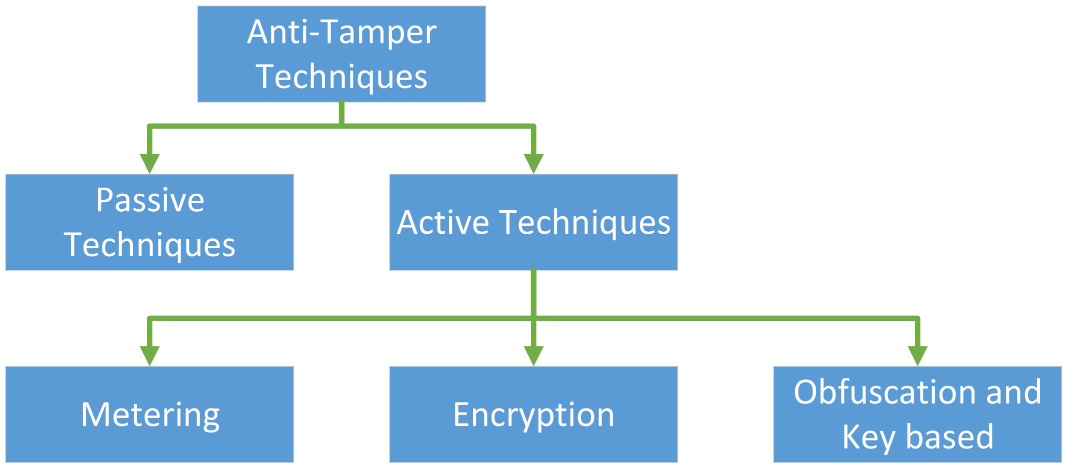

Figure 2.2: Classification of Anti-Tamper Techniques

2.2.3 Side Channel Analysis (SCA) [30]

Some device characteristics are also useful in providing information such as power, time, electromagnetic radiations, path delays, etc. Side channel analysis is the use of these characteristics to understand and/or extract the internal secrets. This information is used to locate the areas where specific data is stored. SCA can also suggest when certain information is available for theft. SCA improves the ability of attacker to relate actual values and assumed outputs. Countermeasures against SCA fall into 2 categories including hiding and randomization. Hiding offers protection by reducing the signal to noise ratio of the side channel leakage.

2.3 Classification of techniques for Anti-Tamper

Much work has been done in the past to provide protection against tampering of Integrated Circuits (ICs). Figure 2.2 shows the classification of different defense mechanisms to counter tampering of ICs.

2.3.1 Passive Techniques

Passive Techniques are those in which the circuit does not prevent the user from using it in the functional mode. But there are certain characteristics and properties in the circuit which help the user to prove the copyrights of the design and hence can file a case against the adversary for counterfeiting and/or tampering. One such technique is Watermarking [7–9]. In one such technique, the designer uses the state transition graph to encode a watermark in the circuit to prove the copyright of the circuit. The watermark can be an image, a literature or some encoding algorithm. Watermark techniques only help to prove copyrights / authenticity. The circuit does not resist the attack. A watermark is chosen in such a way that even some modification in the circuit the tamper can be clearly seen.

2.3.2 Active Techniques

Active Techniques are those in which the circuit has built-in capability to protect itself against tampering. In this case the circuit has an embedded protection hardware so that unauthenticated persons cannot have full access to the circuit. One such technique is encryption. In this method the designer uses encryption to lock the hardware. It makes use of some specific tools to encrypt the hardware.

A second type of active techniques is Hardware Metering [13–16]. Metering protects the hardware from attacks at the fabrication stage. These techniques make efficient use of Physical Unclonable Functions (PUFs) to obfuscate the protecting mechanism and make the circuit behave in an unpredictable way. The designer has to find the response of the PUFs to the stimulus and generate input vectors to make the circuit operate correctly. These are then stored in the memory of IC (EEPROM). Such a method cannot provide protection against attacks at the user end because the inputs are store in EEPROM and the circuit is

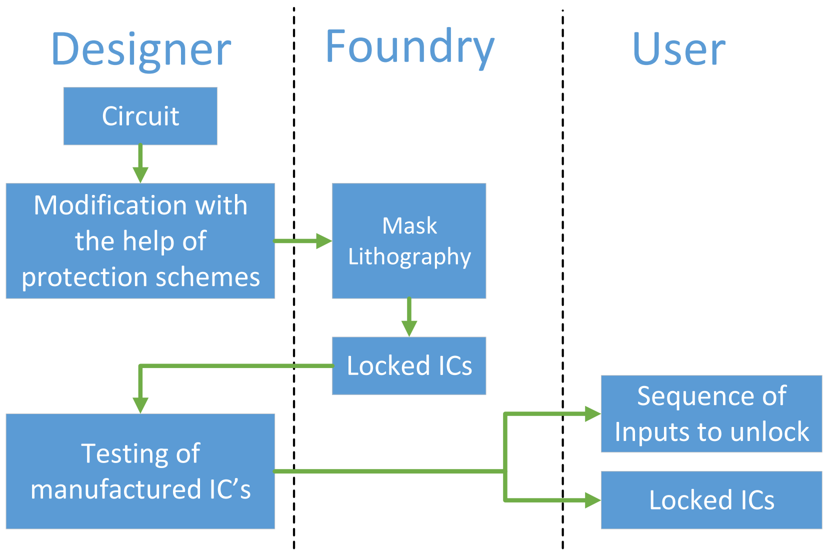
Designer
Foundry
User
Circuit
Modification with the help of protection schemes
Mask Lithography
Locked ICs
Testing of manufactured IC's
Sequence of Inputs to unlock
Locked ICs

Figure 2.3: Production Chain for IP protection using Obfuscation and Key

already operational. Also the design house has to maintain a database of key for each and every IC as each IC has a unique key depending on the PUF challenge response.

A third type of active techniques is obfuscation and key based. In this approach, the design is modified such that the original intent is hidden and/or difficult to extract. The functional mode is entered only when the user authenticates itself by applying a series of inputs called "key".

Figure 2.3 shows the modifications in the production cycle to implement obfuscation and key based protection schemes with a key (sequence of inputs) to make the chip functional. As seen in Figure 2.3, modifications are made in the design of the integrated circuits (ICs) so they are not functional unless an expected sequence of inputs is first applied to them. This design is then fabricated in the foundry which produces chips which are still locked. Hence, the design resists the attack at the manufacturer's end. The attacker must understand the mechanisms to unlock the circuit in order to understand the design. Likewise, a small alteration to the circuit during fabrication may render the circuit non-functional. The manufactured ICs are then tested and a sequence of inputs is generated as per the design to make the IC functional.

Users who purchased the IC are supplied with the chips and the sequence of vectors to unlock. This sequence of vectors can be supplied via software, such as IC burners to load programs. This methodology also resists attacks at the user end as only the authenticated users are supplied with the sequence of vectors to activate. The success of this method depends on the level of obfuscation achieved and the proper use of obfuscated hardware for protection. In this method we cannot make use of any process variables to control the circuit as the sequence key for each IC would be different.

2.4 Related work on Anti-Tamper Techniques

Obfuscation as means of hardware protection was first put forth in [34]. They used netlist level obfuscation for hardware protection. It contained the use of key to drive a dedicated FSM which protects the hardware. The output of this dedicated FSM was injected in various internal signals with the help of XOR gates. In [34], the FSM output was required to be all zero for the proper functioning of the circuit. The FSM proposed in [34] acted as a separate entity altogether and can be easily recognized by the adversary. Although susceptible to attacks and weak defenses, it nevertheless showed the usefulness of obfuscation for protecting hardware.

2.4.1 HARPOON [31] Methodology

The authors in HARPOON suggest an improvement over their previous work [34] by making the protection hardware more close to the IP core. With this formulation, the level of obfuscation has increased. Figure shows the implementation used in [31]. It introduces new state elements in the core logic to increase the total state space. The new states formed are

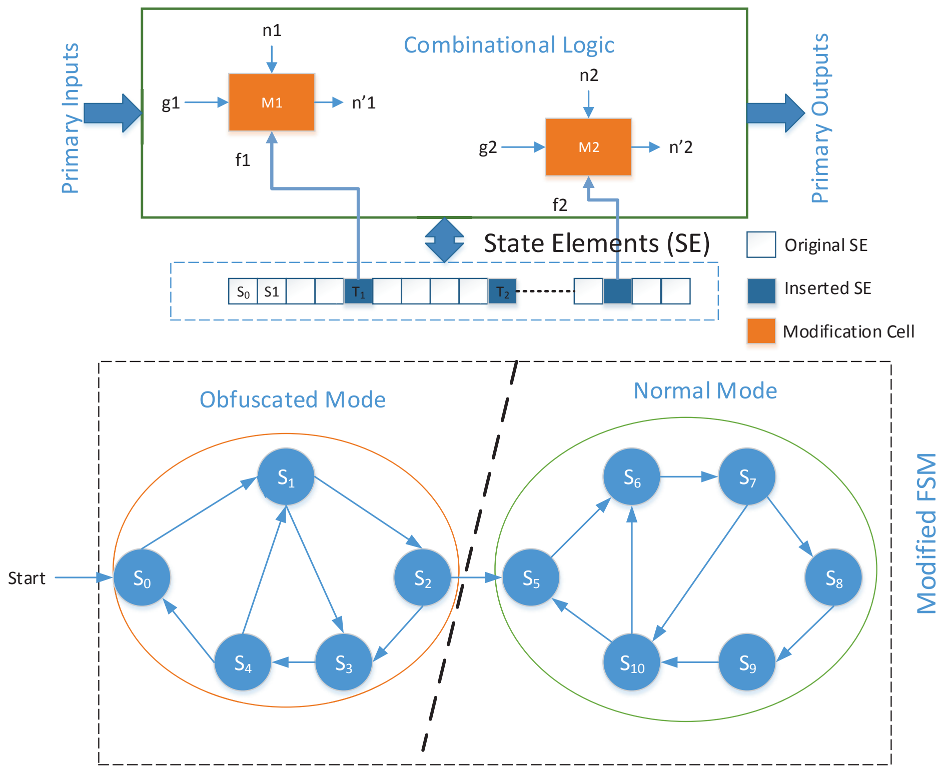

Figure 2.4: HARPOON Methodology.

used to obfuscate the hardware. There are two modes in the circuit: Obfuscated mode and Normal mode. On applying the global reset, the circuit goes directly in a state which is in obfuscated mode. From this initial state, a set of inputs must be applied to drive the circuit to a state in the functional mode. This set of input makes up the needed key for the circuit.

The added state elements are placed at strategic points. These points depend on the output cone. Induction in the circuit takes place with the help of hardware as in Figure 2.4, where M1, M2 and M3 are the induction points. The induction points are designed such that they do not alter the signal in the circuit if all the inserted elements are low, i.e., logic “0”. The state encoding for the states in the normal mode is done such that all the inserted state elements are “0”.

Drawbacks of HARPOON

The above method can be reverse-engineered with some amount of effort by the adversary. Starting from the initial state, the circuit loops in the obfuscated mode until the correct sequence is applied. For example, the correct sequence for the circuit illustrated in Fig 3.1 is $P0 \rightarrow P1 \rightarrow P2$. This is the only path from the initial state to states in the normal mode. Due to area constraints the number of transitions in the obfuscated mode is kept low. After a number of trials the adversary can obtain a list of states that the circuit is continuously running through in this loop. Thus the adversary can know a portion of the invalid states. The number of states in the obfuscated mode is low due to area constraints. So knowing the invalid states can increase the probability of picking a state in normal mode. The adversary can confirm that the state picked is a valid state by traversing through and finding that the circuit no longer loops in the obfuscated space. The adversary can compute other reachable states from this state and the path that can reach to this state using state reachability computation tools.

2.4.2 STEP [35] Methodology

Secure Test and IP Protection (STEP) tries to address both secure test and IP core protection as a joint objective. Figure 2.5 shows the design methodology followed. The design contains a special block called the "Hardware for key Integrated Security" which contains a Pseudo Random generator "PRBS Generator" and the key checker.

In this design "Dummy Flip-Flops" are introduced in the scan chain. These Dummy Flops have to be loaded with the correct value of key. The key checker has an integrated 'N' bit key. During the test phase the key checker compares the value of integrated key and the value of the dummy flops to give a secure signal which is fed to the output multiplexers. The

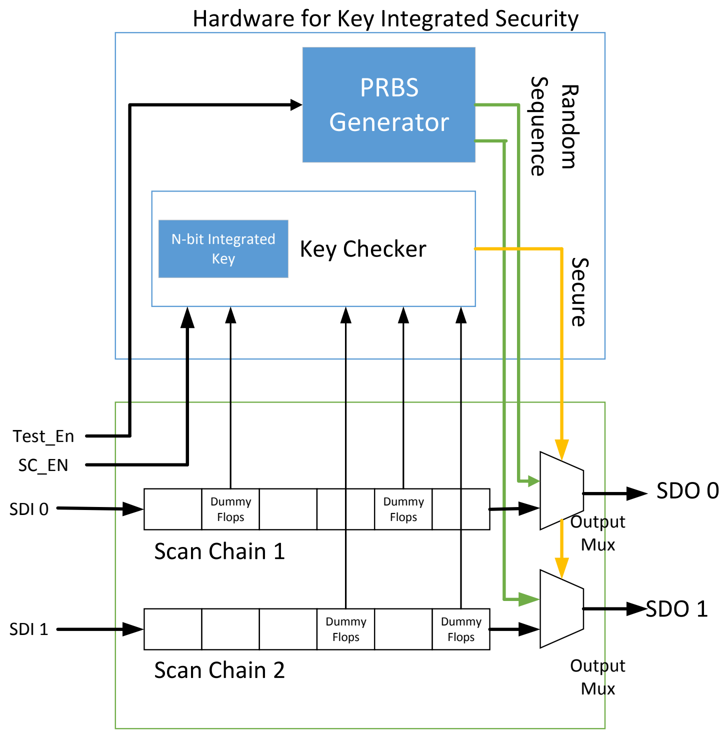

Figure 2.5: STEP design methodology

secure signal is *Low* if the key matches thus giving out the value of the scan chains ($SD0$ and $SD1$) from the multiplexers. If the key comparison fails the secure signal is *High* which outputs the value of PRBS generator. Thus the value of *Secure Signal* determines the value of "Output Multiplexers." Authors claim that the adversary will require $1.4e115$ trials to successfully make the circuit operational. The area overhead required for this method is 9% for AES implementation.

The paper highlights the importance of test circuit protection but has a few drawbacks as follows:

Drawbacks

1. The integrated key is stored inside the key checker block. As in the discussion in Section 2.2.2, White box analysis can reveal the key inside the checker. The use of comparators makes the key more prominent to the adversary.

2. The most important signal which is the *Secure Signal* controlling the value of output multiplexers can be located easily by the adversary as it is common to all the output multiplexers. Forcing a constant stuck-at-low value at *Secure Signal* will make the security mechanism ineffective.

2.4.3 Hardware IP Protection during Evaluation using Embedded Sequential Trojan [32]

The authors provide this methodology during the evaluation phase of an IC. The protection mechanism is based on obfuscation of states similar to HARPOON. However, in this work, the implementation of obfuscated states is such that once on missing the sequence of inputs, the core logic goes in a sequence of states and goes on looping in this sequence. After a certain number of incorrect inputs, the IP core goes into extended states which activate certain hardware to force error in the output. The hardware used to force error in the circuit output is called a Trojan. The idea of having different paths till the Trojan activation states is interesting. The sequence of inputs to the IP core is controlled by another circuit (part of entire circuit) which has an inbuilt timer in it. The controlling timer circuit sends error inputs to the IP core under protection at the end of prescribed time. Thus the chip is operational only for prescribed time. With reverse engineering, the adversary knows about the IP core to which the timer output is connected. To find the sequence of inputs to make the core functional would be difficult as the core traverses through many states before making the Trojan output high.

Drawbacks

It will take some effort for the reverse engineer to find the safe states in the circuit. If the adversary pulls out a safe state, he can backtrack to get the sequence to reach there. The protection mechanism does not operate in the functional mode of the core. Hence, the two modes of operation are separable. This stands as a weakness of this implementation.

2.4.4 RTL Hardware IP Protection Using Key-Based Control and Data Flow Obfuscation [33]

This work aims at protecting the hardware by making changes in the data flow and control flow of the original circuit. The state elements of mode control FSM are planted along with the original state elements. The state elements of mode control FSM are used to calculate the values of the modification signals. The circuit reaches the proper functional mode on application of input sequence, i.e. Key. If an incorrect sequence is entered, the control and data flow of the circuit are altered with the help of conditions on the modification signals. Thus the circuit is not stuck at any state but moves in a sequence of states producing an incorrect output. Only when the modification signals are correct, the circuit produces proper output. This ensures that correct output is received only in the functional states.

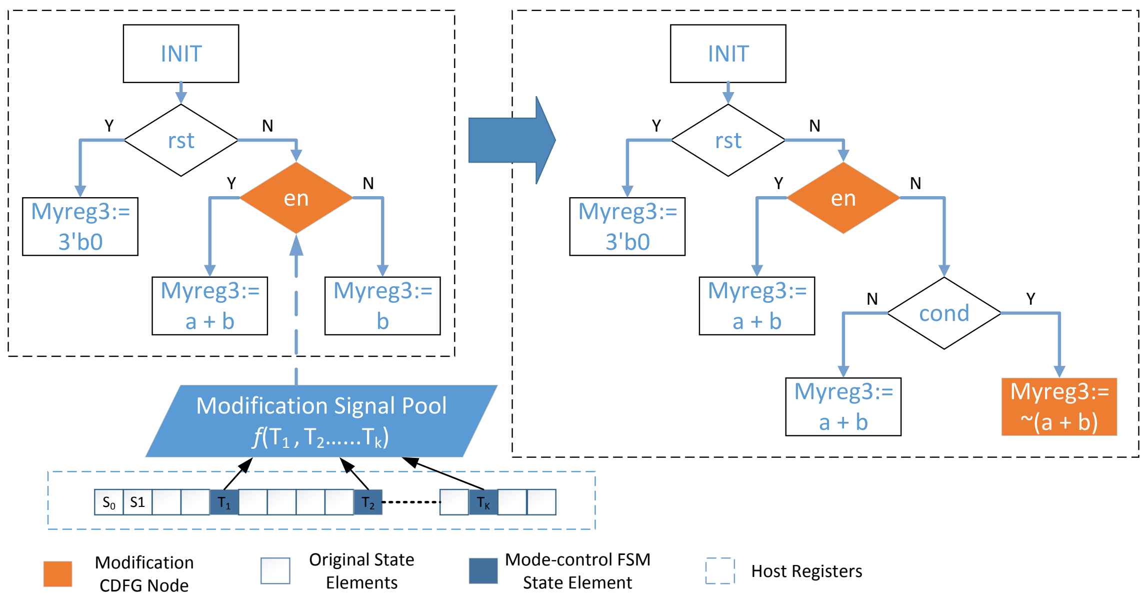

Figure 2.6: Key-Based Control and Data Flow Obfuscation

Advantages

The strategy provides a higher level of obfuscation relative to the other implementations as the circuit is not stuck in a single loop or group of states. Also the implementation is at the RTL level. RTL level implementation blends the protecting hardware into the rest of the logic. The design procedure goes through synthesis and optimization. This causes a restructuring in the final hardware structure. The resultant circuit at the netlist level does not contain a definite pattern of protecting structures, e.g., XORs which can be spotted out easily. The amount of time for reverse engineering will be higher but it will still be achieved in following way.

Drawbacks

In this methodology the modification signal from the pool are used to decide which operations to perform when in a state. The modification signals are derived from the newly introduced state elements to protect the hardware. The control flow graph of the circuit thus contains two types of control signals: original control signals and the modification signals. The value of each signal used for decision can be true or false. A comparator is used to compare the value of modification signals and perform operations accordingly. The operations performed in a state can thus be divided into two categories: operations performed when modification signal is true and operations performed when they are false. Separating the modification signals from the original control signal is possible, albeit difficult. If the input sequence is wrong, no matter what the sequence is all the modification signals will be constant (either true or false). The original state elements will change during this process. As the original control signals are derived from the original state elements, these control signals will not be constant. Thus by applying many runs of incorrect inputs the adversary can figure out the control signals that do not change and hence uncover the modification control signals. Then, the adversary can revert the value of modification signals to find the value of state elements which are introduce to protect the hardware.

2.5 Related work on Anti-Counterfeit Techniques

Counterfeit IC detection methods can be classified according to ICs on which they can be implemented. These methods are applicable on future deigns and existing designs.

2.5.1 Counterfeit detection techniques for existing designs

Traditional techniques for counterfeit detection such as printing serial numbers or barcodes to prevent counterfeiting of ICs in integrated circuits are ineffective as they offer very weak resistance to attacks and can be easily cloned or faked. X-ray and Infrared microscopy [10] inspection techniques are deployed for anti-counterfeit purposes. Special high magnification machinery is needed for it and the reliability of the technique is questionable.

Some of these methods depend on statistical data and aging effect of components in a chip. In [12] the beat frequency of two ring oscillators, one stressed and the other unstressed is measured, to achieve 50 higher delay sensing resolution than that of prior techniques. This technique involving differential frequency measurement eliminates the effect of common-mode environmental variation such as temperature drifts between each sampling points. This technique makes a precise measurement of digital circuit degradation thus obtaining the aging parameters of the chip.

Path delay as an age detection criterion is shown in [11]. The authors show that a single path delay cannot directly reveal the age, as it is also greatly influenced by process variation and this could result in large error in classifying ICs as authentic or counterfeit. Instead, they establish that the relationship between the delays of two or more paths. The results show that this is a great indicator of age.

2.5.2 Counterfeit detection techniques for future designs

The downside of these methods in circuits without proper circuit dedicated for counterfeit detection is that they depend upon statistical data and could result in some false-positives and false-negatives. For future designs the designer can dedicate certain parts of the circuit for counterfeit detection purpose.

Watermarking [7–9] have been used for detecting a class of counterfeits as there are certain characteristics and properties in the circuit which help the user to prove the copyrights of the design. These watermarks often need to be changed with each design for proper identification leading to a high design overhead. In addition, checking for the correct watermark for each device from the vendor may require millions of instructions per device, making it both infeasible and impractical.

Most of the metering techniques [13–16] use Physical Unclonable Functions (PUFs) to produce locked ICs. Each and every IC has a different input sequence to unlock it in order to bring it to functional use. This helps the design house to maintain a database of ICs manufactured and their respective ages. However, keeping a database of all the ICs adds up a huge overhead in IC lifespan and may be infeasible. Anti counterfeit design by PUF-based implementation [17] is also an effective method to keep a track of manufactured ICs. Nevertheless, the measuring of the age also requires a challenge-response database lookup which is cumbersome and time-consuming.

All the above techniques lack a simple, yet effective method to determine if a chip has been used and/or give out the manufactured date for the IC in question. Furthermore, a non-trivial amount of changes are required to implement the previous methods. In this work, we aim to provide a simple and efficient way for ICs such that counterfeit chips are easily detectable. We propose a novel Time-stamp design which uses a strategy based on Obfuscation and Key techniques.

2.6 Related Work on Hardware Trojan Detection

A hardware trojan is a tiny and stealthy modification to the circuit done intentionally by an adversary. The activity of the trojan is suppressed for most part of the operation of IC but gets triggered only at certain conditions. These stealthy hardware trojans can be introduced to bring down the functioning of the IC, alter the specification, extract critical information or even transmit information with the help of an antenna. Trojans can be introduced into a design in any phase of the production chain from RTL design to fabrication [19].

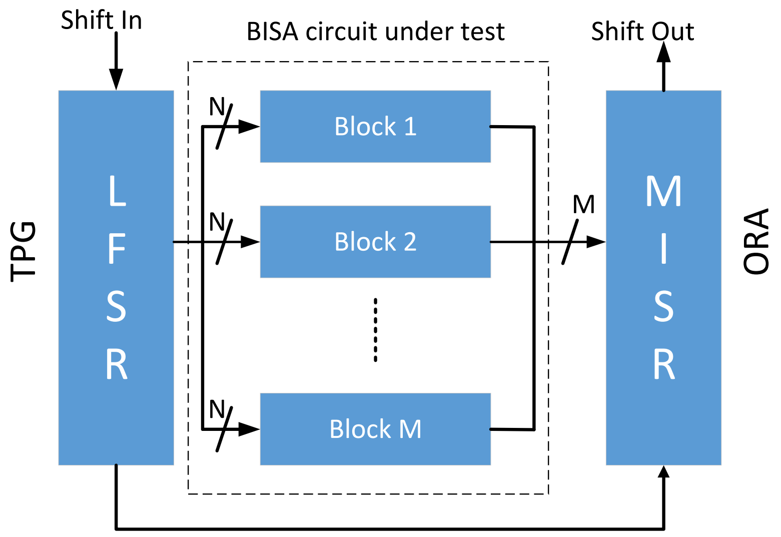

Figure 2.7: BISA cell structure

There have been many previous works which target detection of trojan in the post-silicon phase. These methods include statistical analysis of delay [20], leakage current [21] and also power measurement [22]. In [23, 24] authors use side channel power analysis to obtain the power signature to detect malicious additions to the circuit.

Many design-for-trust techniques in the design phase have been suggested for trojan detection which make insertion of trojan difficult. In [25] many strategies involving formal verification, code coverage analysis and techniques to reduce suspicious signals by redundancy removal, equivalence analysis and sequential ATPG are studied. In [26] an approach which involves N-detect full-scan ATPG method, combined with Suspect Signal Guided Sequential Equivalence Checking (SSG-SEC) is presented. The method involves identification of malicious signals corresponding to hard-to-detect faults.

Authors in [27] present a novel method to fill the unoccupied space in a chip with functional standard cells rather than filling with nonfunctional cells during the layout phase. These filled up functional cells called as BISA (Built in Self Authentication) cells are interconnected to form a circuit that is independent of the original circuit. Figure 2.7 shows the structure of these BISA cells. Any alterations to this circuit can be detected by functional test as the circuit formed is completely testable. As complete area of the chip is filled the adversary has to replace these BISA cells with self constructed circuit affecting the signatures generated by the circuit.

BISA cells are designed such that the adversary cannot distinguish between main circuit of the chip and the introduced BISA cells. The only downside of this method is there is a possibility that alterations are made to the original design and not the BISA cells. In this case the presented method does not provide any detection mechanisms to such alterations. This method thus can be used to complement design-for-trust methods. In this proposed work the real intent of the deign is hidden from the adversary, thus making it extremely difficult for the adversary to make successful alterations in the circuit.

Chapter 3

Interlocking Obfuscation for Anti-Tamper Hardware

Anti-Tamper techniques are used to protect Integrated circuits from tampering and reverse engineering and as a side benefit to detect malicious alterations.

Our proposed method is a low-complexity anti-tamper design at the RTL. Similar to previous methods, it implements obfuscated hardware elements for protection. The hardware can be unlocked only with a correct sequence of keys that is given to the appropriate user. However, by looking at the internals of the circuit, it is nearly impossible to separate the anti-tamper logic from the rest of the circuit or to derive the key.

3.1 Shortcomings in Reverse Engineering Techniques

The adversary has some of the limitations on his side too. Knowing these limitations can help us strengthen circuit's defense mechanisms. The following are some of the hindrances in the path of discovering entire functionality of the chip:

3.1.1 Complete functionality unknown

The adversary has to first figure out the working of the chip. The best possible way to figure out the functionality is with the help of previously released versions of similar type of chip. The reverse engineer can use various data collected from different methodologies to compare and find the similarities and differences.

3.1.2 Limited resources

The resources available to the adversary are dependent on the type of chip being reverse engineered. Chips of crucial importance and military applications would be of greater interest. Resources like time, money, sophisticated labs, man-hour, etc. will be high for such chips. But still these will be in limited amount. There will be no point in spending more resources than required for the design of the chip.

It is highly impossible to design a chip that can never be reverse engineered. But by understanding the limitations of adversary, it may be sufficient to design a methodology that resists the attack for duration till resources of adversary run out. Thus it implies that highly important chips should have higher protection circuitry guarding the overall functionality.

3.1.3 Dependence on algorithmic approach

The reverse engineer tries to find some patterns in the circuit to distinguish between core logic and protected hardware. As the circuit is large, the adversary tries to find the protection mechanism with the help of software tools like test pattern generator. Thus this methodology is more an algorithmic approach.

3.2 Requirements of Effective Anti-Tamper Methodologies

Relating the methodologies used by adversary, we can lay out some requirements for desirable protection mechanisms. The requirements can also be derived from the designer's compatibility point of view.

1. Obfuscated hardware used for protection should be placed such that it is very hard to differentiate it from the core logic of the function. This can be done by placing the obfuscated hardware as close to the core logic as possible and increasing the interaction between them. For our benefit, the core function and the obfuscation logic should not be separable to enhance the level of protection.

2. The ability to change the behavior of the circuit dynamically in the obfuscated mode. As the adversary tries to find the protecting mechanism in an algorithmic way, the dynamic behavior of circuit to such unauthenticated users will make the protection mechanisms stronger. It will require significantly more permutations and combinations of the inputs to attack the system.

3. The design of the anti-tamper mechanisms should not change the specifications of the

chip as seen by the user. Once the user has authenticated himself the chip should behave in the exact same way as it would have without protection mechanisms.

4. The obfuscated design should have the same inputs and outputs as the original design, with the area and power overheads to be as minimum as possible.

5. The complexity of designing the protection mechanism should be low. This is to ensure that the designers of the chip spend minimum amount of time for protection mechanisms. The insertion of anti-tamper mechanisms will thus cause a minimum delay in production life cycle of the chip.

3.3 Methodology

3.3.1 Expansion of States

Similar to previous anti-tamper methods we expand the number of states to support obfuscated hardware. The strength of the protection can be increased by the expansion of states, which is done by increasing the number of present state elements in a Finite State Machine (FSM) in core logic. The overall functionality of the circuit is likewise divided into two modes: Entry mode (obfuscated) and Functional mode. The number of states in the original FSM and Functional mode of new FSM are same but their state encoding is different. This makes our approach different from the previous metering and anti-tamper techniques where the state encoding is kept the same.

Expansion of states by increasing the number of state elements requires very little area overhead. Just increasing the number of state elements by ‘1’, doubles the total number of states available. This gives a very good support for the tamper resistant hardware to be

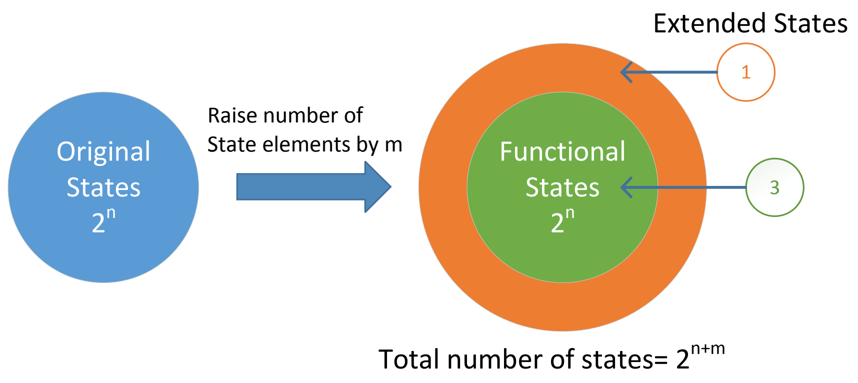
Extended States
1
Original States 2^n
Raise number of State elements by m
Functional States 2^n
3
Total number of states= 2^{n+m}

Figure 3.1: State space expansion of the core FSM.

obfuscated inside the circuit, with very less area overhead. As the expansion of states is done in the Register Transfer Level (RTL) it is extremely difficult for the reverse engineer to separate the functional states and entry mode states from the hardware formed after synthesis.

Expansion of states is shown in Figure 3.1 which shows a state machine with 2^n states is expanded to 2^{n+m} states. Out of these 2^{n+m} states 2^n are used as functional mode states eg. state “3” and remaining as entry mode states eg. state “1”.

3.3.2 Code-Word

The Code-Word is a novel concept which links the system in entry mode and the functional mode. It also help protect the functional mode depending on the sequence of states traversed by the system and inputs applied in entry mode. A Code-Word encodes the path executed by the circuit from the inputs applied in the entry mode in order to reach the functional mode. In each stage of the entry mode, Code-Word is gradually formed. Code-Word is used for protection of the circuit in the functional mode, but note that the expected value of the Code-Word is not stored anywhere on chip. Instead, the Code-Word is integrated into the transition functions of the state machine. For example, the transition function in the functional mode is modified and the next state for a number of states is a function of both the present state and Code-Word.

Therefore, the value of the obtained Code-Word plays a key part in determining the transition relation in the functional mode. The two modes of operation i.e. the entry mode and the functional mode are integrated and not easily separable.

Some key points regarding the code-word are:

1. The desired value of the Code-word is not stored anywhere on the chip.
2. Irrespective of the value of the Code-word formed during the entry mode the functional mode is always entered. Thus the circuit does not loop into only the entry mode states which is the case in previous methods.
3. Correct Code-word is formed only when a desired path is traced applying specific inputs.
4. Value of the Code-word formed while entering the functional mode decides critical transition functions in the functional mode.
5. Value of the Code-word remains constant in the functional mode. Thus once authenticated on boot the user does not have to authenticate again and again.

3.3.3 State Transition Graph (STG)

A State Transition Graph (STG) if used to represent the state transitions and output functions of a Finite State Machine (FSM), with the nodes reflecting the states and the edges defining the input/output conditions for a state-to-state transition. Figure 3.2 shows a State Transition Graph for a circuit with starting state “3”.

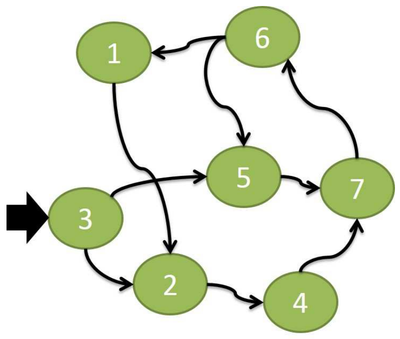

Figure 3.2: State transition graph.

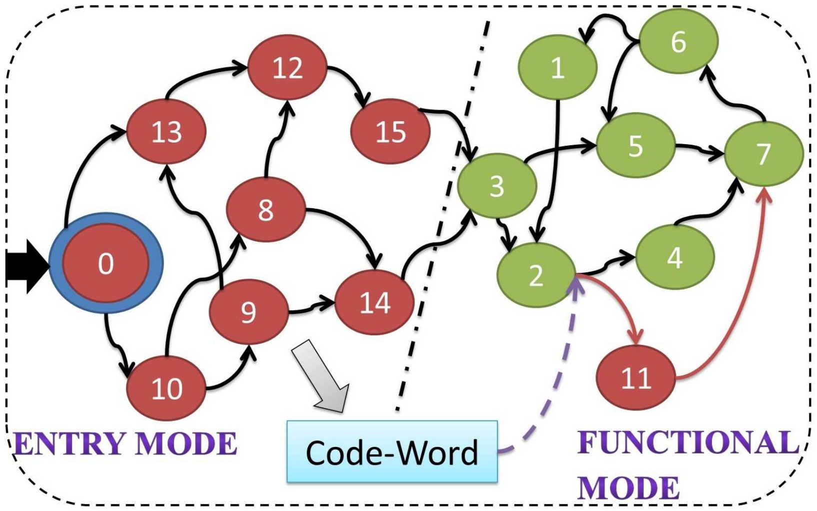

Figure 3.3: Modified State transition graph.

Figure 3.3 shows the state transition graph of a modified circuit using proposed methodology. The states shown in red are the invalid states used in the entry mode and to force faulty path in functional mode. The valid states in the functional mode are shown in green. Figure 3.3 also shows the embedding of the Code-Word to compute the correct next state in state “2”. Code-Word is thus used to protect an important state “4”. If the Code-Word while in the functional mode is correct as per the designer the system will transition from state “2” to “4”, else the state “4” is bypassed by other states depending on the value of the Code-Word.

3.3.4 Entry Mode

With such a setup as mentioned above, entry mode starts with a fixed initial state. The correct Code-Word is formed only when the path traversed is as desired (via a correct input sequence). As shown in the Figure 3.3 there are many paths by which we can reach state "3" form state "0" e.g. 0-13-12-15-3, 0-10-8-14-3 etc. But the correct code-word is formed only when functional mode is entered via desired sequence of states. Irrespective of the value of the Code-Word, the functional mode is always entered. This differentiates our method from existing methods where an invalid key disallows entry to the functional mode. However, the behavior in the functional mode depends on the correct value of the Code-Word. The implementation of the Code-Word logic is interlocked with the original transition function and is protected from the adversary by increasing the interaction with the FSM state elements. The more the use of Code-Word in the state transition function the better is the obfuscation metric of the Code-Word hardware.

Black holes and Gray holes [13] can also be formed in the Entry mode to confuse the attacker if the designer wishes to do so. Black holes and Gray holes have been shown to be effective against reverse engineering. Black holes are a set of states in which the circuit loops once it goes inside. There is no transition which leads out of these set states. Gray holes are set of states in which there is only one pattern when applied takes you out of the loop of these states.

3.3.5 Functional Mode

In the functional mode, the nodes with important computations and assignments are reached only with the correct value obtained in the Code-Word. In other words, the value of the Code-Word is not needed to compute the correct next state for every present state, but

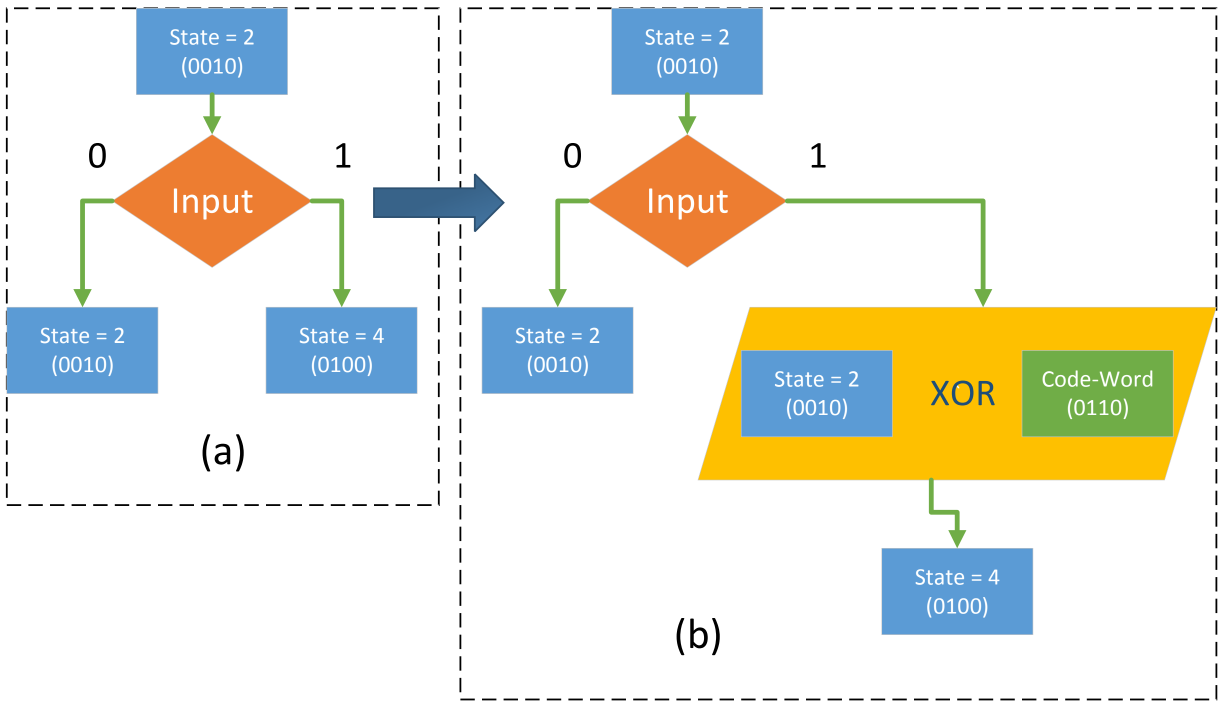

Figure 3.4: Functional Mode Modification

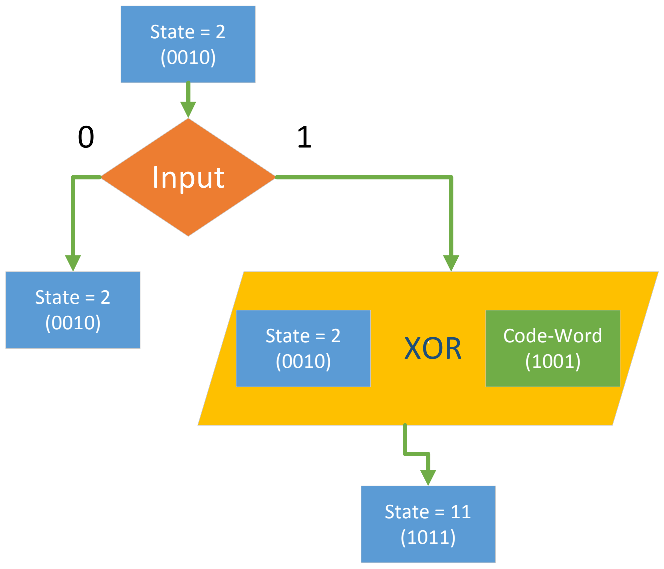

Figure 3.5: State Transition for Incorrect Code-Word

only required for modified states. If the adversary reaches a valid state in the functional mode, he/she is unaware of the next state and also the dependence on the Code-Word. So even if adversary reaches the functional mode via a different input sequence, the next states reached may be different as the value of the Code-Word is different. The functional mode of the circuit adapts a different form with any change in Code-Word. Hence the circuit behaves in a dynamic way according to different values of Code-Word formed. Thus we integrate the Code-word in the functional mode without using any form of comparators which weaken the obfuscation of hardware.

Figure 3.4 shows the modification of state "2" of a circuit with State Transition Graph shown in Figure 3.3. Figure 3.4(a) shows the original control flow and Figure 3.4(b) shows the modified control flow with the usage of Code-Word to decide the next-state. For simplicity the state transition function is taken to be a simple XOR. Correct value of Code-Word in this example "0110" will transition the circuit to correct next state i.e. 4. Figure 3.5 shows transitioning of the circuit to wrong next state "11" due to the incorrect value of the Code-Word. Thus the adversary will not be able to unleash the complete functionality of the IC unless the correct Code-Word is formed by the specific path traversal in the Entry mode.

Figure 3.6 shows a RTL transformation of the code for the transition graph in Figure 3.3. The Figure 3.6(a) is the original design and Figure 3.6(b) shows the modified design. Modified design shows the formation of Code-Word in entry mode marked in blue. Code marked in red shows the usage of Code-Word in state "0010" of functional mode. The original design used 3 state elements thus having a total number of 8 states. In the modified design the sate expansion is done by increasing the number of state elements from 3 to 4 thus having a total of 16 states. Here 8 states are to be used in functional mode and remaining 8 can be used for the entry mode. The state encoding for states in the original design and the functional mode in modified design is kept the same for understanding purpose. Note that

(a) Original Design

```
case STATE  is
        when "011" => if(in1 = ‘1’) then STATE <= "101";
                else STATE <= "010"; end if;
        when "010" =>        STATE <= "100";
        when "101" =>        STATE <= "111";
        when "111" =>   STATE <= "110";
        when "110" => if(in1 = ‘1’) then STATE <= "001";
                else STATE <= "101"; end if;
        when "001" =>        STATE <= "010";
        when "100" =>        STATE <= "111";
        when others =>        null;
end case;
```

(b) Modified Design

```
case STATE  is
  when "0000" => if(in1 = ‘1’) then STATE <= "1101";
      code_word <="0011";
    else STATE <= "1010"; code_word <="0110"; end if;
  when "1010" => if(in1 = ‘1’) then STATE <= "1001";
      code_word <= STATE or code_word;
    else STATE<="1000"; code_word<=STATE and code_word; end if;
  when "1000" => if(in1 = ‘1’) then STATE <= "1100";
      code_word <= STATE(3) & code_word(2 downto 0);
    else STATE <= "1110"; code_word <= not code_word;end if;
  when "1001" => if(in1 = ‘1’) then STATE <= "1110";
      code_word <= STATE(3 downto 1) & in1;
    else STATE <= "1101"; end if;
  when "1110" => STATE <= "0011";
      code_word <= in1 & code_word(3downto1);
  when "1101" => STATE <= "1100";
      code_word <= code_word(1 downto 0) & STATE (3 downto 2);
  when "1100" => STATE <= "1111";
      code_word<= code_word(3)&(not in1) & code_word(1downto0);
  when "1111" => STATE <= "0011";
      code_word <= (not in1) & code_word(2 downto 0);
  when "1011" => STATE <= "0111";
  when "0011" => if(in1 = ‘1’) then STATE <= "0101";
    else STATE <= "0010"; end if;
  when "0010" =>        STATE <= STATE xor code_word;
  when "0101" =>        STATE <= "0111";
  when "0111" =>  STATE <= "0110";
  when "0110" =>        if(in1 = ‘1’) then STATE <= "0001";
    else STATE <= "0101"; end if;
  when "0001" =>        STATE <= "0010";
  when "0100" =>        STATE <= "0111";
  when others =>        null;
end case;
```

Figure 3.6: Example obfuscated RTL design.

in this example, a variable "code_word" is used for clarification purposes. In a real design if code has to be given to untrusted entity, one can easily replace the variable name with an obfuscated name. In such a scenario software obfuscation techniques can be applied to the RTL code to protect the Intellectual Property (IP).

3.4 Design Flow

The complete implementation is at the RTL level and can be modeled using any Hardware Description language. For demonstration of the idea, the proposed method was implemented using designs from OpenCores [1]. Figure 3.7 shows the steps followed in the design process for the proposed technique.

Once the cores are obtained, the first step is to identify an FSM which is the heart and controls most of the assignments and computations. The states of this FSM are expanded and classified into extended states and functional states. State encoding for the functional states is kept as required for the design. Entry mode state transition graph is designed such that it contains loops of states and the correct path is difficult to figure out. The number of transitions and states used in the entry mode depends on the area overhead acceptable. Area overhead increases with the increase in number of transitions.

Functional mode is modified and the value of the Code-Word to be formed is calculated as per the function and the state encoding. Size of the Code-Word depends on the area overhead allowed by the designer/user. Longer Code-Words imply that the circuit can sustain higher levels of brute force attack. The computations and assignments in the entry mode graph are chosen such that the Code-Word changes for any change in the input sequence and also state transition. Finally, a set of inputs is then obtained to form the key to make the circuit functional.

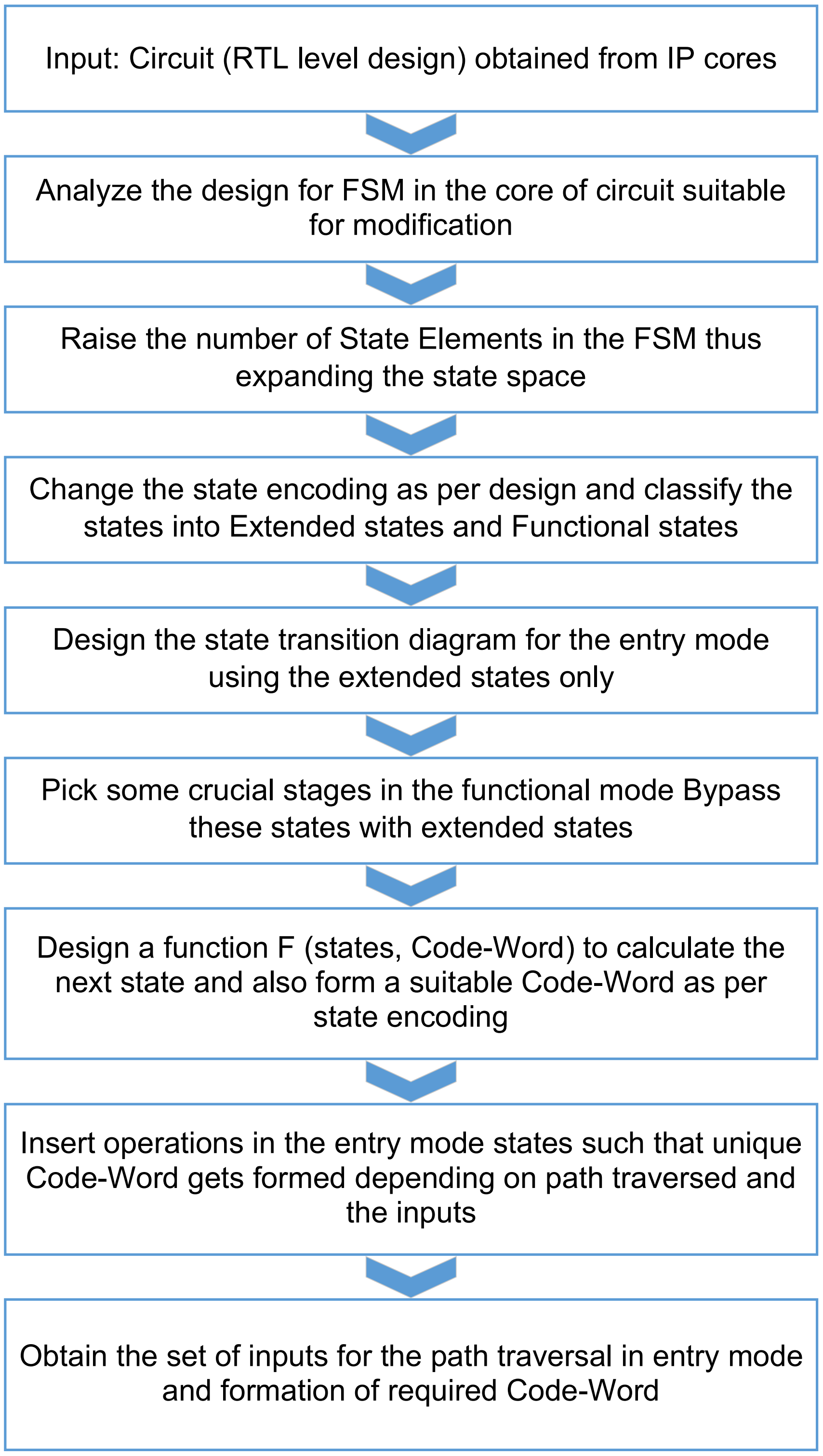
Input: Circuit (RTL level design) obtained from IP cores
Analyze the design for FSM in the core of circuit suitable for modification
Raise the number of State Elements in the FSM thus expanding the state space
Change the state encoding as per design and classify the states into Extended states and Functional states
Design the state transition diagram for the entry mode using the extended states only
Pick some crucial stages in the functional mode Bypass these states with extended states
Design a function F (states, Code-Word) to calculate the next state and also form a suitable Code-Word as per state encoding
Insert operations in the entry mode states such that unique Code-Word gets formed depending on path traversed and the inputs
Obtain the set of inputs for the path traversal in entry mode and formation of required Code-Word

Figure 3.7: Design Flow for Anti-Tamper Techniques

3.5 Implementation

For implementation purposes, an open source core from Open Cores [1] was taken. Circuit chosen for test was an AES design. The following changes were made in the design:

(1) Only the core FSM of the AES design was padded up with three more bits thus raising it from 5 to 8. Now the number of states available in the extended state is 256-32 = 224.

(2) Code-Word was chosen to be of size 48 bits. For the state transition in the entry mode, 64 inputs (out of 130 inputs present in the design) were used to reduce the hardware overhead. Design was implemented in Virtex 7, and simulations were performed using Xilinx ISE.

Implementation of the Code-Word based control in the functional mode helps to achieve dynamic nature of the circuit to various incorrect input sequences. It is very difficult to distinguish between the Code-Word, the FSM state elements and other state elements in the whole circuit. Due to implementation in the RTL level, the components protecting circuit can be said to be highly obfuscated.

3.6 Results

One of the worst possible scenarios is when the adversary has a previous working model of a chip. The designer has launched a new model which has improved performance metric. In this case the adversary knows circuit completely. The adversary can perform known answer tests to verify whether he/she has unlocked the IC or not.

Consider that the adversary has obtained information of other state elements in circuit except the main FSM and the Code-Word as we have not altered the other parts of the circuit. The best way to attack the chip is to separate the Code-Word (48 state elements) and core FSM (8 State Elements). The adversary knows the assignments made in the states of the original circuit. So, it would be best for him to find the main state elements and then the valid states accordingly. Since there are 56 state elements (8 + 48) to be searched for the number of combinations to find state elements is

$$\sum_{k=1}^{56} \binom{56}{k}$$

where, ‘k’ is all possible sizes of Code-Word

Next, he/she will try to find a state in which the assignments performed are similar to the original circuit assignments. The adversary has to identify the obfuscated 32 functional states.

No. of states in the old circuit(state elements 5) = n = 32

Number of computations required for comparison for predicting states with similar assignments= n. Thus the total number of iterations involved is given by

$$\left(\sum_{k=1}^{56} \binom{56}{k}\right) * (n) = 2.30e18$$

To find the next state, the best possible way is to find the next state in the original circuit and map it to the state in new circuit. These comparisons have been done in an earlier step so there is no increase in computations. In calculating this case, we have considered all the best possible scenarios in terms of the attacker. But still the number of computations required is 2.3e18, which is enormous.

In practice, the number of computations required would be larger than this, especially if the adversary does not have any other circuit to compare with. At each stage of the combination he/she will have to guess the value of the Code-word which requires $2^{(56-k)}$ number of iterations. In that case, the total number of iterations will be

$$\sum_{k=1}^{56}\left(\binom{56}{k} * 2^{(56-k)}\right) = 5.233e26$$

Table I shows the hardware overhead in the implementation of these protection mechanisms. For the AES design, the area overhead of our method is 7.8%, compared to a recent technique [33] that needed 8.6%. The strength of our method provides a higher level of protection, in which 5.233e26 combinations is needed, while an average of 7.13e15 as needed in [33] using similar calculations. Average values are considered here as the implemented designs are different.

Table 3.1: ANTI-TAMPER RESULTS

Design	Area Overhead		Number of Combinations Required		
	Ours	[33]	Ours		Average in [33]
			(min)	(max)	
AES	7.8%	8.6%	2.3e18	5.233e26	7.13e15

3.7 Advantages Interlocking Obfuscation Technique for Anti-Tamper Hardware

The advantages of the proposed method are listed below.

1. As the Code-Word is embedded within the transition function itself, it creates a new dynamic nature to the circuit behavior making reverse engineering more difficult. The

circuit response to incorrect sequences is different due to the dynamic Code-Word formed.

2. The method provides active defense mechanism to the circuit even in the functional mode as the important states are bypassed with the help of Code-word.

3. Once the user is authenticated the specifications of the chip become identical to the original hardware. There is no change in the input and output specifications.

4. Method proposed implements protection mechanism at the RTL level. Thus any HDL can model this technique making it language and platform independent.

5. As a side benefit any minor changes in the circuit by untrustworthy parties get magnified due to the nature of the obfuscated design. Thus this method is also helpful to fight attacks.

Chapter 4

Anti-counterfeit Integrated Circuits Using Tamper-Resistant Time-stamp Circuitry

In this chapter we extend our methodology used to tamper proof Integrated circuits to tackle the problem of counterfeit ICs. We address the counterfeit problem with a novel **time-stamp** methodology as a hybrid anti-tamper and counterfeit detection. The time-stamp by itself can offer the manufacturing date, and the surrounding anti-tamper circuitry makes it difficult even for more sophisticated attacks to modify the time-stamp.

4.1 Motivation

Let us analyze counterfeiting benefits in the adversary's point of view. The adversary will make alterations in the circuit to modify the time-stamp if he/she has some benefit from it. By making alterations in the time-stamp circuit the adversary would be interested if the new manufacturing date produced is realistic. In today's world of fast changing technology, lifetime of a particular IC is around 10 years. Few important ICs used in defense system have a lifetime of 20-25 years. Lets consider max lifetime of an IC is to be 30 years. The adversary will only be interested to have the time-stamp circuit produce an output ±30 years from the original date. If the time-stamp circuit is tampered and produces a date which is more than 30 years apart from the original, it will not do the adversary any profit and is highly unrealistic. Thus the aim is to have a time stamp circuit such that even if tampered, it does not give out a date which is ±30 years from the original date.

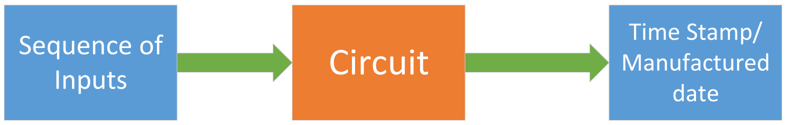

Figure 4.1: Timestamp process

The process to obtain the time-stamp from the circuit is seen as a challenge response type system. Figure 4.1 shows the process to obtain the time-stamp from the circuit under test. A input sequence to drive the circuit in time-stamp mode is applied. At the end of input sequence the circuit output is taken as the response to the query which is the manufactured date.

4.2 Time-Stamp Circuit

Our proposed method involves the use of boosted FSM as in the anti-tamper method, entry mode path traversals [18] and a low area-overhead Linear Feedback Shift Register (LFSR) circuit with a custom tap configuration that will generate the manufactured date. A specific path in the entry mode is traversed so that the circuit gives out the embedded manufacturing date. In addition to the date, the time-stamp can also include information such as version, fab, etc.

4.2.1 LFSR Circuit

A Linear Feedback Shift Register (LFSR) is a register whose next state is a linear function of its previous state. The commonly used linear function for building LFSR is the XOR function. The initial value of the LFSR is called a "seed." The next state of the LFSR is completely determined by its present state. The bit positions in the shift register that affect the feedback function are called "taps." A LFSR has a finite number of states, hence it eventually enters in a cycle of repeating states. However, the length of the cycle is generally very long. For example,an n-bit LFSR with a properly chosen configuration can loop through a cycle of up to $2^n - 1$ states. LFSRs are deterministic in nature, the sequence of states produced by the register is completely determined by its current state.

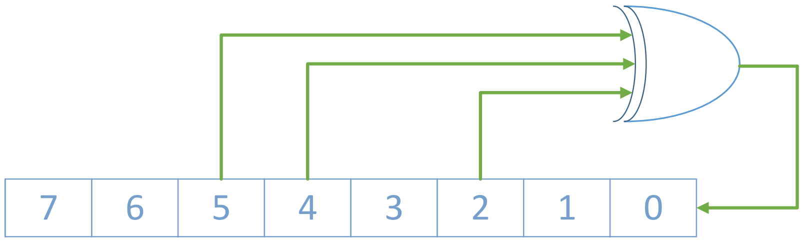

Figure 4.2: Linear Feedback Shift Register

Figure 4.2 shows a 8 bit LFSR with XOR as the feedback function. Bits "2", "4" and "5" are taps in this example. The bits in this LSFR are left shifted at each clocking instance and the vacant position in bit "0" is filled by the XOR result of the tap bits. Thus a sequence of states is obtained from the initial state termed as the "seed." For a conventional tapping for a 8 bit LFSR the cycle loops through a finite $2^8 - 1 = 255$ states.

LFSR circuit have been extensively used for Logic Built In Self Test (LBIST) [36, 37]. They are also useful for random number generation [38]. LFSR structure is easy to design and implement as the designer has lot of options in terms of the initial seed to choose and also the tapping. But it is extremely difficult to reverse engineer because a large combination of seeds and taping can lead to a certain state. Backtracking from a current state of LFSR is thus difficult.

Our time-stamp uses a LFSR in a non-conventional manner where the configuration does not allow for the maximum number of possible states ($2^n - 1$). The tap configuration of the LFSR is designed such that the possible number of states of the LFSR is limited. The key feature of this configuration is that the values that can be produced in the LFSR at the end of 'm' transitions is a finite set of values.

Let 'S' be the set of states of the LFSR during conventional tapping for max states i.e all $2^n - 1$ states. 'V' be the set of states reached by LFSR after'm' transitions in our constrained tap configuration.

$$V \subseteq S$$

$$V = \{v_1, \quad v_2, \quad v_3, \quad ..., \quad v_k\}$$

Let these values v_1, v_2, ..., v_k denote the year for simplicity of discussion. Then, one of the values v_i is 2013 which is the output of the time-stamp circuit.

$$v_i \in V$$

And all other values in the set has a distance from v_i of at least d. That is, the nearest year that can be represented from this set is either 2013+d or 2013$-d$. In constrained tapping the value of 'd' can be set according to the designers requirement.

Figure 4.3 shows a LFSR capable of both normal and constrained configurations. A mode signal to the multiplexer selects the configuration to be used normal or constrained configuration of tapping. The normal configuration will cycle the LFSR in all $2^n - 1$ states and can be used for Logic Built-in-self-test purposes [36, 37] or random number generation [38].

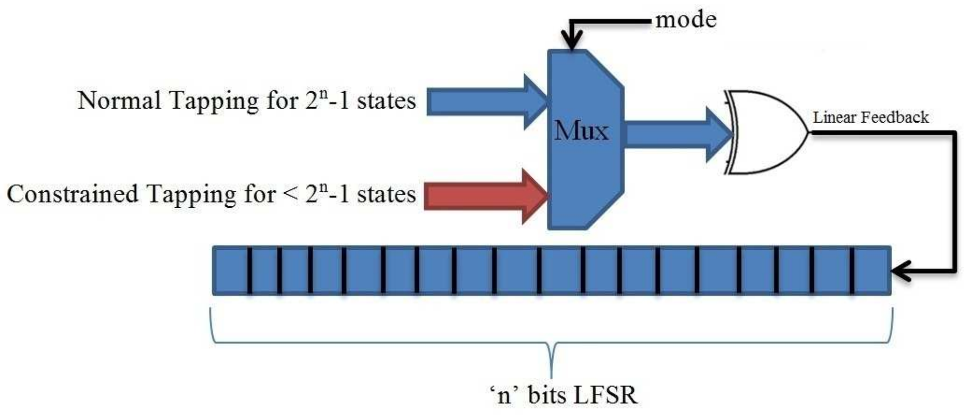
mode
Normal Tapping for 2ⁿ-1 states
Constrained Tapping for < 2ⁿ-1 states
Mux
Linear Feedback
'n' bits LFSR

Figure 4.3: Multi-mode Use of LFSR-based Time-stamp Circuit

The constrained configuration is used for time-stamp mode of operation. Such a multi-mode allows us to embed the time-stamp without paying additional hardware cost, especially if BIST capabilities are already present in the circuit.

4.2.2 Path Traversal in Time-stamp mode

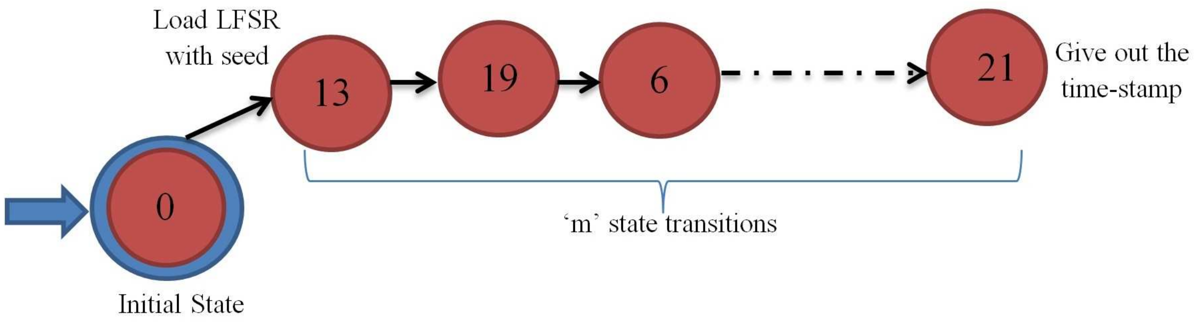

Figure 4.4: Path traversal in entry mode for time-stamp

A fixed path is to be traversed by the circuit in the time-stamp mode for 'm' transitions. The state at the end of this traversed path reads out the time-stamp from the LFSR. Changing the functionality of the circuit to have this path is not a feasible option. For this path we utilize entry mode states obtained by anti-tamper methods in Chapter 3. Expansion of states is done by increasing the number of state elements in core FSM.

A fixed path for the circuit to traverse can be assigned with the help of the invalid states as shown in Figure 4.4. The seed for the LFSR is stored in memory. A transition to state '13' loads the seed into the LFSR. The mode of the multiplexer for the LFSR circuit as in Figure 4.3, is switched to the time-stamp mode and constrained tap configuration is fed to linear function XOR. The circuit is driven through the fixed path for the time-stamp having 'm' transitions. The LFSR automatically brings out the embedded info (date) at the m^{th} state.

Considering the above setup, the adversary would wish to have a different output date at the end of the traversed path. In case of the adversary the beneficial outputs would be ± 30 years from the original date i.e. $d < 30$. For example, if the embedded date is 2013, the desired date in case of adversary would be between 1983 and 2043. Anything above and below that is highly unrealistic and the adversary's aim would not be to achieve such a date. So the aim of our time stamp circuit is not to allow any change to the time-stamp such that output date is between 1983 to 2043 excluding 2013 (the original embedded date).

Since it is very difficult to make changes in the LFSR circuit once the manufacturing is complete such that it can give a different date at the end of the n^{th} state, it may be easier to attack the seed instead of the LFSR configuration as the seed is stored in the memory. So for all the possible seed values must be tried by the adversary in the worst case. Furthermore, we design our LFSR time-stamp configuration such that no seed can yield a year that is within $d = 30$ years of the current year. So in our example where the embedded date is 2013, the configuration is such that there is no seed which can yield an output a valid year from 1983 to 2043 except 2013.

In the preceding discussion, changes to neither the seed nor LFSR configuration can result in a successful change in the resulting "date", even if the date is more than $d = 30$ years away. This serves as a first layer of tamper resistance. However, what happens if the adversary attempts to attack the time-stamp by changing *both* the seed and the LFSR configuration?

The next section proposes adding a second layer of tamper-resistance to the time-stamp that would guard against such more sophisticated attacks.

4.3 Tamper-Resistant Time-Stamp Circuit

In this section, we present an approach to make the time-stamp circuit resistant to more sophisticated attacks. Such attacks imply the addition, modification, or deletion of hardware to bring about changes beneficial for the adversary. The purpose of attack on time-stamp circuit would be to change the embedded info (date).

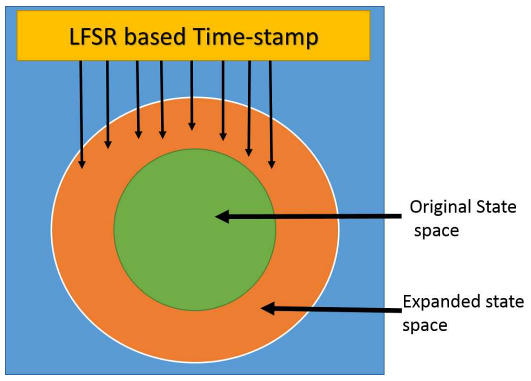

Figure 4.5: Integration for tamper-resistant time-stamp

In the case where the adversary launches a combined change to both the seed and the LFSR configuration, we propose a few additions to the time-stamp circuit to make it more tamper-resistant. It should be clear that in this approach we make only the time-stamp circuit (a small fraction of the circuit) tamper-resistant and not the entire circuit.

In order to ensure the validity of the date given by the time-stamp, we need to obfuscate the hardware used to detect tampering of the time-stamp. To successfully tamper the time-stamp circuit, the adversary will have to know completely the tamper detection hardware.

Clever obfuscation of the tamper detection circuit will make it extremely difficult for the adversary to bring about meaningful and fruitful changes in time-stamp circuit.

In Chapter 3 and [18] we have proposed a method that increased the level of protection against tampering by an expansion of states, which is achieved by increasing the number of present state elements in a Finite State Machine (FSM) in its core logic. The overall functionality of the circuit is divided into two modes: Entry mode and Functional mode. Building on this idea, we use a few paths from the entry mode for time-stamp having 'm' transitions. On traversal of particular time-stamp path of 'm' transitions the circuit reaches a state which gives the time-stamp value to the output.

The main idea we use follows in a similar line of thought, except that we obfuscate only the tamper detection circuitry using the original logic of the chip. The number of states present in the original logic of the circuit is boosted by adding up a few more state elements. These added states are then used to obfuscate the tamper detection logic.

Figure 4.5 shows the integration of LFSR based time-stamp circuit with the main circuit logic. As shown in this figure, the number of states in the main logic of the circuit are expanded by adding few more state elements.

In a normal time-stamp scenario the value of the next state in the path traversal is a function of present state and inputs. Thus state transitions during the time-stamp mode is given by:

$$next\_state = f(present\_state, inputs)$$

For the tamper resistant time-stamp, we make some of the transitions during the time-stamp mode dependent on the LFSR value. Thus for these modified transitions the value of the next state is computed by:

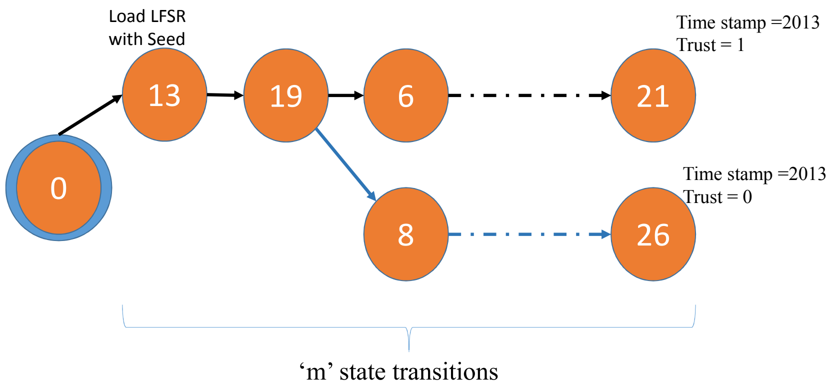

Figure 4.6: Path traversal for tamper resistant time-stamp

$$next\_state = f(present\_state, inputs, LFSR\_value)$$

In this approach, we also define another parameter called the "Trust Bit". This bit is accompanied with the time-stamp output which is at the end of 'm' transitions in time-stamp mode. It is used to convey to the user whether the time-stamp circuit is tampered or not. A logic one of this bit at the end of 'm' transitions denotes that the time-stamp provided is valid and not tampering has been done.

Figure 4.6 shows the path traversal for the tamper-resistant time-stamp in two different scenarios. The correct time-stamp output for the implementation shown is "2013". For the simplicity of discussion only one state, i.e., state '19' has been modified such that its next state is dependent on the value of LFSR. In reality many states have to be modified to cover all possibilities of tampering. When the circuit reaches state '19' the computation of the next state is performed on the present value of a few bits in the LFSR and inputs. Note that no comparator is used here so as to achieve higher obfuscation. The next state is purely a function of the current state and the LFSR values. If the value of LFSR is correct the circuit will transition to the correct next state which is state '6'. Thus, if no tampering is made, the circuit reaches the desired state '21' at the end of 'm' transitions, giving out the time-stamp and making the trust bit high. However, if the computation of the next state when circuit is at state ‘19’ is incorrect the circuit may take one of the many possible paths. An incorrect path from state ‘19’ is shown in the Figure 4.6 in blue where the circuit transitions to state ‘8’ and then finally ‘26’ after ‘m’ transitions. But ‘26’ was not the correct state to be in after ‘m’ transitions and thus outputs the tampered value of LFSR which could be ‘2013’ but the trust bit does not go high.

The main working principle is, to successfully tamper the circuit the adversary has to make sure that the tampered circuit has the intermediate states same as the old circuit. In addition the adversary does not know which states of the LFSR are used for state transition in the entry mode. It is impossible to have a different state of LFSR at the end of ‘m’ transitions which is the time-stamp while keeping the intermediate states the same. Results show that adversary has to go through a lot of combinations to tamper the circuit and thus will not be successful in realistic amount of time.

4.4 Design Flow

The complete design of the hardware is done at the Register Transfer Level. Designer can pick up any hardware description language of his preference. Intellectual Property (IP) cores can be obtained from OpenCores [1]. Figure 4.7 shows the design process followed for time stamp circuit and also the tamper proof time-stamp circuit.

As explained in Chapter 3 the initial step is state expansion and classification into Entry mode states and the Functional mode states. The LFSR circuit is designed with proper tapping and seed to produce proper time-stamp at the end of ‘m’ transitions. A path is dedicated in the Entry mode for time-stamp to traverse for ‘m’ transitions. The LFSR

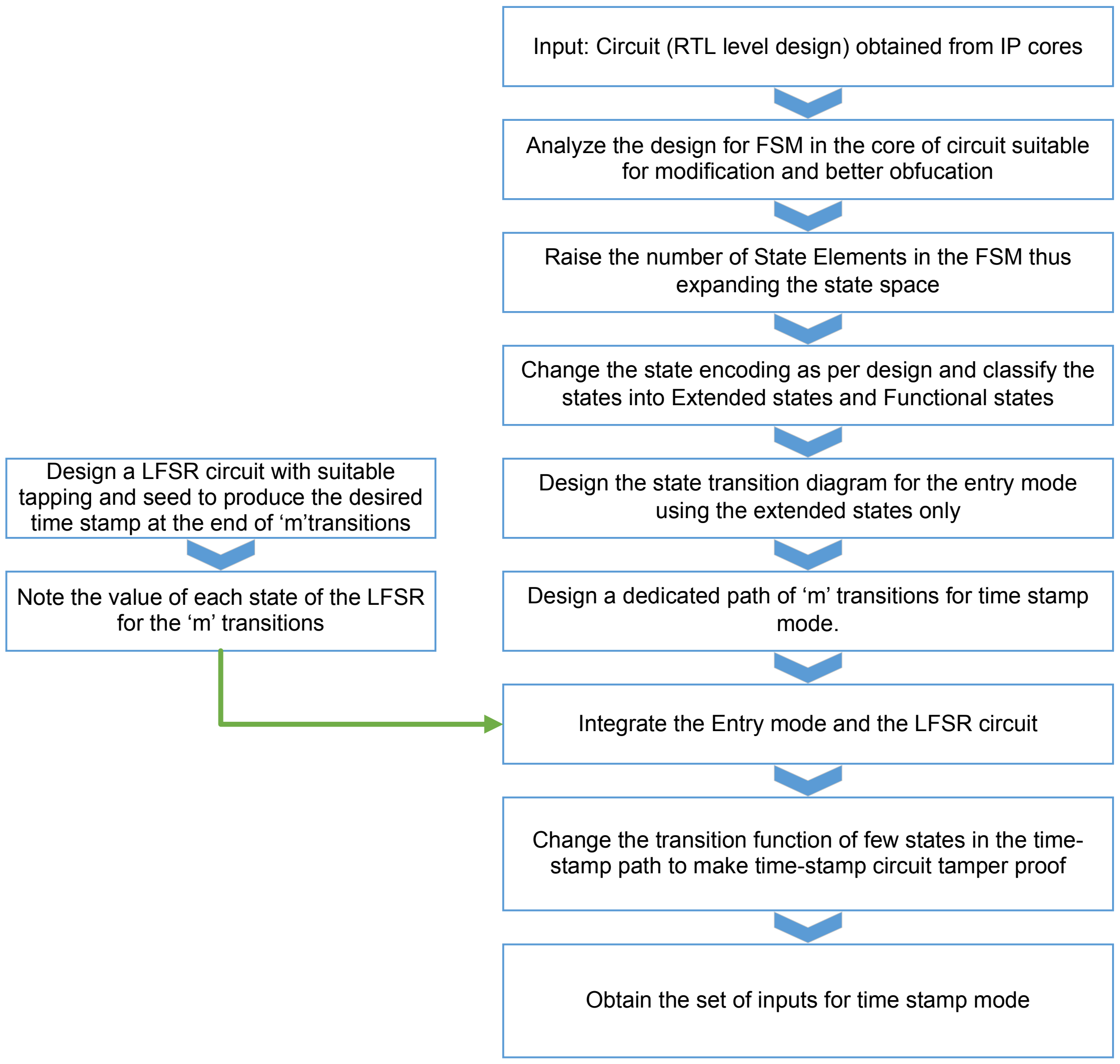
Input: Circuit (RTL level design) obtained from IP cores
Analyze the design for FSM in the core of circuit suitable for modification and better obfucation
Raise the number of State Elements in the FSM thus expanding the state space
Change the state encoding as per design and classify the states into Extended states and Functional states
Design the state transition diagram for the entry mode using the extended states only
Design a dedicated path of 'm' transitions for time stamp mode.
Design a LFSR circuit with suitable tapping and seed to produce the desired time stamp at the end of 'm'transitions
Note the value of each state of the LFSR for the 'm' transitions
Integrate the Entry mode and the LFSR circuit
Change the transition function of few states in the time-stamp path to make time-stamp circuit tamper proof
Obtain the set of inputs for time stamp mode

Figure 4.7: Anticounterfeit Design Method

circuit is integrated with the Entry mode logic such that at the end of this path the circuit gives out the time-stamp.

The Time-Stamp circuit can be made tamper proof by changing the transition function in the Entry mode as explained in Section 4.3. Acceptable area overhead plays an important role in the design. Larger allowed area overhead implies larger obfuscation of the time-stamp circuit as large number of states in the Entry mode can be modified to take into account the value of LFSR circuit.

4.5 Implementation

4.5.1 Time Stamp Circuit

The time-stamp circuit in our setup was implemented with a 16-bit LFSR to encode the value of the manufactured year. Each digit of the year having a 4-bit representation. The output was taken at the end of 50 cycles. The average number of tapping/year to embed = approx. 15, i.e., there are approximately 15 different taps which can be used by designers to encode each year. There are different possible seeds for that year for different tapping. For year = 2012, the tap configuration is 0x05b6.

The **closest** possible years with different seeds are 1941 and 2062. Thus, the adversary cannot modify the seed to bring a small change to year 2012+d where d is a small value. Considering a constant seed and changing the tap configuration is also impossible. For year = 2012, seed = 0x2004. If the adversary tries to change the tap configuration with hamming distance of 1 bit from 0x5b6 the closest the time-stamp will get is 1326 or 4979. Changing the tap configuration with more than 1 bit is a more difficult task, both computationally and physically. Still, any change would result in the time-stamp to exceed 2060. To bring about a change in the time-stamp while giving a batch for manufacturing, only the constrained tap configuration of the circuit and seed in memory need to be simultaneously attacked.

4.5.2 Tamper Resistant Time Stamp Circuit

For the implementation of tamper-resistance circuit around the time-stamp, the following modifications were made:

(1) Only the core FSM of design was padded up with additional bits.

(2) The number of states available as extended state is computed.

(3) Using the extended states, several paths were designed for the time-stamp mode. Transitions were designed such that any deviation in the LFSR output would result in an altogether different state at the end of '50' transitions.

(4) Test Benches were written to check the proper functioning of the implementation.

The time-stamp circuit was implemented on two benchmark circuits, AES and DES, from the Opencores benchmark suite [1]. Each design was implemented in Virtex 7, and simulations were performed using Xilinx ISE. Area overhead was computed to have an idea of the cost of implementation.

4.6 Results

Table 4.1: Area Overhead for the Time-Stamp

Design	Area overhead for only time-stamp	Area overhead for Tamper resistant time-stamp
AES	0.3%	8.8%
DES	1.189%	18.66%

Table 4.1 shows the area overhead for the proposed time-stamp circuitry as a percentage of the overall circuit. From Table 4.1 we can conclude that the area overhead for just the time-stamp circuit is almost negligible. But to make the time-stamp tamper-resistant requires a significant overhead. This cost is incurred due to adding state elements and transitions to the design.

Among the two designs, AES is a comparatively larger design compared with DES. Hence we can see the rise in area percentage for both implementations is smaller for AES. The number of transitions required to be introduced to obfuscate the LFSR based time-stamp will be approximately the same for all designs as the time-stamp circuit does not change with design. Hence with an increase in size of the design the area overhead for tamper resistant circuits will be comparatively lower.

Adversary will have to do a brute force attack to counterfeit an IC. Consider the scenario where the adversary has a chip which has a time-stamp of 2010 and his/her intent is to produce a time-stamp output 2013.

The adversary has to find a feasible path in the time-stamp mode such that after the end of 'm' cycles the state of IC with time-stamp 2010 is equal to the state of IC with time-stamp 2013.

It will be profitable for the adversary to know the best favorable path during the time-stamp mode of the 2010 IC such that least modifications in the circuit can make the IC give out a time-stamp of 2013 with Trust bit 1.

Let us consider the adversary knows the number of cycles 'm' ($i.e.$50) that the designer has used. So the adversary has to find a path of 50 cycles in the extended FSM where least modifications are required to be done.

The extended FSM has 8 state elements in both AES and DES implementations. Thus the total number of states in extended FSM $= 2^8 = 256$

While choosing a path the order of states is important. Also, in a path repetition of states is allowed.

Thus the total number of computations the adversary has to compare is given by

$$No.\ of\ computations\ required\ =\ \frac{n!}{(n-m)!\,(m!)}$$

$$where\ n\ =\ total\ number\ of\ states\ =\ 2^8\ =\ 256$$

$$m\ =\ number\ of\ cycles\ in\ time\ stamp\ mode\ =\ 50$$

$$No.\ of\ computations\ required\ =\ \frac{256!}{(256-50)!\,(50!)}$$

$$=\ 1.0008e^{118}$$

Thus the adversary does not have a realistic chance to tamper the time-stamp circuit in limited time and the limited number if available resources.

4.7 Advantages of Anti-Counterfeit methods with Tamper-Resistant Time-stamp Circuitry

The advantages of the proposed method are listed below.

1. A novel time-stamp method is introduced in this Chapter for Counterfeit detection.
2. Small change in the LFSR circuit has a great impact on the end result rendering it useless for the adversary.
3. This method provides much more beneficial compared to traditional methods of bar-codes/serials or metering.
4. Methods do not require the Original Equipment Manufacturers (OEM) to maintain any database to detect counterfeits which is practically impossible.
5. To guard the circuit against sophisticated attacks, another layer of protection is added to the circuit.

6. The techniques are easy to embed in a system and can be implemented easily by the designer at the Register Transfer Level. The methods are language and platform independent.

7. The proposed method does not require changing of the functionality of the circuit. Only an additional time-stamp mode is added to prove the authenticity of the IC.

Chapter 5

Conclusion

In this thesis we present techniques for making circuit resistant to Intellectual property theft and also detection of counterfeit Integrated Circuits (ICs). It is important to safeguard the circuits against reverse engineering process aimed at obtaining information from the chip. Also, it is extremely important to prove the authenticity if the chips and be able to distinguish between counterfeits and genuine ICs.

A new interlocking obfuscation technique is proposed for active defense mechanisms of the circuit against tampering. The area overhead for implemented design is less than 8%. The circuit response to incorrect sequences is different due to the dynamic Code-Word formed. As the Code-Word is embedded within the transition function itself, it creates a new dynamic nature to the circuit behavior making reverse engineering more difficult. Our method suggests bypassing of the important states with extended states to add up to the confusion of the adversary. The specifications of the chip are not altered and is identical to original specifications once user is authenticated. Inputs and outputs are kept the same.

Static algorithms for detecting the correct path to functional mode may yield certain results but the method ensures that only one path in the entry mode when traversed helps unlock the circuit. Higher levels of protection can be achieved if one allows for a larger area overhead. Thus, there is a tradeoff between area overhead and level of protection achieved.

We propose an efficient time-stamp having an area overhead of only 0.74% for counterfeit detection for future designs. A small change in the LFSR results in a date which is of no use to the adversary, i.e., the resultant date is in distant past or future beyond the life expectancy of an IC. The proposed methods have high merits over traditional serial or barcodes and even on high area overhead watermark techniques. These methods do not require to have a database of ICs as is required by most of the metering techniques.

Furthermore, a second layer of tamper-resistance is proposed to the time-stamp which guards it against more sophisticated attacks that attack both the seed and the LFSR configuration. The cost for implementing tamper resistant circuit is an area overhead of 8.8% for the AES circuit. However, with increase in circuit size, the area overhead drops down as the complexity of the time-stamp circuit does not change with design.

Both the proposed techniques for anti-tamper and anti-counterfeit are easy to embed and practical to use, and it can be implemented directly at the RTL level. Thus, any HDL can model this type of circuit making it language and platform independent. Both the methods complement each other and can co-exist in the same design.

The designs presented use obfuscated circuit to solve the problems. Thus the adversary will not be able to completely understand the complete functionality of the circuit. As a side benefit any minor change in the circuit by third party gets magnified due to the obfuscated design. This helps protect against any malicious hardware trojans being inserted by adversary.

5.0.1 Future Work

In the future, other methods can be used to decrease the area overhead. The use of PUFs can provide high dynamic nature to circuit output, hence a higher level of obfuscation. Computational parameters such as resource utilization can be used for further obfuscation. Further analysis can be performed on the LFSR configuration such that it is extremely hard to tamper both the tapping and seed even with standalone time-stamp circuit. Also, a mathematical formulation to give the best suitable tapping and seed for a given date can be conducted.

Bibliography

[1] URL: http://opencores.org/

[2] Foreign infringement of intellectuaal property rights implications on selected U.S. industries. *available online at* http://www.usitc.gov/publications/332/working_papers/id_14_100505.pdf.

[3] National Strategies for Smart Grid, Cybersecurity and Supply Chain (Chapter 8)*available online at* http://www.usresilienceproject.org/workshop/participants/pdfs/USRP_Resources_Chapter_8_030212.pdf

[4] Inquiry into Counterfeit Electronic parts in the Department of Defense supply chain by Committee on Armed Services United States Senate *available online at* http://www.levin.senate.gov/download/?id=24b3f08d-02a3-42d0-bc75-5f673f3a8c93

[5] Defense Industrial Base Assessment: Counterfeit Electronics by U.S. Department of Commerce *available online at* http://www.bis.doc.gov/defenseindustrialbaseprograms/osies/defmarketresearchrpts/final_counterfeit_electronics_report.pdf

[6] Tony Bent, "Counterfeit IC Components: An Industry Perspective," *National Semiconductor Corporation, available online at* https://www.navalengineers.org/SiteCollectionDocuments/2011%20Proceedings%20Documents/CTC/Bent.pdf

[7] A. T. Abdel-Hamid, S. Tahar, and E. M. Aboulhamid, "A Public-Key Watermarking Technique for IP Designs," *Proc. of the Conf. on Design, Automation and Test in Europe (DATE)*, 2005, vol. 1, pp. 330–335.

[8] F. Koushanfar, I. Hong, and M. Potkonjak, "Behavioral synthesis techniques for intellectual property protection," *ACM Transactions on Design Automation of Electronic Systems*, 2005, vol. 10, pp. 523–545.

[9] R. Majumdar and J. L. Wong, "Watermarking of SAT using combinatorial isolation lemmas," *Proceedings of the 38th annual Design Automation Conference (DAC)*, 2001, pp. 480–485.

[10] DAGE - A Nordstorm Company, "Detecting Counterfeit Components," *Chip Scale Review,* 2008.

[11] R. Moudgil, D. Ganta, L. Nazhandali, M. S. Hsiao and C. Wang, "A novel statistical and circuit-based technique for counterfeit detection in existing ICs,"*Proceedings of the 23rd ACM international conference on Great lakes symposium on VLSI, GLSVLSI'13*, 2013, pp. 1-6.

[12] T. H. Kim, R. Persaud, and C. Kim, "Silicon odometer: An on-chip reliability monitor for measuring frequency degradation of digital circuits," *Solid-State Circuits, IEEE Journal of*, 2008, vol. 43, pp. 874-880.

[13] F. Koushanfar, "Provably Secure Active IC Metering Techniques for Piracy Avoidance and Digital Rights Management," *IEEE Transactions on Information Forensics and Security*,2012, vol. 7, pp. 51-63.

[14] F. Koushanfar, "Integrated circuits metering for piracy protection and digital rights management: An overview," p. Invited Paper, 2011.

[15] F. Koushanfar and G. Qu, "Hardware metering," in *Design Automation Conf. (DAC)*, pp. 490 – 493, 2001 2001.

[16] Y. M. Alkabani and F. Koushanfar, "Active hardware metering for intellectual property protection and security," *Proc. USENIX Security Symp.*, 2007, pp. 20:1-20.

[17] S. Devadas, E. Suh, S. Paral, R. Sowell, T. Ziola and V. Khandelwal, "Design and implementation of puf-based "unclonable" rfid ics for anti-counterfeiting and security applications," in *IEEE International Conference on RFID*, 2008 , pp. 58-64.

[18] A. R. Desai, M. S. Hsiao, C. Wang, L. Nazhandali and S. Hall, "Interlocking Obfuscation for Anti-Tamper Hardware," *Proc. Annual Workshop on Cyber Security and Information Intelligence Research (CSIIRW)*, 2012.

[19] M. Tehranipoor and C. Wang, "Introduction to Hardware Security and Trust," Springer, August 2011.

[20] D. Rai and J. Lach, "Performance of Delay-Based Trojan Detection Techniques under Parameter Variations," in *Proc. of IEEE International Workshop on Hardware-Oriented Security and Trust (HOST),* 2009, pp. 58-65.

[21] J.Aarestad, D. Acharyya, R. Rad, and J. Plusquellic, "Detecting Trojans through leakage current analysis multiple supply pad IDDQs," in *IEEE Transactions on Information Forensics and Security,* 2010 vol. 5, pp. 893-904.

[22] S. Wei, S. Meguerdichian, and M. Potkonjak, "Gate-level characterization: foundations and hardware security applications," *Proceedings of the 38th annual Design Automation Conference (DAC)*, 2010, pp. 222-227.

[23] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi and B. Sunar, "Trojan Detection using IC Fingerprinting," *IEEE Int0l Symp. on Security & Privacy,* 2007, pp. 296-310.

[24] R. Rad, X. Wang, J. Plusquellic and M. Tehranipoor, "Power Supply Signal Calibration Techniques for Improving Detection Resolution to Hardware Trojans," in *Proc. of the Int0l Conf. on Computer-Aided Design,* 2008, pp. 632-639.

[25] X. Zhang and M. Tehranipoor, "Case study: Detecting Hardware Trojans in third-party digital IP cores," in *Proc. of IEEE International Workshop on Hardware-Oriented Security and Trust (HOST),* 2011, pp. 67-70.

[26] M. Banga and M. S. Hsiao, "Trusted RTL: Trojan detection methodology in pre-silicon designs," in *Proc. of IEEE International Workshop on Hardware-Oriented Security and Trust (HOST),* 2010, pp. 56-59.

[27] Kan Xiao and Mohammed Tehranipoor, "BISA: Built-In Self-Authentication for Preventing Hardware Trojan Insertion," in *Proc. of IEEE International Workshop on Hardware-Oriented Security and Trust (HOST),* 2013. Used under fair use, 2013

[28] R. Porter, S. J. Stone, Y. C. Kim, J. T. McDonald and L. A. Starman, "Dynamic Polymorphic Reconfiguration for anti-tamper circuits," in *International Conference on Field Programmable Logic and Applications (FPL)*, 2009, pp. 493-497.

[29] K. Nohl, D. Evans, S. Starbug and H.Plotz, "Reverse-engineering a cryptographic RFID tag," in *the Proceedings of the 17th conference on Security Symposium (SS'08)*, 2008, pp. 185-193.

[30] F. Junfeng, G. Xu, E. De Mulder, P. Schaumont, B. Preneel and I. Verbauwhede, "State-of-the-art of secure ECC implementations: a survey on known side-channel attacks and countermeasures," in *IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, 2010, pp. 76-87.

[31] R. Chakraborty and S. Bhunia, "HARPOON: An Obfuscation-Based SoC Design Methodology for Hardware Protection," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2009, vol. 28, no. 10, pp. 1493-1502. Used under fair use, 2013

[32] S. Narasimhan, R. Chakraborty and S. Bhunia, "Hardware IP Protection During Evaluation Using Embedded Sequential Trojan," in *IEEE Design Test of Computers,* 2011, vol. 29, no. 3, pp.70-79.

[33] R. Chakraborty and S. Bhunia, "RTL Hardware IP Protection Using Key-Based Control and Data Flow Obfuscation," in *23rd International Conference on VLSI Design*, 2010, pp. 405410. Used under fair use, 2013

[34] R. S. Chakraborty and S. Bhunia, "Hardware protection and authentication through netlist level obfuscation," in *IEEE/ACM International Conference on Computer-Aided Design, ICCAD*, 2008, pp. 674-677.

[35] P. Yeolekar, R. A. Shafik, Jimson Mathew, Dhiraj Pradhan and S. P. Mohanty, "STEP: A Unified Design Methodology for Secure Test and IP Core Protection" in *21st ACM/IEEE Great Lakes Symposium on VLSI (GLSVLSI)*, 2012, pp. 333-338. Used under fair use, 2013

[36] N.-C. Lai and S.-J. Wang, "A reseeding technique for LFSR-based BIST applications," in *Proceedings of the 11th Asian Test Symposium, ATS'02*, 2012, pp. 200-205.

[37] M. Arai, H. Kurokawa, K. Ichino, S. Fukumoto and K. Iwasaki, "Seed selection procedure for LFSR-based BIST with multiple scan chains and phase shifters," in *13th Asian Test Symposium*, 2004, pp. 190-195.

[38] I. Zarei Moghadam, A. S. Rostami, M. R. Tanhatalab, "Designing a random number generator with novel parallel LFSR substructure for key stream ciphers," in *Computer Design and Applications (ICCDA)*, International Conference on , vol. 5, pp. 598-601.

[39] G. Palumbo and D. Pappalardo, "Charge Pump Circuits: An Overview on Design Strategies and Topologies," in *Circuits and Systems Magazine, IEEE*, 2010, vol. 10, pp. 31 -45.

[40] R. Pelliconi, D. Iezzi, A. Baroni, M. Pasotti and P.L. Rolandi, "Power efficient charge pump in deep submicron standard CMOS technology," in *IEEE Journal of Solid-State Circuits*, 2003, pp. 1068-1071.