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

Avinash R. Desai, Dinesh Ganta, Michael S. Hsiao, Leyla Nazhandali, Chao Wang, and Simin Hall
Department of Electrical and Computer Engineering, Virginia Tech,
Blacksburg VA 24061, USA
Email: {aviraj, diganta, mhsiao}@vt.edu

***Abstract*—Counterfeit Integrated Circuits (ICs) 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. In this paper, we propose two novel methods to validate the authenticity of ICs. First, a fuse with a charge pump is proposed to serve as a "seal" for the IC, in which any functional use will break the seal, and the broken seal is extremely hard to replace. Second, a novel time-stamp is proposed that can provide the date at which the IC was manufactured. The time-stamp circuitry is constructed using a Linear-Feedback Shift-Register (LFSR) such that any small change to the circuit would result in an entirely different date either in a distant past or future, beyond the lifetime of a typical IC. 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 with about 8.8% area overhead in AES implementation, the adversary requires more than 10^{118} different trials to successfully tamper time-stamp circuit. These techniques are easy to implement and embed into the circuit using todays technologies, while extremely difficult to modify or tamper with by the adversary. Finally, the method can be combined with additional hardware to detect malicious alteration made in the circuit.**

I. 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. However, in such a model, a chip that resembles another visually may be used by an adversary as a counterfeit device. 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 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 [2] 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 [3].

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 [3]. 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 [4] 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. 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 [3].

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: first, whether the IC is new or used; second, the batch information when the IC was manufactured, for example, the date of manufacturing.

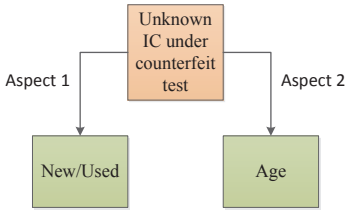

Fig. 1: Two Aspects for Counterfeit detection

While many existing techniques today, such as watermarking, metering and PUF (Physically Unclonable Function) based techniques, can be used as a mechanism to tackle against counterfeiting, they all require a substantial modification to either the original design or the registration process. In addition, the identification of authentic and counterfeit ICs would be a tedious process using such techniques. For example, one would need to query a database of PUF challenge-response pairs to determine if the the response matches the registered value. Likewise, revealing the watermark embedded in the design can be a costly process. An ideal counterfeit detection technique would be one in which there is an universal standard to prove the authenticity of IC. Furthermore, with this standard, one would prefer that the designer should not have to make large modification in the design to make the chips compliant.

Our strategic goal is to provide a universal standard to detect counterfeit ICs with the help of two novel methodologies

978-1-4799-1535-4/13/$31.00 ©2013 IEEE

which can offer protection against counterfeits and yet be easily implemented:

1) We introduce a fuse with a charge pump method to tackle the first problem to distinguish between new and used ICs. In this technique, a fuse is used as a seal which breaks only on functional use of the device.
2) Second, we present a LFSR-based time-stamp circuit which gives out the date the IC is manufactured with a query. The query is a sequence of input vectors to retrieve the manufactured date. The time-stamp circuit is embedded in the circuit such that it is extremely difficult to make it return a date that differs from the original time-stamp even just slightly. We also add an additional layer of tamper-resistance guard against more sophisticated attacks to the time-stamp circuit.

Both of the two techniques are easily implementable and consumes a small fraction of the chip real-estate. Our results indicate an average area overhead of 0.74% in implementing only the time-stamp circuit.

The rest of the paper is organized as follows. Section 2 presents the related work done in counterfeit detection and motivation for work. Section 3 describes fuse and basic time-stamp methodology. Section 4 explains an obfuscated hardware based approach for tamper resistant time-stamp. Section 5 describes additional implementation details. Section 6 presents the experimental results. Section 7 concludes the paper and discusses directions for future work.

II. Related Work

Traditional techniques 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. There have been much effort in the past to protect IP theft and cloning of ICs. Watermarking [5], [6], [7] 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 [9], [10], [11], [12] use 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. PUF-based implementation [13] and use of stressed ring oscillators [8] are also methods to estimate age of 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.

A. Reverse Engineering

Reverse engineering is the process through which adversary obtain details of the circuit for a given IC. Adversary takes advantages of tools and techniques specifically designed for the intended purpose. Techniques used in reverse engineering can be generally classified as Black Box Testing [15], White Box Testing [15] and Side Channel Analysis (SCA) [16]. Adversary can use these techniques to create gate level library and using it to figure out the complete circuit [15].

III. Methodology

A. Fuse Methodology

At low cost, seals provide a simple way to distinguish used ICs from new ones. Analogous to the seals used in medicine bottles to determine if the bottle has been opened and used, we propose an inexpensive technique to implement a seal via a standard fuse and a charge pump. Although fuses have been used for various applications on ICs, to the best of our knowledge, no work has been proposed in the context of anti-counterfeiting.

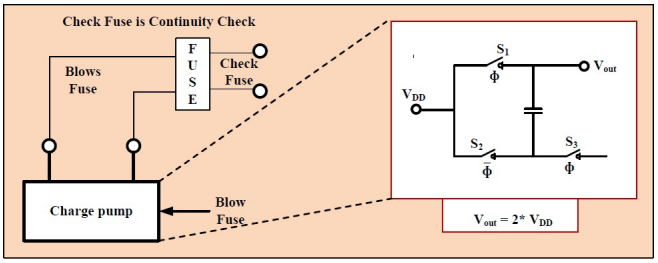

Fig. 2: Use of fuse

A fuse is a standard component provided by the foundry. The voltage that is required to blow the fuse can be easily produced using a simple charge pump circuitry. Figure 2 shows a fuse-based design with a basic charge pump. Although bridging a fuse might be possible with techniques like sputtering deposition using focused ion beams, the associated costs outweigh the benefits the counterfeiter gets from fake ICs.

When an IC is equipped with a fuse and a charge pump, the process of evaluating if an IC is used or new can be incorporated into the DFT mechanism of the chip or it can directly be accomplished through the pins of an IC. The basic actions that are essential are: (1) check the status of the fuse and to (2) blow the fuse.

Check Fuse This command is used to check if the fuse is intact or blown. Check fuse is as simple as performing a continuity check across fuse terminals. This is analogous to checking if the seal of the medicine bottle is intact.

Blow Fuse Once an IC equipped with a fuse becomes a functional part of a board, the fuse is automatically blown on its first use. A charge pump is used to blow the fuse. A variety of charge pumps are presented in the literature and Figure 2 shows a one stage charge pump circuit. The operation of the

charge pumps is presented in [22], and [23] shows a low cost implementation of a charge pump in a CMOS process.

One problem with "blow fuse" is that the fuse has to stay intact during the post manufacturing test of the IC and it should only be blown after it becomes part of a working board. To accomplish this, a circuit such as the pseudo-nmos circuit shown in Figure 3 can be leveraged. During the testing of the IC by the manufacturer and by the original equipment manufacturer (OEM), the blow fuse signal remains low as PMT (Post Manufacturing Test signal) is an active high signal and as PMT dominates the weak pull down resistor R1. When the IC becomes part of a board, PMT is a high impedance signal and R1 keeps the transistor N1 OFF. As a result, blow fuse is set high as C_L charges to VDD. The charge pump output voltage can be transferred to the fuse terminal through N2 pass gate as shown in Figure 3.

To ensure that fuse blows, PMT should not be accessible or controlled by the user/board that the IC becomes part of. So, instead of making PMT as an input pin, it can be set through the DFT elements of the IC which the user never can get access to. The control sequence to assert PMT is only provided to the related testing parties.

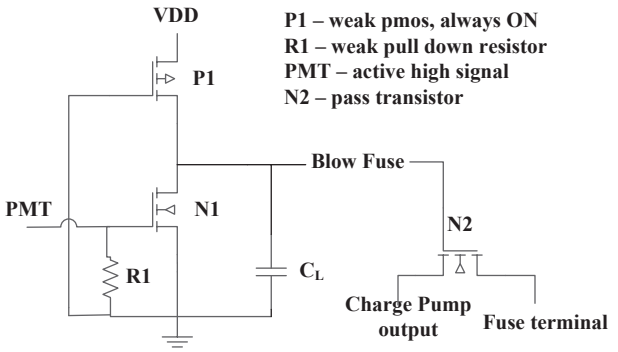

Fig. 3: Circuit enabling blow fuse

A blown fuse indicates that the device has been used, thus it is not fresh out of fab. The cost for an adversary to repair/replace the fuse should be much higher than the benefits they would get in return for selling counterfeits. In the proposed fuse, it is exactly that. The cost of repairing a blown fuse would outweigh the benefits of selling a counterfeit device.

In addition to the fuse, we would also like to know if the device was in fact manufactured in the latest batch and not from a pool of devices manufactured a number of years ago. In both fresh chip pools, the fuses would remain intact since neither has been used. To determine if a chip is newly manufactured or not, a time-stamp is proposed, which is discussed next.

B. Time-Stamp Circuit

In this section, 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. Our proposed method involves the use of an entry mode path traversals as in Anti-Tamper methods [14], [17], [18], [19] 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.

A 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. 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. They have been useful for Logic Built In Self Test (LBIST) [20], [21] and random number generation. LFSR structure is easy to design but it is extremely difficult to reverse engineer because a large combination of seeds and tap configuration can lead to a certain state.

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, $\{v_1, v_2, ..., v_k\}$. Let these values denote the year for simplicity of discussion. Then, one of the values v_i is 2013. 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 $2013-d$.

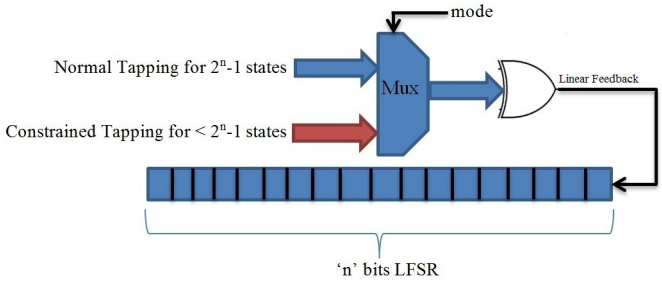

Fig. 4: Multi-mode Use of LFSR-based Time-stamp Circuit

Figure 4 shows a LFSR capable of both normal and constrained configurations. A mode signal to the multiplexer selects the configuration to be used. The normal configuration will cycle the LFSR in all $2^n - 1$ states and can be used for Logic Built-in-self-test purposes [20], [21]. 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.

A fixed path for the circuit to traverse can be assigned with the help of the invalid states generated by expanding the number of state elements as shown in Figure 5. 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 is switched to the time-stamp mode and constrained tap

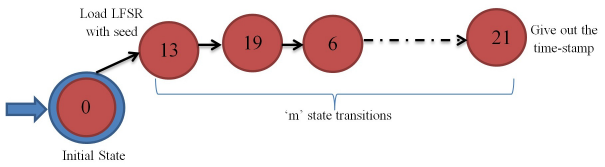

Fig. 5: Path traversal in entry mode for time-stamp

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. 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.

IV. 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).

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.

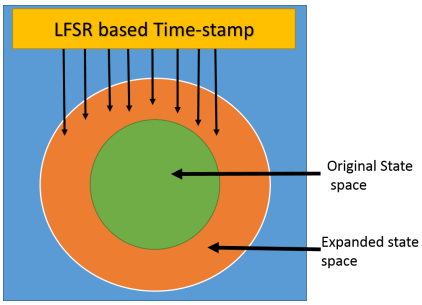

Fig. 6: Integration for tamper-resistant time-stamp

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.

The authors in [14] 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 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 6 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 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:

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

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.

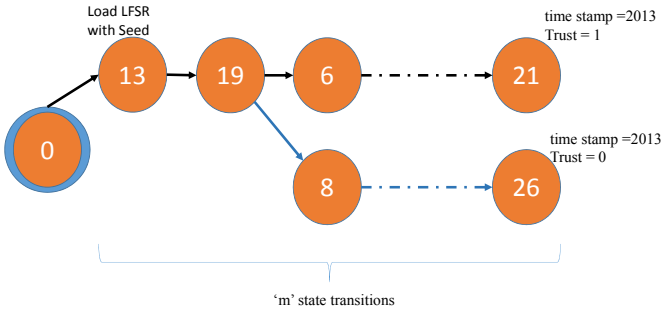

Fig. 7: Path traversal for tamper resistant time-stamp

Figure 7 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 7 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.

V. Additional Implementation Details

A. 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.

B. 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.

VI. Results

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.

TABLE I: 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 I shows the area overhead for the proposed time-stamp circuitry as a percentage of the overall circuit. From Table I 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

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

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

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

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

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

VII. Conclusions and Future Work

Two novel methods have been proposed for counterfeit detection to cover the two aspects in validating the authenticity of an IC. First, To determine if the IC is new or used, we have proposed a fuse with a charge pump methodology. This method works as a "seal" which is extremely hard to replace once broken. Using fuses provide a secure and inexpensive first line of defense to detect counterfeits.

Second, we propose an efficient time-stamp having an area overhead of only 0.74%. 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. The proposed techniques 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.

In the future, various directions can be pursued to decrease the area overhead for the tamper-resistant time-stamp circuit. 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. Computational parameters such as resource utilization can be used for further obfuscation.

References

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

[2] 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

[3] 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

[4] 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

[5] 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.

[6] 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.

[7] 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.

[8] 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*, vol. 43, pp. 874 –880, April 2008.

[9] 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.

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

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

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

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

[23] 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.