Digital PUF using Intentional Faults

Teng Xu and Miodrag Potkonjak
Computer Science Department
University of California, Los Angeles
{xuteng, miodrag}@cs.ucla.edu

Abstract

Digital systems have numerous advantages over analog systems including robustness, resiliency against operational variations. However, one of the most popular hardware security primitive, PUF, has been an analog component. In this paper, we propose the concept of digital PUF where the core idea is to intentionally use high-risk synthesis to induce defects in circuits. Due to the effect of process variation, each manufactured digital implementation is unique with high probability. Compared to the traditional delay based PUF, the induced defects in circuit are permanent defects that guarantee the fault-based digital PUF resilient against operational variations. Meanwhile, our proposed design takes advantage of the digital functionality of the circuits, thus, easy to be integrated with digital logic.

We experiment on the standard array multiplier module. Our standard security analysis indicates ideal security properties of the digital PUF.

Keywords

Physical Unclonable Function (PUF), Intentional Faults, Security, Testing

1. Introduction

There are two well-known wisdoms, testing and security that are widely and strongly established. The first is that integrated circuit (IC) defects and their functional faults are intrinsically bad phenomenon that should be detected, diagnosed and, if possible, eliminated. An exciting research and engineering field, testing, has been built with tremendous practical importance. In summary, faults are unwanted.

The second canon is related to emerging security and the exceptionally popular primitive, the physical unclonable function (PUF). A great variety of PUFs that employ different physical entities (e.g. delay and leakage energy), different architectures (e.g. ring oscillator, feed-forward, obfuscated parallel, differential, SRAM), and target different types of security protocols (e.g. secret key and public key) have been proposed and evaluated. Nevertheless the common denominator is that all proposed PUFs are analog systems. The common belief is that the digital PUF is unachievable because any digital system is easy to simulate, emulate, and fabricate.

Our objective is to simultaneously rebut these two well-established postulates. Specifically, we demonstrate how we can take advantage of process variation to intentionally induce faults in circuits and use the faulty circuit as a natural digital PUF. Three key observations are that (i) parts of large VLSI ICs with faults can produce highly un-predictable outputs; (ii) faults can be intentionally induced because of process variation, e.g., when wires in circuits are intentionally put close to each other, bridge faults between the wires can happen; (iii) it is difficult to form an IC that contains exactly a specified list of faults because of process variation. The first and the second observations are essential for creating digital PUFs. The last observation prevents a large family of security attacks and serves as a starting point to create and to use a unique piece of digital PUF.

Our basic idea is surprisingly simple. The starting point is to use process variation to intentionally inject faults in an IC. Process variation is defined as the deviation of integrated circuit (IC) parameters (e.g. threshold voltage, effective length) from the nominal specifications that manifest as a result of manufacturing processes. Then we intentionally induce faults in circuits, e.g., design wires to be close to each other. As a result, even with exactly the same design, different implementations have different faults because of process variation. We directly use the faulty circuit as a digital PUF. A PUF is a physical system with multiple inputs and at least one output, whose outputs are prohibitively difficult to predict for a given set of inputs. The digital PUF has numerous advantages over its traditional analog realization, including operational and environmental stability. While predicting the output of a circuit with one or a few faults may be easy, as the number of faults increases, the prediction becomes exceptionally hard. The essential step in exploiting faults is the creation of structures so that the faults in circuits can maximize output randomness. Using extensive simulations we analyze digital PUFs based on standard XOR network in terms of their security properties. Based on this, we demonstrate the fact that digital PUF shows even better security properties than the published analog PUF. In order to establish this claim, we use both standard PUF tests as well as looking into their resistance against different types of attacks.

Before we summarize our contributions, we claim that faulty IC is an ideal PUF. The first and most important support for this claim is related to their digital nature, the consequential benefits, and the unclonability because of process variation.

2. Related Work

2.1 PUF

Pappu et al. introduced the concept of the first PUF and demonstrated it using mesoscopic optical systems [1]. Devadas' research group at MIT developed the first family of silicon PUFs through the use of intrinsic process variation in deep submicron integrated circuits [2]. Guarardo and his coworkers at Philips Research in Eindhoven demonstrated how PUFs create unique startup values in SRAM cells [3]. Although a variety of PUF structures have been proposed,

978-1-4799-7581-5/15/$31.00 ©2015 IEEE

arbiter-based (APUF) [2], SRAM PUFs [3], and ring oscillator-based (RO-PUF) [4] are by far most popular. More recently, Xu et al. first proposed digital PUFs based on LUT networks on FPGA [5][6]. PUFs can be applied in sensor networks, lightweight protocols, and the security of Internet of things [7][8].

2.2 Testing PUFs

A number of approaches have been demonstrated for testing PUF security properties [9]. Many technologies are also proposed for the simulation of faulty circuits [10]. In addition to using them, our approach maps the testing of PUF security to standard randomness tests of random number generators. Therefore, if one succeeds in breaking a PUF that passes the outputs randomness test it is equivalent to break the widely used statistical test. Such event is unlikely but would be of major importance for many fields for science and engineering.

2.3 Fault Injection

Since at least 1997 fault injections have been recognized and demonstrated as a powerful security attack on cryptographic devices [11]. Numerous fault injection- based security attacks have been reported and have been surprisingly successful [12]. Models to evaluate the circuit sensitivities to random defects are proposed in [13]. The key difference between the surveyed research and our efforts is that for the first time we intentionally induce faults in circuits and advocate positive use of faults for security.

3. Preliminaries

Due to the effect of process variation, the wires that are close to each other have high potential to cause defects in the circuit. According to the position of where the faults occur, we simulate two types of faults in our digital PUF, stuck-at faults and bridge faults.

3.1 Stuck-At Faults

In this case, we assume that process variation effects the wire connection inside a gate that eventually causes the functionality of the gate to be changed. In our simulation, under such circumstance, we suppose that the output of the gate is stuck at logic high or logic low. Figure 1 shows an example of a stuck-at fault caused by the process variation inside a gate.

3.2 Bridge Faults

Another type of fault that locates between wires which connect gates are bridge faults. When we intentionally put wires in a layout close to each other, two or more normally distinct lines would have relatively high probability to be shorted together. In this paper, we use this wired-and fault model to simulate the bridge fault between gate wires as depicted in Figure 2. To be more specific, whenever a bridge fault occurs, if one of the wires is logic low, the other wire would be forced to logic low.

Finally, we consider the real layout of a full adder as shown in Figure 3. When we intentionally put wires close to each other in the process of manufacturing, as the example shown in Figure 3, both stuck-at faults and bridge faults can occur. However, the positions of the faults are completely uncontrollable because of process variation.

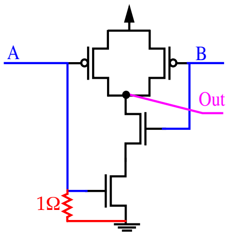

Figure 1: The schematic of a 2-input NAND gate. A, B are inputs and Out is the output. Suppose the red wire is the bridge caused by the process variation, as a result, this NAND gate is stuck at 0. The figure is cited from [14].

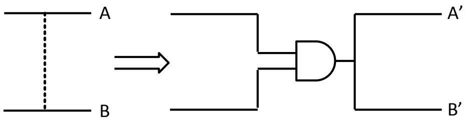

Figure 2: Wired-and mechanism for bridge fault simulation.

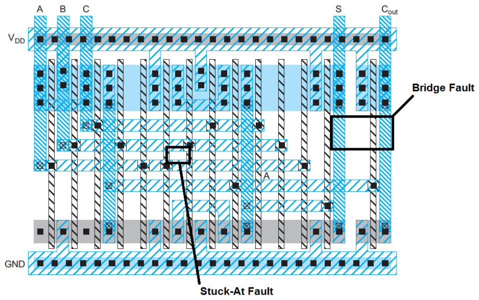

Figure 3: Full adder layout with stuck-at fault and bridge fault, A, B, C is inputs and S and C_{out} are outputs. The figure is cited from [15].

4. Concept

4.1 A Motivational Example

Figure 4 shows the gate-level full adder with 4 potential faults (G1 to G4). Among the faults, G1 and G2 are stuck at faults, G3 and G4 are bridge faults and we use wired- and for simulation. Every time we assume that only a single fault occurs in the circuit and the corresponding outputs are compared with the fault-free circuit outputs given the same inputs. The result in Table 1 indicates that even with a single fault in the circuit, the outputs change dramatically. This provides three observations: (i) Single faults can already alter the circuit outputs in such a way that is completely different from the fault-free outputs. (ii) Different faults have different impact on the circuit outputs. (iii) There are multiple faults on a big circuit.

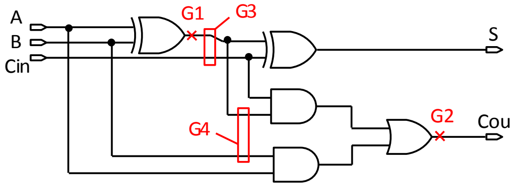

Figure 4: tuck-at and bridge faults in a full adder. G1, G2 are stuck-at faults and G3, G4 are bridge faults.

A / B / Cin	Cout / S						
	Fault-Free	$G1 \rightarrow 1$	$G1 \rightarrow 0$	$G2 \rightarrow 1$	$G2 \rightarrow 0$	$G3(bridge)$	$G4(bridge)$
0 0 0	0 0	0 1	0 0	1 0	0 0	0 0	0 0
0 0 1	0 1	1 0	0 1	1 1	0 1	0 0	0 1
0 1 0	0 1	0 1	0 0	1 1	0 1	0 0	0 1
0 1 1	1 0	1 0	0 1	1 0	0 0	1 0	1 0
1 0 0	0 1	0 1	0 0	1 1	0 1	0 0	0 1
1 0 1	1 0	1 0	0 1	1 0	0 0	1 0	0 0
1 1 0	1 0	1 1	1 0	1 0	0 0	1 0	0 0
1 1 1	1 1	1 0	1 1	1 1	0 1	1 0	0 1

Table 1: Single fault impacts on the outputs of a full adder. Values in red indicate the different bits in faulty outputs compared to fault-free outputs.

4.2 Digital PUFs

In order to create the digital PUF, we have two key operations. The first is to intentionally design the circuit in such a way that faults are easily induced by process variation, e.g., put wires close to each other to induce bridging between wires. Note that we do not manually inject faults to certain positions, but only to take advantage of the intentional design defects to induce faults. As a result, for different implementations, the position and type of the faults would be different due to process variation. The second is that since the faults are randomly created due to intrinsic process variation, it is only by gate level characterization that the position and the type of the faults can be measured and, thus, potentially enable an attacker to clone the de- vice. We eliminate this possibility by physically removing (e.g. burning) those pins on the circuit, which enable gate level characterization. Therefore, the physical unclonablity of the faulty circuit is guaranteed.

Now consider an attacker who attempts to clone our digital PUF. He is not able to execute a hardware level attack to look into the structure of digital PUF due to the burning of the pins. What he can do is to test all the possible input vectors on the faulty circuit to get the corresponding outputs and further create a mapping between the inputs and outputs. Due to the difficulty of reverse engineering, he cannot reverse engineer the corresponding hardware architecture just by acquiring this mapping.

Another core idea of the design is to run the faulty circuit for iterations. Essentially, the outputs of the faulty circuit can be iteratively utilized as the inputs in the next iteration after keep repeating. Therefore, after rounds of iterations, the influence of the circuit faults on the final outputs can be propagated and enlarged, thus making the outputs to be completely unpredictable.

5. Security Analysis

We analysis the security properties of faulty circuits based on the standard multiplier module. We use 16-bit inputs array-multiplier with 32 bits of outputs. We also assume that process variation will cause 1 percentage of the circuits have fault. We feed back the outputs as inputs iteratively for 10 iterations. For each of the following analysis, we simulate to test on 50 instances.

The security digital PUF comes from the unpredictability of its outputs. When given the inputs, without acquiring the structure of faulty circuits, attackers should not be able to deduce the corresponding outputs. According to the way of prediction. The attacks to PUF can be categorized in two types: prediction from fault-free circuit and prediction from statistical model.

5.1 Predication from Fault-free Circuit

In this type of attack, the attacker tries to predict the outputs of a digital PUF by using the fault-free circuit. As the circuit with exactly the same fault cannot be reproduced, the attacker replaces the faulty circuit in digital PUFs with fault-free circuit, and then tries to predict the faulty outputs according to the fault-free outputs. Two criterions can be analyzed to decide whether our digital PUF is resilient against this type of attack.

1) Hamming distance distribution: In this criterion, we compare the hamming distance between the faulty outputs and fault-free outputs. Ideally, the result should be in form of polynomial distribution with peak on the half of the output number.

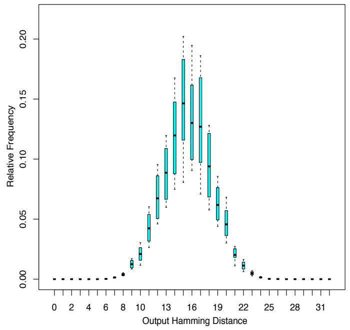

Figure 5: Prediction from fault-free circuit:(a) Hamming distance distribution (the error bar shows the distribution of max, 75%, mean, 25% and min)

2) Conditional probability: The other criterion is that the attacker tries to build a conditional probability model between every bit of fault-free outputs and faulty outputs. The goal of the attacker is to predict P(O1i = c1/O2j = c2),c1, c2=1 or 0 where O1 is the faulty output and O2 is the fault-free output. When the probability is always distributed around 0.5, it means the correlation between faulty outputs and fault-free outputs are weak, which indicates ideal security property.

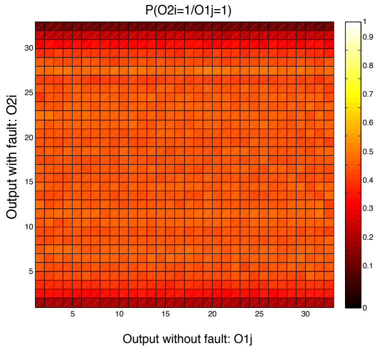

Figure 6: Prediction from fault-free circuit: Conditional probability between fault-free output O2 and faulty output O1.

5.2 Predication from Statistical Model

In this type of prediction, the attacker tries to predict the faulty outputs by building statistical model of the digital PUF to increase the correct rate of prediction. The statistical model can involve the following three aspects.

1) Frequency Prediction: In this attack, the attacker collects the data from previous outputs from digital PUFs and builds a probability distribution for each output being a particular value. The ideal situation would be that each bit of the output has equal probability to be 1 or 0 that provides no clue for the attacker to make the prediction.

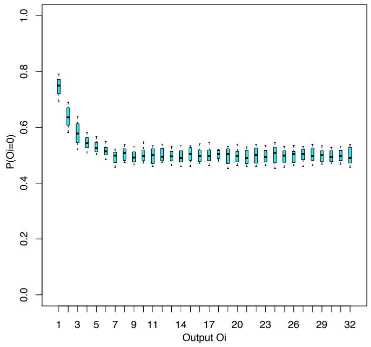

Figure 7: Prediction from statistical model: Frequency prediction: the probability that a bit in faulty output is equal to 1.

2) Avalanche Criterion: The notion of avalanche effect refers to that when the inputs of a digital PUF change slightly, the output changes significantly. If the avalanche effect is not exhibited to a significant degree, it indicates the digital PUF shows low randomness, thus easy to be predicted the other output given some similar output. We use the hamming distance between two similar outputs to indicate the avalanche effect. The ideal situation should be that the result is in the form of polynomial distribution.

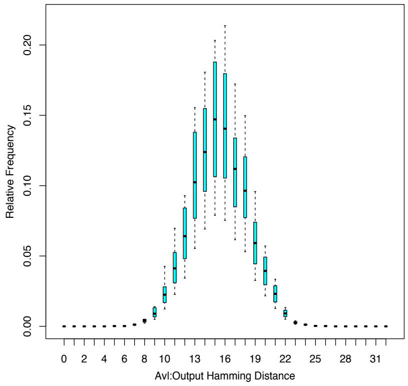

Figure 8: Prediction from statistical model: Hamming distance distribution of avalanche effect.

3) Conditional Correlation: Another type of attack is that the attacker tries to look into the correlation between an output bit Oi and an input bit Ij of a particular digital PUF. The goal of the attacker is to predict P (Oi = c1|Ij = c2), c1, c2=1 or 0. When the value is equal to 0.5, the correlation comes the lowest. Figure 9 shows the correlation results.

Reference

[1] R. Pappu, B. Recht, J. Taylor, and N. Gershenfeld, "Physical one-way functions," Science, vol. 297, no. 5589, pp. 2026–2030, 2002.

[2] B. Gassend et al., "Silicon physical random functions," in Computer and Communications Security, pp. 148–160, 2002.

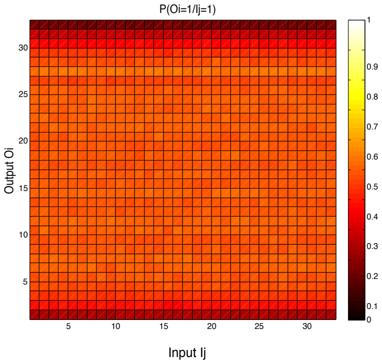

Figure 9: Prediction from statistical model: Conditional probabilities between Output bits Oi and input bits Ij.

[3] J. Guajardo, S. S. Kumar, G.-J. Schrijen, and P. Tuyls, "FPGA intrinsic PUFs and their use for IP protection," in CHES, pp. 63–80, 2007.

[4] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in DAC, pp. 9–14, 2007.

[5] T. Xu, J. B. Wendt, and M. Potkonjak, "Digital bimodal function: an ultra-low energy security primitive," in ISLPED, pp. 292–296, 2013.

[6] T. Xu, M. Potkonjak, “Robust and Flexible FPGA-based Digital PUF,” International Conference on Field Programmable Logic and Applications (FPL), 2014.

[7] T. Xu, J. B. Wendt and M. Potkonjak, "Secure Remote Sensing and Communication using Digital PUFs," ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS), pp. 173-184, 2014.

[8] T. Xu, J. B. Wendt, and M. Potkonjak, "Security of IoT Systems: Design Challenges and Opportunities," IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 417-423, 2014.

[9] Y. Hori, T. Yoshida, T. Katashita, and A. Satoh, “Quantitative and statistical performance evaluation of arbiter Physical Unclonable Functions on FPGAs,” Reconfigurable Computing and FPGAs - ReConFig, International Conference on, pp. 298-303, 2010.

[10] F. Hapke, et al. ”Defect-oriented cell-aware ATPG and fault simulation for industrial cell libraries and designs,” IEEE International Test Conference, 2009.

[11] D. Boneh, R.A. DeMillo, and R.J. Lipton, “On the importance of checking cryp- tographic protocols for faults,” Advances in Cryptology - EUROCRYPT, pp. 37-51, 1997.

[12] A. Barenghi, L. Breveglieri, I. Koren, and D. Naccache, “Fault injection attacks on cryptographic devices: theory, practice, and countermeasures,” Proceedings of the IEEE, vol. 100, no. 11, pp. 3056-3076, 2012.

[13] C.H. Stapper, “Modeling of integrated circuit defect sensitivities,” IBM Journal of Research and Development, vol. 27, no. 6, pp. 549-557, 1983.

[14] J. Plusquellic, CMPE 646: VLSI Design Verification and Test Course Notes, University of New Mexico, 2007, Lecture Notes.

[15] N. Weste, and D. Money, CMOS VLSI Design, Pearson/Addison Wesley, 2005.