# A Technique to Build a Secret Key in Integrated Circuits for Identification and Authentication Applications

Jae W. Lee, Daihyun Lim, Blaise Gassend, G. Edward Suh, Marten van Dijk, and Srini Devadas

MIT Computer Science and Artificial Intelligence Lab (CSAIL), Cambridge, MA 02139
{leejw, daihyun, gassend, suh, marten, devadas}@mit.edu

### Abstract

This paper describes a technique that exploits the statistical delay variations of wires and transistors across ICs to build a secret key unique to each IC. To explore its feasibility, we fabricated a candidate circuit to generate a response based on its delay characteristics. We show that there exists enough delay variation across ICs implementing the proposed circuit to identify individual ICs. Further, the circuit functions reliably over a practical range of environmental variation such as temperature and voltage.

### Introduction

Secret keys are essential to many emerging applications such as intellectual property protection and software licensing as well as conventional ID cards and smart cards. However, storing secret information in digital form is known to be vulnerable to invasive physical attacks as well as non-invasive attacks that discover internal connections or charges [1].

As an alternative we propose Physical Unclonable Functions (PUFs)[1], which extract a secret key from hidden timing or *delay information* rather than digital information. While ICs can be reliably mass-manufactured to have identical digital functionality, it is well known that each IC is unique in its delay characteristics due to inherent manufacturing variations. We exploit this statistical delay variation of transistors and wires across ICs to build secret information unique to each IC.

A simple application illustrates our technique. A particular circuit, that we term a PUF, is fabricated on multiple IC's. Given an IC with a PUF on it, we apply a set of secret inputs (challenges) to the PUF and observe the outputs (responses), which depend on the delays of excited paths in the PUF. The set of challenge-response pairs are known to the "lock" and the PUF itself is the "key." When a key is presented to a lock, the lock queries the key for the response to a particular challenge. If the key responds correctly, the lock opens, otherwise it does not. For security reasons, no challenge is repeated. We will argue in this paper that by using simple circuits fabricated using conventional manufacturing technology, we can create a key that cannot be cloned. (We note that the restriction on not repeating challenges can be removed using the notion of a controlled PUF; the interested reader is referred to [5].)

We will assume that the adversary cannot guess the challenges that correspond to the lock – these can be stored in a remote secure location, and only sent to the physical lock on demand. An adversary with physical access to a PUF key implemented on an IC can try to clone the PUF using a variety of different attacks. We describe these attacks and argue that they will not be successful in a later section.

Delay of a wire or a transistor is, of course, dependent on environmental fluctuations such as temperature variation, voltage variation and ambient noise. The reliability of a PUF is therefore a question mark. Rather than using absolute delays that can vary significantly with environmental changes, we use *relative delay comparisons* in our PUFs. As we will demonstrate, this dramatically improves to the reliability of PUFs, and they stay reliable even under significant temperature and voltage variation.

Previous researches have proposed the addition of specific circuits that produce unique responses due to manufacturing variation of MOSFET drain currents such that ICs can be identified [6]. However, these techniques focus simply on assigning one unique identifier to each chip without having security in mind. Therefore, they can be applied to *identify* an IC, but not to *authenticate* it. In our approach, a PUF has exponentially large number of unique challenge-response pairs, from which a lock can randomly choose a pair to authenticate the key.

To summarize, in this paper we explore the possibility of a PUF to be a viable alternative to a digital key stored in on-chip non-volatile memory. The paper is organized as follows. First, we present a candidate implementation of PUF, followed by the security issues including possible attacks to break its security. Then we provide experimental results with fabricated PUF test chips to quantitatively evaluate the security and reliability of the implementation. Finally, the last section summarizes our work.

### Circuit Implementation

Figure 1 depicts the block diagram of a candidiate PUF implementation, which is called an *arbiter-based* PUF[2]. In

[1]Alternatively, PUF is referred to as Physical Random Functions in some literatures [4, 5].

[2]Experiments on a PUF prototyped on a self-oscillating loop using FPGAs were described in [4].

this implementation, we excite two delay paths simultaneously and make the transitions race against each other. Then the arbiter block at the output determines which rising edge arrives first and sets its output to 0 or 1 depending on the winner. This circuit takes an $n$-bit challenge ($b_i$) as an input to configure the delay paths and generates a one-bit response as an output. In our test chip, $n$ is 64.

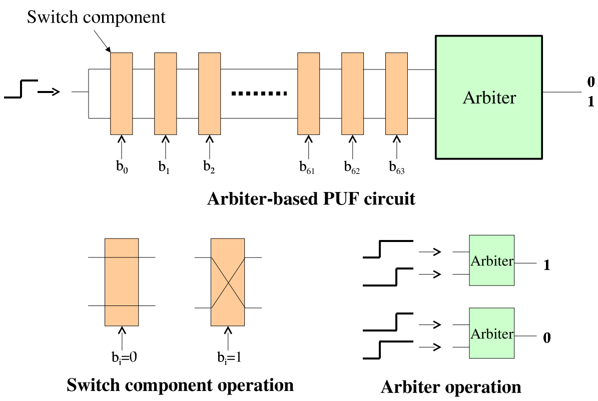


Fig. 1. Arbiter-based PUF circuit (basic arbiter scheme).

Figure 2 shows the details of the switch component. This block interconnects its two input ports ($i_0$ and $i_1$) to the output ports ($o_0$ and $o_1$) with different configurations depending on the control bit ($b_i$); for $b_i$=0 the paths go straight through, while for $b_i$=1 they are crossed. It is simply implemented with a pair of 2-to-1 muxes and buffers. For the arbiter, we used a simple transparent latch with an active-low enable input in our test chip. (We note that this arbiter favors the path to output 0 because it is preset to 0 and requires a setup time constraint to switch to 1. This skew was compensated for by fixing a small number of most significant challenge bits to effectively lengthen the path for 0.)

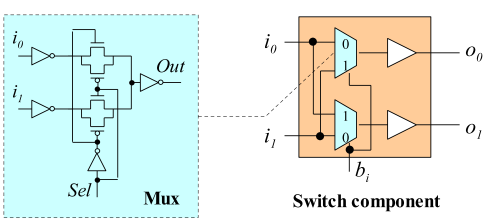


Fig. 2. Implementation of a switch component.

If the racing paths are symmetric in the layout regardless of the challenge bits and the arbiter is not biased to either path, a response is equally likely to be 0 or 1, and its value is determined only by the statistical delay variation in the fabricated ICs. Consequently, we wish to have paths to be as symmetric as possible to give a PUF enough randomness to function as a security primitive.

In our test chip, we carefully placed and routed the cells, for symmetric paths, and also so that the wires on the delay circuits effectively cover the entire chip. This layout technique makes it extremely hard for an adversary to probe the internal nodes to read out a logic value without breaking the PUF, i.e., without changing the delays of wires or transistors.

## Security of Arbiter-based PUF

There are at least two requirements for PUF to be a viable alternative to a digital key – *security* and *reliability*. For security, measurable delay variation must be large enough to distinguish chips fabricated with the same technology, masks, and even on the same wafer. If there is very little random variation of delay across ICs, an adversary can easily predict the behavior of a specific PUF either by fabricating another IC or by building a model for it, resulting in duplication of the key.

For reliability, the response of the circuit to a given challenge should be consistent for repeated measurements over a practical range of environmental variation such as temperature and power supply voltage. Otherwise, even with a valid key we would have difficulties in checking its authenticity.

The most plausible attacks are described below.

1. An adversary could attempt to clone a PUF by exhaustively enumerating all challenge-response pairs, but this requires applying an exponentially large number of challenges and observing their responses.

2. An adversary can attempt to fabricate a counterfeit PUF that produces exactly the same responses as the original PUF for all challenges. However, if there is enough process variation that cannot be easily controlled or eliminated by manufacturers, he will have to fabricate a huge number of ICs.

3. An adversary can open up the package of a PUF chip and attempt to measure device delays of the circuit by probing or monitoring internal devices. Then he may use these measured delays in a sophisticated timing model to predict the behavior of the circuit to a given random challenge. However, probing with sufficient precision is likely to be very difficult because interaction between the probe and the circuit will directly influence the behavior of the circuit. Besides, any damage on the delay paths affects capacitive coupling between wires to change delay characteristics, effectively destroying the PUF key.

4. Non-invasive model building attacks are also possible. First, an adversary can use a publicly available mask description of IC/PUF to build a timing-accurate model with a polynomial number of parameters. Then, he can apply many random or chosen challenges, and monitor the responses to calculate those parameters. If his model can predict the response of a real chip with a very high probability, it breaks the security of PUF by building a "virtual counterfeit." In case that the modeling attack

is a concern, we can make the modeling task much harder by adding more nonlinearity [2]. One of them is the feed-forward arbiter scheme depicted in Figure 3, where one or more challenge bits are determined by the racing result in an intermediate stage instead of being provided by a user.

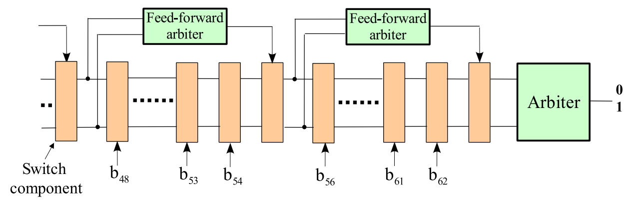


Fig. 3. Adding unknown challenge bits using feed-forward arbiters (feed-forward arbiter scheme).

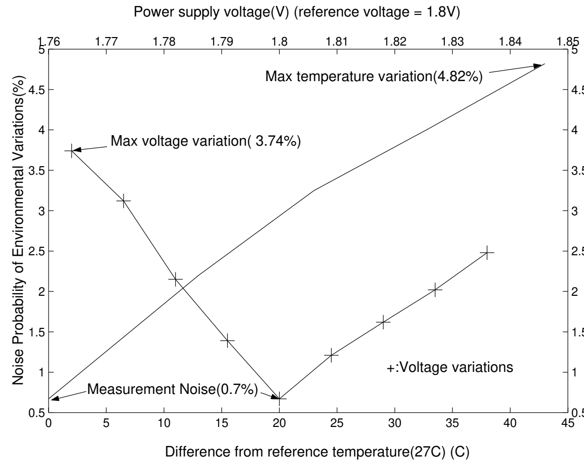


Fig. 4. The amount of noises from temperature and power supply voltage variations with the basic arbiter scheme.

## Experimental Results

This section demonstrates the security and the reliability of our PUF circuit with experimental results from fabricated ICs.

### *A. Interchip Variation*

To quantify the delay variation across ICs, we define the *interchip variation* ($\tau$) as the probability that the first measured response for a given challenge on a first chip is different from the first measured response for the same challenge on a second chip. We call these first measured responses *reference responses*.

To reduce the measurement noise, the PUF computes each response as the majority out of 11 repeated measurements. In this set up, we used 37 test chips to estimate $\tau \approx 23\%$ for the basic arbiter scheme. For the feed-forward arbiter scheme, $\tau$ increases to 40%.

### *B. Reliability*

To be useful, a PUF should reliably generate the same response for a given challenge. Unfortunately, environmental variations, instabilities in the circuit, and aging may cause a PUF to produce a response different from the reference response. To quantify the effect of these variations, we define the *noise* ($\mu$) as the probability that a newly measured response is different from the corresponding reference response.

First, temperature or supply voltage variations can significantly change the circuit delay, and lead to unexpected responses. In fact, the delay variations due to environmental variations can be orders of magnitude larger than the manufacturaing variations.

Fortunately, the arbiter-based PUF is very robust to environmental variations because the response relies on the difference of delays between two adjacent paths, instead of their absolute values. Consequently, the response remains consistent unless the faster path becomes slower.

Figure 4 shows the amount of noise introduced by the temperature ($\mu_t$) and voltage variations ($\mu_v$) for the basic arbiter scheme. The reference responses are recorded at 27 degrees Celsius and 1.8V power supply voltage. The noise is evaluated using 10000 challenges. Even if the temperature increases 40 degrees to 67 degrees Celsius, $\mu_t \approx 4.8\%$. Also, with $\pm$ 2% power supply voltage variation, $\mu_v \approx 3.7\%$. Both $\mu_t$ and $\mu_v$ are well below the interchip variation of 23%.

Second, a PUF response may change due to variations within the circuit even without environmental variation, which is called measurement noise ($\mu_m$). For example, junction temperatures or internal voltages may slightly fluctuate as the circuit operates, and change the delay characteristics. For some challenges, a setup time violation for the arbiter latch may lead to an unpredictable response. In the reference environment, we estimated $\mu_m \approx 0.7\%$ for the basic arbiter scheme and $\mu_m \approx 2.2\%$ for the feed-forward arbiter scheme.

Finally, aging may change the PUF characteristics. Electromigration and hot-carrier effects cause long-term degradation of the reliability of interconnects and transistors [3]. In most applications, the PUF circuits will age much slower than other processing elements because they only get activated to generate a secret. While we believe that the effect of aging is not a major problem compared with the environmental variations, we plan to run long-term aging experiments whose results are not available at this time.

### *C. Modeling Complexity*

Currently, for the basic arbiter scheme, our best non-invasive model building attack achieves about 97% accuracy in predicting the response of a given random challenge using a machine learning algorithm. Note that the error rate of 3% is still significantly higher than $\mu_m \approx 0.7\%$ but that it is less than $\mu_t$ or $\mu_v$. This means that the basic arbiter scheme can only be used for authentication purposes in the reference environment.

The feed-forward arbiter scheme appears to have significantly higher modeling complexity due to the nonlinearities introduced by the additional arbiters. So far, we have not been able to find a successful modeling attack that results in accurate predictions.

### D. Identification/Authentication Capability

Given the interchip variation $\tau$ and noise $\mu$, we can estimate the identification/authentication capability of the proposed PUF implementation. First, the probability that at least $2t+1$ out of $k$ reference responses differ between two chips is equal to

$$a = 1 - \sum_{i=0}^{2t} \binom{k}{i} \tau^i (1-\tau)^{k-i}. \quad (1)$$

For a single chip, the probability that at most $t$ out of $k$ responses differ from the corresponding reference responses is

$$b = 1 - \sum_{j=t+1}^{k} \binom{k}{j} \mu^j (1-\mu)^{k-j}. \quad (2)$$

From (1) and (2), the probability of being able to identify $N$ chips using a set of $k$ challenges is at least[3]

$$p = a^{\binom{N}{2}} b^N \approx (1 - \binom{N}{2}(1-a))(1 - N(1-b)).$$

For the basic arbiter scheme in a constant environment, $\tau \approx 0.2$ and $\mu \approx 0.007$, and we can distinguish $N = 10^9$ chips with probability at least $p \approx 1 - 5 \cdot 10^{-10}$ by trying $k = 1100$ challenges and allowing $t = 43$ incorrect responses. For the feed-forward arbiter scheme, where $\tau \approx 0.4$ and $\mu \approx 0.02$, we can distinguish the same number of chips with $k = 800$, $t = 64$, and $p \approx 1 - 7 \cdot 10^{-12}$.

### E. Performance

For a given 64-bit challenge, it takes an order of 50 ns for an input rising edge to transmit across the 64-stage parameterized delay circuit and evaluate an output at the arbiter[4]. Therefore, if we want to generate 1100 CRPs to distinguish 1 billion chips for the basic arbiter scheme, it takes about 55 $\mu s$. This is fast enough for most applications since a PUF is evaluated infrequently only to obtain a secret. We can also boost the performance by replicating multiple delay paths and arbiters and evaluate the responses in parallel.

## Conclusion

We proposed a candidate implementation of PUF, fabricated it, and investigated its security and reliability. The test chip was built in TSMC's 0.18 $\mu m$, single-poly, 6-level metal process with standard cells, whose die photo is shown in Figure 5. It contains 8 sets of the arbiter circuit generating an 8-bit response for a challenge and a JTAG-like serial interface, and measures a total area of 1212 $\mu m$ x 1212 $\mu m$. It operates at 100 MHz.

Our evaluation indicates that there exists significant delay variation of wires and transistors across ICs implementing this circuit, and that the idea of leveraging this variation to uniquely identify and authenticate an IC is promising. However, there are open issues that should be addressed for PUFs to be deployed in real applications. For example, it should be shown to be resistant against more elaborate modeling and other types of attacks, and additional reliability issues (e.g., the effects of aging) should be resolved.

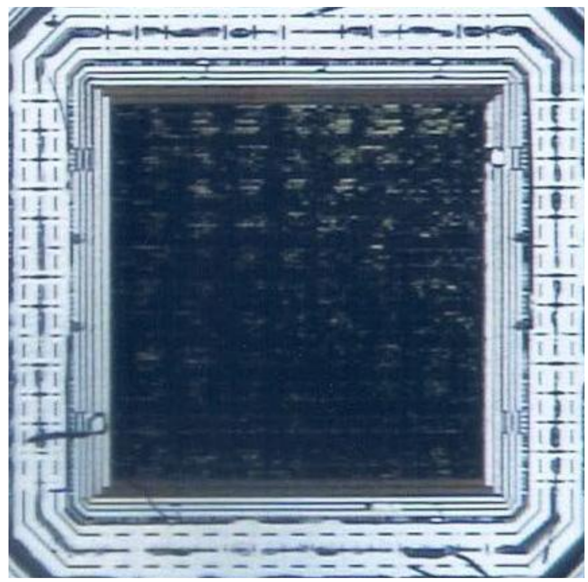

Fig. 5. A die photo of the fabricated chip

[3] There are $\binom{N}{2}$ possible chip pairs. The two chips in each pair generate at least $2t+1$ different reference responses with probability $a$. There are $N$ chips. Each chip measures at most $t$ responses corrupted by measurement noise with probability $b$. Hence, with probability $a^{N(N-1)/2} b^N$, any two chips do not agree on at least one (equals $(2t+1)$ minus two times $t$) measured response and can be identified from one another.

[4] This delay corresponds to 500-800 FO4 delay depending on the sources.

## References


[1] R. J. Anderson. *Security Engineering: A Guide to Building Dependable Distributed Systems*. John Wiley and Sons, 2001.

[2] B. Gassend, D. Lim, D. Clarke, M. van Dijk, and S. Devadas. Identification and Authentication of Integrated Circuits. *Concurrency and Computation: Practice and Experience*, 2003.

[3] A. Chandrakasan, W. Bowhill, and F. Fox. *Design of High Performance Microprocessor Circuits*. IEEE press, 2000.

[4] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas. Silicon Physical Random Functions . In *Proceedings of the Computer and Communication Security Conference*, May 2002.

[5] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas. Controlled physical random functions. In *Proceedings of* $18^{th}$ *Annual Computer Security Applications Conference*, December 2002.

[6] K. Lofstrom, W. R. Daasch, and D. Taylor. IC Identification Circuit Using Device Mismatch. In *Proceedings of ISSCC 2000*, pages 372–373, February 2000.