Literature Survey of Physical Unclonable Functions on FPGAs

Michael Patterson

Computer Engineering 583

Physical Unclonable Functions

A Physical Unclonable Function, or PUF as they are generally referred to, is a function with certain desirable properties. First of all, the function must be embodied in a physical structure. This may seem obvious, but it's important to clarify that PUF only refers to the function of a physical unit and not something like a software implementation. Secondly, a PUF must be easy to evaluate, but hard ("impossible") to predict. It is similar in this way to a mathematical one-way function. A given input should repeatedly lead to the same output, but it should be impossible to predict the output of a given input without previously observing it. Finally, a PUF should be easy to make, but "impossible" to duplicate. It's necessary that it be easy to make, or it would not be economically reasonable. It must be "impossible" to duplicate, otherwise it wouldn't really serve any purpose at all. An attacker could just manufacture or purpose a duplicate, provide the same input, and generate the same output. The response of such a PUF would basically prove nothing at all.

The terminology for discussing PUFs is pretty simple, but a few terms need to be defined before summarizing the existing relevant literature. The type of authentication that is being done here is known as "Challenge Response Authentication". To initiate this type of authentication procedure, a physical stimulus (input) is applied to the PUF. This is known as the "Challenge". The PUF then reacts in an unpredictable but consistent way and generates an output, known as the "Response". Together, these are often referred to as a Challenge Response Pair, or CRP.

Additionally, it should be mentioned that this literature survey is focused only on the types of PUFs that can be implemented on an FPGA. There are many other PUF designs that rely on physical units that do not lend themselves well to integration with an FPGA. While some of these designs are very interesting and likely useful, they are outside the scope of this paper and class.

With this basic understanding of functionality and terminology, it should be fairly easy to take a high-level look at the existing research in this area, and that's exactly what this paper attempts to do. There are basically two main strategies that have been proposed for designed a PUF to be implemented on an FPGA. The first relies on the timing characteristics, delays, and inconsistencies of all integrated circuits. It will be described in the first section of this paper, and several implementations based on these characteristics will be described. The second major strategy relies on certain circuits or elements of an FPGA that tend towards one of two stable states. This behavior is exploited in several PUF designs, and these will be examined in the second section of this paper.

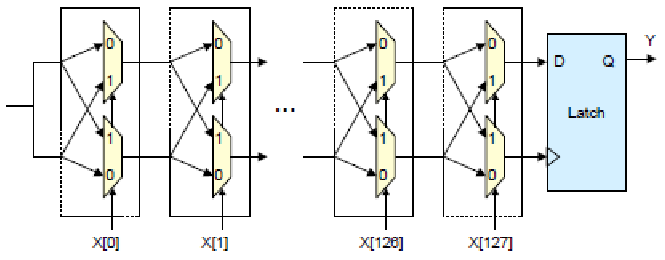
(I should also point out that often the name of a paper doesn't indicate the type of PUF design being discussed. I have structured this paper based on the name of the design, but the powerpoint tree should prove to match papers with designs.)

Timing Characteristics and Delays

It's a common fact that elements in integrated circuits have slightly varying characteristics, even when they are manufactured in the same exact manner. Even the slightest environmental variations during manufacturing lead to gates and wires with slightly different delays. These "random" characteristics can be used to generate inherently random output, and the following designs rely on this to serve as a PUF.

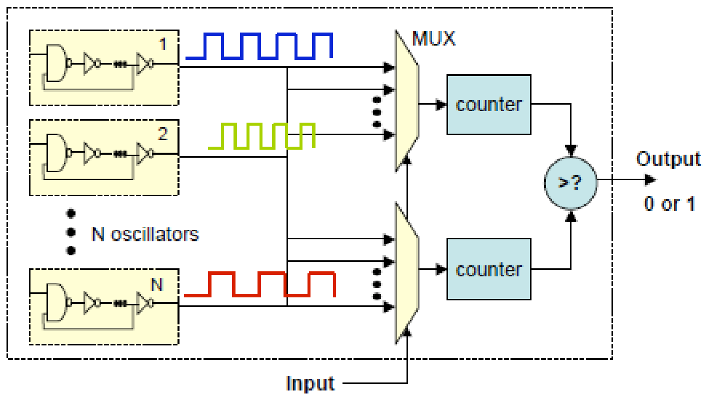
Delay PUF with MUXes and Arbiter

This type of PUF was one of the first proposed designs. It basically consists of two paths that are created by connecting a number of "switch delay elements" in series. Each element uses a two-to-one multiplexer to control which path is taken by each input. The challenge bits X are used as the select bits to the multiplexers, and the response of the PUF is based on which path leads to the faster signal propagation. The idea is that each wire and gate will have a slightly different delay, and so one of the two signals will reach the end of its path before the other one does. These signals are connected to an arbiter, and this unit determines which signal propagated more quickly along its path. The two paths are identical in theory and design, so the only factors that impact the speed of signal propagation are the inherently random delays present in the circuit elements. Since the input bits are used as the select signals to the multiplexers, each input leads to a different path which in turn leads to a different outcome. The delay characteristics are consistent for any one specific path, so a certain set of challenge bits should always result in a certain set of output bits. This PUF design can be seen in the following figure.



Ring Oscillator and Counters

Another PUF design that relies on the random delays inherent to circuit elements is the ring oscillator PUF. This design is composed of many delay loops that oscillate with a particular frequency. They are laid out identically, but the minor variations in manufacturing lead to loops with slightly different frequencies. These loops drive counters which are used to produce the response bits to a given challenge. One of the two counters will reach a given number more quickly than the other, even when the design and implementation of the circuits are the same. This PUF design is displayed in the following figure.



Reconfigurability of the FPGA

While the last two PUF designs were originally published several years ago, a newer design has recently been developed that also relies on the random delay characteristics of circuits. However, instead of implementing a specific circuit such as the previous two designs, this design actually takes advantage of the reprogrammability of an FPGA. The challenge in this case is an entire bitfile that is used to program the FPGA. Based on which cells are used as PUFS, the timing characteristics of the entire FPGA serve as the response. The design is presented in the following image showing multiple FPGA configurations.

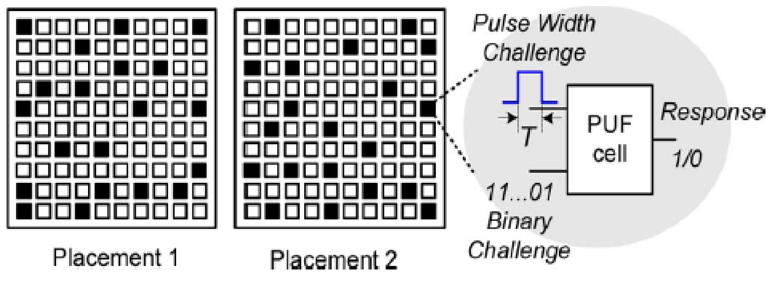


Fig. 6. Two random placement of PUF cells on FPGA.

This design is rather unique in its approach, but there are certain liabilities that may make it less than optimal. The FPGA must be carefully analyzed prior to it being used as a PUF. All possible input bitfiles must be programmed on the board, and the unique timing characteristics of the board must be stored in a database and matched with the appropriate bitfile. This is a limitation of all PUF designs, but the information needed for this design is much more substantial than just a set of input bits and the corresponding output. Also, there is the issue of the challenge being a reprogramming of the FPGA. While it is often possible to reprogram only certain portions of an FPGA, this approach is certainly more invasive of the normal functioning of the board than the other designs discussed. It's likely that this will prevent this design from being used in more commercial applications where the original functioning of the board cannot be affected.

Two Stable States

The second set of PUF designs that will be examined in this paper rely on interesting circuit elements that have only two stable states. Such elements tend to move towards one of the two states in a seemingly random manner when they are left in an uninitialized state. While it's "impossible" to predict which state will be taken by a given element, the actual functioning of each individual element is rather consistent. In this way, PUFs can be created based on the characteristics of which state a given implementation of a circuit assumes.

SRAM PUF

Unlike the previously discussed designs, this design does not depend on laying out a certain circuit and programming it onto an FPGA. Instead, it relies on the SRAM that is present on most modern FPGAs. Since an SRAM bit assumes a "random" value of 0 or 1 when it is originally given power, these bits can be used to provide a unique response. These bits also have the beneficial characteristic that they tend to repeatedly assume the same value. This is necessary for a successful PUF design, because the output must be repeatable to correctly identify a given device.

Butterfly PUF

Another design fairly similar to the SRAM PUF is the Butterfly PUF. It is based on unstable cross-coupled circuits, and largely serves the same purpose as the SRAM PUF. When forced into an unstable state, each circuit tends to either the '0' state or the '1' state. This tendency is inherently random due to minor variations in the manufacturing process, but it is also consistent across repeated iterations. This makes it a good circuit for use in a PUF. The design is displayed in the following figure.

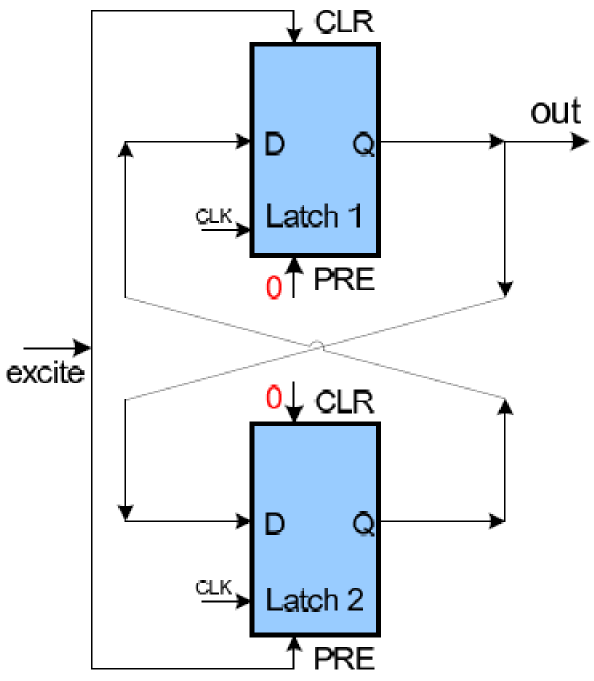


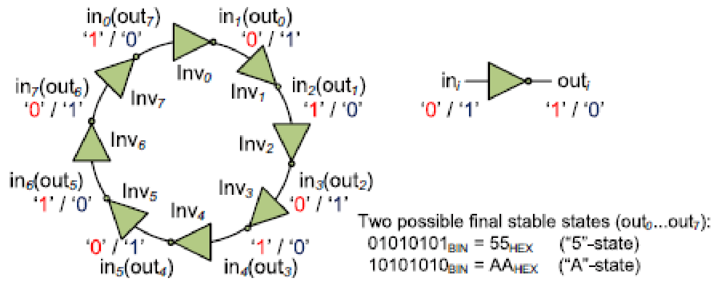
Fig. 3. Butterfly PUF: Cross-coupled Latches

So what purpose does this Butterfly PUF serve since it seems to function similarly to the SRAM PUF? The main benefit is that it is implemented as a circuit and not as a memory unit. This makes it possible to implement this PUF on any FPGA, not just those with SRAM units. It also means that no memory needs to be sacrificed in order to use this design. That doesn't mean it's better than the SRAM PUF, it just means that it is a better fit for certain situations.

Bi-stable Ring PUF

The final design examined in this paper is the Bi-stable Ring PUF, or BR-PUF. This design is based on the fact that an inverter ring made up of an even number of inverters only has two possible states. It tends towards one of these states when it is released from an unstable state. Again, the tendency of the BR-PUF is inherently random, but it is consistent across several identical challenges. This design results in a success PUF and can be seen in the following figure.

A. The Basic Idea: Bistable Rings



Testing and Evaluation

In addition to explanations of various PUF designs, many of these papers also included testing methodology and evaluation criteria for determining how successful a given design is. Common testing criteria involves testing a PUF under a variety of environmental conditions and measuring both the intra-PUF variation and also the inter-PUF variation. The intra-PUF variation is the number of bits in the response that change when a PUF is repeatedly given the same challenge. The inter-PUF variation is the number of bits in the response that vary between devices when given the same challenge. Ideally, a PUF would have an intra-PUF variation of zero and an inter-PUF variation of 50 percent. It is also desirable that the Hamming distance of the output is large when the input is changed.

Paper References

Blaise Gassend, Dwaine Clarke, Marten van Dijk, and Srinivas Devadas. 2002. Silicon physical random functions. In Proceedings of the 9th ACM conference on Computer and communications security (CCS '02), Vijay Atluri (Ed.). ACM, New York, NY, USA, 148-160. DOI=10.1145/586110.586132 http://doi.acm.org/10.1145/586110.586132

Guajardo, J.; Kumar, S.S.; Schrijen, G.-J.; Tuyls, P.; , "Physical Unclonable Functions and Public-Key Crypto for FPGA IP Protection," Field Programmable Logic and Applications, 2007. FPL 2007. International Conference on , vol., no., pp.189-195, 27-29 Aug. 2007

doi: 10.1109/FPL.2007.4380646

URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4380646&isnumber=4380602

Kumar, S.S.; Guajardo, J.; Maes, R.; Schrijen, G.-J.; Tuyls, P.; , "Extended abstract: The butterfly PUF protecting IP on every FPGA," Hardware-Oriented Security and Trust, 2008. HOST 2008. IEEE International Workshop on , vol., no., pp.67-70, 9-9 June 2008

doi: 10.1109/HST.2008.4559053

URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4559053&isnumber=4559030

Majzoobi, M.; Koushanfar, F.; , "Time-Bounded Authentication of FPGAs," Information Forensics and Security, IEEE Transactions on , vol.6, no.3, pp.1123-1135, Sept. 2011

doi: 10.1109/TIFS.2011.2131133

URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5737786&isnumber=5983314

Suh, G.E.; Devadas, S.; , "Physical Unclonable Functions for Device Authentication and Secret Key Generation," Design Automation Conference, 2007. DAC '07. 44th ACM/IEEE , vol., no., pp.9-14, 4-8 June 2007

URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4261134&isnumber=4261114

Qingqing Chen; Csaba, G.; Lugli, P.; Schlichtmann, U.; Ruhrmair, U.; , "The Bistable Ring PUF: A new architecture for strong Physical Unclonable Functions," Hardware-Oriented Security and Trust (HOST), 2011 IEEE International Symposium on , vol., no., pp.134-141, 5-6 June 2011

doi: 10.1109/HST.2011.5955011

URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5955011&isnumber=5954984