POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

TERO PUF Implementation on Xilinx Artix7 FPGA

Project Report for Embedded Systems Course

Aspesi Andrea, 962608
Putortì Alessandro, 953258

Professors:
Prof. Fornaciari William
Prof. Zoni Davide
Prof. Galimberti Andrea

Academic year:
2021-2022

Abstract: Physical Unclonable Functions (PUFs) are becoming important in recent years for various purposes such as Device Identification, True Random Number Generators, Cryptographic Key Generators, IP Protection. Different ways of realizing PUFs have been proposed in literature, exploiting process variations occurring inside chips. Among them, some can be efficiently implemented in FPGAs: in our project, after analyzing the state of the art, we chose to implement a Transient Effect Ring Oscillator (TERO) PUF. TERO PUF offers high resistance to surrounding logic influence and temperature changes, providing at the same time high uniqueness and reliability. As target, we chose Basys3 boards from Digilent, shipped with Xilinx Artix7 XC7-A35T.

1. Introduction

A wide variety of Physical Unclonable Functions have been proposed in literature, exploiting a very wide range of random and unpredictable properties of electronic chips to generate unique and reliable responses required by specific application like chip Identification, cryptographic key generation, logic obfuscation and IP protection. Among the ones which are suitable to be implemented on FPGA, Transient Effect Ring Oscillator (TERO) PUFs have been proved to have very good properties in terms of uniqueness, reliability and resistance to environmental changes.

In [2] by A.Wild et al., an in-depth analysis of RO, Loop and TERO PUFs has been performed comparing the responses coming from 100 Basys3 boards on which 1280 PUFs were implemented.
Our project consists in a further analysis of the data acquired by the authors of the paper, followed by a SystemVerilog implementation of the **TERO PUF for device identification purposes**, targeting both Basys3 and CmodA7-35t boards (both mounting XC7-A35T FPGA), tested at 100Mhz.

In order to design and test the PUF modules Xilinx Vivado 2020.2 was used. The analysis of the PUF responses and the generation of the IDs has been performed through some dedicated Python scripts and jupiter notebooks.

2. Design

Lorem ipsum **dolor** sit amet, consectetur adipiscing elit. Suspendisse facilisis ac erat vitae luctus. Aliquam in faucibus urna. Nulla ullamcorper tristique odio eu euismod. Nunc porta euismod orci vel ornare. Suspendisse suscipit non orci non blandit. In est massa, blandit sed metus non, commodo euismod leo. Nunc nec purus sed est imperdiet lacinia. Nullam convallis lectus neque, nec gravida **dolor** semper vel.
Donec quam odio, pulvinar sit amet ipsum et, volutpat ullamcorper ex. Quisque porta, justo nec tempor ornare, leo mauris iaculis ex, sit amet pellentesque quam mi eget neque. Phasellus maximus bibendum sem non pharetra. Morbi elementum tempor sem in scelerisque. Curabitur egestas pharetra libero eget bibendum.

Aenean sit amet molestie risus. Quisque ac sodales elit. Nulla tincidunt ac augue vitae malesuada. Proin eget sagittis arcu, non molestie nibh. Nullam in nulla sit amet lacus semper molestie eget eget turpis. Nunc vel urna sit amet nisl facilisis pulvinar in fermentum ligula. Morbi sollicitudin laoreet est eget pharetra. Quisque laoreet porta sem, quis fermentum leo mollis at.
In facilisis viverra felis, a dapibus mauris maximus sed. Curabitur ut eros a odio mollis fermentum vel vel eros. Donec id vestibulum erat, sit amet laoreet urna. Suspendisse aliquam, sapien quis suscipit dapibus, ligula nisl lacinia sem, in tempus neque quam et turpis. Nulla facilisi. Donec malesuada odio sit amet est egestas, in sollicitudin lorem vehicula. Quisque nec dictum quam, vitae tincidunt est. Curabitur id elit fringilla, tincidunt augue ac, pellentesque **dolor**. Vivamus leo sapien, eleifend eget diam a, porttitor ullamcorper arcu. Donec dignissim ipsum vel sapien porttitor, tincidunt placerat **dolor** condimentum. Aenean non elementum orci, id molestie nunc. Donec pharetra arcu sem, mollis maximus massa pretium ut. Duis sed congue libero, eu laoreet tortor. Phasellus sed consectetur nisl. Nulla interdum ligula eu placerat porttitor.

3. ID Generation

As stated in *Section 2*, the generation of the ID from the PUF responses can be made in different ways. This choice, naturally, influences the way the frequencies must be exposed in output (which responses to expose and in which order, given the input challenge). In order to compare different algorithms we decided to expose in output all the 1280 frequencies, regardless of the challenge. Before computing the actual IDs, we split the responses in 80 batches of 16 frequencies each. Each batch contains frequencies coming from PUFs of the same type. In the following some further details about the main algorithms:

3.1. 2Compare

2Compare algorithms takes two frequencies, f_i and f_j from each batch, and returns 1 if $f_i > f_j$, else 0. Thus, the resulting ID has 80 bits, one for each batch.

Below the pseudo-algorithm:

Algorithm 1 2Compare Algorithm

Require: $i, j \in [1, 16]$, $frequencies(batchIndex, freq)$
1: **for** $batchIndex = 1$ *to* 80 **do**
2: $\quad ID[batchIndex] = frequencies(batchIndex, i) > frequencies(batchIndex, j)$
3: **end for**
4: **return** ID

After our analysis we stuck to this algorithm since it's easy to implement and the resulting uniqueness and reliability are are quite good.

3.2. 2Comp Neighbor

2Compare neighbor is a variation of the previous one. Instead of taking two frequencies for each batch, it compares the "neighbor" frequencies of the same batch. We decided not to use this algorithm since, with 16 frequencies batches, the number of bits for each ID is 8, way too low to identify a sufficient number of devices. Increasing the number of frequencies per batch partially solves this problem, since (for the same number of TERO instances) the number of batches (and thus of challenges) is consequently reduced.

3.3. Lehmer-Gray

The response generation through Lehmer-Gray encoding is the best in terms of extracted entropy. allowing to extract 49 bits from a 16 bits batch. It was previously used in [1], basing on the method proposed by Yin and Qu in [3] to extract maximum entropy.
The algorithm consists in computing the Lehmer coefficients s_i for each frequency of a batch, with the following formula:

$$s_i = |\{j > i : freq(j) < freq(i)\}| \tag{1}$$

After that, the obtained coefficients are Gray encoded, and some of the bits of the encoded coefficient are taken as a response.
We did not choose this algorithm since the choice of the bits to take is not trivial, and depends on the particular implementation.

4. Results and Conclusion

After having realized the design, we transferred it to our two Basys3.
With the help of a utility script, we collected frequencies in various configurations. In particular, we wanted to verify the best number of repetitions and evaluation time, in order to have a high reliability, but waiting a reasonable amount of time for the PUF response.

- TERO loop are expected to oscillate for a short period of time, then settle in a stable state. Using a too short evaluation time truncates the oscillation measurement: although the resulting read frequency values display a unique pattern, we can better capture the uniqueness if we wait for most of the loops to stop oscillating. At the same time, a high evaluation time can significantly slow down the process, and make the PUF unusable.
- Repeating the measurement several times (and averaging the values) is necessary as the oscillation frequency of a TERO loop depends, apart from the instance-dependent process variations, on environmental noise (e.g., due to temperature or supply voltage differences) and random temporal noise (e.g., temporal fluctuations in the power supply). The latter type is not constant and can be averaged out. This approach has been used several times, as it has been proved to increase the reliability (as we verified via experiments).

4.1. Evaluation time and Repetitions

Using the data we collected, we got to the same conclusions of the paper: using more than 2^5 clock cycles for evaluation does not produce a distinguishable improvement in reliability nor uniqueness (as this latter is little influenced by evaluation time). Instead, increasing the number of repetition from 1 to 2^{12} adds an extra 5% to the reliability indicator (but slows the response time of the PUF).
Using an evaluation time of 2^5 clock cycles and 2^{12} repetitions is the best compromise between reliability and response time of the PUF.

4.2. Reliability

To calculate the reliability of our implementation, we used 100 measurements each at a different configuration, comparing the same challenge number response:

$$Reliability_i = 100\% * (1 - \frac{1}{s}\sum_{t=1}^{s}\frac{HD(Xr_i, X_{i,t})}{n}) \tag{2}$$

were i is the i-th challenge (among the 120 available), HD is the Hamming Distance, Xr_i is the first response measured for that challenge, $X_{i,t}$ represents each of the 100 responses generated for the i-th challenge, $s = 100$ are the measurements taken and $n = 80$ is the number of bits of each response.

Using the chosen configuration (evaluation=2^5, repetitions=2^{12}), we got a mean reliability of 97.4% for the first PUF ($TERO\_AA$), 96.9% for the second PUF ($TERO\_AP$).
The ideal value for the reliability is 100% (otherwise stated: for each evaluation of the same PUF, fixed the challenge, we would like the same exact response bitwise)

4.3. Uniqueness

With the same 100 measurements for the two PUFs of before, we tried to estimate the uniqueness of the response for each given challenge. Of course to get to a more accurate result, lots of PUFs are required.
The formula used for the uniqueness is the following:

$$Uniqueness_i = 100\% * \frac{k}{k(k-1)}\sum_{i=1}^{k-1}\sum_{j=i+1}^{k}\frac{HD(X_i, X_j)}{n} \tag{3}$$

were i is the i-th challenge (among the 120 available), $k = 2$ is the number of PUFs used, HD is the Hamming Distance, X_i is the response of the challenge for the i-th PUF, X_j is the response of the challenge for the j-th PUF and $n = 80$ is the number of bit of each response.

Using the chosen configuration (evaluation=2^5, repetitions=2^{12}), we got a mean uniqueness of 50.25%. The ideal value is 50% (otherwise stated: the probability of a bit flip between two responses of different PUFs, fixed the challenge, is 50% - the toss of a coin).

4.4. Conclusion

Seen such results, we consider the implementation of the TERO PUF successful, and in line with the results of the paper.
Further work can be done to use the challenge directly in the FPGA to send only the needed frequency measurements, and in this way avoid exposing the whole CRP (i.e. Challenge Response) space. As anticipated in the design, little effort is required to achieve this with our implementation.

5. Introduction

This document is intended to be both an example of the Polimi LaTeX template for Master Theses in article format, as well as a short introduction to its use. It is not intended to be a general introduction to LaTeX itself, and the reader is assumed to be familiar with the basics of creating and compiling LaTeX documents (see [? ?]).
The cover page of the thesis in article format must contain all the relevant information: title of the thesis, name of the Study Programme, name(s) of the author(s), student ID number, name of the supervisor, name(s) of the co-supervisor(s) (if any), academic year.
Be sure to select a title that is meaningful. It should contain important keywords to be identified by indexer. Keep the title as concise as possible and comprehensible even to people who are not experts in your field. The title has to be chosen at the end of your work so that it accurately captures the main subject of the manuscript.
It is convenient to break the article format of your thesis (in article format) into sections and subsections. If necessary, subsubsections, paragraphs and subparagraphs can be used. A new section is created by the command

```
\section{Title of the section}
```

The numbering can be turned off by using `\section*{}`. A new subsection is created by the command

```
\subsection{Title of the subsection}
```

and, similarly, the numbering can be turned off by adding an asterisk as follows

```
\subsection*{}
```

It is recommended to give a label to each section by using the command

```
\label{sec:section_name}%
```

where the argument is just a text string that you'll use to reference that part as follows: *Section 1 contains* INTRODUCTION ....

6. Equations

This section gives some examples of writing mathematical equations in your thesis.
Maxwell's equations read:

$$\begin{cases} \nabla \cdot \boldsymbol{D} = \rho, & \text{(4a)} \\ \nabla \times \boldsymbol{E} + \dfrac{\partial \boldsymbol{B}}{\partial t} = \boldsymbol{0}, & \text{(4b)} \\ \nabla \cdot \boldsymbol{B} = 0, & \text{(4c)} \\ \nabla \times \boldsymbol{H} - \dfrac{\partial \boldsymbol{D}}{\partial t} = \boldsymbol{J}. & \text{(4d)} \end{cases}$$

Equation (4) is automatically labeled by `cleveref`, as well as Equation (4a) and Equation (4c). Thanks to the `cleveref` package, there is no need to use `\eqref`. Equations have to be numbered only if they are referenced in the text.
Equations (5), (6), (7), and (8) show again Maxwell's equations without brace:

$$\nabla \cdot \boldsymbol{D} = \rho, \tag{5}$$
$$\nabla \times \boldsymbol{E} + \frac{\partial \boldsymbol{B}}{\partial t} = \boldsymbol{0}, \tag{6}$$
$$\nabla \cdot \boldsymbol{B} = 0, \tag{7}$$
$$\nabla \times \boldsymbol{H} - \frac{\partial \boldsymbol{D}}{\partial t} = \boldsymbol{J}. \tag{8}$$

Equation (9) is the same as before, but with just one label:

$$\begin{cases} \nabla \cdot \boldsymbol{D} = \rho, \\ \nabla \times \boldsymbol{E} + \dfrac{\partial \boldsymbol{B}}{\partial t} = \boldsymbol{0}, \\ \nabla \cdot \boldsymbol{B} = 0, \\ \nabla \times \boldsymbol{H} - \dfrac{\partial \boldsymbol{D}}{\partial t} = \boldsymbol{J}. \end{cases} \tag{9}$$

7. Figures, Tables and Algorithms

Figures, Tables and Algorithms have to contain a Caption that describes their content, and have to be properly referred in the text.

7.1. Figures

For including pictures in your text you can use `TikZ` for high-quality hand-made figures [**?**], or just include them with the command

`\includegraphics[options]{filename.xxx}`

Here xxx is the correct format, e.g. `.png`, `.jpg`, `.eps`, ...

Figure 1: Caption of the Figure.

Thanks to the `\subfloat` command, a single figure, such as Figure 1, can contain multiple sub-figures with their own caption and label, e.g. Figure 2a and Figure 2b.

(a) One PoliMi logo.

(b) Another one PoliMi logo.

Figure 2: Caption of the Figure.

7.2. Tables

Within the environments `table` and `tabular` you can create very fancy tables as the one shown in Table 1.

Example of Table (optional)

	column1	**column2**	**column3**
row1	1	2	3
row2	α	β	γ
row3	alpha	beta	gamma

Table 1: Caption of the Table.

You can also consider to highlight selected columns or rows in order to make tables more readable. Moreover, with the use of `table*` and the option `bp` it is possible to align them at the bottom of the page. One example is presented in Table 2.

7.3. Algorithms

Pseudo-algorithms can be written in LaTeX with the `algorithm` and `algorithmic` packages. An example is shown in Algorithm 2.

Algorithm 2 Name of the Algorithm

```
Initial instructions
for for − condition do
    Some instructions
    if if − condition then
        Some other instructions
    end if
end for
while while − condition do
    Some further instructions
end while
Final instructions
```

8. Some further useful suggestions

Theorems have to be formatted as follows:

Theorem 8.1. *Write here your theorem.*

Proof. If useful you can report here the proof.

Propositions have to be formatted as follows:

Proposition 8.1. *Write here your proposition.*

How to insert itemized lists:

- first item;
- second item.

How to write numbered lists:

1. first item;
2. second item.

9. Use of copyrighted material

Each student is responsible for obtaining copyright permissions, if necessary, to include published material in the thesis. This applies typically to third-party material published by someone else.

	column1	**column2**	**column3**	**column4**	**column5**	**column6**
row1	1	2	3	4	5	6
row2	a	b	c	d	e	f
row3	α	β	γ	δ	ϕ	ω
row4	alpha	beta	gamma	delta	phi	omega

Table 2: Highlighting the columns

10. Plagiarism

You have to be sure to respect the rules on Copyright and avoid an involuntary plagiarism. It is allowed to take other persons' ideas only if the author and his original work are clearly mentioned. As stated in the Code of Ethics and Conduct, Politecnico di Milano *promotes the integrity of research, condemns manipulation and the infringement of intellectual property*, and gives opportunity to all those who carry out research activities to have an adequate training on ethical conduct and integrity while doing research. To be sure to respect the copyright rules, read the guides on Copyright legislation and citation styles available at:

`https://www.biblio.polimi.it/en/tools/courses-and-tutorials`

You can also attend the courses which are periodically organized on "Bibliographic citations and bibliography management".

11. Conclusions

A final section containing the main conclusions of your research/study and possible future developments of your work have to be inserted in the section "Conclusions".

12. Bibliography and citations

Your thesis must contain a suitable Bibliography which lists all the sources consulted on developing the work. The list of references is placed at the end of the manuscript after the chapter containing the conclusions. It is suggested to use the BibTeX package and save the bibliographic references in the file `bibliography.bib`. This is indeed a database containing all the information about the references. To cite in your manuscript, use the `\cite{}` command as follows:
Here is how you cite bibliography entries: [?], or multiple ones at once: [? ?].
The bibliography and list of references are generated automatically by running BibTeX [?].

References

[1] Roel Maes, Anthony Van Herrewege, and Ingrid Verbauwhede. Pufky: A fully functional puf-based cryptographic key generator. In Emmanuel Prouff and Patrick Schaumont, editors, *Cryptographic Hardware and Embedded Systems – CHES 2012*, pages 302–319, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[2] Alexander Wild, Georg T. Becker, and Tim Güneysu. A fair and comprehensive large-scale analysis of oscillation-based pufs for fpgas. In *2017 27th International Conference on Field Programmable Logic and Applications (FPL)*, pages 1–7, 2017.

[3] Chi-En Daniel Yin and Gang Qu. Lisa: Maximizing ro puf's secret extraction. In *2010 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 100–105, 2010.

A. Appendix A

If you need to include an appendix to support the research in your thesis, you can place it at the end of the manuscript. An appendix contains supplementary material (figures, tables, data, codes, mathematical proofs, surveys, ...) which supplement the main results contained in the previous sections.

B. Appendix B

It may be necessary to include another appendix to better organize the presentation of supplementary material.

Abstract in lingua italiana

Qui va l'Abstract in lingua italiana della tesi seguito dalla lista di parole chiave.

Parole chiave: qui, le parole chiave, della tesi, in italiano

Acknowledgements

Here you might want to acknowledge someone.