INDIAN INSTITUTE OF TECHNOLOGY, KANPUR

SURGE 2023

Design and Analysis of Physically Unclonable Functions on Artix-7 FPGA.

Prepared By

SUVRAT PAL
Dept. of Material Science and Engineering

Mentored by

Dr. Urbi Chatterjee
Dept. of Computer Science and Engineering

12 JULY 2023

Contents

Indian Institute of Technology, Kanpur

Certificate

This is to certify that **SUVRAT PAL**, a 2^{nd} year BTech student of Material Science and Engineering, at IIT Kanpur has successfully completed his Surge 2023 Intern on **Design and Analysis of Physically Unclonable Functions on Artix-7** under the guidance of **Dr. Urbi Chatterjee** Dept. of CSE.

Dr. Urbi Chatterjee
(Project Guide)

Date: 12 July 2023

Acknowledgement

I would like to express my heartfelt gratitude and appreciation to all those who have contributed to the successful completion of our project on the design and analysis of FPGA on Artix-7. I extend my sincere acknowledgments to Dr. Urbi Chatterjee, my esteemed guide from the Department of Computer Science and Engineering. Her guidance, expertise, and unwavering support have been instrumental in shaping this project. I would also like to extend my thanks to the Ph.D. scholars, Neelofar Hassan , Vishesh MIshra, and Suraj Mandal, for their valuable assistance throughout the project. Their expertise, constructive feedback, and dedication have greatly contributed to the overall quality of my work. I am grateful to the institution for providing me with the necessary resources and infrastructure to conduct my research effectively. Access to the required tools and equipment has been crucial in the implementation and analysis of the FPGA design. I am deeply indebted to all the individuals mentioned above, as well as anyone else who has contributed to this project in any way. Your support and guidance have been pivotal in the successful completion of this endeavor.

Date:
12/07/2023

SUVRAT PAL
Dept. of Material Science and Engineering

1 Abstract

Physically Unclonable Functions (PUFs) are emerging as promising security primitives that harness the inherent variations in the physical properties of electronic devices to generate unique and unpredictable identifiers. PUFs exploit the microscopic differences in hardware components, such as transistors, resistors, or capacitors, to create device-specific responses to challenges or stimuli. The main idea behind PUFs is that these physical variations are difficult to reproduce or clone accurately, making them suitable for secure hardware implementations. PUFs demonstrate uniformity by consistently generating all possible responses in equal proportion when subjected to multiple challenges, reliability by producing the same responses over time and under diverse operating conditions, and uniqueness by signifying that each device generates a distinct and distinguishable response. All the PUFs are designed keeping these three properties in mind and inherently claim the validity of these properties. However, the claims made by proposed PUFs may not be feasible at the hardware level. This work aims to assess the feasibility of the claimed properties of PUFs, namely uniformity, reliability, and uniqueness, at the hardware level. To accomplish this, the study focuses on analyzing a machine learning attack resilient delay-based PUF **Arbiter PUF**. The implementation of these PUFs is carried out using Verilog HDL, and the designs are synthesized and validated on the Basys3 board using Xilinx Vivado Software. This setup is utilized to generate responses for a large number of random challenges, thus constituting the challenge response pair (CRP) database. Thereafter, the CRP database is used to evaluate the PUFs using C running Ryzen5 running on 16GB of RAM.

2 Introduction

In an era where securing sensitive data and protecting hardware systems against unauthorized access is a critical concern, Physical Unclonable Functions (PUFs) have emerged as a promising solution. PUFs leverage inherent physical variations within hardware components to generate unique and unpredictable identifiers, making them a powerful tool for device authentication, secure key generation, and data protection.

This project presents the implementation of a strong PUF on a Field-Programmable Gate Array (FPGA) using the Basys3 development board equipped with a MicroBlaze processor. The Basys3 board, featuring Xilinx Artix-7 FPGA, offers a versatile and cost-effective platform for designing and evaluating PUFs in real-world applications.

The primary objective of this project is to design, implement, and evaluate a robust PUF architecture that can withstand various attacks, ensuring the integrity and security of hardware systems. Leveraging the physical variations present within FPGA resources, such as gate delays or power consumption discrepancies, the PUF will generate a unique response for each challenge input, making it highly resistant to cloning and tampering attempts.

The PUF architecture will be described using a hardware description language (HDL) **Verilog**, utilizing the Xilinx Vivado toolchain for synthesis, place, and route. Additionally, a MicroBlaze soft processor will be integrated into the FPGA design to facilitate communication with the PUF and provide a platform for secure key management and authentication protocols.

3 PUF

PUF stands for Physical Unclonable Function. It is a concept in computer science and cryptography that refers to a physical device or component with unique properties that cannot be easily duplicated or cloned. PUFs exploit the inherent variations found in physical systems to generate unique and unpredictable responses to input stimuli.

A PUF operates by exploiting the manufacturing variations that naturally occur during the fabrication process of electronic devices. These variations can be caused by factors such as temperature, voltage, material properties, and process imperfections. By measuring and characterizing these variations, a PUF can generate a unique and unpredictable response, often referred to as a "fingerprint" or "silicon DNA," for each instance of the device.

The output of a PUF is typically used as a cryptographic key or a means of device authentication. Since the response of a PUF is specific to the individual device and cannot be easily replicated, it can provide a high level of security against various attacks, such as cloning, counterfeiting, and tampering.

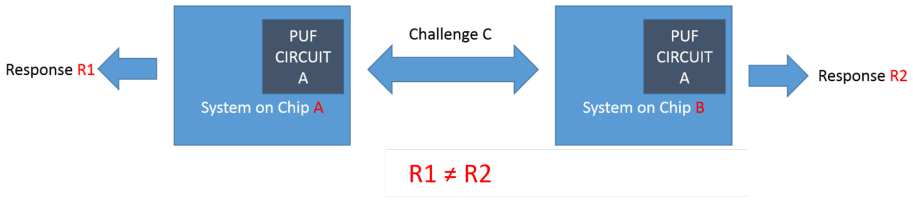

Figure 1: Illustration of a Physically Unclonable Function

PUFs have applications in various areas, including secure key generation, hardware-based authentication, anti-counterfeiting measures, and secure device identification. They are particularly useful in scenarios where the security of a cryptographic system relies on the uniqueness and unpredictability of the key or authentication mechanism.

There are different types of PUFs based on the physical properties they exploit. Some common types include SRAM PUF, Butterfly PUF, Ring Oscillator PUF, Arbiter PUF, and Magnetic PUF. Each type utilizes different physical characteristics and measurement techniques to generate unique responses.

4 Arbiter PUF

An Arbiter PUF (Physical Unclonable Function) is a type of PUF that is based on the delay differences in an arbiter circuit. It exploits the manufacturing variations in the delay paths of the circuit to generate a unique response for each instance of the PUF.

1. **Circuit Configuration:** An Arbiter PUF consists of a symmetric circuit, usually composed of a series of interconnected multiplexers. The circuit takes in a challenge input, which is a pair of signals or bits, and produces a response output based on the relative delays of the signals in the circuit.

2. **Delay Variations:** Due to manufacturing variations, each path in the circuit can have slightly different delays. These variations arise from differences in the fabrication process, material properties, and other factors. As a result, the signals arriving at the multiplexers experience different delays, leading to different response patterns.

3. **Challenge-Response Generation:** To generate a response, the Arbiter PUF compares the arrival times of the two input signals at each multiplexer. The relative delay determines which input is selected as the output. The response is then derived from the sequence of selected inputs across the entire circuit.

4. **Uniqueness and Stability:** The response pattern of an Arbiter PUF is unique to each instance due to the manufacturing variations. The delay differences introduce a form of randomness that is difficult to reproduce or predict. However, it's worth noting that the stability of Arbiter PUFs can be affected by environmental factors and aging effects, which may cause the delay paths to change over time.

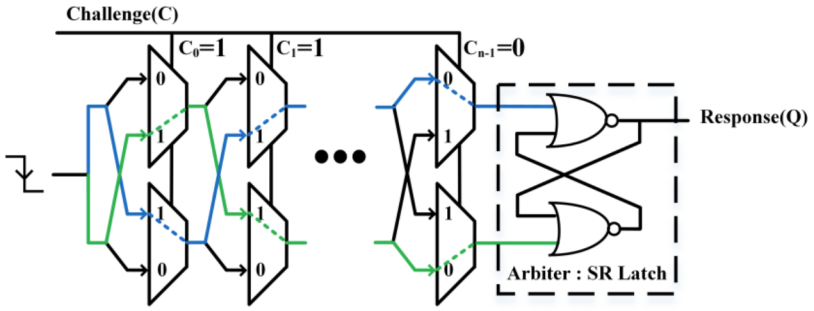

Figure 2: Schematic Diagram of Arbiter PUF

5 Basys3

The Basys 3 is a development board designed by Digilent Inc. It is based on the Xilinx Artix-7 field-programmable gate array (FPGA) and is primarily used for learning and prototyping digital circuit designs.

5.1 Key Features:

1. **FPGA:** The Basys 3 board features the Xilinx Artix-7 FPGA, which provides a reconfigurable hardware platform for implementing digital designs. The Artix-7 FPGA offers a range of logic cells, digital signal processing (DSP) slices, and input/output (I/O) capabilities.

2. **I/O Interfaces:** The board includes various I/O interfaces, such as switches, buttons, LEDs, seven-segment displays, VGA, USB-UART, and Pmod connectors. These interfaces enable interaction with external devices and provide options for user input and output.

3. **Connectivity:** The Basys 3 board supports USB connectivity for programming and power supply. It also offers built-in USB HID and USB-UART bridges, allowing for communication with a host computer.

4. **Power:** The board can be powered through the USB connection or an external power supply.

5. **Development Tools:** The Basys 3 is compatible with Xilinx Vivado Design Suite, a comprehensive development environment for FPGA design. Vivado provides tools for designing, simulating, synthesizing, and programming FPGA designs onto the Basys 3 board.

6. **Educational Use:** The Basys 3 is widely used in educational settings, such as universities and colleges, for teaching digital design, computer architecture, and FPGA programming. Its user-friendly features and versatile I/O interfaces make it suitable for hands-on learning and prototyping.

7. **Project Prototyping:** With its FPGA and various I/O interfaces, the Basys 3 board is ideal for prototyping digital circuits and projects, including embedded systems, digital signal processing, image processing, and more.

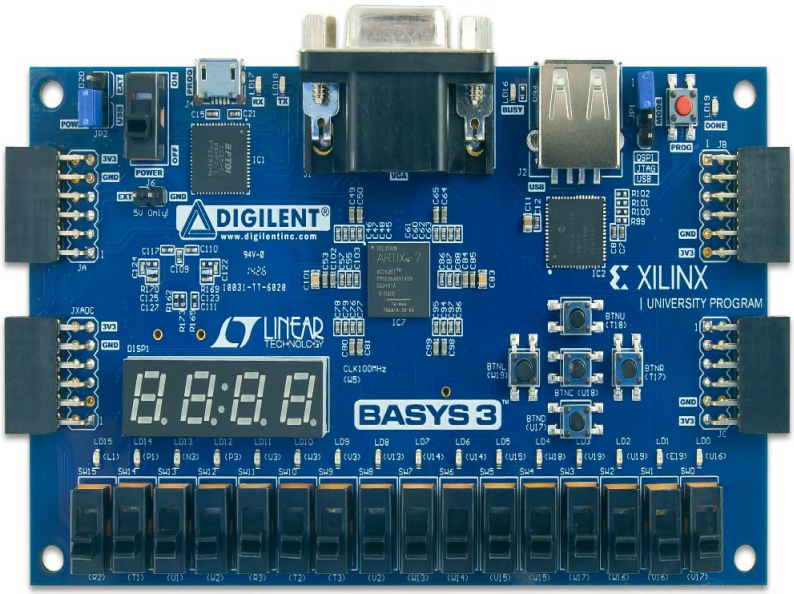

Figure 3: Basys 3 Board

6 Serial communication with microblaze

MicroBlaze is a soft processor core provided by Xilinx for their FPGAs. It is a configurable and customizable processor that can be implemented within an FPGA design. To establish serial communication with MicroBlaze, we use a UART (Universal Asynchronous Receiver-Transmitter) module.

1. **Hardware Configuration:** Ensure that the UART module is properly connected to the MicroBlaze processor within your FPGA design. This typically involves connecting the UART module's transmit (TX) and receive (RX) lines to appropriate pins on the FPGA and making the necessary connections to the MicroBlaze processor.

2. **Software Initialization:** In your MicroBlaze software application, initialize the UART module by configuring its registers. This involves setting the baud rate, data bits, parity, and stop bits according to your communication requirements.

3. **Transmitting Data:** To send data over the serial connection, write the data to the UART's transmit buffer. The UART module will take care of sending the data out through the TX line. You can use UART-specific functions or write to the appropriate UART registers to perform the data transmission.

4. **Receiving Data:** To receive data, read from the UART's receive buffer. The UART module will store incoming data in the receive buffer as it arrives. You can periodically check the receive buffer for new data or use interrupts to handle incoming data in real-time.

5. **Software Handling** Once data is received or transmitted, you can process it as per your application requirements. This may involve parsing received data, responding to specific commands, or performing other necessary operations.

7 Methodology

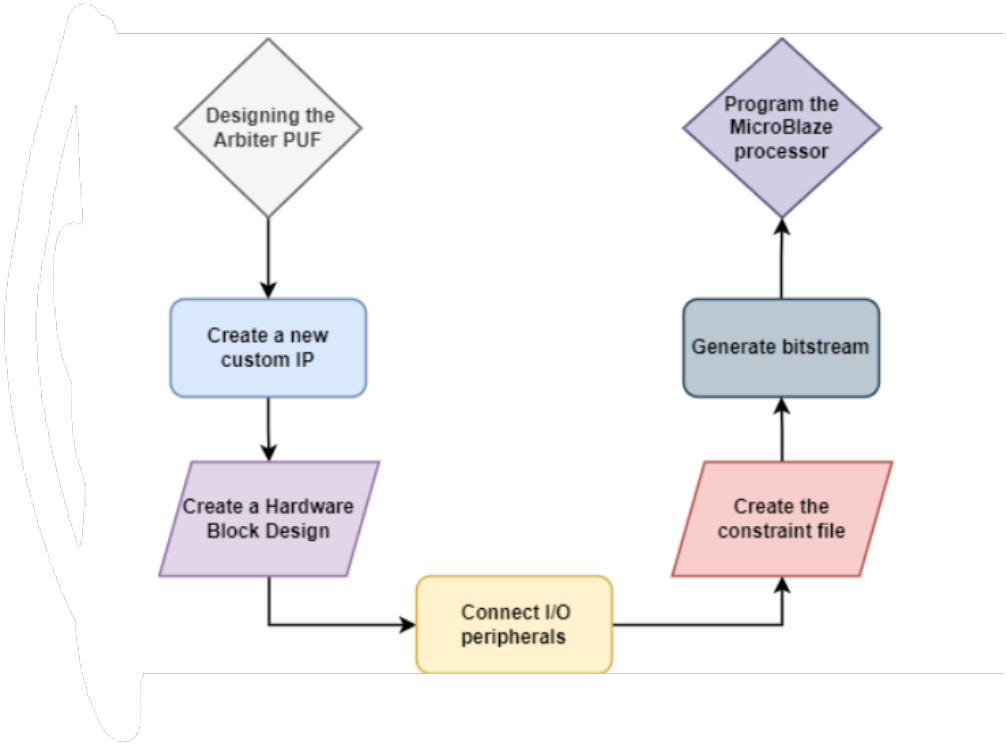

Figure 4: Workflow of assessment methodology

In this section, we discuss the steps taken to implement and analyze the arbiter on FPGA.

7.1 Designing the Arbiter PUF

1. Write the Verilog(or VHDL) Code for the Arbiter PUF using Xlinix Vivado 2019.2. This code defines the behavior and structure of the PUF.

2. Specify the number of arbiter stages and the desired output length for the PUF.

7.2 Create a new custom IP

1. Create a new AXI4 Peripheral and configure its parameters, including number of registers, Data width.

2. Edit the newly created IP. Declare outputs as wire and rename inputs as slv_reg0, slv_reg1, ... and so on. Add the Verilog Code to it.

7.3 Create a Hardware Block Design

1. **Add MicroBlaze processor:** This entails selecting the MicroBlaze IP from the Vivado IP catalogue and configuring its parameters, including clock frequency, cache capacity, and memory mapping.

2. Add the newly created IP to the design.

3. **Add Uarlite IP:** From Vivado IP catalogue, add the Uarlite IP and customize it for I/O peripherals.

7.4 Connect I/O peripherals:

1. Connect I/O peripherals and UART ports to the MicroBlaze processor as the next phase. This requires allocating FPGA pins to the proper peripherals and configuring their parameters.

7.5 Create the constraint file

1. Create the HDL wrapper of the block design.
2. Synthesis and implement the design.
3. Create the .xdc file as per the Basys3 board design and required setup for PUF.

7.6 Generate bitstream:

1. Generate the bitstream file of HDL wrapper and export the hardware including bitstream.

7.7 Program the MicroBlaze processor:

1. Create a Platform project on Vitis IDE and import the generated bitstream file to it.
2. Writing code in a high-level programming language, such as C/C++, and compiling it into an executable binary file that can be loaded onto the MicroBlaze processor.
3. Build the application project and after that, launch the hardware to get the serial output on the terminal.

8 Result

In this section, we present the analysis details for Arbiter PUF.

8.1 Evaluation Metrics for PUFs:

1. **Uniqueness (U):** It is the measurement of the ability of PUF to uniquely distinguish two identical devices. The same challenge is applied to two similar PUF instances, and their hamming distance is calculated as uniqueness. For I PUF instances, uniqueness can be calculated as follows:

$$u = \frac{2}{I(I-1)} \sum_{i=1}^{I-1} \sum_{j=i+1}^{I} AHD(P_i, P_j) X 100\%$$

   where AHD(P_i, P_j) denotes the average Hamming distance between the response of the PUF instance P_i and P_j . Ideally, the value of the uniqueness (u) should be 50

2. **Reliability (R):** It denotes the stability of the PUF response across repeated measurements in uncontrolled environmental conditions. Reliability is measured as:

$$R = 1 - \frac{1}{N} \sum_{i=1}^{N} AHD({}^{r}P_j, {}^{i}P_j) X 100\%$$

   where ${}^{r}P_j$ denotes the response of PUF instance P_j measured in reference environmental condition r and ${}^{i}P_j$ denotes the response during i^{th} measurement using a same challenge. N is the number of different measurements. Ideally, reliability (R) should be 100%.

3. **Uniformity (U_n):** It indicates the probability of 0 and 1 in the PUF response. For better security of PUF 0 and 1 should be equiprobable. For a particular PUF instance uniformity can be calculated as:

$$U_n = \frac{1}{N} \sum_{i=1}^{n} r_i X 100\%$$

   where r_i is the i^{th} response bit (i.e. 0 or 1) and n is the number of response bit. Ideally, uniformity should be 50%.

Uniqueness	Uniformity	Reliability	Challenge Length
51.34	57.64	97.57	64

Resource	Utilization	Available	Utilization(%)
LUT	1898	63400	2.99
LUTRAM	174	19000	0.92
FF	2621	126800	2.07
BRAM	2	135	1.48
IO	4	210	1.90
BUFG	3	32	9.38
MMCM	1	6	16.67

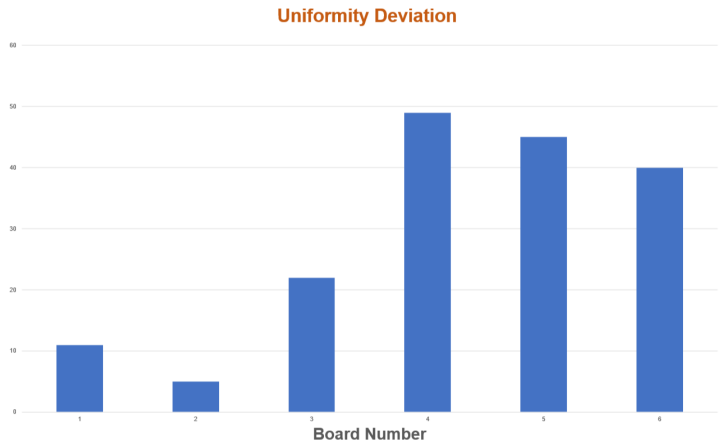

Figure 5: Uniformity deviation for PUFs

Figure 6: Schematic diagram of Arbiter PUF

9 Conclusion

In this project, we successfully designed and analyzed a Physical Unclonable Function (PUF) on the Artix-7 FPGA. Our evaluation of the PUF's performance focused on three key aspects: uniformity, reliability, and uniqueness. The PUF exhibited a statistically uniform distribution of responses, indicating its ability to generate unpredictable and diverse keys. Furthermore, it demonstrated robustness and stability under varying environmental conditions and operating parameters. The probability of collision among generated keys was found to be extremely low, emphasizing the uniqueness of the PUF. Overall, our findings validate the effectiveness of the PUF design on the Artix-7 FPGA, making it a promising solution for secure key generation and authentication applications. Future research could delve deeper into addressing potential sources of non-uniformity and exploring advanced techniques to further enhance the PUF's performance.

References

[1] M. H. Mahalat, S. Mandal, A. Mondal and B. Sen, "An Efficient Implementation of Arbiter PUF on FPGA for IoT Application," 2019 32nd IEEE International System-on-Chip Conference (SOCC), Singapore, 2019, pp. 324-329, doi: 10.1109/SOCC46988.2019.1570548268.

[2] D. P. Sahoo, D. Mukhopadhyay, R. S. Chakraborty and P. H. Nguyen, "A Multiplexer-Based Arbiter PUF Composition with Enhanced Reliability and Security," in IEEE Transactions on Computers, vol. 67, no. 3, pp. 403-417, 1 March 2018, doi: 10.1109/TC.2017.2749226.

[3] Ünsalan, Cem, and Bora Tar. 2017. Digital System Design with FPGA: Implementation Using Verilog and VHDL. 1st ed. New York: McGraw-Hill Education. https://www.accessengineeringlibrary.com/content/book/9781259837906

[4] N. N. Anandakumar, M. S. Hashmi and M. A. Chaudhary, "Implementation of Efficient XOR Arbiter PUF on FPGA With Enhanced Uniqueness and Security," in IEEE Access, vol. 10, pp. 129832-129842, 2022, doi: 10.1109/ACCESS.2022.3228635.