Aidan Nowakowski (aidannow@bu.edu)

EC544 Project Report

Introduction/Project Design

In the world of IoT devices, many of these devices are not resource rich and can lack in cryptographic capabilities. Despite this, authenticating, authorization, and other important security measures are still vital to the proper, and secure, function of these IoT devices. Therefore it is necessary to implement an alternative to storing or generating a cryptographic key on the device. Instead physical unclonable functions (PUFs) can be used. A PUF is a function that generates a device-unique response to a challenge by utilizing the naturally occurring variations in a device generated in the manufacturing process. Since PUFs are unique to each device, they can be used for cryptographic keying in IoT devices since they are less resource-dependent than standard key generation.

This project aimed to implement a PUF that exploited the variations in the manufacturing of FPGA boards that affect the propagation delay of a bitstream through a circuit. PUFs of this type are known as arbiter PUFs. The name arbiter comes from the use of a flip-flop or a latch at the end of the series of switch components to store an output of a race condition. This implementation utilized a data-type flip-flip (DFF) to act as the arbiter. Both the input and clock pins were attached to a series of 128 switch components that utilized two two-to-one multiplexers to continually alternate or maintain the values of each line based on the select line generated from the challenge input. Both lines receive the same input signal 1 at the same time, so the race condition is created as if the clock pin receives the 1 first, the value stored will be 0, but if the input pin receives the 1 first, the value stored will be 1 once the clock pin receives the signal. This arbiter thus acts like a coin flip, where the output is determined based on the delays created by the switching throughout the circuit. However, one bit is not enough to generate a unique signature, so this process is repeated 128 times through 128 different arbiter units to generate the 128-bit unique signature in the PUF module.

Finding these unique signatures must be done on the FPGA board implementations, not the Vivado simulations. To do so a method of communication between the computer terminal and FPGA needs to be implemented to pass challenge values and receive the response outputs. To do so the UART (universal asynchronous receiver-transmitter) protocol was implemented in Verilog. UART is widely used in the embedded system and microcontroller fields to allow for device-to-device hardware communications and only requires the use of two wires. This project's UART communication passed information in 8-bit sections with no parity bit. Ideally, this system would allow for fast communication when transferring the 128-bit challenge and response values and allow for multiple tests to be done quickly on the same board. However, the communication between the board and the computer terminal would prove to cause a myriad of issues for the lab.

Missing Functionality

As stated prior, the UART protocol would not be able to be implemented in the current setup. Due to using the provided Artix-7 FPGA boards in the Photonics 115 laboratory, the available programs and connections were limited. The lab computers connected to the boards ran on Windows which did not have any of the UART terminal programs installed on them which meant that sending and receiving signals over the UART protocol would not be possible with this lab setup. An alternative approach was taken using the provided Vivado program. A debugging profile could be set up to access the current implementation on the board and probe for the propagation delay across the circuit. However, the functionality and accuracy of this method could be questioned as within a single board, using the same bitstream binary and debugger binary would result in propagation delays varying by over 10ns with similar ranges between multiple boards. The variation seen however could potentially imply that with a more thorough measuring system, this PUF implementation could generate unique signatures. Due to the inability to implement an effective communication protocol or a method to accurately assess the propagation delay through the board's debugging core, determining how effective the implemented arbiter PUF was at generating unique and stable signatures was not possible.

Additional Implementation

While the UART protocol implementation ended up not being used directly with the PUF, its implementation was still done correctly through the use of a receiver module and transmitter module, which were tested with a testbench to prove that the communication between the two modules was working as expected. Implementing the UART protocol was not initially presented in the proposal, however, implementing it was still useful as learning how to implement a UART protocol will be extremely helpful in later embedded systems and microcontroller works and related heavily to the communication modules and plays a role in machine-to-machine interactions. Additionally, an implementation utilizing a set-reset (SR) latch was briefly implemented, however, to induce the race case, first an input of 1 would need to be passed to the S and R pins and then followed by an input of 0 to both pins. The DFF implementation was deemed more streamlined and easier to induce a race condition but comparing the two arbiters yielded further insight into how the PUF operated and what would be best to generate the unique signatures.

Conclusions/Lessons

While ultimately the project failed to execute its main objective of generating unique signatures for the Atrix-7 FPGA boards, the process of researching and implementing such a function still provided useful knowledge and development of technical skills. PUFs are an interesting topic that plays a significant role in hardware security, especially as technology advances toward more prevalent edge computing. A way to authenticate devices quickly based on the intrinsic hardware will be important for reliable security and preventing device tampering from going unnoticed. Through researching PUFs and their implementations, a greater knowledge of hardware security can be gained, and technical skills can be further developed through working on a uniquely challenging project in a hardware description language. As well as learning how to implement the UART protocol. Due to the potential implications of the variations in the propagation delay, this project warrants further work that will cultivate the aforementioned skills and hopefully yield unique response signatures.

Youtube link to the presentation: https://youtu.be/wSXjGG4UKLw

GitHub repository link: https://github.com/AidanNowa/EC544_PUF_Project

Sources

1. https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9982457 (Implementation of Efficient XOR Arbiter PUF on FPGA With Enhanced Uniqueness and Security. N. Nalla Anandakumar, Mohammad S. Hashmi, and Muhammad Akmal Chaudhary)
2. https://eprint.iacr.org/2014/802.pdf (Physical Characterization of Arbiter PUFs. Shahin Tajik, Enrico Dietz, Sven Frohmann, Jean-Pierre Seifert, Dmitry Nedospasov, Clemens Helfmeier, Christian Boit, Helmar Dittrich)
3. https://www.researchgate.net/publication/224222003_Implementation_of_a_PUF_circuit_on_a_FPGA (Implementation of a PUF circuit on a FPGA. Mehmet Soybali, Berna Ors, Gokay Saldamli)
4. https://www.diva-portal.org/smash/get/diva2:1413111/FULLTEXT01.pdf (Characterization of FPGA-based Arbiter Physical Unclonable Functions. Jingnan Shao)
5. https://digitalcommons.usf.edu/cgi/viewcontent.cgi?article=8866&context=etd (SR Flip-Flop Based Physically Unclonable Function (PUF) for Hardware Security. Rohith Prasad Challa)