Design and Implementation of a Secure Physical Unclonable Function (PUF)

Project Overview

This project focuses on enhancing FPGA hardware security by generating **unique and device-specific secret keys** using a **delay-based Ring Oscillator Physical Unclonable Function (RO-PUF)**. The design exploits unavoidable **manufacturing (silicon) variations** present in FPGA logic elements. These variations cause small differences in signal propagation delays, which are practically impossible to replicate, making the design resistant to **cloning, counterfeiting, and hardware attacks**.

The implementation is carried out purely at the **circuit design and simulation level** using **Verilog HDL** and **Xilinx Vivado 2018.2**, targeting an **Artix-7 FPGA** (simulation only). No physical hardware deployment or GUI development is involved.

Design Methodology

The project follows a structured hardware design flow similar to safety-critical FPGA designs, focusing on correctness, repeatability, and performance analysis.

Step 1: Design Simulation

A Verilog module is created with clearly defined **input and output ports**, including:

- System clock
- Reset signal
- Enable signal
- PUF response output (binary signature)

Functional simulation is performed to validate correct logic behavior before synthesis.

Step 2: Parameterization of Constants

Key constants (A and B) are defined as **Verilog parameters** to control:

- Number of ring oscillators
- Length of delay paths
- Sampling duration

Parameterization allows scalability and easy tuning of the RO-PUF architecture without modifying the core logic.

Step 3: State Machine Design

A **finite state machine (FSM)** is implemented to control the PUF operation. The FSM manages:

- Initialization phase
- Ring oscillator activation
- Frequency measurement

- Comparison and response generation

Multiple ring oscillators are paired, and each pair forms one PUF comparison unit.

Step 4: Oscillation and Frequency Measurement

Each ring oscillator generates a free-running oscillation. Due to silicon delay variations:

- Different ROs oscillate at slightly different frequencies

Counters are used to **measure oscillation counts** over a fixed time window. The FSM ensures synchronized measurement across all RO pairs.

Step 5: Binary Response Generation

For each RO pair:

- If RO1 frequency > RO2 frequency → output bit = 1
- Else → output bit = 0

The collection of these bits forms a **PUF response vector**, which acts as a **unique hardware fingerprint** for the FPGA.

These scalar bits can be:

- Stored in internal registers
- Used for device authentication
- Compared against a reference signature for security validation

Step 6: Clock and Reset Management

A global clock drives all synchronous logic, ensuring stable timing behavior. The reset signal:

- Initializes counters and FSM states
- Ensures deterministic startup behavior

Dedicated **Verilog testbenches** are written to:

- Apply clock and reset stimuli
- Validate response stability
- Observe waveform correctness

Verification and Simulation

Simulation is carried out using **Xilinx Vivado 2018.2**. The verification process includes:

- Functional simulation of FSM transitions
- Frequency comparison validation
- Response repeatability testing
- Reset and enable condition testing

Waveforms are analyzed to ensure correct oscillation counting and response generation.

Performance Analysis

After synthesis, the following metrics are evaluated:

1. Area

Measured in terms of FPGA resource utilization.

2. Delay

Critical path delay affecting maximum operating frequency.

3. Power

- Static power: Leakage power of FPGA resources
- Dynamic power: Switching activity during RO oscillation

4. LUT Count

Number of Look-Up Tables used by RO, FSM, and counters.

5. Synthesis and Simulation Waveforms

- FSM operation
- Counter values
- Binary PUF output stability

6. Power Breakdown

- Static power consumption
- Dynamic power during PUF evaluation phase.

Applications

- Hardware authentication
- Anti-cloning protection
- Secure key generation
- FPGA IP protection

This figure-1 illustrates the systematic design flow adopted for impulementing a Secure RO-PUF on FPGA. The process begins with the creation and simulation of a Verilog module where delay-based Ring Oscillator (RO) pairs exploit silicon variations to generate unique binary keys. Following this, frequency oscillations from paired ROs are processed to derive scalar responses. A sequence of steps including clock management, testbench writing, and performance analysis follows, ensuring the RO-PUF functions securely and efficiently.

Figure 1: Systematic Design Flow for Secure RO-PUF Implementation

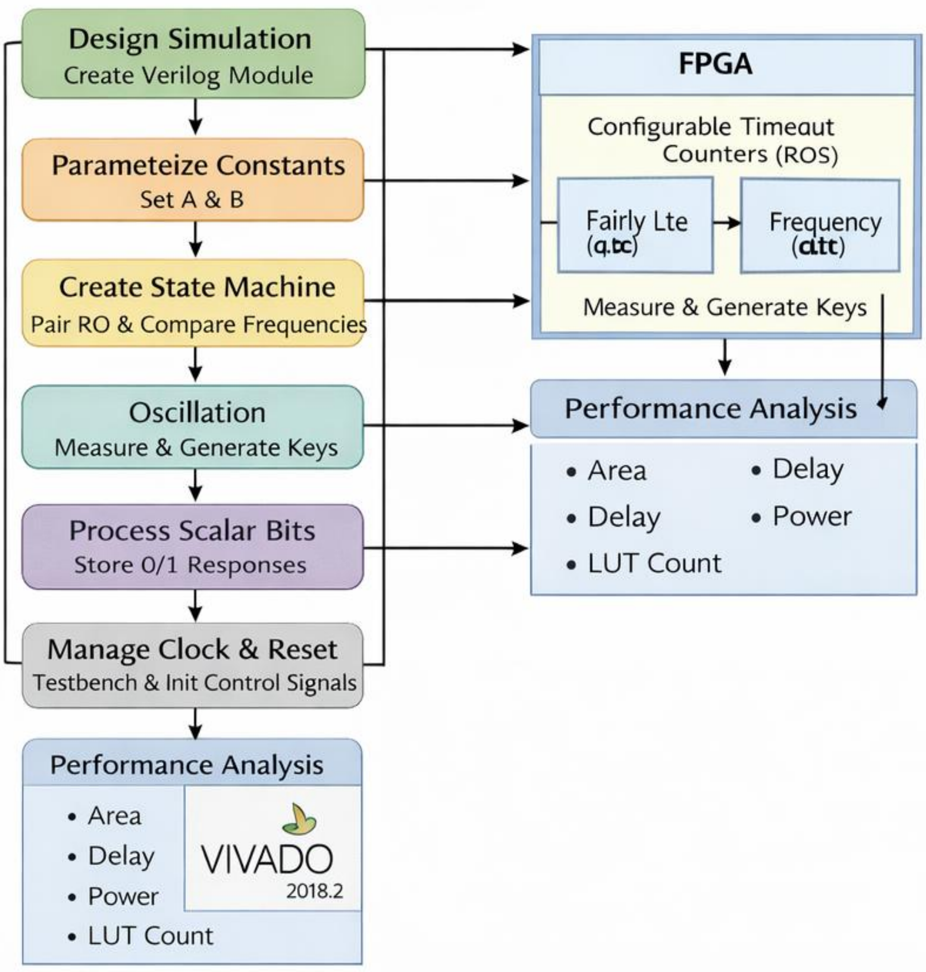

Figure 1: Systematic Design Flow for Secure RO-PUF Implementation