1 Introduction

Hash functions are being used in modern cryptography to provide authentication in a cryptographic system by generating a fixed message digest which is unique to any given message regardless of the message size. A change in a single bit in the message will completely change the message digest and this is how hash functions provide authentication. One of the most recognizable hash functions that is being used in cryptosystems is Secure Hash Algorithm (SHA) which was published by The National Institute of Standards and Technology (NIST) under the name of SHA-0 in 1993. Due to serious flaws in SHA-0, it was shortly replaced by SHA-1 algorithm which was designed by the National Security Agency (NSA) and published in 1995. And as computational power increased, there was a need for a new SHA function with a larger message digest space and that resulted in publishing a hash function with a message digest up to 512 bits as the new SHA-2 standard in 2001 which was also designed by NIST. In 2005, security flaws were found in SHA-1 which resulted in successful partial attacks on it and because SHA-2 had common features with SHA-1, NIST was ready to introduce a new SHA but this time through an international competition where designer all over the world submit their hash function’s design specifications and the design winning the competition will be announced as the SHA-3 standard in 2012.

The competition is composed of three rounds where in each design had to go through different and thorough testing phases and depending on the overall results of the tests in a round, a decision is being made by NIST whether the design should pass to the next round or not. Currently the competition is in its final round (round 3) where there are only five designs left to compete for the title as the new SHA-3 standard. The five finalists are “Blake”, “Grostl”, “JH”, “Keccak” and “Skein”. At this point of the competition, a losing design does not mean that it is a bad design or has a security flaw, in fact, all finalist designs are considered to be secure and fast hash functions that can be used by anyone but there can only be one winner to be the standard for SHA-3. Some cryptosystems use “Serpent” or “RC6” as the secret key cipher in the system instead of using the Advanced Encryption Standard (AES) ”Rijndael” although it was announced the winner by NIST in the competition to choose the AES standard where Serpent came in second and RC6 finished in fourth place. For this purpose, we would like to design a system where the user can choose between some or all of the SHA-3 finalists, to use as the hash function
algorithm in their cryptosystem without affecting the remaining functionality of the cryptosystem.

Partial Reconfiguration (PR) is a method of configuring only part of an FPGA while the remaining FPGA is functioning properly. This method can be used to implement a cryptosystem that uses a hash function and the user can switch between several hash functions to be used in the system.

2 Motivation

The purpose of this project is to design a reconfigurable system that allows the user to switch between different hash function designs to be used with different applications and user demands. This will allow the user to have a low area FPGA platform that is capable of performing authentication or non repudiation with different Hash function algorithms. Having a PR system can also be used to test and measure the five candidates performance in a cryptosystem under the same testing conditions without having to implement the same system five times (one for each SHA-3 finalist). Investigate how all five candidates will perform under the same conditions in terms of speed, power consumption and reconfiguration time for a specific platform.

3 Project Description

We will be designing an embedded system controlled by a Microprocessor (Microblaze or PowerPC) in which a partial reconfigurable region is defined and the FPGA is floor planned so that SHA-3 designs fit in this reconfigurable region.

4 Questions to be answered

The purpose of the project is to create the authentication module in a cryptographic system using different hash function algorithms from SHA-3 finalists and through different project phases from designing to implementation to downloading on the board, we will seek answers to the following questions.

- How much area can be saved building the system using PR compared to a regular implementation?
- Is it worth it to have a single system with all five SHA-3 candidates or having five different designs might be better?
- Does having the same reserved area known as the Partial Reconfigurable Region (PRR) for all the design will affect the speed (or the throughput) of some of the designs compared to others?
- How much time does reconfiguring the design consumes? and is it the same for all the designs regardless of the resources they use?
- Is it worth it to have a single system with all five SHA-3 candidates or having five different designs might be better?
5 Project Timeline

- 03/31 - 04/04: Literature reviews and testing SHA-3 designs.
- 04/05 - 04/12: Creating the Static system with a Microprocessor in EDK.
- 04/13 - 04/21: Interfacing SHA-3 designs with the static design.
- 04/21 - 04/26: Planning PRR to all the reconfigurable SHA-3 designs.
- 04/27 - 05/03: Writing the software part and debugging.
- 05/04 - 05/10: Testing designs on board / Final Report.

6 Possible changes in Project Specifications

Due to time constraints and other issues that can occur within the project, we might have to include some of the SHA-3 finalists and not all of them as we expect, and based on issues related to area constraints, the tools and the board that will be used in the project, it will be determined later which designs will be excluded if any.

7 Literature Search

The following references are described in the reference section:

8 Tentative Final Table of Contents

1. Introduction
   - Overview
   - Method
   - Outline
2. Background
   - Hash function
   - SHA-3
   - Partial Reconfiguration
3. Implementation methodology
   - Design Description
   - Hardware
   - Software
4. Results
5. Conclusion
6. References
References