LPS-SHA1:Low Power and Simple Implementation of Secure Hashing Algorithm (SHA1)using VHDL Implemented on FPGA

Choi Tim Antony Yung, Laurice Sattouf, Dimitri Garcia, **Mohamed El-Hadedy**
Department of Electrical and Computer Engineering
College of Engineering, California Polytechnic State University, Pomona
Pomona, California
Email: {choiyung, lsattouf, dimitrig, mealy}@cpp.edu

ABSTRACT

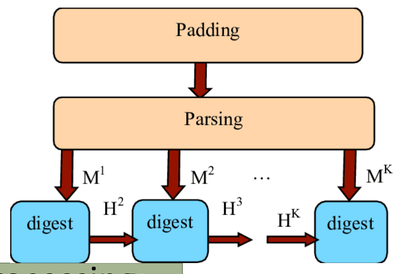
To test the functionality of the SHA1 module in a real-life setting, a system was designed to feed message as byte stream from a computer via UART to a Diligent Nexys A7 development board and display resulting SHA1 digest by VGA output.

INTRODUCTION

It is obvious that the rapid evolution of the communication standards that include message authenticity and integrity verification, require SHA-1 hash functions implementations optimized in terms of performance, power dissipation and size. This can be partially achieved by modifying the embedded hash function. Therefore, the purpose for this paper is to compare between an existing implementation and a proposed one in order to get the simplest and least power consumption.

MATH BEHIND ALGORITHM

SHA1 algorithm can be described in two stages:



1- Pre-processing:

a- padding the message.
b- parsing the message into blocks.
c- setting the initial hash value:

$$H_0^{(0)} = 67452301$$
$$H_1^{(0)} = efcdab89$$
$$H_2^{(0)} = 98badcfe$$
$$H_3^{(0)} = 10325476$$
$$H_4^{(0)} = c3d2e1f0$$

2- Hash Computation:

a. *Preparing the Message schedular:*

$$W_t = \begin{cases} M_t^{(i)} & 0 \le t \le 15 \\ ROTL^1(W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16}) & 16 \le t \le 79 \end{cases}$$

b. *Initialize the five working variables, a, b, c, d, and e, with the (i–1)st hash value:*

$a = H_0^{(i-1)}$
$b = H_1^{(i-1)}$
$c = H_2^{(i-1)}$
$d = H_3^{(i-1)}$
$e = H_4^{(i-1)}$

c. *For t=0 to 79:*

{
$T = ROTL^5(a) + f_t(b, c, d) + e + K_t + W_t$
$e = d$
$d = c$
$c = ROTL^{30}(b)$
$b = a$
$a = T$
}

d. *Compute the i^{th} intermediate hash value $H^{(i)}$:*

$H_0^{(i)} = a + H_0^{(i-1)}$
$H_1^{(i)} = a + H_1^{(i-1)}$

IMPLEMENTATION OF SHA-1 CORE

A- SHA-1 Core:

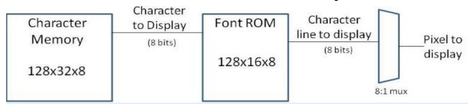
As it is common to transmit and store data in multiple of bytes, this implementation integrated the circuitry of byte to word conversion to optimize specifically intake of byte stream. a 5-bit counter and a 2-bit counter was used to count 80 iterations for each message block such that division circuit can be waived i.e., simple 4x1 multiplexers can handle control of f_t and K_t using 2-bit counter as select. The message schedule was generated and stored as needed, i.e., only the past 16 W_t processed was stored in a 16-word shift register, among them four word was used as input to XOR gate and rotated left by one by rearranging wires from the XOR gate output to provide W_t when $16 \le t \le 79$.
D- 7-sement display unit.

B- UART Controller:

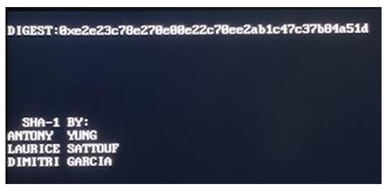
To facilitate use of the SHA-1 core, a UART controller is developed. The UART controller takes a variable input from a keyboard and sends it to the SHA-1 core to be interpreted as the message via terminal. As previously mentioned, the message size of SHA-1 is up to 2^64 bits. Traditional inputs such as switches and buttons are impractical when dealing with large message sizes. With UART, it is possible to quickly send out large amounts of data either manually or with scripting.

C- VGA Controller:

To facilitate use of the SHA-1 core, a UART controller is developed. The UART controller takes a variable input from a keyboard and sends it to the SHA-1 core to be interpreted as the message via terminal. As previously mentioned, the message size of SHA-1 is up to 2^64 bits. Traditional inputs such as switches and buttons are impractical when dealing with large message sizes. With UART, it is possible to quickly send out large amounts of data either manually or with scripting.

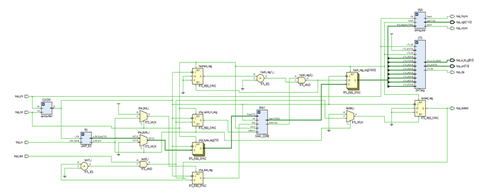


1. A character memory to create a grid-like map of the monitor and assign each location a value.
2. *A font ROM to assign each letter/number a binary value.*
3. *A multiplexer to determine whether a given location should contain a character, and if so, assign the proper value at each pixel.*



RTL SCHEMATIC

The following is the RTL schematic for the implemented using Vivado:



ANALYSIS AND RESULTS:

The integration of byte to word conversion circuit and specialization of performing SHA1 may contribute to the reduced LUTs and Flip-Flops used in this design comparing to a multifunctional design. However, there are room for improvement on power consumption for this design. As W_t are generated by a combinational circuit, a possible improvement is to limit transition of signal on that logic to reduce power consumption, either by converting it to a sequential circuit or mask the activity of the circuit with an enable signal possibly from states signals to reduce signal transitions during idle states.

CONCLUSION

A hardware SHA1 digest core designed to reduce power consumption and resource usage was successfully implemented at register-transfer level by integrating byte to word conversion circuitry and eliminating usage of division circuit by separating message schedule counter into a 2-bit and 5-bit counters. A companion system was designed and implemented to handle serial byte stream message input via UART and output the resulting SHA1 digest to a VGA display. Future work includes further reduction of power consumption possibly by reducing idle state signal transitions

Algorithm	Power Consumption	LUT used	Flip Flops
[10]	0.219	1171	1034
Proposed Design	0.222	727	816

REFRENCES

1. I. Kawazome, "ikwzm/SECURE_HASH: SHA-1,SHA-256,SHA-512 Secure Hash Generator written in VHDL(RTL) for FPGA(Xilinx and Altera).," 13 October 2017. [Online]. Available: https://github.com/ikwzm/SECURE_HASH. [Accessed 17 May 2021].
2. P. P. Chu, "VGA Controller II: Text," in FPGA Prototyping by VHDL Examples, John Wiley & Sons, Inc., p. 291–319.
3. J. M. Diez, S. Bojanić, L. Stanimirovicć, C. Carreras and O. NietoTaladriz, "Hash algorithms for cryptographic protocols : FPGA implementations," 10th Telecommunications Forum TELFOR'2002, No