ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 6, June 2013

Effective Implementation of SHA-1Algorithm using FPGA

Sunil Kumar Tudumu[1], Syamala.K[2]

Student (M.Tech, VLSI-SD), Dept., of ECE, Avanthi Institute Of Engineering And Technology, Visakhapatnam, India [1]

Assistant Professor, Department of ECE, Avanthi Institute Of Engineering And Technology, Visakhapatnam, India [2]

Abstract: In this report Secure Hash Algorithm (SHA-1) is proposed and it is implemented to produce a single output 160-bit message digest (the output hash value) from an input message suing FPGA. The input message is composed of multiple blocks. For providing this output it uses many arithmetic and logical operations. The Secure Hashing Algorithm is issued by Federal Information Processing Standards Publications (FIPS PUBS) and National Institute of Standards and Technology (NIST) after approval by the Secretary of Commerce pursuant to Section 5131 of the Information Technology. "cryptography" by Information Security. The Modelsim.se.v6.2c is used for functional simulation of the SHA-1.

Keywords: SHA-1,algorithm, FPGA, modelSim

I. INTRODUCTION

Very-large-scale integration (VLSI) is the process of creating integrated circuits by combining thousands of transistor-based circuits into a single chip. VLSI began in the 1970s when complex semiconductor and communication technologies were being developed. The microprocessor is a VLSI device. The term is no longer as common as it once was, as chips have increased in complexity into the hundreds of millions of transistors.

The first semiconductor chips held one transistor each. Subsequent advances added more and more transistors, and, as a consequence, more individual functions or systems were integrated over time. The first integrated circuits held only a few devices, perhaps as many as ten diodes, transistors, resistors and capacitors, making it possible to fabricate one or more logic gates on a single device. Now known retrospectively as "small-scale integration" (SSI), improvements in technique led to devices with hundreds of logic gates, known as large-scale integration (LSI), i.e. systems with at least a thousand logic gates. Current technology has moved far past this mark and today's microprocessors have many millions of gates and hundreds of millions of individual transistors.

At one time, there was an effort to name and calibrate various levels of large-scale integration above VLSI. Terms like Ultra-large-scale Integration (ULSI) were used. But the huge number of gates and transistors available on common devices has rendered such fine distinctions moot. Terms suggesting greater than VLSI levels of integration are no longer in widespread use. Even VLSI is now somewhat quaint, given the common assumption that all microprocessors are VLSI or better.

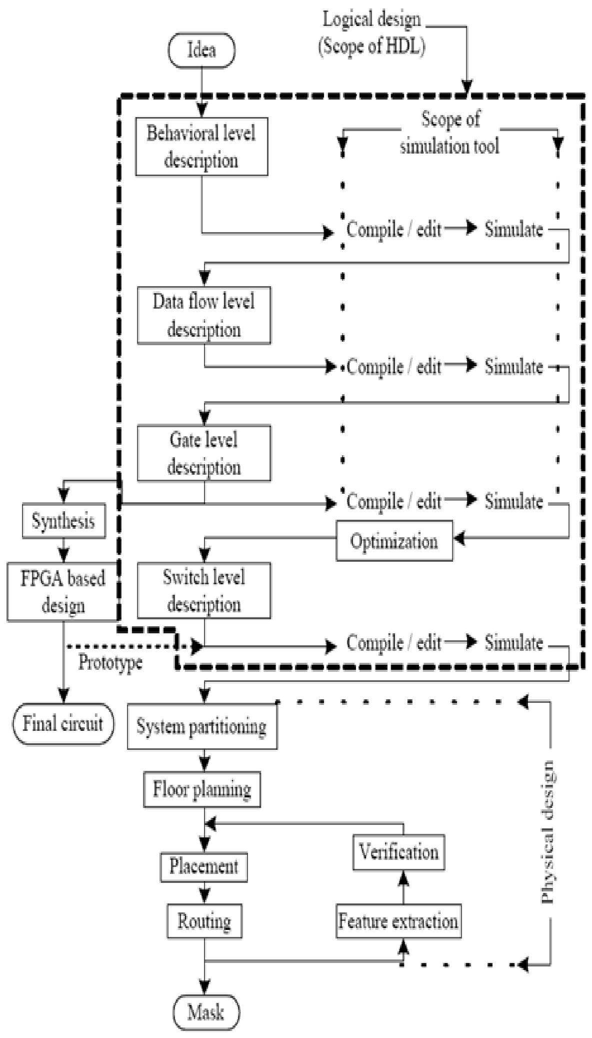
As of early 2008, billion-transistor processors are commercially available, an example of which is Intel's Montecito Itanium chip [1]. This is expected to become more commonplace as semiconductor fabrication moves from the current generation of 65 nm processes to the next 45 nm generations (while experiencing new challenges such as increased variation across process corners). Another notable example is NVIDIA's 280 series GPU.

This microprocessor is unique in the fact that its 1.4 Billion transistor count, capable of a teraflop of performance, is almost entirely dedicated to logic (Itanium's transistor count is largely due to the 24MB L3 cache). Current designs, as opposed to the earliest devices, use extensive design automation and automated logic synthesis to lay out the transistors, enabling higher levels of complexity in the resulting logic functionality. Certain high-performance logic blocks like the SRAM cell, however, are still designed by hand to ensure the highest efficiency (sometimes by bending or breaking established design rules to obtain the last bit of performance by trading stability)

II. TECHNICAL DETAILS

The design descriptions are tested for their functionality at every level – behavioral, data flow, and gate [2,3]. One has to check here whether all the functions are carried out as expected and rectify them. All such activities are carried out by the simulation tool. The tool also has an editor to carry out any corrections to the source code. Simulation involves testing the design for all its functions, functional sequences, timing constraints, and specifications. Normally testing and simulation at all the levels – behavioral to switch level – are carried out by a single tool; the same is identified as "scope of simulation tool".

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021



With the availability of design at the gate (switch) level, the logical design is complete. The corresponding circuit hardware realization is carried out by a synthesis tool. Two common approaches are as follows:

• The circuit is realized through an FPGA. The gate level design description is the starting point for the synthesis here. The FPGA vendors provide an interface to the synthesis tool. Through the interface the gate level design is realized as a final circuit. With many synthesis tools, one can directly use the design description at the data flow level itself to realize the final circuit through an FPGA. The FPGA route is attractive for limited volume production or a fast development cycle.

• The circuit is realized as an ASIC. A typical ASIC vendor will have his own library of basic components like elementary gates and flip-flops. Eventually the circuit is to be realized by selecting such components and interconnecting them conforming to the required design. This constitutes the physical design. Being an elaborate and costly process, a physical design may call for an intermediate functional verification through the FPGA route. The circuit realized through the FPGA is tested as a prototype. It provides another opportunity for testing the design closer to the final circuit.

III. SHA -1 DESCRIPTION AND FORMULATION

Where <<<denotes circular shift to the left by *s* bits and ⊕is a logical xor operation. Let K_t be a constant value for step *t*. The values of *K* are set as follows:

$$K_t = \begin{cases} x''5a827999'' & 0 \le t \le 19 \\ x''6ed9eba1'' & 20 \le t \le 39 \\ x''8f1bbcdc'' & 40 \le t \le 59 \\ x''ca62c1d6'' & 50 \le t \le 79 \end{cases}$$

A function *F*(*X*,*Y*,*Z*) depending on the step *t* is defined as follows :

$$F(X,Y,Z) = \begin{cases} (X \wedge Y) \oplus (\neg X \wedge Z) & 0 \le t \le 19 \\ X \oplus Y \oplus Z & 20 \le t \le 39 \\ (X \wedge Y) \oplus (X \wedge Z) \oplus (Y \wedge Z) & 40 \le t \le 59 \\ X \oplus Y \oplus Z & 60 \le t \le 79 \end{cases}$$

where ⊕ and ^ are bitwise logical and, xor and complement, respectively. The message is processed for 0 ≤*t* ≤ 79 with the following function, which is here called the SHA-1 step function:

$$T = (A \lll 5) + F(B,C,D) + W_t + K_t + E$$

where + denotes an addition modulo 2^{32}. After each step, the values of the registers are set as follows:

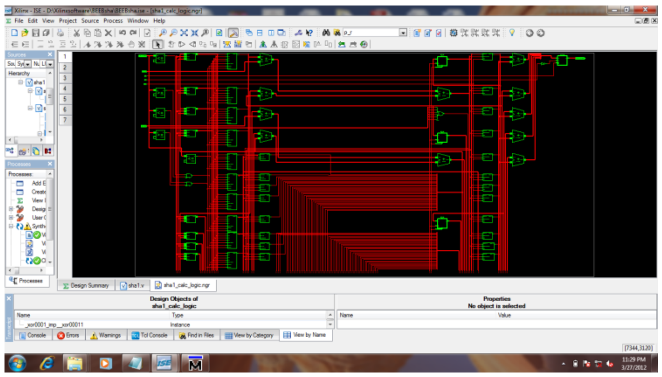
$$\begin{aligned} A &\leftarrow T \\ B &\leftarrow A \\ C &\leftarrow B \lll 30 \\ D &\leftarrow C \\ E &\leftarrow D \end{aligned}$$

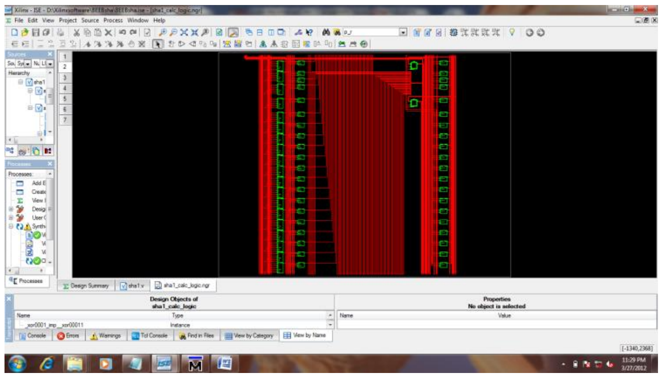
Finally, when all 80 steps have been processed, the following operations are performed:

$$\begin{aligned} H_0 &\leftarrow H_0 + A \\ H_1 &\leftarrow H_1 + B \\ H_2 &\leftarrow H_2 + C \\ H_3 &\leftarrow H_3 + D \\ H_4 &\leftarrow H_4 + E \end{aligned}$$

IV. RESULTS

The simulation of the can be performed through the Modelsim software tool. The version used here is 6.2c. Initially before entering into the coding part the IC name on to which we are going to dump the project is to be specified first.


(a)


(b)

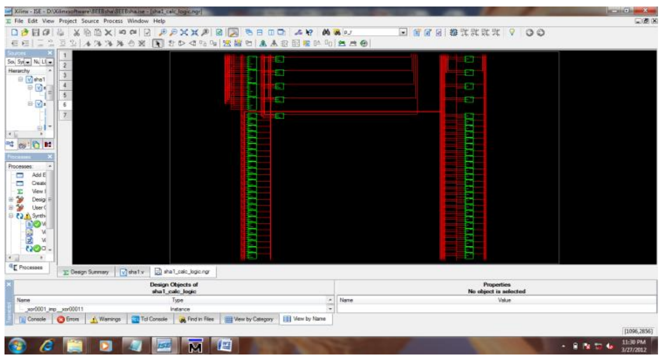
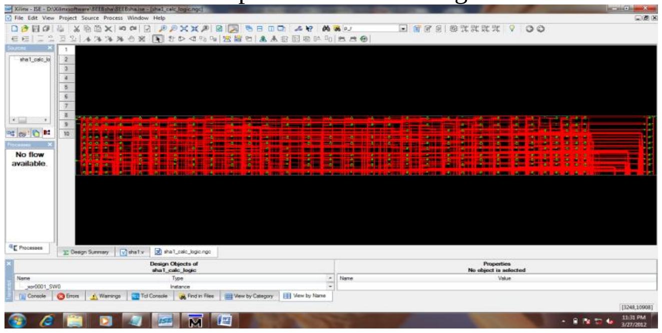
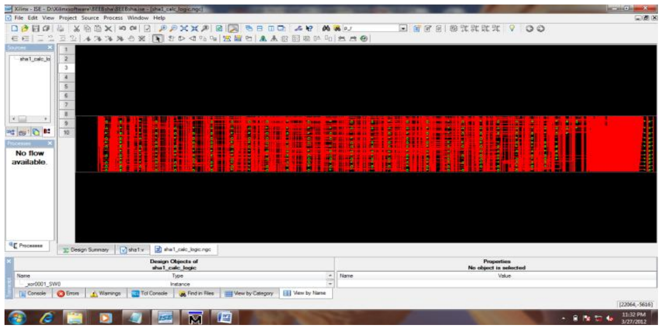

(c)

Fig.2. (a) RTL Schematic Part-1 (b) RTL Schematic Part-2 (c) RTL Schematic Part-3

Fig.2 (a)-(c) describe the schematic representations of the RTL. The same are depicted in the these figures.


(a)


(b)

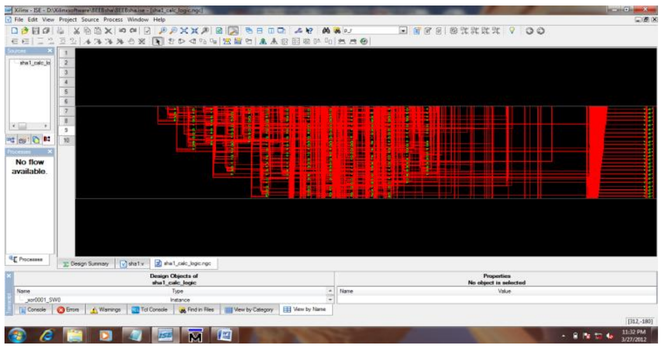

(c)

Fig.3. (a) Technology schematic Part-1 (b) Technology Schematic Part-2 (c) Technology Schematic Part-3

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

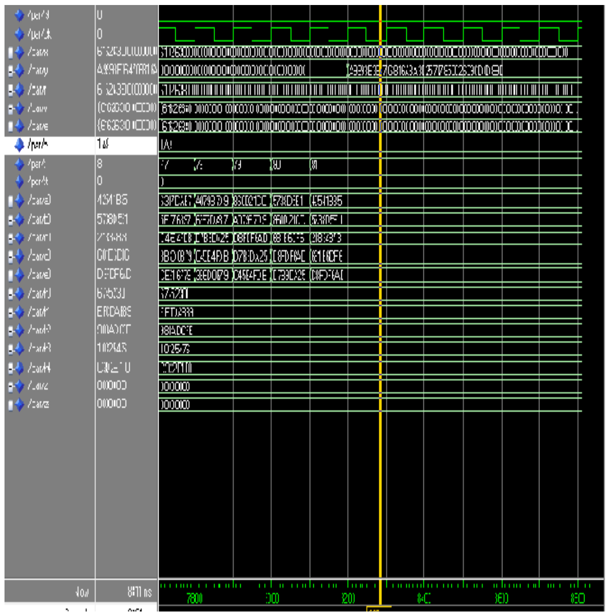
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 6, June 2013

The technology schematics are presented in the figures Fig.3(a)-(c). The output waveforms of the synthesised SHA-1 are shown in the figures Fig.4(a)&(b). The output waveforms are obtained from the test bench are depicted here in these figures.
The definition of different symbols are as follows.

Bit	:	A binary digit having a value of 0 or 1.
Byte	:	A group of eight bits.
FIPS	:	Federal Information Processing Standard.
Word	:	A group of either 32 bits (4 bytes) or 64 bits (8 bytes), depending on the algorithm.

The corresponding function is given as follows.

$$f_t(x, y, z) = \begin{cases} Ch(x, y, z) = (x \wedge y) \oplus (\neg x \wedge z) & 0 \leq t \leq 19 \\ Parity(x, y, z) = x \oplus y \oplus z & 20 \leq t \leq 39 \\ Maj(x, y, z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z) & 40 \leq t \leq 59 \\ Parity(x, y, z) = x \oplus y \oplus z & 60 \leq t \leq 79. \end{cases}$$


(a)

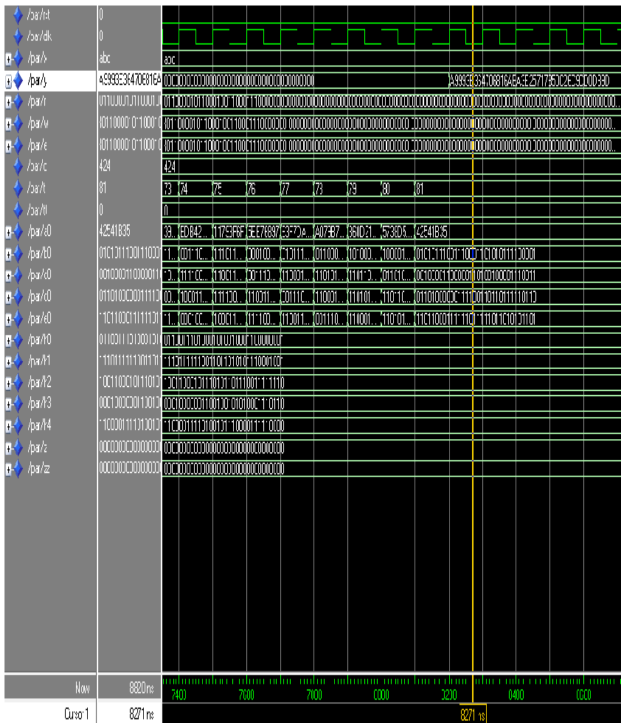

(b)

Fig.4.(a) Output waveforms-1 (b) Output waveforms-2

The applications of the proposed SHA-1 are listed as

1.Recharge Cards: Here we are using SHA-1 algorithm in this In this applications to produce a digested message. Digested message is nothing but the secret number present in the recharge card.
2. **Electronic Money Transfer:** In money transfer also we are using SHA-1. We are using this application in ATM's and DD forms.
2. **Security applications that requires authentication:** Here the message will secured and transferred from source to destination in a compressed manner.
3. **Software distribution:-**
This application is applicable in recharge cards to give them the sequence.
4. **Data storage:-**
The original data will be present in the data storage even if the data is in digest form.

V. CONCLUSION

SHA is famous message compress standard used in computer cryptography. The improved version SHA-1 algorithm has been analysed in this paper, and Implimented by VHDL .Using the SHA-1 module, a long message can be generated a short and Safe message abstract in a very short time i.e message digest. This algorithm used in many authentication applications. It provides more security compared other cryptography algorithms.

REFERENCES

[1] Deng An-wen.cryptography (in Chinese).china water power press 2006:150-15
[2] Wang Yu-Min.Information hidden: Theory and Technology (in Chinese).Tsinghai Publicizing houses.2006:30-35
[3] S Wenbo Mao. Modern cryptography: Theory and practice Pearson education 9.
[4] Zhang Fang-Guo.The Research on Hyperrlliptic curve crpto system (in chinese) Xidian university.2001:222-30
[5] Wade Trappe,Lawrencec.Washington.Introduction to cryptography with coding theory (2nd edition).Pearson Education.2006:133-136
[6] Deng Jian-Zhi,chang Xiao-hui,Gui Qiong, “Design of Hyper Elliptic curve Digitial Signature”, proc.IEEE international conference on information technology And computer science 2008(ITCS09),IEEE press, Jul-2009,pp.45-47

BIOGRAPHIES

Sunila Kumar Tamudu is currently pursuing his M.Tech with specialization in Very Large Scale Integration - (VLSI-SD) at Avanthi Institute of Engineering and Technology, Visakhapatnam affiliated to JNTUK, Kakinada. His research interest include VLSI and Embedded Systems.

Mrs Syamala Kanchimani is currently working as Assistant Professor in the Department of Electronics & Communication Engineering, Avanthi Institute of Engineering and Technology. She has wide teaching experience and guided many undergraduate and post graduate projects.