

Design SHA-2 MIPS Processor Using FPGA

Safaa S. Omran
Computer Engineering Techniques,
College of Electrical and Electronic Techniques
Baghdad, Iraq
Omran_safaa@ymail.com

Laith F. Jumma
Computer Engineering Techniques,
College of Electrical and Electronic Techniques
Baghdad, Iraq
Laith1993@gmail.com

Abstract

According to the wide developments in the area of communications, there is a demand for secure system for data transmissions. Hence, a new algorithm and security standards are developed. One of these algorithms and standards are the Hash function. In this paper, a Hash system SHA-2 MIPS (Microprocessor without Interlocked Pipelines) Processor (single cycle) is designed using Xilinx Spartan-3AN interfaced with keyboard and Video Graphics Array (VGA) display. The implementation of the MIPS processor by choosing a certain number of instructions that is necessary to invoke the SHA-384 and SHA-512 algorithm. A keyboard is interfaced with the Xilinx Spartan-3AN kit, to enable the user to enter the data and the result is shown on a VGA Display.

Keywords: VHDL; SHA-2; MIPS Processor

Introduction

A variation on the message authentication code is the one- way hash function. A hash function accepts a variable-size message M as input and produces a fixed-size output, referred to as a hash code $H(M)$. Hash functions are used in conjunction with symmetric ciphers for digital signatures. In addition, hash functions are used for message authentication [1].

VHDL (Very High Speed Integrated Circuit Hardware Description Language) is a hardware description language. It describes the behaviour of an electronic circuit or system, from which the physical circuit or system can then be attained[2].

One of the earliest hardware designs of hash function is reported by selimis in [3]. The authors in [4] propose a processor for both SHA-1 and MD5 algorithms, and implemented on an Altera Apex 20k. While in [5] a high- performance SHA-1 design in presented and implemented on a Xilinx Vertex-E FPGA. The authors in [6] implement a SHA-1, HAS-160 and MD5 in one chip. In [7] a processor is designed to

support the algorithm of SHA-1 and SHA-256 hashing. However, in this research SHA-384 and SHA-512 hash MIPS processor with keyboard and VGA display is designed and implemented using Xilinx-Spartan-3AN.

SHA-2 Algorithms

The SHA-2 namely SHA-384, and SHA-512 have several commonalities. SHA-2 algorithms are one-way hash functions that process a message to produce a message digest.

This paper was implemented SHA-384 and SHA-512 hash algorithms. Each algorithm can be classified into two stages.

The first stage is the Pre-processing stage and the second is Hash Computation.

A. Pre-processing

SHA-2 input block size various depends on what algorithm is used. In this stage the message is padded into block size called chunk. The input blocks size of SHA-384 and SHA-512 is equal to 1024-bits divided into 16 words $W[0-15]$ of 64-bits. Then extending the 16 words $w[0-15]$ depending on a certain algorithm. Table 1 shows features of the two hash functions.

B. Hash Computation

Hash Computation various depending on the algorithm used if it is SHA-384 or SHA-512.

• SHA-512 Computation

The message digest is 8 hash variables each of 64 bits are used. It take 80 rounds to completed SHA-384 and SHA-512 calculation. SHA-512 algorithm is done as follows:

In the first step hash variables shown in Table II and are initialized.

In the second step message words $W[0-15]$ extended into 80 words $W[16-79]$

$$S0 = (W[t-15] \ggg 1) \oplus (W[t-15] \ggg 8) \oplus (W[t-15] \gg 7)$$

$$S1 = (W[t-2] \ggg 19) \oplus (W[t-2] \ggg 61) \oplus (W[t-2] \gg 6) \quad (1)$$

$$W[t] = W[t-16] + S0 + W[t-7] + S1$$

Where >>> is right rotate and >> is right shift

Table I. Hash Function Comparison

Algorithms	Rounds & number of W	Block Size (bits)	Hash output Size (bits)	Max Size of input (bits)
SHA-384	80	1024	384	$2^{128} - 1$
SHA-512	80	1024	512	$2^{128} - 1$

Table II. Initialized SHA-512 Variables

<i>H0</i>	<i>H1</i>	<i>H2</i>	<i>H3</i>
<i>6A09E667</i> <i>F3BCC908</i>	<i>BB67AE85</i> <i>84CAA73B</i>	<i>3C6EF372</i> <i>FE94F82B</i>	<i>A54FF53A</i> <i>5F1D36F1</i>
<i>H4</i>	<i>H5</i>	<i>H6</i>	<i>H7</i>
<i>510E527F</i> <i>ADE682D1</i>	<i>9B05688C</i> <i>2B3E6C1F</i>	<i>1F83D9AB</i> <i>FB41BD6B</i>	<i>5BE0CD19</i> <i>137E2179</i>

In the third step 8 variable (*A to H*) is added which are equal to (*H0 to H7*)

The fourth step is the main loop where (*A to H*) will be calculated in 80 Round as follow:

$$TEMP1 = H + S1 + CH + K[t] + W[t] \quad TEMP2 = S0 + MAJ$$

$$H = G \quad G = F \quad F = E$$

$$E = D + TEMP1 \quad (2)$$

$$D = C \quad C = B \quad B = A$$

$$A = TEMP1 + TEMP2$$

where $K[t]$ is round constants and ($S0, S1, CH, MAJ$) equation are below

$$V0 = (A \ggg 28) \oplus (A \ggg 34) \oplus (A \ggg 39) \quad V1 = (E \ggg 14) \oplus (E \ggg 18) \oplus (E \ggg 41)$$

$$CH = (E \& F) \oplus ((\neg E) \& G) \quad (3)$$

$$MAJ = (A \& B) \oplus (A \& C) \oplus (B \& C)$$

The final step is adding $(A$ to $H)$ with $(H0$ to $H7)$ which produce the message digest of 512-bit.

$$H0 = H0 + A \quad H1 = H1 + B \quad H2 = H2 + C$$

$$H3 = H3 + D \quad (4)$$

$$H4 = H4 + E \quad H5 = H5 + F \quad H6 = H6 + G \quad H7 = H7 + H$$

- *SHA-384 Computation*

SHA-384 and SHA-512 are identical algorithms except that: The message digest in 384-bit is constructed by omitting $(H6-H7)$.

The initial hash values for SHA-384 are different from that of SHA-512 $H0$ through $H7$ are shown in Table III.

Table III. Initialized SHA-384 Variable

$H0$	$H1$	$H2$	$H3$
CBBB9D5D	629A292A	9159015A	152FECD8
C1059ED8	367CD507	3070DD17	F70E5939
$H4$	$H5$	$H6$	$H7$
67332667	8EB44A87	DB0C2E0D	47B5481D
FFC00B31	68581511	64F98FA7	BEFA4FA4

Fig. 1 shows a block diagram for SHA-2 computation.

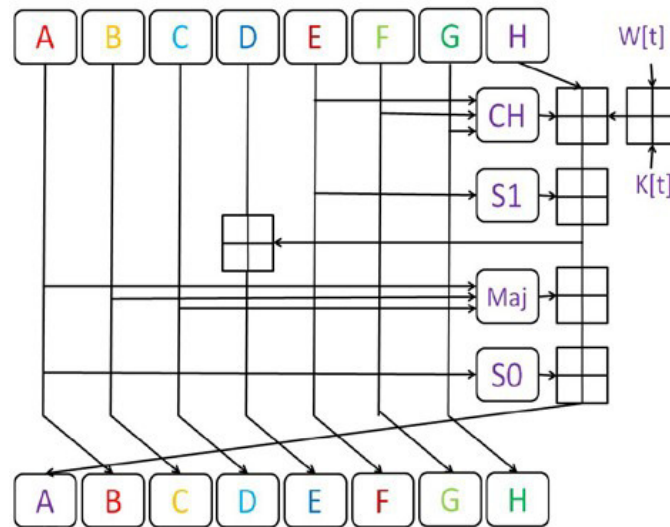


Fig. 1 SHA-2 Computations

Instruction Set Architecture Size

MIPS processor uses 32-bit for each instructions. MIPS defining three instruction formats: R-type, I-type, and J-type. *R-type* instructions have three registers. *I-type* instructions have two registers and a 16-bit immediate value. *J-type* (jump) instructions have one register and a 26-bit immediate.

• *R-type Instructions*

R-type instructions have three registers as operands: two as sources and one as a destination.

It has six fields: funct(function), op(opcode), rs(source register), rt(target register), rd(destination register) and sham(shift operations). Each field is five or six bits, as indicated in fig.2a.

op	rs	rt	rd	sham	funct
6-bit	5-bit	5-bit	5-bit	5-bit	6-bit

Fig. 2a R-type

• *I-Type Instructions*

I-type instructions have two register operands and one immediate operand. It has four fields: imm, op, rs and rt. The first three fields, op, rs and rt, are like those of R-type instructions. The imm field holds the 16-bit immediate, as indicated in fig.2b.

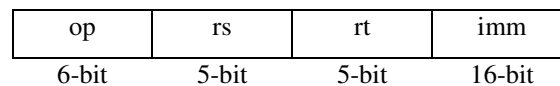


Fig. 2b I-type

• *J-type Instructions*

This format is used only with jump instructions. This instruction format uses a single 26-bit address operand, addr. J-type instructions begin with a 6-bit opcode. The remaining bits are used to specify an address addr, as indicated in fig.2c.

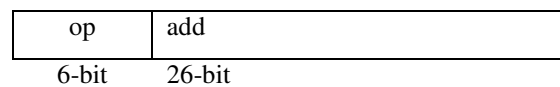


Fig. 2c J-type

Processor Design

The single-cycle microarchitecture executes an entire instruction in one cycle. It is easy to explain and has a simple control unit. Because it completes the operation in one cycle it does not require any nonarchitectural state. However, the cycle time is limited by the slowest instruction [8-9].

In this research few instructions were implemented in the design only the instructions that requires to invoke the algorithm of SHA-384 and SHA-512 were implemented using VHDL in Xilinx ISE software environment .

Since SHA-384 and SHA-512 hash variable is 64-bit so in this paper a MIPS single-cycle processor of 64-bit is designed for SHA-384 and SHA-512.

The simple design of a 64-bit, single cycle MIPS processor consists of two parts:

- 64-bit Datapath
- Control unit.

A. 64-bit Datapath

A 64-bit single cycle MIPS processor requires a 64-bit datapath. It contains element such as registers, memories, ALUs, sign, extenders and multiplexers. A description for each datapath is shown below:

a. Program counter (PC): is a register represents the address of instruction to execute which has 32-bit size.

b. Instruction memory: store instructions used to execute hash function and consist of a 64-bit data as its output

c. Register file : consist of 32 registers, each has 64-bit in size. It has two read port and one write port.

d. Data memory : has one input write port (WR) and two output read ports (RD).The input is written in memory which specified by the address (A) if (WE) signal is 1 at the raising edge of (clk) signal, each memory location is 64-bit size.

e. Sign extension simply copies the most significant bit of a 16-bit input size.

f. ALUs is used in order to execute the arithmetic and logical instructions. ALUs take alucontrol(2:0) as input and generate corresponding function . Note that there are modification in some instruction.

Table IV. MIPS instruction and where it used

Instruction	SHA-84	SHA-512
JMP	Y	Y
LOAD	Y	Y
STORE	Y	Y
ADDI	Y	Y
BEQ	Y	Y
XOR	Y	Y
AND	Y	Y
NOT	Y	Y
OR	Y	Y
ADD	Y	Y
RR 28	Y	Y
RR 34	Y	Y
RR 39	Y	Y
RR 14	Y	Y
RR 18	Y	Y
RR 41	Y	Y
RR 1	Y	Y
RR 8	Y	Y
RR 19	Y	Y
RR 61	Y	Y
SR 7	Y	Y
SR 6	Y	Y

B. Control unit

The control unit receives from the instruction memory the current instruction funct(5:0) and Opcode(31:26) of the datapath which tell it what to do to execute the instruction.

Fig .3 shows the complete design of the MIPS processor of 64-bit and Table IV shows all instructions required to invoke the SHA-384 and SHA-512 algorithms.

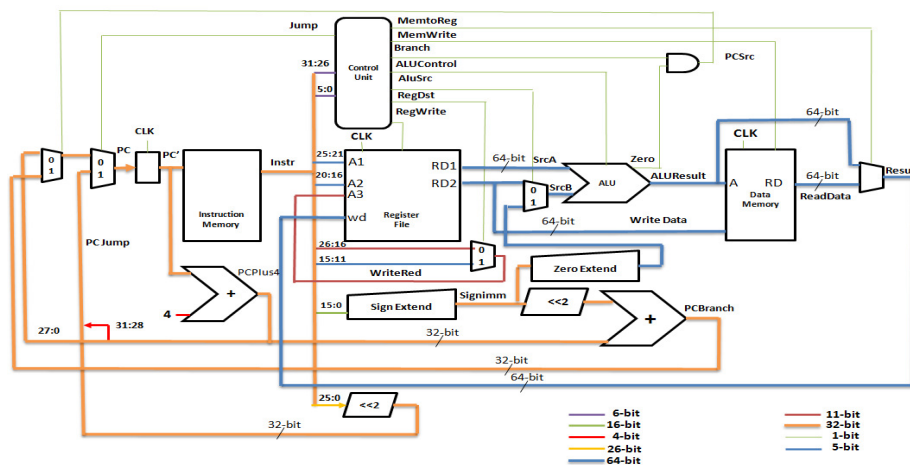


Fig.3 Block diagram of MIPS Processor

Keyboard & VGA Display Interface

A keyboard consists of a matrix of keys and an embedded microcontroller that monitors (i.e., scans) the activities of the keys and sends *scan code* accordingly [10].

For example, when we press and release the N key, the keyboard first transmits its make code (31) and then the break code (F0 31).

The information is transmitted as an 11-bit “packet” that contains a start bit, 8 data bits, an odd parity bit, and a stop bit, as shown in Fig 4.

In this research take White Resulting black to show the character and screen background. The patterns of the tiles is 8- column-by-16-row for each character since the resolution screen is 640-by-480, 80 tile s can be fitted into a horizontal line and 30 tiles can be fitted into a vertical line. Fig .5 shows the ROM content of the letter “F”.

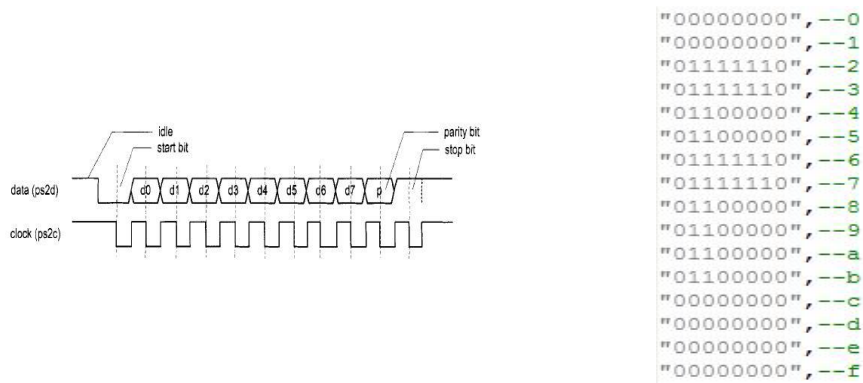


Fig. 4 PS/2 Bus Timing Waveforms

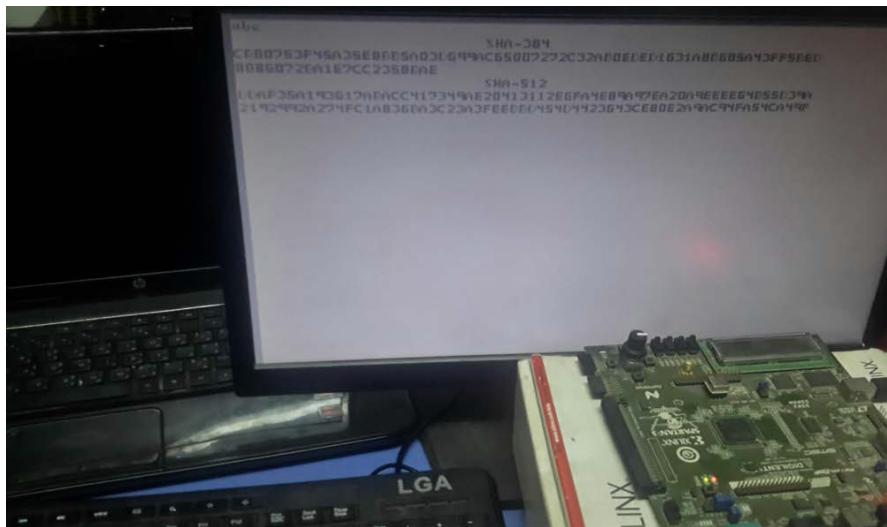


Fig 5 Font pattern for the letter “F”.

The FPGA application have eight colour corresponding to VGA port with a three-bit interface as shown in Table V.

Table V. Resulting color

VGA_R[3:0]	VGA_G[3:0]	VGA_B[4:0]	Resulting Color
0000	0000	0000	Black
0000	0000	1111	Blue
0000	1111	0000	Green
0000	1111	1111	Cyan
1111	0000	0000	Red
1111	0000	1111	Magenta
1111	1111	0000	Yellow
1111	1111	1111	White



Results

10

Conclusion

A hash SHA-384 and SHA-512 was implemented using FPGA Spartan-3an. A MIPS processor was designed using VHDL Xilinx ISE software language with only instructions that Hash need. This study is a starting point for future studies and can be extended to invoke the algorithm for SHA-384 and SHA-512 by using a superscalar MIPS processor.

References

- [1] William Stallings, Cryptography and Network Security Principles and Practices.: Prentice Hall, November 16, 2005.
- [2] Volnei A. Pedroni, Circuit Design with VHDL. London, England: MIT Press, 2004.
- [3] N.Sklavos, and O.koufopavlou G.Selimis, "VLSI implementation of the keyed-hash message authentication code for the wireless ," in 10th IEEE international Conference on Electronics, Circuit, and System (ICECS'03), Dec. 2003, pp. 24-27.
- [4] C.P.Su, C.T.Huang, and C.W.Wu M.W.Wong, "An HMAC processor with integrated SHA-1 and MD5 algorithms," in In Proceeding of the 2004 Conference on Asia South Pacific Design Automation, 2004, pp. 456-458.
- [5] A.Kakarountas, A.Miliadonis, and C.Coutis H.Michail, "Efficient implentation of the keyed-hash message authentication code (HMAC) using the SHA-1 hash Function," in 11th IEEE International Conference on Asia South Pacific Design Automation (ASP-DAC'04),2004, pp. 567-570.
- [6] D.W.Kim, T.W.Kwon, and R.J.Choi Y.K.Kang, "An efficient implementation of Hash function processor for IPSEC," in Proceeding of IEEE Asia-Pacific Conference,2002, pp. 93-96.
- [7] M.Askar and T.S.Celebi, "Design and FPGA implementation of Hash processor," in Information Security and Cryptology Conference (ISC 2007), Ankara, Turkiya, Dec. 2007, pp. 85-89.
- [8] J.L.Hennessy D.A.Patterson, Computer Organization And Design The Hardware/Software Interface, 4th ed.: ELSEVIER, 2012.
- [9] D.M.Harris and S.L Harris, Digital design and computer architecture, 1st ed.: Elsevier, 2007.

- [10] P.P.Chu, FPGA Prototyping By VHDL Examples. Canada: John Wiley & Sons, Inc., Hoboken, New Jersey.,2008.