Volume No 03, Special Issue No. 01, March 2015 ISSN (online): 2348 – 7550

ANALYSIS OF SHA-1 IMPLEMENTATION USING BASELINE ARCHITECTURE AND MULTI-INPUT ADDING

Deepika Sharma

Student, Department of Electronics and Communication Engineering,

School of Engineering and Technology, Poornima University, Jaipur (India)

ABSTRACT

Due to the rapid developments in the wireless communications area and personal communications systems, providing information security has become a more and more important subject. This security concept becomes a more complicated subject when next-generation system requirements and real-time computation speed are considered. In order to solve these security problems, lots of research and development activities are carried out and cryptography has been a very important part of any communication system in the recent years.The hardware is described in VHDL and verified on Xilinx FPGAs. The advantages and open issues of implementing hash functions using a processor structure are also discussed. 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.

Keywords: Cryptography, Hash functions

I. INTRODUCTION

Cryptography is the branch of computer science that deals with security. It supports operations such as encryption and decryption. The cryptography is implemented in the form of hash functions, symmetric key algorithms, and public key algorithms. The symmetric and public key algorithms are used for encryption and decryption while hash functions are one way functions as they don't allow the retrieval of processed data. As MD5 and SHA are the two mostly used algorithms in the industry, this paper focuses on secure hash algorithm. Hash algorithms, also commonly called as message digest algorithms, are algorithms generating a unique fixed-length bit vector for an arbitrary-length message *M*. The bit vector is called the hash of the message and it is here denoted as *H*. The hash can be considered as a fingerprint of the message.

The hash function *H* must have the following properties:

- *One-way property*: for any given value *h*, it is computationally infeasible to find *x* such that $H(x) = h$.
- *Weak collision resistance*: for any given message *x*, it is computationally infeasible to find $y \neq x$ with $H(y) = H(x)$.
- *Strong collision resistance*: it is computationally infeasible to find any pair (x, y), such that $H(x) = H(y)$.

II. SHA-1 ALGORITHM

Secure Hash Algorithm (SHA) is described in the National Institute of Standards and Technology's (NIST) Federal Information Processing Standard (FIPS) 180-2: Secure Hash Standard (SHS) [3]. SHS describes the following algorithms: SHA-1 (SHA-160), SHA-256, SHA-385 and SHA-512, where the number is the length of the hash *H* in bits. In this report, only SHA-1 (SHA-160) is considered. SHA-1 is widely used in various public-key cryptographic algorithms, e.g. in Digital Signature Algorithm (DSA) [6]. SHA-1 calculates a 160-bit *H* for a *b*-bit *M*. The algorithm consists of the following steps:

1. *Appending Padding Bits:* - The *b*-bit *M* is padded in the following manner: a single 1-bit is added into the end of *M*, after which 0-bits are added until the length of the message is congruent to 448, modulo 512.
2. *Appending Length:* - A 64-bit representation of *b* is appended to the result of the above step. Thus, the resulted message is a multiple of 512 bits.
3. . *Buffer Initialization:* - Let H0, H1, H2, H3 and H4 be 32-bit hash value registers. These registers are used in the derivation of a 160-bit hash H. At the beginning, they are initialized as follows:

H0 = x″67452301″
H1 = x″e f cdab89″
H2 = x″98badc f e″
H3 = x″10325476″
H4 = x″c3d2e1 f 0″

1. *Hash Calculation:* SHA1 may be used to hash a message, M, having a length of *l* bits, where $0 \le l \le 264$. The algorithm uses:

- A message schedule of 80x32-bit words. The words of the message schedule are labeled W0, W1.
- Five working variables of 32-bits each. The working variables are labeled as: A, B, C, D and E.
- A hash value of five 32-bit words. The words of the hash value are labeled as: H0(i), H1(i), H2(i), H3(i), H4(i) which will hold the initial hash value H(0), replaced by each intermediate hash value (after each message block is processed) H(i) where i denotes the number of 512 bit block being processed in the message M, and ending with the final hash value, H(N) where N is the number of the last 512 bit block in the message M.
- A single temporary word, T. Previously defined constants which are labeled Kt, where t is the round number.

The calculation is carried out as follows:

The message schedule is prepared, i.e. the message word that is going to be used in that round is prepared. This computation is done as described in the following formula:

$Wt = Mti \; 0 \le t \le 15$

$Wt = ROTL(Wt\text{-}3 \oplus Wt\text{-}8 \oplus Wt\text{-}14 \oplus Wt\text{-}16) \; 16 \le t \le 79$

In the above formula Mit denotes the tth 32-bit message word of the ith 512-bit message block in the message M. The 5 working variables A, B, C, D and E that are going to be used in the computation are prepared as follows:

A = H0 (i–1) B = H1 (i–1)

C = H2 (i–1) D = H3 (i–1) E = H4 (i–1)

After these initializations, the final values of the working variables for that round are calculated as described below:

$T = S^5(A)+f(t;B,C,D) + E + Wt + Kt$

$E = D$

$D = C$

$C = S^{30}(B)$

$B = A$

$A = T$

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

$H0 \leftarrow H0+A$

$H1 \leftarrow H1+B$

$H2 \leftarrow H2+C$

$H3 \leftarrow H3+D$

$H4 \leftarrow H4+E$

4. *Output:* - When all *Mj* have been processed with the above algorithm, the 160-bit hash *H* of *M* is available in *H*0, *H*1, *H*2, *H*3 and *H*4.

III. VERILOG IMPLEMENTATION

In this study the aim is to implement the designed hash function core on Verilog. The whole package and separate modules were synthesized and analyzed using Xilinx ISE 12.1 tool.

- The Verilog implementation was divided into five modules: *Initial module:* - It collects the serial input bits and sends 512 bit blocks to the next module.
- *Round module:* - It performs the hashing calculations and operations on the input message block and previous hash output to generate a new hash value.
- *Last Block module:* - At the end of the message bit stream the final message block of 512 bits has to be prepared by adding 64 bits of message length at the end of 448 bits of input message block, padded accordingly to suffice the word size requirement. This final message block does this function of preparing the last message block.
- *Final module:* - This module computes the hash value by adding the previous hash value to the new hash value achieved from the Round module. Then it sends the 160 bit hash value, bit by bit (serially).

Top module: - This module is the control unit for controlling the functioning of the rest of the modules and to ensure that the SHA1 algorithm flow is followed and maintained

IV. METHODOLOGIES/SOLUTION APPROACHES

[Aianhua He, et-al, 2009] has proposed the FPGA based and performance analysis of a compatible SHA series design. The novelty of the design is that it can provide all of the SHA functions: SHA-1, SHA-256, and SHA-512 with limited trade- off. Block RAM, the base element of FPGA, is used to diminish the logic block consumption, while the scalable Wt generator and the arithmetic logic reuse decrease the proneness of the extra resource consumption. The Mixed-SHA is able to achieve 442Mbps. Two other assumptions, SHA_1_256,

SHA_1_512, are made to discuss the bottleneck of the design, and providing two other optional compatible designs, at the same time, to face different requirements of the application. The throughput then raise to 1.2Gbps

With using about 1100 Slices, less than 1% of the total resources in xc5vlx220. In all, by comparing with the performance of the design that quote from other authors, the Mixed-SHA implementation has a comparable performance, and better compatibility than other design [2].

[Brian Baldwin, et-al, 2010] has presented a methodology for fair and accurate comparisons of the SHA-3 hash functions. Author implemented and tested as many designs as was necessary to obtain full coverage of all of the hash variants as required by NIST. Author developed a hardware wrapper to allow inclusion of padding and interfacing, to obtain the full timing and area analysis, and for completeness compared the area and speed of the hash designs both internal and external of this wrapper. Finally author presented throughput results for both long and short hash messages inclusive of this wrapper [3].

[Bernhard Jungk, et-al, 2011] has focused on area-efficient FPGA implementations of the SHA-3 finalists. The performance of Gristle is considered the best alongside Keycap in terms of the throughput-area ratio; BLAKE follows closely, while Skein and JH trail behind. If the focus lays on pure area consumption, the situation reverses and it is much easier to implement a really small JH or BLAKE design.Keccak, Gristle and Skein are much bigger. There is still room for improvements of almost all implementations. For example, the area of Keycap could probably is further reduced by making a design which usesonly4slices in parallel and JH could be much faster, if more parallelism is used [5].

[C.P Arsenal, et-al, 2007] has discussed different design architectures of Blake-256 implemented on FPGA. Design uses large hardware resources gives maximum throughput i.e. 8G design requires only 14 clock cycles for Hash value calculation resulted throughput of 2.6 Gbps. 1G design gives most efficient results in terms of number of slices utilized. The optimized delay path is utilized in 4G design with respect to Virtex 5 architecture. That's gives maximum TPA of 2.1. Appropriate selection of number of slice LUTs and Slice registers and their placement according to the Virtex 5 Device Architecture Resources gives the optimized results. The selection of architecture is dependent upon type of application either high speed requirements or low area constraints, suitable optimization could be performed in a particular domain to achieve best design results [4].

[Fatma Kahri, et-al, 2013] has presented an architecture and efficient hardware implementation of SHA -256. Author reported the implementation results of SHA-256 and new hash function on Xilinx Virtex 2, Virtex 5, Virtex 6 and Virtex 7 FPGAs. The performance of the implementation in terms of area, throughput, frequency and efficiency and compared the standard with the newest hash function [9].

V. STRENGTHS AND LIMITATIONS

- MD5 and SHA-1 both have very fast bitwise operators that make up the function , so they can be computed very, very quickly on a modern processor.
- Hash functions are the most important cryptographic algorithms and used in several fields of communication integrity and signature authentication.
- It is used in Digital signature algorithms and Throughput improvement.

- Rather than storing a user's password, a system will typically store the hash of the password instead. When a user enters their password, the hash is then computed and compared with the stored hash. If the hash matches, due to the collision resistance property of hashing algorithms, it implies that the passwords match.
- The sender can hash a file before sending to the recipient. The recipient will then hash the file received and check the hashes match. This can also be used for the storage of files, to ensure files have not been corrupted or modified.

VI. IMPLEMENTATION RESULTS

As shown in Table 1 the proposed SHA-1 implementation reduces the area by 24.5% and increases the speed by 13.6%.Meanwhile my implementation with the same clock speed reduces the area by 39.7% which meant significant decrease in area.

Table 1 Implementation Results

	Frequency(MHz)	Area
Proposed	714	6067.8
	625	5469.5
[3]	625	9064.5

VII. CONCLUSION

It is stated that SHA-1 can be efficiently implemented on FPGAs with minimal logic resources. In this study hash functions SHA1 is implemented in a processor structure using the same hardware blocks. These hash functions are examined in detail and a new instruction set is proposed. Hash processor is fully designed and captured using VERILOG HDL in Xilinx ISE software environment. The design is also implemented on Xilinx FPGA and implementation results are given. The designed hash function core has serial interface that makes communication with the external units such as a personal computer possible.

VIII. ACKNOWLEGDEMENT

I would like to express my deep gratitude and thanks to **Prof. Mahesh Bundele (Coordinator, Research), Poornima University** for giving me an opportunity to work under his guidance for review of research papers and his consistent motivation & direction in this regard. I would also express my sincere thanks to **Mr. Dipesh Patidar (Asst.Professor, ECE), PIET** for their guidance and support.

REFERENCES

[1] Dai Zibin; Zhou Ning, "FPGA implementation of SHA-1 algorithm," ASIC, 2003. Proceedings. 5th International Conference on, vol.2, no., pp.1321, 1324 Vol.2, 21-24 Oct. 2000

[2] Docherty, J.; Koelmans, A., "A flexible hardware implementation of SHA-1 and SHA-2 Hash Functions," Circuits and Systems (ISCAS), 2011 IEEE International Symposium on, vol., no., pp.1932, 1935, 15-18 May 2011

[3] Eiroa, S.; Baturone, I., "Hardware authentication based on PUFs and SHA-3 2nd round candidates," Microelectronics (ICM), 2010 International Conference on, vol., no., pp.319, 322, 19-22 Dec. 2010

[4] El-Hadedy, M.; Gligoroski, D.; Knapskog, S.J., "Low Area Implementation of the Hash Function "Blue Midnight Wish 256" for FPGA Platforms," Intelligent Networking and Collaborative Systems, 2009. INCOS '09. International Conference on, vol., no., pp.100, 104, 4-6 Nov. 2009

[5] Eun-Gu Jung; Daewan Han; Jeong-Gun Lee, "Low area and high speed SHA-1 implementation," SoC Design Conference (ISOCC), 2011 International , vol., no., pp.365,367, 17-18 Nov. 2011

[6] Guoping Wang, "An Efficient Implementation of SHA-1 Hash Function," Electro/information Technology, 2006 IEEE International Conference on , vol., no., pp.575,579, 7-10 May 2006

[7] Iumoka, A.A., "Efficient prediction of hash function in VLSI using neural networks," Electrical Performance of Electronic Packaging, 2000, IEEE Conference on. , vol. no., pp.87, 90, March 2000

[8] Junmou Zhang; Friedman, E.G., "Power Estimation for Cycle-Accurate Functional Descriptions of Hardware," Circuits and Systems, 2004. ISCAS '04. Proceedings of the 2004 International Symposium on, vol.2, no., pp.II, 529-32 Vol.2, 23-26 May 2004

[9] Kayu, R.; Maheshwari, V.; A.K., "ASIC-hardware-focused comparison for hash functions MD5, RIPEMD-160, and SHS,"Computing Communication and Networking Technologies (ICCCNT), 2010 International Conference on, vol., no., pp.1,6, 29-31 July 2010

[10] Kitsos, P.; Sklavos, N.; Koufopavlou, O., "An efficient implementation of the digital signature algorithm," Electronics, Circuits and Systems, 2002. 9th International Conference on, vol.3, no., pp.1151, 1154 vol.3, 2002

[11] Khalil-Hani, M.; Nambiar, V.P.; Marsono, M.N., "Hardware Acceleration of OpenSSL Cryptographic Functions for High-Performance Internet Security," Intelligent Systems, Modelling and Simulation (ISMS), 2010 International Conference on, vol., no., pp.374, 379, 27-29 Jan. 2011

[12] K.K. Murthy, N.S.; Rao, N.B., "An efficient power estimation model for high speed VLSI," Advances in Computing, Communications and Informatics (ICACCI), 2013 International Conference on, vol., no., pp.1358, 1362, 22-25 Aug. 2013

[13] K.M. Banerjee, and Amit Mehrotra, "A Novel Power Estimation Method for On-chip VLSI Distributed RLCG Global Interconnects Using Model Order Reduction Technique," IEEE Transactions On Computer-Aided Design of Integrated Circuits And Systems, Vol. 21, No. 8, August 2013

[14] Lao, R.; Maheshwari, V.; Agarwal, V.; Choudhary, A.; Singh, A.; Mai, A.K.; Bhattacharjee, A.K., "Accurate estimation of delay for step input," Computer and Communication Technology (ICCCT), 2010 International Conference on , vol., no., pp.673,677, 17-19 Sept. 2010

[15] L. V.; Mondal, S.; Maqbool, M.; Mal; Bhattacharjee, A.K., "A Novel Power Estimation Method for On-chip VLSI," Advances in Computer Engineering (ACE), 2010 International Conference on, vol., no., pp.105, 109, 20-21 June 2010

[16] Lin Zhou; Wenbao Han, "A Brief Implementation Analysis of SHA-1 on FPGAs, GPUs and Cell Processors," Engineering Computation, 2009. ICEC '09. International Conference on , vol., no., pp.101,104, 2-3 May 2009

[17] Li Hong-Qiang; Miao Chang-yun, "Hardware Implementation of Hash Function SHA-512," Innovative Computing, Information and Control, 2006. ICICIC '06. First International Conference on, vol.2, no., pp.38, 42, Aug. 30 2006-Sept. 1 2006

[18] Michail, H.; Goutis, C., "Holistic methodology for designing ultra high-speed SHA-1 hashing cryptographic module in hardware," Electron Devices and Solid-State Circuits, 2008. EDSSC 2008. IEEE International Conference on, vol., no., pp.1, 4, 8-10 Dec. 2008

[19] Michail, H.E.; Kakarountas, A.P.; Milidonis, A.; Goutis, C.E., "Efficient implementation of the keyed-hash message authentication code (HMAC) using the SHA-1 hash function," Electronics, Circuits and Systems, 2004. ICECS 2004. Proceedings of the 2004 11th IEEE International Conference on, vol., no., pp.567, 570, 13-15 Dec. 2004

[20] Michail, Harris; Kakarountas, A.P.; Koufopavlou, O.; Goutis, C.E., "A low-power and high-throughput implementation of the SHA-1 hash functions," Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on, vol., no., pp.4086, 4089 Vol. 4, 23-26 May 2005

[21] M. L. L. X. Wang, H. Yu, "Finding collisions in the full SHA-1," in Advances in Cryptology, Proceedings Crypto'05, IEEE Transactions on, pp. 17-36, May 2005

[22] N. Ahmad and A. S. Das, "Analysis and detection of errors in implementation of SHA-512 Algorithms on FPGAs," Computer Journal, vol. 50, pp. 728-738, May 2007

[23] N. Ferguson, B. Schneier, and T. Kohno, "Cryptography engineering: design principles and practical application", IEEE Transactions on, vol., no., pp.673, 677, 17-19 Sept. 2004

[24] N.K. Satoh and T. Inoue, "ASIC-hardware-focused comparison for hash functions MD5, RIPEMD-160, and SHS," in International Conference on Information Technology: Coding and Computing, ITCC, 2005, pp. 532-537 May 2004

[25] N. I. o. S. a. Technology, "Announcing the Standard for Secure Hash Standard," 50th Midwest Symposium on, vol., no., pp.21, 24, 5-8 Aug. 2007

[26] P. K. Yuen, Practical cryptology and web security. Harlow: Addison-Wesley, IEEE Transactions on, vol., no., pp.673, 677, 17-19 March 2005

[27] P. Chang, Y. Chu, and C. Jen, "Low-cost subsystem power estimation", IEEE Trans. Circuits Syst. II, vol. 47, pp. 137–145, Feb. 2007

[28] R. Puri and C.T. Chuang, "SO1 Digital Circuits: Design Issues," Thirteenth International Conference on VLSI Design, pp. 474-479, 2008

[29] Roy M. Pelella, "Hysteresis in Floating- Body PD/SOI CMOS Circuits," International Symposium on VLSI Technology Systems and Applications, pp. 278-281, March 2008

[30] Putri Ratna, A.A.; Dewi Purnamasari, P.; Shaugi, A.; Salman, M., "Analysis and comparison of MD5 and SHA-1 algorithm implementation in Simple-O authentication based security system," QiR (Quality in Research), 2012

[31] S. Steinke, M. Knauer, L. Wehmeyer, and P. Marwedel,"An Accurate and Fine Grain Instruction- Level Energy Model Supporting Software Optimizations," in Proc. Int. Wkshp Power & Timing Modeling, Optimization & Simulation (PATMOS), March 2008

[32] Sinha and A. P. Chandrakasan, "Joule Track - A Web Based Tool for Software Energy Profiling," in Proc. Design Automation Conf., pp. 220–225, April 2009

[33] S. B. Kamble and K. Ghose, "Analytical Models for Energy Dissipation in Low Power Caches," in Proc. Int. Symp. Low Power Electronics & Design, pp. 143–148, March 2009

[34] Selimis, G.; Sklavos, N.; Koufopavlou, O., "VLSI implementation of the keyed-hash message authentication code for the wireless application protocol," Electronics, Circuits and Systems, 2003. ICECS 2003. Proceedings of the 2003 10th IEEE International Conference on, vol.1, no., pp.24, 27 Vol.1, 14-17 Dec. 2003

[35] Sklavos, N.; Dimitroulakos, G.; Koufopavlou, O., "An ultra high speed architecture for VLSI implementation of hash functions," Electronics, Circuits and Systems, 2003. ICECS 2003. Proceedings of the 2003 10th IEEE International Conference on, vol.3, no., pp.990,993 Vol.3, 14-17 Dec. 2003

[36] Tang Qiong; Ye Jianwu, "Implementation and Optimization of the High Performance SHA-1 Model Based on FPGA," Computer Science & Service System (CSSS), 2012 International Conference on, vol., no., pp.687, 690, 11-13 Aug. 2012

[37] Thang Qiong; Ye Jianwu, "Implementation and Optimization of the High Performance SHA-2Model Based on FPGA," Computer Science & Service System (CSSS), 2012 International Conference on, vol., no., pp.687, 690, 11-13 Aug. 2011

[38] Thamrin, N.M.; Ahmad, I.; Khalil Hani, M., "A secure field programmable gate array based System-on-Chip for Telemedicine application," Information Society (i-Society), 2011 International Conference on, vol., no., pp.105, 109, 27-29 June 2011

[39] Yu Ming-yan; Zhou Tong; Wang Jin-xiang; Ye Yi-zheng, "An efficient ASIC implementation of SHA-1 engine for TPM," Circuits and Systems, 2004. Proceedings. The 2004 IEEE Asia-Pacific Conference on, vol.2, no., pp.873, 876, 6-9 Dec. 2004

[40] Zhou Hua; Liu Qiao, "Hardware design for SHA-1 based on FPGA," Electronics, Communications and Control (ICECC), 2011 International Conference on, vol., no., pp.2076, 2078, 9-11 Sept. 2011