International Journal of Applied Engineering Research
ISSN 0973-4562 Volume 10, Number 4 (2015) pp. 10921-10932
© Research India Publications
http://www.ripublication.com

Performance Study of Enhanced SHA-256 Algorithm

Gowthaman A[1], M Sumathi[2]
[1]IInd Year M.Tech VLSI Design, Sathyabama University, Chennai, India
[2]Professor, Electronics and Control, Sathyabama University, Chennai, India
Email id: gowthamaece@gmail.com, sumagopi206@gmail.com

Abstract

The prolong growth of wired and wireless communication has spark off the revolution for the generation of new cryptographic algorithms. Hash functions are similar and important cryptographic fields, which are very complicate for data integrity assurance and data security services. This paper looks into optimization technique in new VLSI architecture for the SHA-256 hash functions are presented. This paper combines different techniques namely rescheduling of Carry Select Adder and Non linear Pseudo code random generator for improving the functions in an inner loop of hashing algorithm. The new system can minimize the critical path by rescheduling of inner part of the SHA architecture. The proposed system can be suitable to achieve less area utilization, improved security and fastest data throughput and higher performing frequency. The proposed architecture is developed using Verilog HDL and synthesized in Quartus V9.0 and Modelsim V6.4 on a common platform for achieving good results.

Keywords: Encryption, Decryption, SHA algorithms, SHA-256, Pseudo code generator, LFSR, Carry Select Adder.

Introduction

Cryptography means Secure Writing. It discovers the context of message from sender and the recipient. Cryptography refers to the science of encompassing the method of converting a graspable message into one that ungraspable and reconverting the message back to its original form. Today our modern world utilizes various electronic operations: E-mail, Internet banking, document transfer, online shopping. Cryptography has inclined a vital role for safeguard of data conversion. Hash task mapping the message of erratic length to a string of fixed length, called the message Hash or digest. In 2002 the national institute of science and technology (NIST) published the SHA, which specifies three new secure hash algorithms SHA-224,256, SHA-384, and SHA-512.Hash task are mainly used to guard function of purity. They also provide the guard of authentication, when they are used in combination with digital signature and MAC algorithms. These algorithms are constant and one-way functions that input message and output message digest. It processes the data in different stages (i) message filler (or) padding, (ii) message extension, (iii) message squeezing [1] [3].

(i) Message filler (or) padding:
The binary message is to be processed, while processing message appended with a "1" and padded with zeros up to its length 448 modulo 512. Named as M (M=512 or 1024). The hash computation uses a data as variable, constant, algebraic operations. SHA -2 algorithms applied for SHA-224,256,512. It differs mostly in the size of operands, using 64-bit words instead of 32-bit. [3]

(ii) Message extension:
SHA-256 algorithm operate on 32-bit words each 512-bit $M^{(I)}$block from the pre-processing stage is 16 32-bit blocks denoted $M^{(I)}$ is $0\leq$ to ≤ 15. The message scheduler takes each M (I) and expands it into 64 32-bit blocks. [3]

(iii) Message Squeezing:
The W_t words from the message extension stage are passé to SHA squeezing function (or) SHA core. The core used to 8 32-bit working variables are A,B,C,D,E,F,G,H. 64 rounds of the Squeezing function are performed [1]. The final 256-bit output is formed by concatenating the final hash value is

$$H^{(N)} = H_0^{(N)} \& H_1^{(N)} \& H_2^{(N)} \& \ldots . H_7^{(N)} \tag{1}$$

Hash Functions (or) task promote at the origin of many famous cryptography approaches in Digital Signature Standard (DSS), Message Authentication Codes (MAC's), Hashing is the building blocks of Secrete-key encryption. Other utilization of hash task includes Random Noise Generation (RNG), fastest encryption and password depot and verification. These are widely spread into hyperlink and wireless protocols (WAP) which have mentioned security layers and cryptographic schemes.

Related Works

N. Sklavos et al., [9] described about three various methods that are implemented with new hashing technique, for improving the operating frequency, throughput and area. It is highlighted that proposed work is better than existing SHA-1 algorithm. Synthesize on place & routing are done in Xilinx FPGA, Vertex II device. The hardware utilization can be reduced in terms of slices, latches from 45% to 35%.

Robert P. McEvoy et al.,[3] concentrated on hardware optimization approach for the SHA-2 family. First time pipelined – unrolled approaches are inspected. Different angle of unrolling is designed, and the simulation work carried out for performance of throughput, area measurements in the RTL level design using VHDL. This method shows the improvement in critical path delay that is a factor of 1.8.

O. Koufopavlou et al.,[5]dealt with several mode operation has been proposed, for improving Security levels in secrete key, Digital signatures. This system achieved less area source and higher operating frequencies. The code was developed by VHDL using structural modeling, and synthesized using Vertex (v200pq240).The performance is expressed in terms of efficiency in %.

Medien Zeghid et al., [4] concentrated on rescheduling the operation, and hardware re usage. This paper allowing a reduction in critical path while tasking and area also reduced. This system can implement in different hardware for producing the realistic outputs. The throughput can be increased by level of 17% to 26%, system can be implemented by two devices Vertex v200pq240 and Virtex2 xcx2v2000.

Sylvain Ducloyer et al., [8] explained about the merging of three architectures for achieving the area reduction. By combining, this architecture produced heat and throughput loss and concluded finally it does not provide flexibility and area resources. Dynamic reconfiguration applied for extra memory for improvement of flexibility. This system implemented on Altera Stratix II device. It can produce throughput level up to 567Mbps.

Marcio Juliato et al.,[10] decided to implement the architecture in both software and hardware. Partitions done in the real time hardware parts and reduce the coding size and the total time of execution and reused the same. In this system does not show the improvement in area reduction. The hardware performed 11 times faster after reducing the program size of 16%.

Existing System

The optimization technique of Secure hash algorithm is designed by function of Choice, Majority and Summing operations. The input of the hash values are processed by the equations (3) (4), the output of the first round hashed value is 8 numbers of 32-bit blocks. The bits are returned to the next set of iteration, for processing the data with new hash values. The carry save adder (CSA) is added to the 32-bit blocks for 64 iterations. Adder saves the values in registers for further addition process. Finally the hashed value is 8 numbers of 32-bit blocks. By merging this data, 256-bit hashed value is produced. CSA separates the sum and carry root and the carry propagation technique is applied for minimizing the delay. Another method can also be applied for reducing the delay by implementing Unrolling and Pipelining.[2] The operations are explained as follows;

(i) Unrolling

This unrolling technique performs multiple rounds of squeezing function in combinational logic, to compute the hash function it will take less number of clock cycles. Ex: core was unrolled once; the hash should be calculated in half of the clock cycles. It decreases the clock frequency.

(ii)Pipelining

The pipelining is used for registers to break the long critical path within the SHA core. External control circuitry is required to enable the registers correctly. Pipelined designs achieve very short critical paths, allowing message hashes to be calculated at high frequencies and high data throughputs.

The SHA-256 algorithm compute 64 iterations over the block of 512-bit messages and hash values of 256-bits, to interpret eight numbers of 32-bit words (A, B, C, D, E, F, G, H), [1]. In SHA-256 algorithm, there are several ways for designing the inner part of the loop, because of the number of additions needed. It is possible to rearrange the inner part for achieving high performance in the data dependencies [4].

The operation in the inner loop of the algorithm was performed by pre-computation, and subtractions of the functions. The variables of 8 numbers of 32-bit blocks are performed by this method. The pre-computation save the sum value during the run time iterations, for previous iteration of δ_k value is, refer equations in [1], the following figure(1) is designed from the equations(2)(3)(4),refer [2].

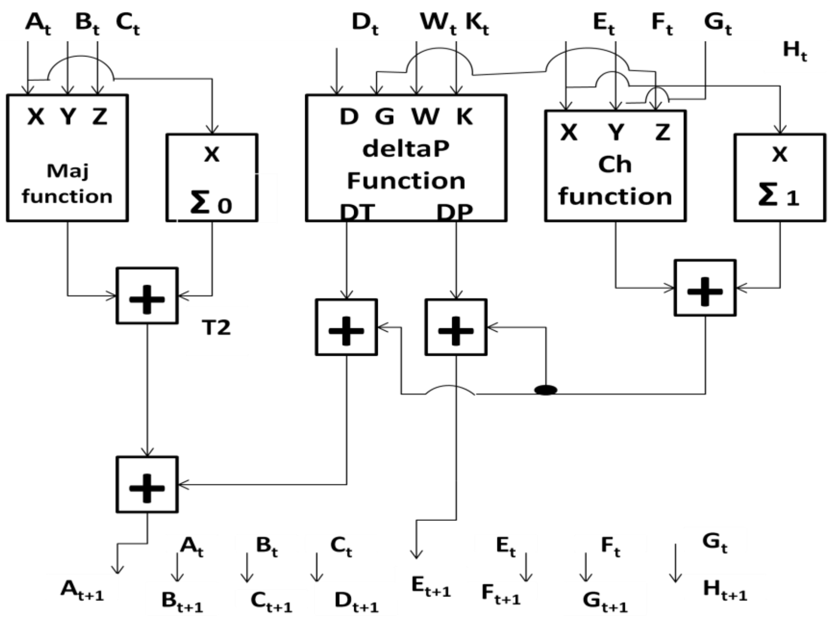

Figure 1: Existing SHA-256 Inner loop Structure

The final equation can be calculated by reordering the equations in [2], and by using the difference between δ_k and τ_k. The main thing is to calculate additions in a pre computation; this is used to reduce the critical delay path. The δ'_k is found using.[4]

$$\delta'_k = \delta_k + D_k \tag{2}$$

$$E_{k+1} = \Sigma_1 (E_k) + CH(E_k ,F_k ,G_k) + \delta'_k \tag{3}$$

$$A_{k+1} = \Sigma_0(A_k) + MAJ(A_k, B_k, C_k) + \Sigma_1(E_k) + CH(E_k, F_k G_k) + \delta_k \tag{4}$$

This final existing architecture can be designed by equation (2) (3), this architecture requires an additional clock cycle to initialize the system for decreasing the data dependency. This system needs more hardware functions to produce high throughput. This method is used to store the computed values from A_{k+1} and E_{k+1} in the registers like intermediate state buffer. The above existing architecture has large area utilization by using more number of adders to perform the linear addition. The delay is a common factor which reduces system performance, and by using the linear pseudo code generator, the security level will also be less. It is in need to revise the architecture for the improvement of performance factors such as area, delay, throughput and efficiency.

Proposed Work

The exertion of hash functions hurting for great amounts of structural and waste resources that may result in less performance and efficiency. This has inspired the research on many approaches for spurring the computation in terms of throughput/area ratio.

The Pseudo random generator describes a function of the SHA-256 algorithm, shown in fig (2), it is same as that of a mod 32-bit linear feedback shift register that generates a pseudo random sequence. The D flip flop, seed value and control register are used to decide the value of code generator. D flip flop is also used to shift the data values to the next stage. The seed value can be the origin of pseudo code sequence generator; initial seed value is must for producing the pseudo code. The general code generator (LFSR) architecture has only one EX-OR gate to perform more number of addition operations. This modified LFSR is a single EX-OR with summation in the feedback of MSB. The summation is used to perform the tap for (M, t_1, t_2, $t_{3\ldots\ldots}$ t_m), the tap decides the shifting bits for further iterations, it is a Non-Linear Pseudo code sequence generator shown in Figure (2),

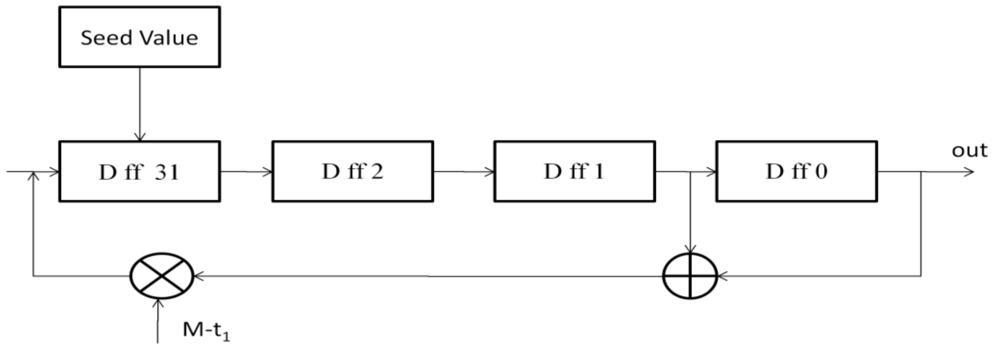

Figure 2: Proposed Pseudo Random Sequence Generator

Where M is the final length of the pseudo code in bits, and t denotes the tap in bits, both are ex-or’ed into the feedback summation. For example pseudo code generates 32-bit, {32, 20, 15, 6} with tap values as bit 32, bit 20, bit 15, and bit 6(note that the MSB bit always EX-OR’ed into feedback summation). The pseudo random generator {32, 20, 25, 6} is translated to {32, 32-20, 32-15, 32-6} it is reordered to {32, 26, 17, 12}; this code generator is same as simple pseudo code generator with tap. The pseudo code sequence generator can produce the values up to 2^n-1. Zero is the missing value for initializing condition. For non-zero condition, the origin seed value should not be a zero value.

The pseudo random generator is performed in two step process: first the seed value is read and feed through the shift register. Secondly the shift register will read the seed data for producing the pseudo value. Performance depends on seed value; it is a type of non-linear code generator for adding more number of input bits. By using this pseudo code sequence generator the security of the Secure Hash Algorithm is improved.

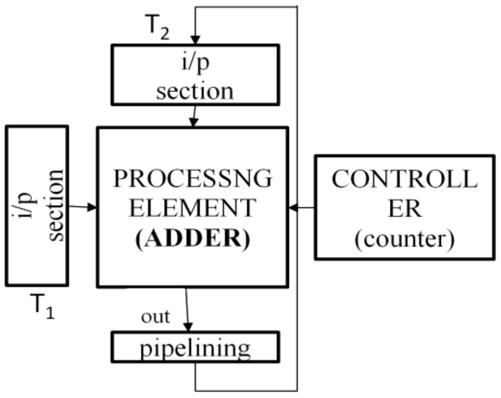

Figure 3: Feedback Carry Select Adder

Carry select adder is one of the fastest adders which perform the fast arithmetic functions in many data-processing processors. The scope of the carry select adder is to reduce the power and to improve efficiency of the module. In general, the speed for addition computation is limited by the time that is required to propagate a carry through the adder. The carry select adder is used in many arithmetic systems to eliminate the problem of carry propagation delay by generating multiple carries independently and select a single carry to produce sum. Nonetheless the CSLA is not area efficient because it uses numerous pairs of Ripple Carry Adders (RCA) to crop sum and carry Cin =0 and Cin =1, at last the sum and carry are chosen by multiplexers. Additionally Binary to Excess Converter is used for reducing the area (BEC-1). For replacement of Cin =1, BEC logic uses less number of logic gates than the full adder of n-bit.

The essential idea of this work is to use the single stage of CSLA, for improving the area and reducing the delay, the first stage of Ripple Carry Adder and Binary to Excess Converter logic with pipelined feedback logic is taken. The general adder diagram is shown in the figure (3).

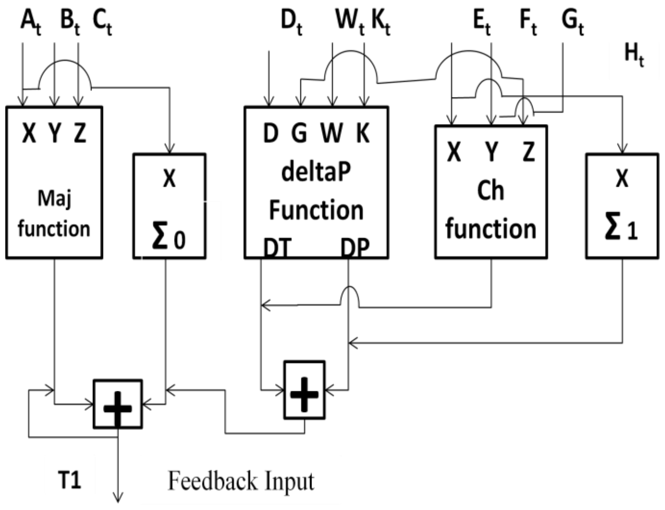

Figure 4: Proposed SHA-256 Architecture

In figure (3) the processing element performs like a Carry Select Adder for 32-bit. For first iteration T_1 and T_2 are the message inputs of 32-bits. From second iteration, T_2 becomes feedback of input message bits from pipelined register. T_1 always perform message inputs for all the iterations. The controller role is to count the 'n' number of rounds (or) iterations. The count is decided using 2^n-1. Here the controller is used as up counter to count 0-63iterations in the process of 8 numbers of 32-bit blocks.

In the proposed system pseudo random generator and Carry select Adder are integrated in existing SHA algorithm to achieve an improved performance parameters and security of the data bits, in Figure(5). By implementing the Pseudo code generator, it provides different data bits to process the input variables, this code can generate n number of values for entire addition process and it is different from normal pseudo code generator. It will generate the code for 0-63 iterations with contrasting values of 32-bits. The code is generated by the initialized seed value, it is a non-zero condition. The pseudo generator is frequent for all the 32-bit blocks.

Initializing seed value is the secrete key for hashing function, only known by exporter and acceptor. Seed value starts from value 1 to n numbers. The generated code is called hash value. This value is fed to the input of the adder block. So its addition output is complicated, when compared with the existing output. Retrieving the original value is very problematic.

The Carry Select Adder block is achieved with less delay and power because of rescheduling the blocks. Only a single stage is going to be use in the modified adder circuit for achieving the goal. The first stage of adder is having Ripple Carry Adder block and Binary to Excess Converter block. It can predict the carry output before coming out of its original carry value and can produce the output in two ways of Cin 0 and Cin 1. The multiplexer is used to select values depends upon the original carry value.

In this proposed work the selected multiplexed value can be feedback to the same input block as performed by the first stage. So the entire addition process will be calculated by the single stage of carry select adder, the final stage of 8 rounds, 32-bits are saved and formed by 256-bit hash output. By this way, the delay and area of the entire architecture is reduced and therefore an improvement can be achieved in high throughput and efficiency. This adder is efficient for all the SHA-128, 224,256 and 512bit functions.

Result and Discussion

In order to appraise the proposed architecture, it is developed using Verilog - HDL and synthesized in Quartus V9.0 and Modelsim V4.1 on a common platform. The aim is to compare the hardware resources like throughput/area and power. The simulated results for SHA-256 architecture are listed below for comparison. The comparison table and graphs are mentioned. The frequency and clock cycles are compared with the existing system. Throughput and efficiency are calculated by the formula.

$$Throughput = \frac{Block\ Size\ X\ Max.Frequency}{Clock\ Cycles} \tag{5}$$

$$Efficiency = \frac{Throughput}{No.of\ Slices} \tag{6}$$

Throughput for proposed system is 1456 Mbps, it is increased one when compared with existing system of 824 Mbps. In the same way, efficiency is also increased to 2.83%, but the existing system value is 0.83%. Maximum clock frequency for proposed system is 183MHz, it is high when compared with Existing system, the existing Maximum Frequency is only 103 Mhz. In new system area will be reduced in 50% of existing system and overall thermal power dissipation is decreased in terms of 5.36% .The proposed systems encrypted output has been shown below Fig.5, and it is more secured when compared with existing system. It's more problematic for retrieving the original hash value. By retrieving the seed value and process data, the plain text can be produced. The rehashed output is also mentioned in Fig.6

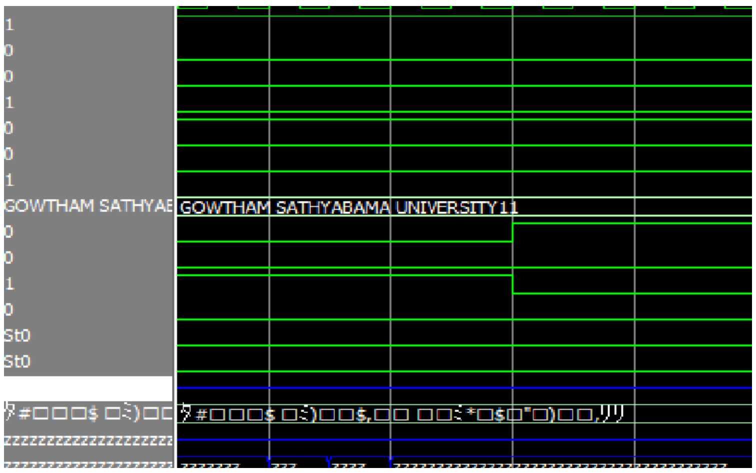

Figure 5: Hashed output of SHA-256 Architecture

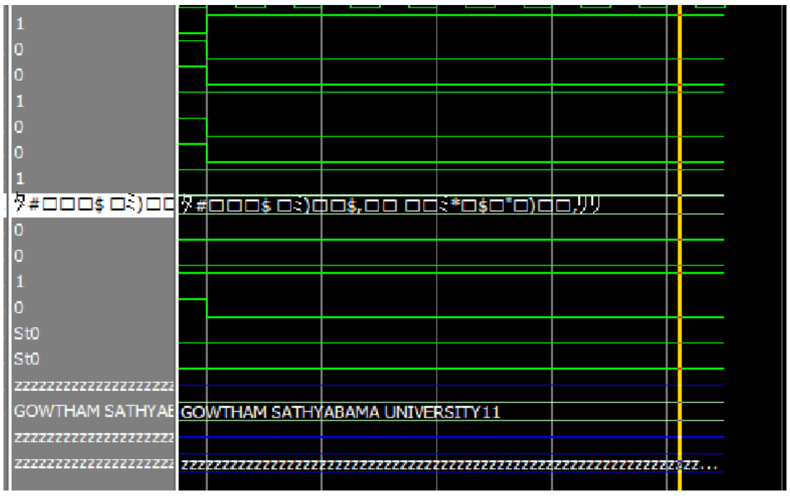

Figure 6: Rehash Output of SHA-256 Architecture

Table 1: Analysis of performanc Factors

Performance Factors	Existing System	Proposed System
Throughput	824Mbps	1456Mbps
Efficiency	0.83%	2.83%
Maximum Frequency	103Mhz	183Mhz

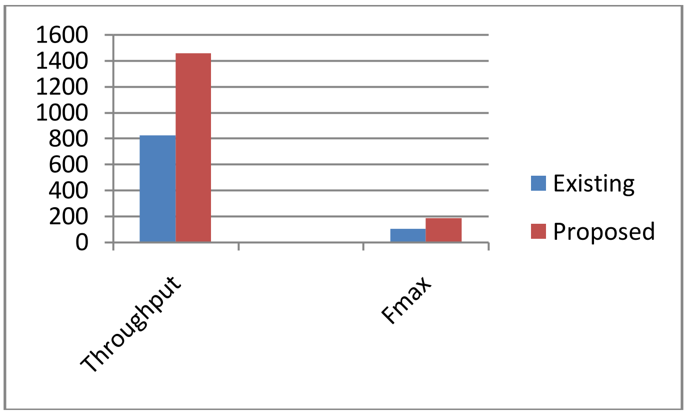

Figure 6: Comparison of Performance Factors

Table 2: Comparison Table of Performance Metrics

Metrics	Existing System	Proposed System
Area (No of Registers)	103	51
Total Thermal Power Dissipation(mw)	430.23	361.75
Total Delay(ns)	26.33ns	17.45ns

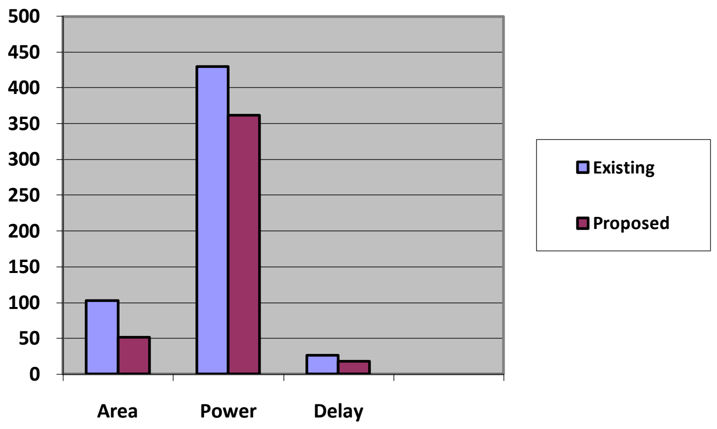

Figure 7: Comparison of Performance Metrics

Conclusion

In this work, the architecture of existing and proposed SHA-256 algorithm is studied and its software implementation done using Quartus-V9.0 software. The proposed / revised architecture exhibits high performance, throughput and efficiency. The hardware design is based on rescheduling of computation in the inner loop of SHA-256 algorithm. The proposed architecture used Non-Linear Pseudo random generator for improving the security of SHA-256 function. The rearrangement of pseudo code sequence generator produces the 8 number of 32-bit problematic hash output. The Carry Select Adder is used for reducing the delay in all the iterations. By resizing the Carry Select Adder, area will be reduced and moreover power dissipation is also decreased. Although the performance factors are focused on the SHA-256 algorithm, the work is prolong with other SHA-2 family by cause of its frequent framework. The work can be extended by modifying the compression function / Maj, Ch operation in the inner loop of the SHA-256 to increase the performance of the system.

References

[1]. W.Stallings , “Cryptography and Network Security”, Prentice Hall, Nj , 2003.

[2]. Algredo-Badillo , C.Feregrino-Uribe , R. Cumplido, M. Morales-Sandoval, “FPGA-based implementation alternatives for the inner loop of the Secure Hash Algorithm SHA-256”, Journal of Microprocessors and Microsystems, vol. 37(2013),pp. 750-757.

[3]. Robert P. McEvoy, Francis M. Crowe, Colin C. Murphy and William P.Marnane, University college cork, Ireland, “ Optimisation of the SHA-2

Family of Hash Functions on FPGAs", Journal of Emerging VLSI Technologies and Architectures (ISVLSI'06).

[4]. M. Zehid, B. Bouallegue, M. Machhout, A. Baganne, R. Tourki, "Architectural design features of a programmable high throughput reconfigurable SHA-2 processor", Journal of information Assurance and security,vol.2(2008),pp. 147-158.

[5]. O. Koufopavlou, "Implemenation of the SHA-2 hash family standard using FPGA's", The journal of Supercomputing Vol. 31(3) (2005) pp227-248.

[6]. R.P Mc Evoy, F.M. Crowe, C.C. Murphy, W.P. Marnane, "Optimization of the SHA-2 Family of hash functions on FPGA's", in: IEEE Computer society Annual Symposium on VLSI: Emerging VLSI Technologies and Architectures(ISVLSI'06), 2006.09.006.

[7]. Helion Technology Limited, Tiny hash core family for Xilinx FPGA,datasheet. Revision 2.0.2010.

[8]. S. Ducloyer, R. Vaslin, G. Gogniat, E.Wanderly, "Hardware implementation of a multi-mode hash architecture for MD5, SHA-1 and SHA-2", in: Workshop on Design and Architectures for Signal and Image Processing, 2007.

[9]. N. Sklavos, O. Koufopavlou, "On the hardware implementations of the SHA-2(256, 384, 512) hash functions", in: Proceedings of IEEE International Symposium on Circuits & Systems (IEEE ISCAS'03), 2003, pp. 153-156.

[10]. M.Juliato, C. Gebotys, "Tailoring a reconfigurable platform to SHA-256 and HMAC through custom instructions and peripherals", in: International Conference on Reconfigurable Computing and FPGA's, IEEE Computer Society, Los Alamitos, CA, USA, 2009, pp. 195-200

[11]. B. Ramkumar and Harish M Kittur, "Low-Power and Area Efficient Carry Select Adder", The Journal of IEEE Transactions on Very Large Scale Integration(VLSI) Systems, Vol.20, No.2, Feb 2012

[12]. Y. Kim and L.S. Kim, "64-bit carry select adder using Single ripple carry adder ," in: Electron. Lett., vol.37, no. 10, pp. 614-615, May 2001.

[13]. Y. He, C.H. Chang, and J.Gu, "An area efficient 64-bit squre root carry select adder for low power applications," in Proc. IEEE Int,. Symp. Circuits Syst., 2005, vol.4, pp. 4082-4085

[14]. Cypress semiconductors 24-bit Pseudo code random generator, document no 001-13577 Rev.*J, Revised november 25, 2014

[15]. K. Chandra Sekhar, K. Saritha Raj, "An Efficient Pseduo Random Number Generator for Cryptographic Applications", in: International journal of Engineering and Advanced Technology(IJEAT), ISSN: 2249-8958, Volume-4 Issue 1, Oct 2014.

[16]. Padma Devi, Ashima Gridher, Balwinder Singh, "Improved Carry Select Adder with Reduced Area and Low Power Consumption", in: International Journal of Computer Applications(0975-8887), volume 3-No.4, June 2010.

[17]. Ms. Archana Kakde, Ms. Manisha Waje "Low power & Area Efficient 16 bit Carry Select Adder Based on Adiabatic Logic", in: International journal of Engineering and Advanced Technology(IJEAT), ISSN: 2278-0181, Volume-2, july 2014.

[18]. Federal Information Processing Standards. Secure hash Standard, FIPS PUB 180-2, 2002.

[19]. Dai Zibin, Zou Ning. "FPGA implementation of SHA-1 Algorithm", in: Inst. Of Electron. Technol., Inf, Eng. Univ., Zhengzhou, China, Oct 2003.

[20]. A. Menezes, P.van Oorchol, and S. Vanstone, "Handbook of applied Cryptography", in: CRC Press, Inc, October 1997.

[21]. "The Keyed hash message authentication code (HMAC)," NIST, FIPS PUB 198, March 2012.

[22]. J. Goodman and A.P. Chandrasekaran. "An energy efficient reconfigurable public-key cryptography processor". IEEE journal of Solid-state circuits 36(11): 1808-1820, 2001.

[23]. Citavicius, A. Jonavicius, "Unpredictible Cryptographic Pseudo-Random Number Generator based on Non-Linear Dynamic Chaotic System",The Journal of Electronis and Electrical Engineering, Elektronika IR Elektrotechnika, ISSN 1392-1215, 2007. No. 7(79)

[24]. A. Regenscheid, R. Perlner, S. Jen Chang, J. Kelsey, M. Nandi, S. Paul, "status report on the first round of the SHA-3 Cryptographic hash algorithm competition", Tech. rep., NIST, September 2009.

[25].