Area Efficient SHA-1 Algorithm

[I]Md. Zakir Hussain, [II]Kazi Nikhat Parvin

[I]Asst. Professor, ECED, Muffakham Jah college of Engineering and Technology
Banjara Hills, Road No.3 Hyderabad, Telangana, India
[II]Asst. Professor, ECED, Bhoj Reddy Engineering College for Women
Vinay Nagar, Santoshnagar Cross roads, Saidabad, Hyderabad, Telangana, India

Abstract

In today's world where IoT(Internet of Things) is one of the most impact creating technologies, cloud computing is making its footsteps, security is of utmost importance and one of the underlying factors for successful working of internet. Loopholes in security could lead to the collapse of the entire system or the bad faith may lead to the extinction of the technology. Cryptography ensures the security by providing features like encryption, decryption, data integrity, confidentiality, authenticity etc. This paper proposes a method to improve performance of the pre-existing secure hash algorithm making use of modulo addition replacing the addition ('+') operator predefined by Verilog HDL. The algorithm is widely used in cryptographic applications, security protocols. SHA-1 is the most used algorithm for internet security.

Keywords

Internet of Things; Data security; Field programmable gate arrays; Data integration

I. Introduction

Information security is one of the major issues in this highly sophisticated world. It is necessary to determine the authenticity of the received data. Authenticity of the confidential data like bank account number, passwords, credit/debit card numbers, and other personal data is mandatory in order to avoid fraud and false representations. Authentication service should assure the recipient that the received information is from the trusted source.

Cryptography is one of the most widely used field in wireless communications and personal communication systems where Information security is the major area of research. Various cryptographic algorithms look after confidentiality and data origin authentication for the security of information. MAC (Message Authentication Code) is the vital element in the authentication schemes. Hash function is used to generate a unique MAC for the given information/data.

For a secure cryptographic electronic device, a trusted and efficient algorithm should be selected to provide unbreakable security for the given information.MD5 algorithm generates a 128 bit unique hash value, its security was compromised due to collision attacks resulting in the threat to the information. In 2004, it was found that two different messages has nearly same hash value i.e. 142 out of 160 bits are same. SHA-1(Secure Hash Algorithm) is a one way hash function which takes the input data and creates an irreversible unique digest of 160 bits long for that given data. In this algorithm, it is impracticable to produce same digest for two different information and also very difficult to find the message that corresponds to a particular digest.

II. Features of Hash Function

1. One way hash: Meaning for a given word, or plain text there exists one and only one unique hash value ,even one character change in input will result in change in entire output
2. Weak collision resistance: For a given message, say x and its hash, f(x), it is not possible to find another message x' which would result in the same hash value f(x).
3. Strong collision resistant: It is not possible to find two non-identical messages that result in the same hash value.

III. Our Proposed System

The earlier methods that were used to implement the SHA-1 would make use of the addition('+') operator provided by Verilog HDL after synthesis, the results that were found when compared with modulo 2^32 addition algorithm the results are far efficient than the previously proposed approached work. Operation performed did not yield the most efficient and best results. We replaced the normal addition with modular addition, the operation performed being mod (2^32) addition. Upon synthesis the process turned out to be more efficient and consumed less number of slices and also improved result. Earlier results would require 732 slices and now it requires 625 Slices. This shows a great improvement in terms of power consumption and delay. The reduction in number of slices used leads to usage of fewer flip flops and LUTs. The simulation and coding was done with FPGA (Spartan-3E) being the target hardware.

1. Algorithm

1. Add the two numbers using '+' operator
2. Let the result be stored in a temporary variable
3. Subtract 2^32 from the resulting sum
4. If the resultant is greater than the temporary variable, then it is the output else the previous sum is the output

IV. Secure Hash Algorithm-1

To obtain a hash, input should be broken down into chunks of 512 bits size, and if the length is greater than that, it is broken into several blocks of each 512.If the message is not a multiple of 512, then it is made into a multiple of 512 by padding. Padding is done by appending the message with 1 followed by zeros and the binary equivalent of the message is given in last 48 bits. This chunk is processed and message abstract is formed using the functions shown in table 1 and constants shown in table 2 which are fed in 160 bit size buffer of 32 bits register each.

A=0x67452301
B=0xefcdab89
C=0x98badcfe
D=0x10325476
E=0xc32d2e1f0

SHA-1 uses non linear function which uses (Wt) constants and Kt (constants), which is different for different rounds.

Table 1 : Functions

Step number	Function
0≤t≤19	F1=f(t,B,C,D)=(B AND C) XOR (NOT B AND D)
20≤t≤39	F2=f(t,B,C,D)=B XOR C XOR D
40≤t≤59	F3=f(t,B,C,D)= (B AND D) OR (B AND D) OR (C AND D)
60≤t≤79	F4=f(t,B,C,D)= B XOR C XOR D

Table 2 : The Various Constants

Step number	Hexadecimal value
0≤t≤19	Kt=0x5a827999
20≤t≤39	Kt=0x6ed9eba1
40≤t≤59	Kt=0x8f1bbcdc
60≤t≤79	Kt=0xca62c1d6

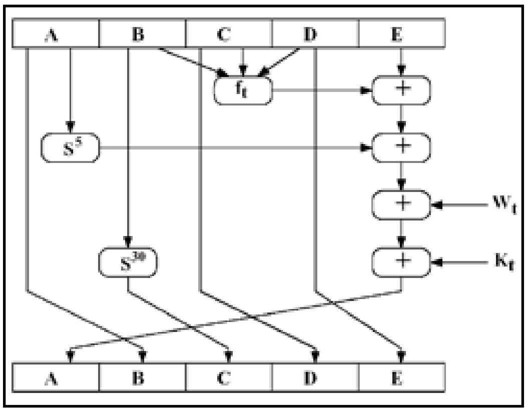

Fig. 1 : SHA-1 step function

The standard values of the registers A, B, C, D, and E are initially given by FIPS. From the above figure, we can understand that the predefined values in the registers A, B, C, D, and E are changed for every round and is used as input for the next round. This process goes on until the final 160 bit hash value is generated.

A constant Kt is used in the generation of hash, which is different for different rounds. Since there are 80 rounds, the Kt value changes for every 20 rounds. As mentioned in the above table, there are 4 Kt values used for rounds 0-19, 20-39, 40-59 and 60 to 79 respectively.

The message should be in congruency with 448 mod 512

V. Synthesis Results

For experimental purpose, we used Verilog code to complete the coding process and the stand alone PC used was a 2.4Ghz i5 Processor. The below table gives a brief comparison of the results obtained previously i.e. without modulo addition and with modulo addition and shows a clear difference in the number of slices reduced and the percentage reduced from 15 to 13 also the number of LUT slices has reduced.

Table 3 : Synthesis Results Comaparison

OPERATION	No. of slices used	No. of flip flops used	Percentage utilization (slices)	Percentage utilization(slice flip flops)
Using('+') operator given by Verilog	732	669	15	7
Using modulo addition	625	662	13	7

Hash function can compress a message of arbitrary length into an output of fixed length, which in case of SHA-1, is 160 bits. After performing the necessary operations on each 512 bit block (further divided into 32 bit words) for 80 rounds, we get the final hash value of 160 bit length which is unique and is impracticable to break.

From the above comparison, we found that the use of modulo addition instead of "+" operator resulted in the improved results. In reference to the previous work, the number of slices used were 732 out of 4656 constituting to 15% and number of slice flip flops used was 669 out of 9312 making it 7%, but usage of modulo operator consumed 625 out of 4656 slices and 662 out of 9312 slice flip flops hence the reduction in number of slices used is 13.3%.

VI. Conclusion

In this paper, SHA-1 algorithm which is a famous message compress standard in cryptography has been analysed. A successful effort has been made to improve the synthesis results by using modulo addition thereby reducing the number of flip flops, LUT's, logic elements as mentioned in the above table. The target hardware was FPGA (Spartan-3E) for simulation and coding of the algorithm.

References

[1] *Alfred M., Oorschot P., and Vanstone S., Handbook of Applied Cryptography, CRC press, 1997.*

[2] *Bruce S., Applied Cryptography: Protocols, Algorithms and Source Code in C, John Wiley and Sons, Canada, 1996.*

[3] *Stallings W., Cryptography and Network Security Principles and Practices, Prentice Hall Press Upper Saddle River, 2010.*

[4] *Goldwasser S., Micali S., and Rivest R., "A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks," Journal on Computing, vol. 17, no. 2, pp. 281-308, 1988.*

[5] *Ilya M., "Hash Functions: Theory, Attacks, and Applications," in Proceedings of Microsoft Research, Silicon Valley Campus, pp. 1-22, 2005.*

[6] *National Institute of Science and Technology, "Secure Hash Standard," Federal Information Processing Standard 180-1, available at: http:/ http://www.itl.nist.gov/fipspubs/fip180-1.htm, last visited 1995.*

[7] *National Institute of Science and Technology, "Secure Hash Standard," Federal Information Processing Standard 180-2,available at: http://www.itl.nist.gov/fipspubs/fip180-1.htm, last visited 2002.*

[8] *National Institute of Standards and Technology, "Secure Hash Standard," "FIPSPUB180www.itl.nist.gov/fipspub/fips180-1.html, last visited 2003.*

[9] *National Institute of Science and Technology, "Implementing Cryptography," NIST SP 800, available at: http://csrc.nist.gov/publications/ nistpubs/800-21-1/sp800-21-1_Dec2005.pdf, last visited 2005.*

[10] *"New European Schemes for Signatures, Integrity and Encryption Project," available at: http://www.cryptonessie.org, last visited 2000.*

[11] *Phan R. and Wagner D., "Security Consideration for Incremental Hash Function Based on Pair Blocking Chaining," in Proceedings of Computers and Security, USA, pp. 131- 136, 2006.*

[12] *Rivest R., Shamir A., and Adleman L., "A Method for Obtaining Digital Signature and Public-Key Cryptosystems," Communication of The ACM, vol. 21, no. 2, pp. 120-126, 1978.*

[13] *Sklavos N., Alexopoulos E., and Koufopavlou O., "Networking Data Integrity: High Speed Architectures and Hardware Implementations," The International Arab Journal of Information Technology, vol. 1, no. 0, pp. 54-59, 2003.*

[14] *United States Department of Commerce, National Bureau of Standards, Data Encryption Standard, Federal Information Processing Standards Publication, 1977.*

[15] *William S., Cryptography and Network Security, Principles and Practice, Prentice Hall of India, 2005.*