Study of Data Security Algorithms using Verilog HDL

M. Sumathi[*], D. Nirmala[**], R. Immanuel Rajkumar [***]
[*]Professor, [**] Assistant Professor, [***] Assistant Professor, Sathyabama University,
Chennai-600119, Tamilnadu, India.

Article Info

Article history:

Received Apr 17, 2015
Revised Jul 16, 2015
Accepted Jul 29, 2015

Keyword:

Cryptograpy algorithms
Data security
Decryption
Encryption
Verilog HDL

ABSTRACT

This paper describes an overview of data security algorithms and its performance evaluation. AES, RC5 and SHA algorithms have been taken under this study. Three different types of security algorithms used to analyze the performance study. The designs were implemented in Quartus-II software. The results obtained for encryption and decryption procedures show a significant improvement on the performance of the three algorithms. In this paper, 128-bit AES, 64-bit of RC5 and 512-bit of SHA256 encryption and Decryption has been made using Verilog Hardware Description Language and simulated using ModelSim.

Corresponding Author:

M. Sumathi
Departement of Electronics and Control Engineering,
Sathyabama University,
Rajiv Gandhi Road, Jeppiaar Nagar, Chennai-119.
Email: sumagopi206@gmail.com

1. INTRODUCTION

The risks found in digital communication systems are non-authorized access, denial of service, data corruption, leakage, monitoring attacks, authentication and trashing, etc. These problems occur while executing different electronic operations such as video/text transfer, communications, online trade, etc. Many procedures/algorithm steps have come out to reduce the risks. We can employ firewalls, detection systems, security via cryptographic algorithms and others [1]. The increasing need for secured data communication has increased to development of several cryptographic algorithms such as DES, 3DES, AES, RSA, SHA, RC5, etc., The basic idea of crypto algorithm is to secure the data while transmitting in the network. The data is to be transmitted from sender to receiver in the network must be encrypted using the encryption algorithm. By using decryption technique, the receiver can view the original data. AES algorithm is proved in many papers [2]-[5] that have to be efficient in both hardware and software implementations. It can utilize data of different length say 128,192 or 256 bits for processing with a private key of same length. Sub-byte, Shift row, Mixed Column and Add Round key are the four basic operations carried out in 10, 12 or 14 rounds/iterations. The Mixed Column operation will not be carried out in the last round [6]-[15].

Hash functions are cryptographic elements used to provide solutions of data integrity and authentication issues. It performs operations in iterative fashion and maps the binary digits of arbitrary length with some fixed length. The existing algorithms for performing hash functions are MD5, SHA-1, SHA-2, Whirlpool, Haval and Ripe MD-160 and so on. Currently, SHA-256 algorithm become popular in performance because it reducing the critical path by reordering the operations in each iteration. It is implemented and validated in the FPGA Virtex-2Xc2VP-7 [16]. The performance of the hardware implementation of SHA-256 exhibit high throughput and efficiency. Though a many hardware and software implementations of hash function algorithms are reported [17]-[22] and still it has been active research topic in both academia and industry in recent years. The hardware architectures reported aim to achieve better performance by customizing hardware elements that compute specific functions and also by using different techniques such as carry save adders, embedded memories, pipelining, unrolling techniques, etc. RC5 is a fast symmetric cryptography algorithm that uses the same key for encryption and decryption. The plain text and cipher text are fixed - length bit sequences, so it is named as block cipher. It uses two's complement addition and subtraction operation, bitwise exclusive-or operation and left-right rotation as primitives in the algorithm. It works well for real-time image encryption by partitioning the image into macro blocks and the resultant histogram of the cipher image is fairly uniform. The key feature of RC5 is the use of data-dependent rotations and to study the evaluation of cryptographic primitives [23]-[31]. It is fast and adaptable to processors of different word-lengths and provides high security.

2. RELATED WORKS

Advanced Encryption Standard (AES) algorithm has been analysed by many of the researchers across the world. AES algorithm is better than its preceding algorithms because it can be applied for more bits of data. 128-bit data can be given as input to the AES algorithm along with either 128-bit or 192-bit or 256-bit key. Manjesh et al., have used pipelining registers after each round in 128-bit AES algorithm in order to improve the speed of the algorithm. They have done the hardware implementation of AES algorithm using pipelining registers. Finally the work is concluded that use of pipelining register may increase the area of hardware architecture and therefore it need to be used carefully. Vanitha et al., have developed an efficient VLSI architecture for AES algorithm. This architecture increases the throughput and security of AES algorithm and has used a combinational circuit in s-box and inverse s-box instead of look up table for reducing the area. Ritu Pahal et al., have used symmetric cryptographic technique AES algorithm having 200-bit block and the key size is 200-bit. The conventional 128-bit AES algorithm is implemented for 200 bit AES algorithm using five rows and five columns of matrix. After the implementation, the proposed work is compared with 128-bit, 192-bit and 256-bit AES techniques on two points. These points are execution time and throughput of encryption and decryption process. Abdul Karim, Amer Shtewi et.al, has made modifications to the Advanced Encryption Standard to reflect a high level security and better image encryption standards. The modification is done by adjusting the shift rows phase. The proposed modification to image cryptosystem is highly secure from the cryptographic view point. The results also prove that when compared to original AES algorithm; the modified algorithm gives better encryption results in terms of security against statistical attacks.

Francis M. Crowe et al., dealt with the hardware utilization approach for the SHA-256 function. Unrolling and pipelining are approached in first time of this algorithm. The various sizes of unrolling method are designed and the output result is taken from the RTL design in the Verilog HDL. This system can show the improvement in the critical path delay. O. Koufopavlou, et al, concentrated on various hashing technique that are implemented in different methods of operation the hardware used for implementation utilized reduced area in terms of slices. Romain Vaslin et al., described on various architectures to be combined in a single module to reduce a area and delay. By improving of this architecture module heat and loss in data will be occur. This module does not suitable for memory flexibility, and area resources. This system is implemented on Altera Stratix device. It can give throughput level up to 500mbps. Prof. Loh, et al., adopted adders function in the cryptography technique. The carry save adder is the fastest adder in the digital adders. The drawback of this adder is that taking more time to produce the sum and carry output and know the result of addition at once in a stage, do not know about the addition of the value is larger or smaller for a given data. Hossam El-din H. Ahmed et al., described about RC5 in highly secure based on real-time image encryption by portioning image into 64 bit macro blocks and resultant histogram of the encrypted image is fairly uniform. The drawback of this system is reconfigurability that not possible, it leads security because of single round of encryption. Ronald L. Rivest et al., dealt with the highly reconfigurable RC5 with variable bit size and key sizes 128, 224,256, are proposed. The key scheduling and pseudo random generator is used for improving the security of the algorithm; different phases are used for shuffling the data. The drawback of this system is speed that depends on word size and provides fewer throughputs.

3. RESEARCH METHOD

3.1. AES Algorithm

The National Institute of Standards and Technology (NIST) have issued a call for an Advanced Encryption Standard (AES) algorithm which can overcome the drawbacks of 3DES in the year 1997. In the span of 5 years NIST received 15 algorithms. In the year 2001, NIST selected Rijndeal as the proposed AES algorithm. Rijndeal was proposed by Dr. Joan Daemen and Dr.Vincent Rijmen. Both of them are cryptographers from Belgium. The AES algorithm is having a block length of 128 bit with different key sizes as 128 bit, 192 bit and 256 bits. The 128 bit requires 10 rounds of operation, 192 bit requires 12 rounds of operation and 256 bit requires 14 rounds of operations. AES uses a symmetric key and a block cipher. Symmetric key is the key which is same for encryption and decryption. Block cipher takes a number of bits and encrypt them as a single unit. AES is not a feistel structure. Infeistel structure used half of the data to modify the other half and then the halves are swapped. The overall AES structure is shown in figure 1.

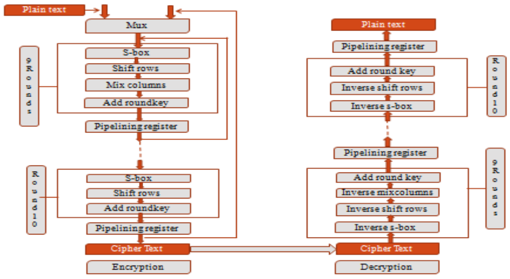

Figure 1. AES Structure

There are four stages present in each round of AES algorithm such as Substitute bytes, Shift rows, Mix columns and Add round key. The structure is simple. Both encryption and decryption starts with add round key. It is followed by 9 rounds which consist of all the four stages and the last round consists of only 3 stages. Mix columns will not be there in last round.

3.1.1. Substitute Bytes

The substitution bytes are the first stage in each round of AES algorithm. It is also known as sub bytes. AES defines a 16*16 matrix of byte values, which is called s-box; it contains a permutation of all possible 256 8-bit values. Each individual byte of state is mapped into new byte in the following way- the left most 4 bits are used as row value and the right most 4 bits are used as column value.

3.1.2. Shift Rows Transformation

In shift rows, the rows are shifted. The first row will not be shifted. Second row will be shifted by one byte circularly in right direction; third row is shifted by two bytes circularly right, fourth row is shifted by 3 bytes circularly in right direction. In inverse shift rows, the rows are shifted circularly in opposite direction. The first row will not be shifted. Second row is circularly shifted by one byte in left direction. For third row, a 2-byte circular right shift is performed. For fourth row, a 3-byte circular left shift is performed.

3.1.3. Mix Columns Transformation

In mix column, the transformation is performed on columns of the matrix. The first column will not be changed. The remaining columns will be changed in the following format.

$$A' = (A*02) \text{ xor } (B*03) \text{ xor } (C*01) \text{ xor } (D*01) \quad (1)$$

$$B' = (A*01) \text{ xor } (B*02) \text{ xor } (C*03) \text{ xor } (D*01) \quad (2)$$

$$C' = (A*01) \text{ xor } (B*01) \text{ xor } (C*02) \text{ xor } (D*03) \quad (3)$$

$$D' = (A*03) \text{ xor } (B*01) \text{ xor } (C*01) \text{ xor } (D*02) \quad (4)$$

The multiplication with (02) can be performed as, 1 bit left shift of the given 8 bit data and then a bitwise xor operation with (0001 1011) if the leftmost bit of the original value is 1 before the shift. If the leftmost bit is not 1 before the shift then the value should be left as it is after the shift. The multiplication of x with (03) is performed as {x xor (x*02)}.

3.1.4. Add Round Key Transformation

In add round key transformation, the 128 bits are XORed with 128 bits of the round key and the operations is given in eqn. (5).

$$S' = S \text{ xor } R \quad (5)$$

where S' represents the state after adding round key
S represents the state before adding round key
and R represents the round key

3.2. RC5 Algorithms

RC5 algorithm is a block symmetric cipher. It uses number of variable rounds, variable keys and variable data bits. In block cipher the same secret key is used in the encryption and decryption. So it should produce more secure when compared to other algorithm. Plaintext and cipher text are fixed length sequence used in the block cipher. RC5 algorithm has a two word size of input and making of 64 bit word cipher text and also same in decrypted plain text. There are four types of parameters are used in the RC5 algorithm such as,

(i) Word (W)

It specifies the variable number of word size in bits. The choices like 16, 32 and 64 are used in the Rc5 algorithm.

(ii) Rounds (r)

It specifies the variable number of rounds.it used to improve the security and speed. The number of rounds is 0, 1, 2, 3……255.

(iii) Byte (b)

It is nothing but the variable number of bits and secret key used in the encryption and decryption. That's why we used 'b' and used 0, 1, 2….255.

(iv) Key (K)

The secret key array used is K[0], K[1]…..K[b-1]. RC5 does not secure while we dint use any number of choices. It should have at least one round to be used. So the minimum number of possible is given as, w-32, r-12 and k-16.These are widely used in the RC5 encryption and it can provide more secure while using these combination. There are three important primitive parameters are given below:

1. Addition: It's used to add two numbers and taking a two's compliment. In reverse operation, it is used to subtract two numbers.

2. Ex-OR operation: It performs the XOR operation after addition. Same XOR operation is used during the decryption side for getting correct combination of the input and output.

3. Shift operation or Rotation: After XOR, it is used shift operation and taking left and right rotation. While encryption, right rotation (x<<<y) and left rotation (x>>>y) is used during decryption. Rc5 algorithm divided into three parts.

1. Key expansion
2. Encryption algorithm
3. Decryption algorithm

3.2.1. Key Expansion

Key expansion depends on the variable number of rounds. Random binary digits are arranged in the S table array. It's given below

$$t = 2(r+1) \quad (6)$$

where t denotes the size of the table and r denotes the number of rounds in the Rc5 algorithm

$$Pw = \text{odd}((e-2)2w) \quad (7)$$

$$Qw = \text{odd}((\varphi-1)2w) \quad (8)$$

Where, e = 2.718281828459… (base of natural logarithms), Φ = 1.618033988749… (golden ratio), Odd(x) = odd integer nearest to x. For w = 16 and 32 in hexadecimal form, the values are P16 is equivalent to b7e1, Q16 is equal to 9e37, P32is equal to b7e15163 and Q32 is equal to 9e3779b9. The three steps are applied as, Secret key conversion from bytes to words, Initialization of array S and Mixing of the secret key.

3.2.2. Encryption Algorithm

The architecture of RC5 encryption is shown in figure 2. The procedure is expressed with arithmetic operations are given.

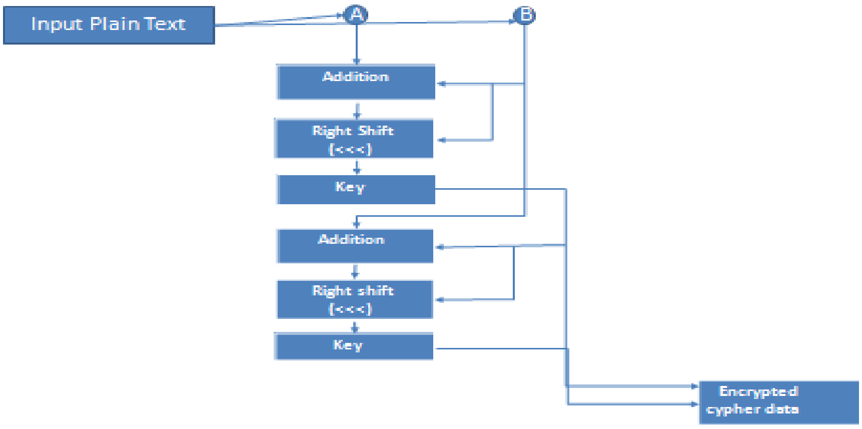

Figure 2. Architecture of RC5 Encryption

Two bit words are divided into A and B and processed in equations

$A = A + S\,[0];$ (9)

$B = B + S\,[1];$ (10)

for i = 1 to r do

$A = ((\,A \oplus B\,) <<< B\,) + S[\,2 * i\,];$ (11)

$B = ((\,B \oplus A) <<< A\,) + S[\,2 * i + 1];$ (12)

The output values are stored in the register and in particular round half of the input is upgraded.

3.2.3. Decryption Algorithm

The decryption algorithm can be derived from encryption algorithm and its architecture is shown in figure 3 for i = r down to 1 do

$B = ((B - S[2 * i + 1]) >>> A) \oplus A;$ (13)

$A = ((A - S[2 * i] >>> B) \oplus B;$ (14)

$B = B - S[1];$ (15)

$A = A - S[0];$ (16)

The output values are stored in the register in each round and the amount of rotation is not found. So, the rounds are different.

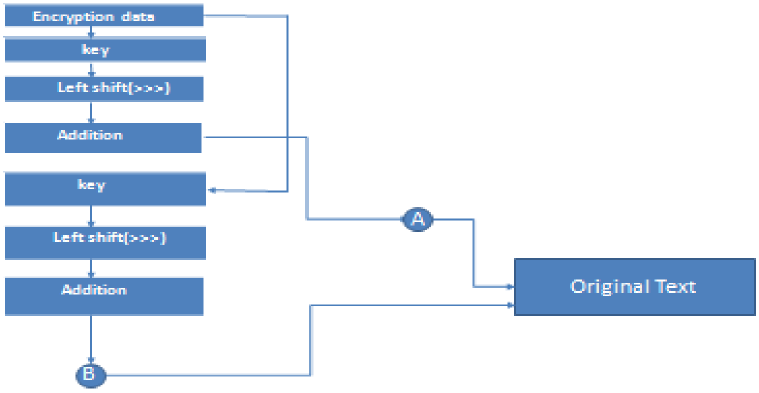

Figure 3. RC5 Decryption

3.3. SHA Algorithms

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. In present day's modern electronic operations used are E-mail, Internet banking, document transfer, online shopping, etc. 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, SHA-512. Hash Functions (or) task promote at the origin of many famous cryptography approaches in Digital Signature Standard (DSS), Message Authentication Codes (MAC's), etc., 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 HiperLAN and wireless protocols (WAP) have mentioned security layers and cryptographic schemes. Hash task is 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

3.3.1. 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 that 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.

3.3.2. 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< to < 15. The message scheduler takes each M(i) and expands it into 64 number of 32-bit blocks. The process is shown in equations (17)-(26).

$$(x_0)=ROT_7(x\ )\ xor\ ROT_{18}(x)\ xor\ SHF_3(x) \tag{17}$$

$$(x_1)=ROT_{17}(x)\ \ xor\ ROT_{19}(x)\ \ xor\ SHF_{10}(x) \tag{18}$$

$$W_t=\{X_1(W_{t-2}) + W_{t-7}+X_0(W_{t-15}) + W_{t-16} \tag{19}$$

Where $ROT_n(x)$ denotes the circular rotation of x by n positions to the right and $SHF_n(x)$ denotes the right shifting of x by n positions. Additions in the SHA-256 algorithm are Modulo 2^{32}.

3.3.3. Message Squeezing

The W_t words from the message extension stage are passed to SHA squeezing function (or) SHA core. The core used 8 number of 32-bit working variables that are A, B, C, D, E, F, G, H. Totally, 64 rounds of the Squeezing function are performed.

T_1=H + X_1(E) + Ch(E,F,G) + K_t + W_t (20)
T_2=X_0(A) + Maj (A, B, C) (21)

The constraints are, H=G G=F
F=E E=D+ T_1
D=C C=B
B=A A= T_1+ T_2

Where

Ch(x,y,z)=(x AND y) xor (x bar AND z) (22)

Maj(x,y,z)=(x AND y) xor (x AND z) xor (y AND z) (23)

(x_0)=ROT_2(x) xor ROT_{13}(x) xor ROT_{22}(x) (24)

(x_1)=ROT_6(x) xor ROT_{11}(x) xor ROT_{25}(x) (25)

The final 256-bit output is formed by concatenating the final hash value is

$H^{(N)} = H_0^{(N)}$ & $H_1^{(N)}$ & $H_2^{(N)}$ & ….$H_7^{(N)}$ (26)

3.3.4. Optimization Technique

CSA separates the sum and carry root and the carry propagation technique is carried out for minimizing the delay. The techniques are Unrolling and Pipelining.

(i) Unrolling

This unrolling technique performs multiple rounds of the squeezing function in combinational logic, for compute the hash function it will take less no of clock cycles. Ex: core was unrolled once; the hash should be calculated in half of the clock cycles. It decreases the clock frequency. The fast pipelining can be 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.

(ii) Pipelining

The structure of SHA algorithm is shown in figure 4 In SHA-256 algorithm, the majority and choice function plays a major role. Both the functions are compression unit of SHA algorithm. It will compress the 512- bits to 256 of the output with cipher text. The LFSR counter operates based on D-Flip flops, Seed values and EX-OR operation, the seed value is the key for generating the sequence of data. The shift register stores the first seed value, remaining registers are zero condition. In the initial condition no operations performed in the registers. The seed value shifted to next register for process the pseudo random output. This type of Pseudo random generator is used for improve the security. Carry Save Adder is one of the fastest adders among the all other adders in arithmetic operations. Adder will process the data depends on the well balanced carry save adder. It processes the output until it receives the carry from previous addition output. This addition took more time to process the entire data.

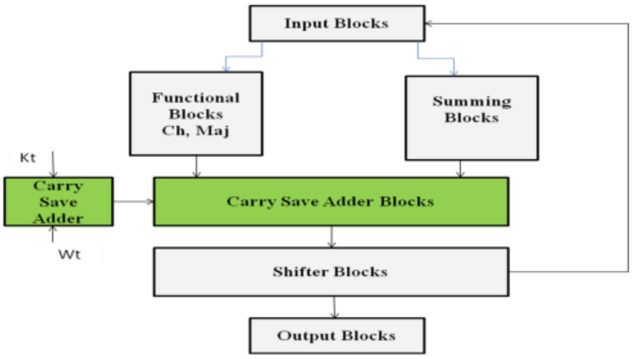

Figure 4. SHA algorithm

4. RESULTS AND DISCUSSIONS

The AES algorithm is implemented using Verilog and simulated using Model Sim 6.3g. The simulation of encryption and decryption of 128-bit AES algorithm has been done. **SATHYABAMA** is a 128 bit data which is considered as the plain text in this simulation. A 128 bit key has been given for this simulation. Both the plain text and key are given as input to the encryption procedure. Both the plain text and the key will undergo all the encryption stages in AES algorithm. The simulation graph for encryption and decryption of AES algorithm is shown in figure 5 and figure 6. The output of encryption procedure is **MTBAOHYBAA**. This text is given as input to the decryption along with 128-bit key. The cipher text and the key will undergo all the decryption stages in AES algorithm. After decryption we can retrieve the original plain text. Use of pipelining registers increases the speed of AES algorithm. This will reduce the power consumption. The RC5 algorithm is implemented in the same platform. The simulation of the RC5 encryption and decryption of 64 bit is shown in figure 7.

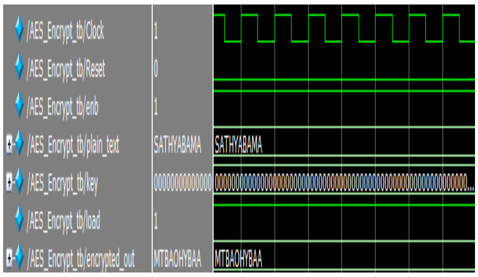
/AES_Encrypt_tb/Clock 1
/AES_Encrypt_tb/Reset 0
/AES_Encrypt_tb/enb 1
/AES_Encrypt_tb/plain_text SATHYABAMA SATHYABAMA
/AES_Encrypt_tb/key 000000000000 0000000000000000000000000000000000000000000000...
/AES_Encrypt_tb/load 1
/AES_Encrypt_tb/encrypted_out MTBAOHYBAA MTBAOHYBAA

Figure 5. AES Encryption result

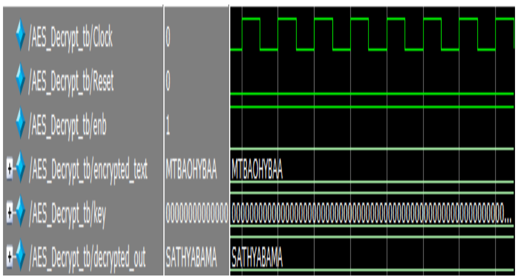
/AES_Decrypt_tb/Clock 0
/AES_Decrypt_tb/Reset 0
/AES_Decrypt_tb/enb 1
/AES_Decrypt_tb/encrypted_text MTBAOHYBAA MTBAOHYBAA
/AES_Decrypt_tb/key 000000000000 0000000000000000000000000000000000000000000000...
/AES_Decrypt_tb/decrypted_out SATHYABAMA SATHYABAMA

Figure 6. AES Decryption result

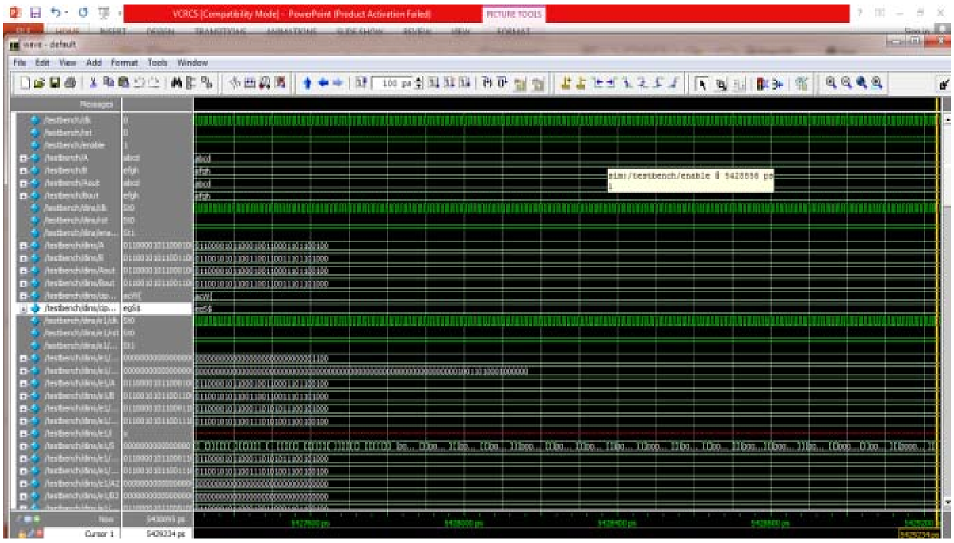

Figure 7. Encryption and Decryption result of RC5

The input of the 64 bit plain text is abcdefgh and it is divided into two 32 bits. In encryption, 32 blocks and 16 rounds are used and the same blocks and rounds are used in the decryption, for getting original 64 bit plain text. This algorithm is implemented in Quartus II V9.0. The Hashing and Rehashing result are mentioned in figure 8 and figure 9, the plain text of 512 -bits given into the input of SHA-256, input is processed by 8 blocks of 32-bits up to 64 iterations, and the data’s are temporarily stored into the 8 blocks (A, B, C, D, E, F, G, H) the block data’s are combined together to form a 256-bits output of a SHA algorithm. The same way cipher text is captured and gave to the input of rehashing, the output of original text will appear in the output of Rehashing SHA function. Finally, the original plain text is converted into cipher text and retrieves to original plain text. This algorithm implemented in Quartus II V9.0 and Modelsim 6.4a for simulations and performance factors.

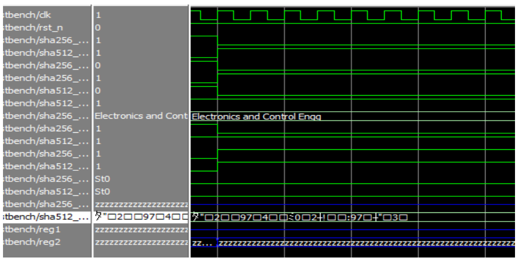
tbench/clk 1
tbench/rst_n 0
tbench/sha256_... 1
tbench/sha512_... 1
tbench/sha256_... 0
tbench/sha256_... 1
tbench/sha512_... 0
tbench/sha512_... 1
tbench/sha256_... Electronics and Cont
Electronics and Control Engg
tbench/sha256_... 1
tbench/sha512_... 1
tbench/sha256_... 1
tbench/sha512_... 1
tbench/sha256_... St0
tbench/sha512_... St0
tbench/sha256_... zzzzzzzzzzzzzzzzzzzz
tbench/sha512_... ㄆ"□2□□97□4□□
ㄆ"□2□□97□4□□:0□2+!□□:97□+"□3□
tbench/reg1 zzzzzzzzzzzzzzzzzzzz
tbench/reg2 zzzzzzzzzzzzzzzzzzzz
zz... zzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzz

Figure 8. Encryption result of SHA-256

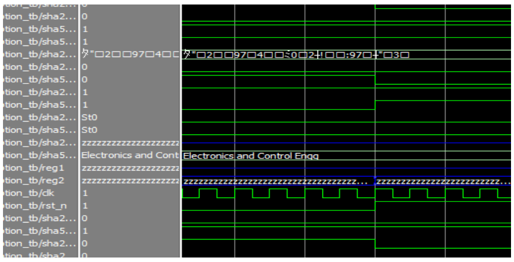
tion_tb/sha2... 0
tion_tb/sha5... 1
tion_tb/sha5... 1
tion_tb/sha2... ㄆ"□2□□97□4□□
ㄆ"□2□□97□4□□:0□2+!□□:97□+"□3□
tion_tb/sha2... 0
tion_tb/sha5... 0
tion_tb/sha2... 1
tion_tb/sha5... 1
tion_tb/sha2... St0
tion_tb/sha5... St0
tion_tb/sha2... zzzzzzzzzzzzzzzzzzzz
tion_tb/sha5... Electronics and Cont
Electronics and Control Engg
tion_tb/reg1 zzzzzzzzzzzzzzzzzzzz
tion_tb/reg2 zzzzzzzzzzzzzzzzzzzz
zzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzz...
tion_tb/clk 1
tion_tb/rst_n 1
tion_tb/sha2... 0
tion_tb/sha5... 1
tion_tb/sha2... 0

Figure 9. Rehashing output of SHA-256

5. CONCLUSION

In this paper, a software framework for the implementation of data security algorithms is described. Three different algorithms have been taken in this study and its implementations are analyzed in Quartus – II software. The encryption and decryption are designed using Verilog HDL and simulated using ModelSim. Among these three algorithms, SHA-256 is more compatible for processing lengthy data and it provides high security. The system satisfies all the requirements and the results proven its reliability for the data transmission.

REFERENCES

[1] Manjesh K.N, R.K. Karunavathi. "Secured High Throughput Implementation of AES algorithm". *IJARCSSE*. 2013; 3: 1193-1198.

[2] M. Vanitha, R. Sakthivel, Subha. "*Highly* Secured High Throughput VLSI Architecture for AES algorithm". *IEEE*. 2012; 1(7): 403-407.

[3] Ritu Pahal, Vikas Kumar. "Efficient Implementation of AES". *IJARCSSE*. 2013; 3(7): 290-295.

[4] Abdulkarim Amer Shtewi, Bahaa Eldon M Hasan, Abd El Fatah A Hegazy. "An Efficient Modified Advanced Encryption Standard (MAES) Adapted for Image Cryptosystems". *IJCSNS*. 2010; 10(2): 226-232.

[5] Xinmiao Zhang, Keshab K Parhi. "High-Speed VLSI Architectures for the AES Algorithm". *IEEE*. 2004; 12(9): 957-967.

[6] B. Santhi, K.S. Ravichandran, A.P. Arun, L. Chakkrapani. "A Novel Cryptographic Key Generation Method Using Image Features". *Research Journal of Information Technology*. 2012; 4(2): 88-92.

[7] Mr. Vikas Tyagi. "Data Hiding in Image Using least significant bit with Cryptography". *IJARCSSE*. 2012; 2(4): 120-123.

[8] Mostafa Abd-El-Barr, Altaf Al-Farhan. "A Highly Parallel Area Efficient S-Box Architecture for AES Byte-Substitution". *IACSIT International Journal of Engineering and Technology*. 2014; 6(5): 346-350.

[9] M. Narasimhulu, S. Mahaboob Basha, P. Chandra Sekhar. "Hardware Implementation of High Performance AES using Minimal Resources". *IJER*, 2014; 3(2): 68-72.

[10] Mr. Shelke R.B, Mrs. Patil A.P, Dr. Pa.il S.B. "VLSI Based Implementation of Single Round AES Algorithm". *IOSR Journal of Electronics and Communication Engineering*. 2009; pp. 63-67.

[11] Pravin Kawle, Avinash Hiwase, Gautam Bagde, Ekant Tekam, Rahul Kalbande. "Modified Advanced Encryption Standard". *IJSCE*. 2014; 6(1): 120-129.

[12] T. Rahman, S. Pan, Q. Zhang. "*Design of a High Throughput 128-bit AES (Rijndeal Block Cipher)*". Proceedings of International Multi Conference of Engineers and Computer Scientists. 2010; 2.

[13] Stallings W. *Cryptography and Network Security*. Third Edition, Pearson Education, 2003

[14] Julia Juremi, Ramlan Mahmod Salasiah Sulaiman Jazrin Ramli. "Enhancing AES s-box generation based on Round key". *International Journal of Cyber-Security and Digital Forensics*. 2012, 1(3): 183-188.

[15] M. Gnanambika, S. Adilakshmi, Dr. Fazal Noorbasha. "AES-128 Bit Algorithm Using Fully Pipelined Architecture for Secret Communication". *International Journal of Engineering Research and Applications*. 3(2): 166-169.

[16] 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*. 2013; 37: 750-757.

[17] Robert. P, McEvoy, Francis.M, Crowe, Colin C. Murphy and William P. Marnane. "Optimisation of the SHA-2 Family of Hash Functions on FPGAs". *International Symposium of Emerging VLSI Technologies and Architectures*. 2006.

[18] 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*. 2008; 2: 147-158.

[19] O. Koufopavlou. "Implemenation of the SHA-2 hash family standard using FPGA's". *The journal of Supercomputing*. 2005; 31(3): 227-248.

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

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

[22] M. Juliato, C. Gebotys. “Tailoring a reconfigurable platform to SHA-256 and HMAC through custom instructions and peripherals”. *International Conference on Reconfigurable Computing and FPGA’s, IEEE Computer Society*. 2009: pp. 195-200.
[23] B. Ramkumar, Harish M Kittur. “Low-Power and Area Efficient Carry Select Adder”. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*. 2012; 20(2).
[24] Y. Kim, L.S. Kim. “64-bit carry select adder using Single ripple carry adder,” *Electron Letters*. 2001; 37(10): 614-615.
[25] Y. He, C.H. Chang, J. Gu. “*An area efficient 64-bit square root carry select adder for low power applications*”. Proceedings of *IEEE* International Symposium on Circuits and Systems. 2005; 4: 4082-4085.
[26] K. Chandra Sekhar, K. Saritha Raj. “An Efficient Pseudo Random Number Generator for Cryptographic Applications”. *International journal of Engineering and Advanced Technology*. 2014; 4(1).
[27] Padma Devi, Ashima Gridher, Balwinder Singh. “Improved Carry Select Adder with Reduced Area and Low Power Consumption”. *International Journal of Computer Applications*. 2010; 3(4).
[28] Ms. Archana Kakde, Ms. Manisha Waje. “Low power & Area Efficient 16 bit Carry Select Adder Based on Adiabatic Logic”. *International journal of Engineering and Advanced Technology*. 2014; 2(2).
[29] A. Menezes, P. van Oorchol, S. Vanstone. “*Handbook of applied Cryptography*”. CRC Press, Inc, October 1997.
[30] J. Goodman, P. Chandrasekaran. “An energy efficient reconfigurable public-key cryptography processor”. *IEEE journal of Solid-state circuits*. 2001; 36(11): 1808-1820.
[31] Citavicius, A. Jonavicius. “An Image encryption using Pseudo-Random Number Generator based on Non-Linear Dynamic Chaotic System”. *WSEAS Transactions on Communications*. 2009; 8(9): 1022-1031.