ISSN 2250-3153

Multi-Layered Cryptographic Processor for Network Security

Pushp Lata*, V. Anitha**

* M.tech Student, E & C Department,Dayananda Sagar College of Engineering, Bangalore, India
** Associate Professor, Dayananda Sagar College of Engineering, Bangalore, India

Abstract- This paper presents a multi-layered architecture for the security of network and data using a layered structure of cryptographic algorithms. This architecture gives benefit of low area and high speed requirement. This paper mainly concern for security so architecture is designed in layers to avoid the possibilities of invasion from outsiders. This paper implements a combination of Private-Public-Private key algorithm for encryption-decryption in two layers and a Public key for key generation. Simulation of codes is carried out on ModelSim 6.3f and design optimization is shown using Xilinx-Project Navigator ISE 13 suite. This paper proposes multi-layered architecture of integrated cryptographic processor which can be used for any byte length. This architecture decreases the use of any one cryptographic algorithm designed for any one specific purposes.

Index Terms- Network Security, cryptography, Multi-layered architecture

I. INTRODUCTION

IN this digital world, with the increment of Internet in human life every step like Banking, payment, financial transaction etc. The importance of network security is also increasing. Security forms the backbone of today's digital world.

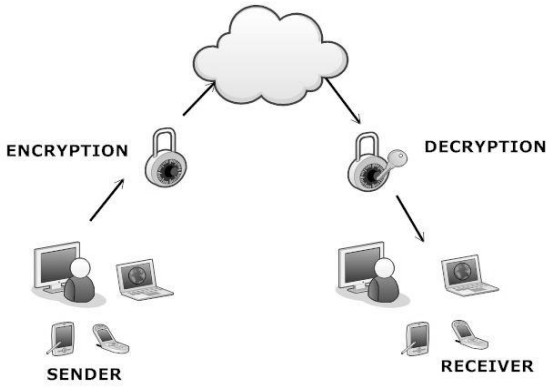

Fig. 1: Cryptographic process

There are mainly two types of cryptographic systems used for security purpose. One is Symmetric key algorithm (also known as Private Key algorithm) and other is Asymmetric key algorithm (also known as Public key algorithm).The symmetric key algorithm uses same key for encryption and decryption of a message (like Advanced Encryption Standard (AES),Data Encryption Standard (DES).The asymmetric key algorithm use one key for encryption of a message and a different key for the decryption of same message. Private Key algorithm due to having high throughput is suitable for data communication, and public key algorithm with much lower throughput are suitable for key exchange. Advanced Encryption Standard (AES), a Federal Information Processing Standard (FIPS), is an approved cryptographic algorithm that can be used as private key algorithm. Data Encryption Standard (DES), selected by the National Bureau of Standard as an official Federal Information Processing Standard (FIPS) for United States in 1976, is also a widespread use candidate for private key algorithm. Rivest-Shamir-Adi (RSA) is a public key algorithm key algorithm invented in 1977 by Ron Rivest, Adi Shamir and Leonard Adleman (RSA).

This paper shows the architecture of implementation of a multi-layered structure of cryptographic algorithm (Private-Public-private key algorithm). Elliptic Curve Cryptography (ECC) is most suitable for key exchange for all the available algorithms.

II. CRYPTOGRAPHIC ALGORITHMS

In this section, introduction and a brief detail of all the three algorithms (Private-Public-private key algorithm) for encryption and decryption of message are provided. This section also explains the key exchange protocol using Elliptic Curve Cryptography (ECC).

A. Advanced Encryption Standard (AES)

The Advanced Encryption Standard (AES) algorithm operates on blocks of 128,192 or 256 bits, by using cipher keys of length 128,192 or 256 bits respectively. The block diagram for Advanced Encryption Standard (AES) is as shown in figure2.

Inputs and Outputs

AES-128 uses 128b key and need 10 rounds.Similarly, AES-192 uses 192b key with 12 rounds and AES-256 uses 256b key with 14 rounds of data processing.

The basic unit for processing in the Advanced Encryption Standard (AES) algorithms is a byte (a sequence of eight bits).The first step in Advanced Encryption Standard (AES) encryption is conversion of input bit sequence into byte sequence, which is arranged in a two dimensional array of bytes (called as States). Each byte in state matrix is an element of a Galois field, GF (2^8). The AES-128 consists of 10 rounds with four main operations in each round, namely, "ByteSub", "Shift Row", "MixColoumn", and "AddRoundKey". States undergoes

to AddRoundKey initially before the round-1.There is no MixColoumn in lastround. The key scheduler block is used for generating 128b key for each round. Key Scheduler block consists of two main operations, firstly, key expansion, which expands the input key bits required by the algorithm, and lastly, the key selection, which selects the required number of bits from the expanded key. The implementation of Advanced Encryption Standard (AES) mainly needs LUT (for ByteSub), MixColoumn need GF (2^8), circularshifter (for ShiftRow), logicalXOR for AddRoundKey.

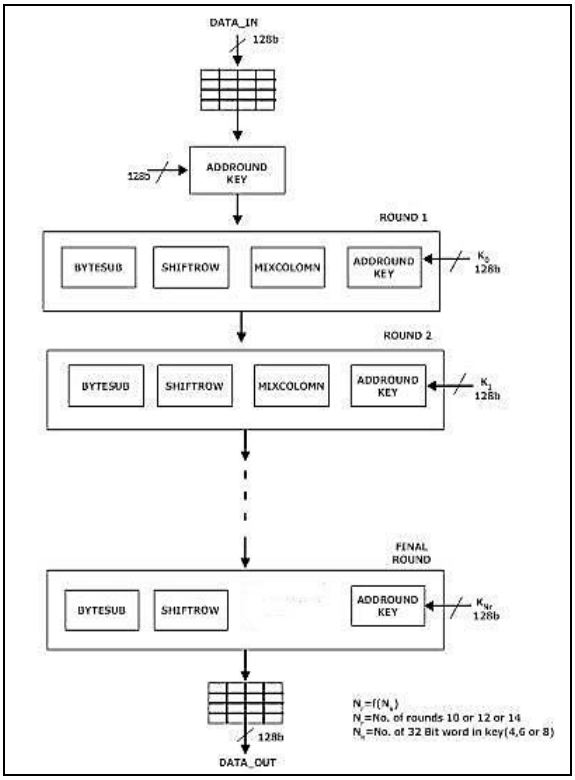

Fig. 2:AES block diagram

B. Data Encryption Standard (DES)

Data Encryption Standard (DES) is the block cipher architecture type of algorithm that takes a fixed length string of plaintext bits and transforms it through a series of complicated operations into another ciphertext bit string of the same length. The block size is of 64-bits; however, only 56-bits of these are actually used by the algorithm, as rest 8 bits are kept for checking parity purpose only and hence discarded. Every 8^{th} bit of the selected key (i.e., 64b) is discarded, that is, positions 8, 16, 24, 32, 40, 48, 56, 64.Data Encryption Standard (DES) consists of 16-rounds. The figure 3 shows the single stage structure of Data Encryption Standard (DES).Data block first go through Initial permutation (IP), then to 16 rounds of complex key-dependent permutation, and finally, to another permutation which is the inverse of the IP called as IP^{-1} as shown in figure2.The Fiestelfunction, f () operates on half of 64b block i.e., 32b at a time and consists of four stages namely,Expansion,Keymixing, Substitution (also known as S-Box) and Permutation (also known asP-Box).

The implementation of Data Encryption Standard (DES) needs XOR, Shift, LUT and Permutation. 3DES is more secure than Data Encryption Standard (DES) considered by National Institute of Standards and Technology (NIST). 3DES (or TDES) can also be implemented in place of Data Encryption Standard (DES).

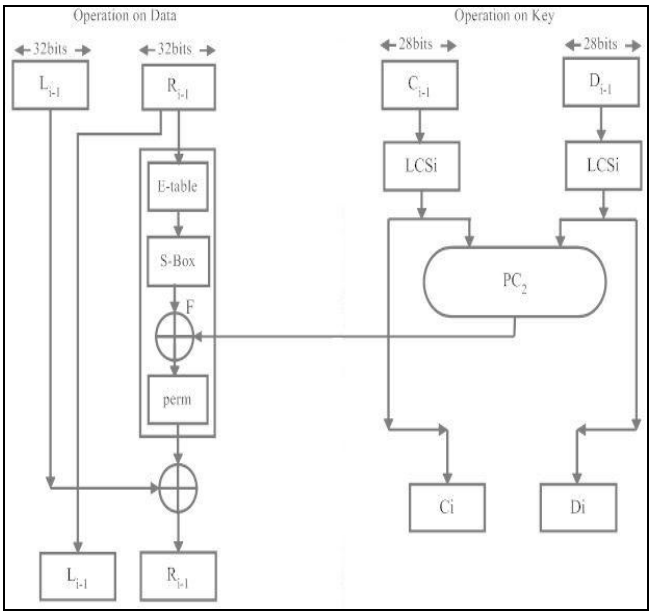

Fig. 3: Single stage operation of DES

C. Rivest-Shamir-Adi (RSA)

Rivest-Shamir-Adi (RSA) algorithm can be classified as three algorithms; the key generation algorithm, encryption algorithm, decryption algorithm.

Firstly,Rivest-Shamir-Adi (RSA) key generation algorithm can be described as follows.

1) Generate two large random and distinct primes P and Q, $P \neq Q$.
2) Calculate N=P.Q and ϕ= (P-1)(Q-1). Here, N is used as the modulusfor both the public and private keys.
3) Choose a random integer, E, $1<E<\phi$, and find greatest common divisor such that $gcd(E,\phi)=1$.Where,ϕ is Euler's totient function.
4) Compute the unique integer D, $1<D<\phi$, such that $ED=1 \pmod{\phi}$, i.e., E and ϕ are coprime.
5) Public key is (N, E) and private key is (N, D).

Secondly, Rivest-Shamir-Adi (RSA) encryption algorithm can be described as follows, $C=M^E \bmod N$

Finally, Rivest-Shamir-Adi (RSA) decryption algorithm can be described as follows, $M=C^D \bmod N$

Where, C is the ciphertext and M is the message.

D. Elliptic Curve Cryptosystem (ECC)

An elliptical curve is defined by an equation in Z variables with coefficients. The cubic equation for elliptical curve takes the form $y^2 +axy+by= x^3+cx^2+dx+e$

Where,a,b,c,d,e are coefficients, and x & y are variables

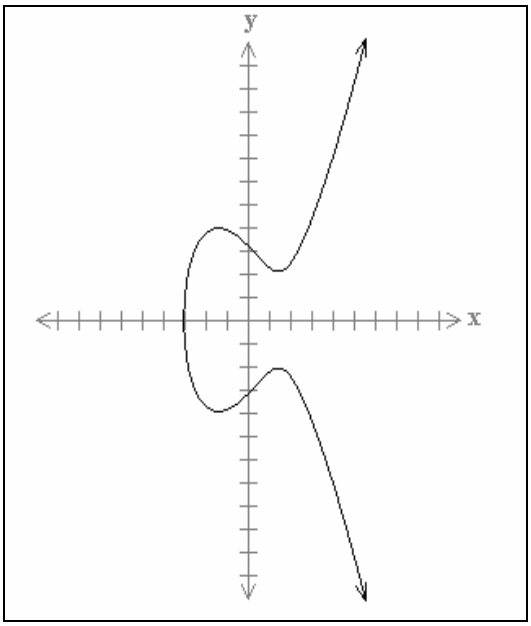

Fig. 4: ECC curve

Elliptic Curve Cryptosystem (ECC) operates over a group of points on an elliptical curve defined over a finite field. Its main cryptography operation is scalar multiplication, which computes Q=kP, i.e., a point P multiplied by an integer k resulting in another point Q on the curve.

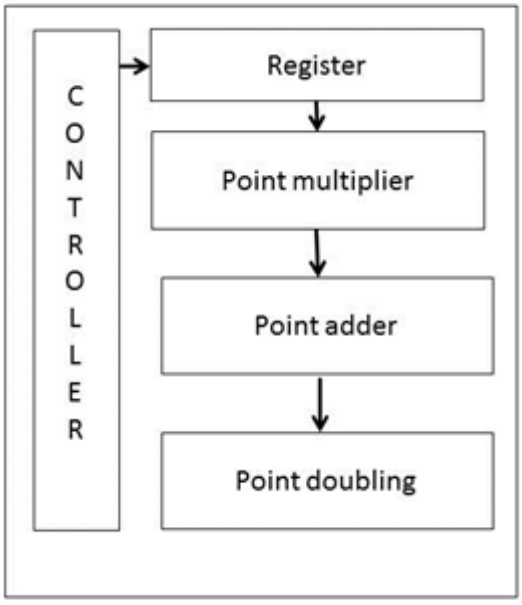

Fig. 5: ECC block diagram

III. ARCHITECTURAL DESCRIPTION

The overall architecture of cryptographic processor for network security is shown in figure5.

A. Key Generation

A public key cryptographic algorithm is implemented for the generation of a secure key for the data encryption and decryption.The host CPU controls the overall processes. The key exchange protocol of Elliptic Curve Cryptosystem (ECC) is implements key for data processing algorithms.

B. Microprocessor

The microprocessor is used to select the cryptographic algorithm among the three according to the type of message/data bits being transmitted. The message goes to the various algorithm processes of respective algorithm resulting in an encrypted message. The decryption is almost similar but in reverse direction by the help of control unit and microprocessor.

C. Processor block

The processor block performs the encryption of data. This block consists of functional blocks of Advanced Encryption Standard (AES), Data Encryption Standard (DES)and Rivest-Shamir-Adi (RSA).

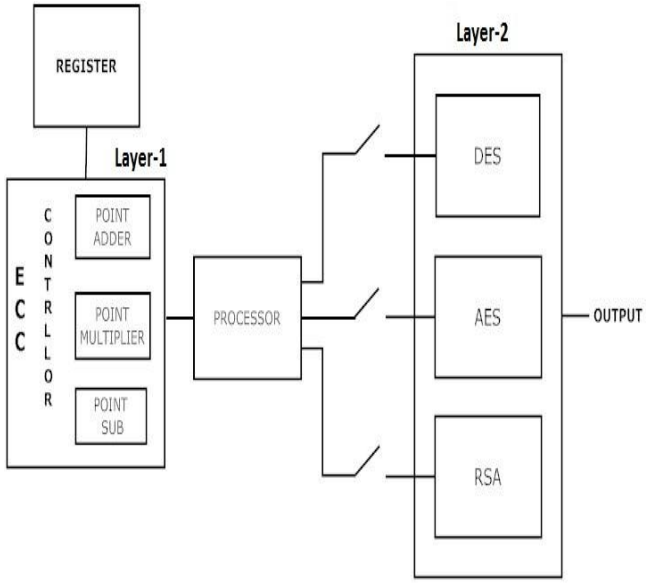

Fig. 6.: Architecture of cryptographic processor for network security

IV. IMPLEMENTATION

The blocks used for Advanced Encryption Standard (AES) [5] [6] and Data Encryption Standard (DES) [7] [8] are commonly selected blocks and area requirement decreases and speed for execution increases. The AddRoundKey and InvAddRoundKey is implemented using simple XOR operation followed by shift and XOR for MixColoumn and InvMixColoumn. The four basic operations of the Data Encryption Standard (DES) are completed by XOR, shift, LUT and permutation blocks.

The architecture is divided into two layer of cryptographic algorithm namely ECC, for generation of key and second layer consists of all the three algorithms structure, which performs according to the size of data being used. The multi-layered

architecture helps in increasing the throughput as shown in comparison table 1.

The architecture of multi-layered cryptographic processor is shown in figure6.

V. RESULT

A. *Advance Encryption Standard (AES)*

Simulation is done for 128-bits input of data which is plain text and the roundkey are placed in register. The decryption followed by encryption consists of encrypted data of 128-bits which is the ciphertext and key given for both encryption and decryption is of 128-bits.As key generated and plain text is given, the encrypted output following each blocks includes s-box generation in round 1 to round 10 and MixColoumn generation in round 1 to round 9 as there is no MixColoumn in last round of execution.

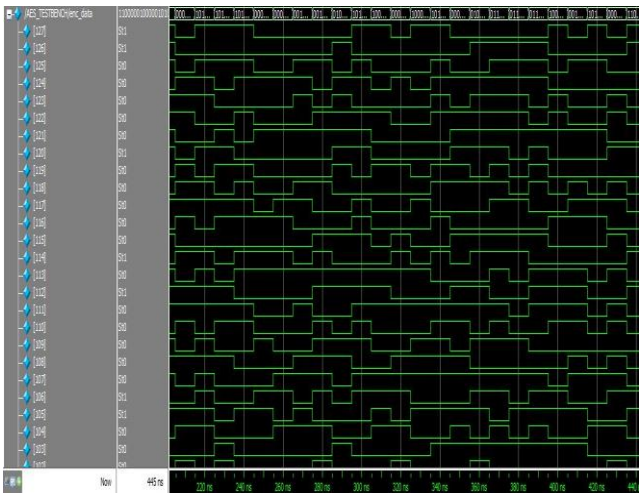

Fig. 7:AES Simulation Result (encrypted_data)

As shown in figure7, the encrypted data is decrypted with same blocks but now in reverse order, letting last block R10 as first block now and similarly R1 as last block.The completion of encryption and decryption can be easily obtained by completion signal i.e. encrypt_complete for encryption completion and decrypt_complete for decryption completion.

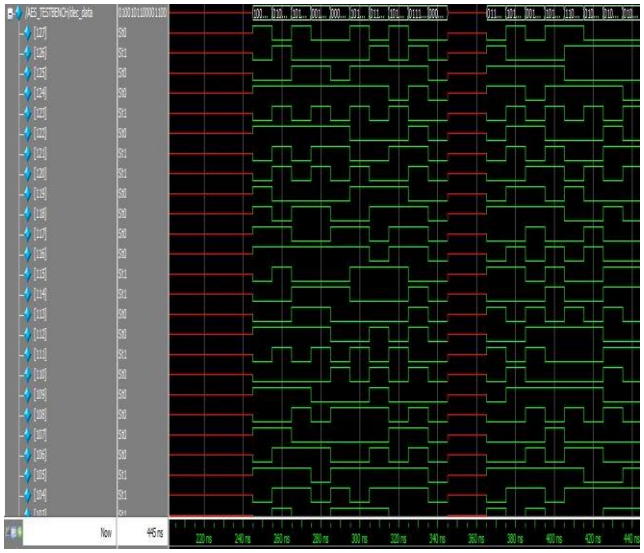

Fig. 8: AES Simulation Result (decrypted_data)

The final simulation is shown in figure 9. The summary generated using Xilinx ISE is shown in figure 10.

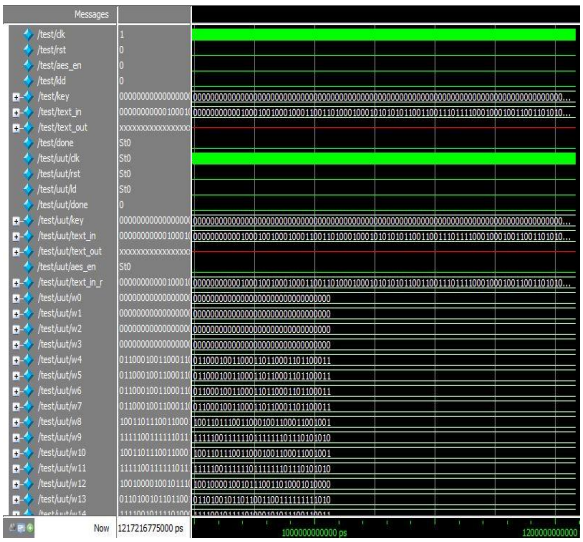

Fig. 9: AES Simulation Result (complete)

The summary for AES shows that there are no errors present in the AES design and the device utilization is according to the specifications showing optimum utilization of all resources available.

Project File:	aes128.xise	Parser Errors:	No Errors
Module Name:	DEVICELEVEL	Implementation State:	Placed and Routed
Target Device:	xc3s400-4pq208	•Errors:	No Errors
Product Version:	ISE 13.2	•Warnings:	No Warnings
Design Goal:	Balanced	•Routing Results:	All Signals Completely Routed
Design Strategy:	Xilinx Default (unlocked)	•Timing Constraints:	All Constraints Met
Environment:	System Settings	•Final Timing Score:	0 (Timing Report)

Device Utilization Summary				
Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Flip Flops	474	7,168	6%	
Number of 4 input LUTs	4,290	7,168	59%	
Number of occupied Slices	2,371	3,584	66%	
Number of Slices containing only related logic	2,371	2,371	100%	
Number of Slices containing unrelated logic	0	2,371	0%	
Total Number of 4 input LUTs	4,346	7,168	60%	
Number used as logic	4,290			
Number used as a route-thru	56			
Number of bonded IOBs	23	141	16%	
Number of BUFGMUXs	2	8	25%	

Fig. 10: AES Simulation Summary

B. Data Encryption Standard (DES)

Simulation is done for 64-bit input of data which is plain text and the roundkey are placed in register. The decryption followed by encryption consists of encrypted data of 64-bits which is the ciphertext and key given for both encryption and decryption is of 56-bits.At first the text_in is divided into two half known as left and right blocks.The key generation for round 1-8 is shown in figure 13.

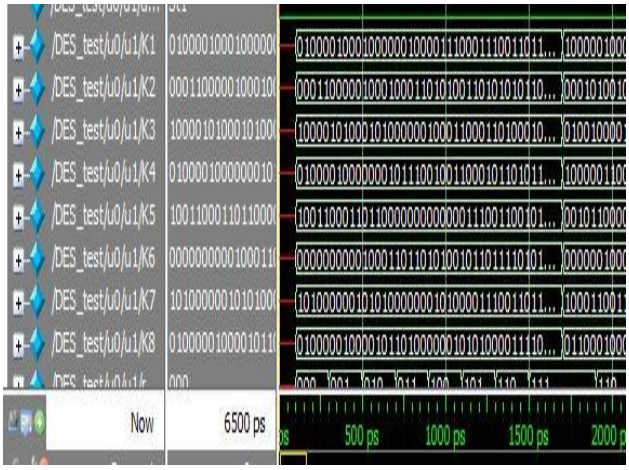

Fig. 11: DES Simulation Result (key- round1-8)

The top_module shown in figure12 shows complete simulation result of DES. In this complete encryption and decryption of data is shown.

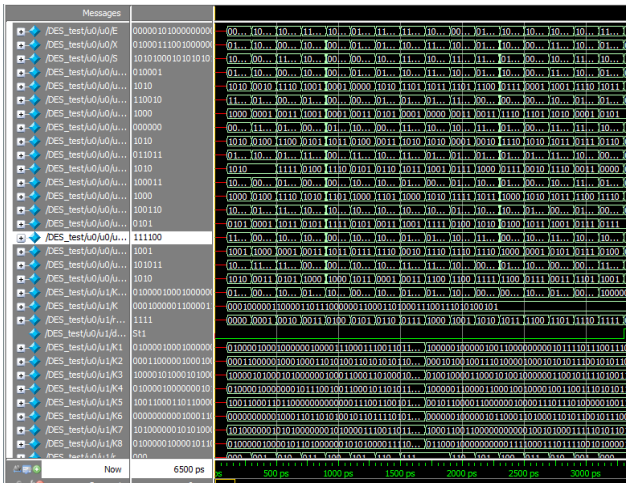

Fig.12: DES Simulation Result (top_module)

Finally figure13 shows summary of DES execution using Xilinx ISE showing no error in the design and optimum utilization of all devices.

des Project Status			
Project File:	DES.xise	Parser Errors:	No Errors
Module Name:	des	Implementation State:	Programming File Generated
Target Device:	xc3s500e-5ft256	•Errors:	No Errors
Product Version:	ISE 13.2	•Warnings:	No Warnings
Design Goal:	Balanced	•Routing Results:	All Signals Completely Routed
Design Strategy:	Xilinx Default (unlocked)	•Timing Constraints:	All Constraints Met
Environment:	System Settings	•Final Timing Score:	0 (Timing Report)

Device Utilization Summary				
Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Flip Flops	64	9,312	1%	
Number of 4 input LUTs	852	9,312	9%	
Number of occupied Slices	449	4,656	9%	
Number of Slices containing only related logic	449	449	100%	
Number of Slices containing unrelated logic	0	449	0%	
Total Number of 4 input LUTs	852	9,312	9%	
Number of bonded IOBs	190	190	100%	
Number of BUFGMUXs	1	24	4%	
Average Fanout of Non-Clock Nets	3.99			

Fig. 13: DES Simulation Summary (using Xilinx)

C. RON, ADI SHAMIR, RIVEST ADI (RSA)

Simulation is done for 512-bit input of data which is plain text and the roundkey are placed in register. The decryption followed by encryption consists of encrypted data of 512-bits which is the ciphertext.

Simulation results

The Random number generator, GCD, Encryption and decryption are written in Verilog Code and simulated in ModelSim 6.3f and results are mentioned below.

1) Random number generator

The 16-bit random number generator implemented using the 16-bit LFSR is shown in Figure14.

Fig. 14:RSA Simulation Result (random number generator)

Figure 14 shows the waveform of the random odd number generator which has generated few odd numbers .

2) gcd

The 16-bit GCD (Extended Euclidean algorithm) implemented is shown in Figure 15.

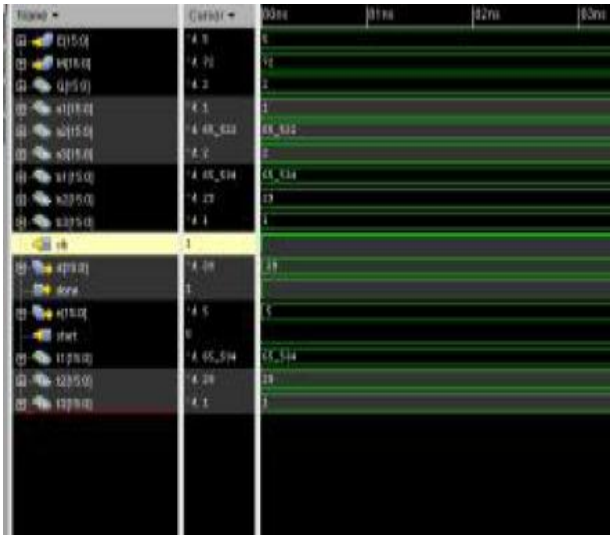

Fig.15:RSA Simulation Result (gcd generator)

Figure15 also shows the waveform for extended Euclidean algorithm with two inputs and the resulted output is the public key e and the private key d.

Encryption and Decryption

The 16-bit Modular Multiplication [12] simulated using ModelSim is shown in Figure16.

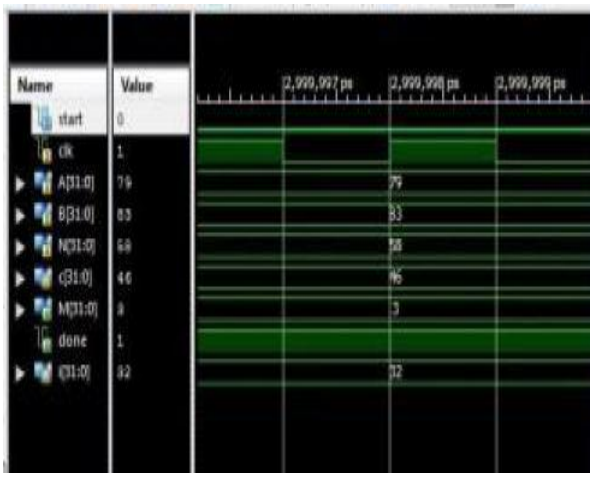

Fig. 16: RSA Simulation Result (modular multiplication)

Figure16 shows the waveform for modular multiplication module in which the input value is A, B and N, the resulted output is M. The 16-bit Encryption module simulation is shown in Figure17.

The waveform for encryption module consists of input values is e, n and plain text p, the resulted output is the encrypted cipher text m. The 16-bit decryption module simulation is shown in Figure18.

Fig. 18: RSA Simulation Result (decryption)

Figure18 shows the waveform for decryption module in which the input value is d, n and cipher text p, the resulted output is the encrypted plain text m.

3) Top Module (RSA)

Figure19 shows the waveform of the Top module of entire RSA. Here the random prime numbers generated are P and Q. After getting P and Q, n and ϕ (n)are calculated. Andpublic key e is selected by random prime number. GCD checks whether e and ϕ (n) arerelatively prime or not by getting the GCD as 1, and using Extended Euclidean algorithm d is calculated. Now the plain text M is given, and by using e and n, after encryption we get the cipher text C. Now by using d, n and the cipher text, after decryption we got back the plain text same as the expected result.

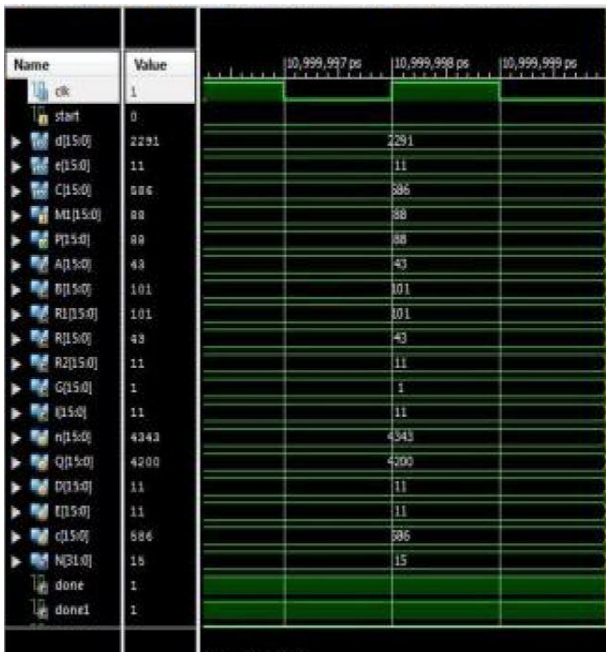

Fig. 19: RSA Simulation Result (top_module)

D. ELLIPTICAL CURVE CRYPTOGRAPHY (ECC)

The point multiplication operation and the point verification operations are implemented by ECC architecture [18]. The shared memory stores the operands, k, Px, Py before the start of the multiplication operation. The results are placed back into the shared memory after operation executed. The result of ECC follows following steps as shown from figure 20 to 23.

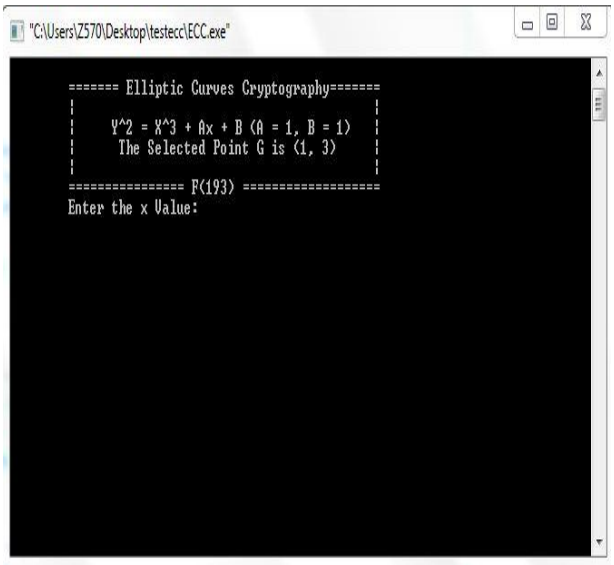

Fig. 20: Console output for ECC (1st step)

complete, the results are placed back into the shared memory. The curve parameters a/b/m as well as the width of memory is fixed for the particular version of the core. As figure20 shows the console showing output window for ECC, asking for input from the user but it can be automated to save time and energy of the user.

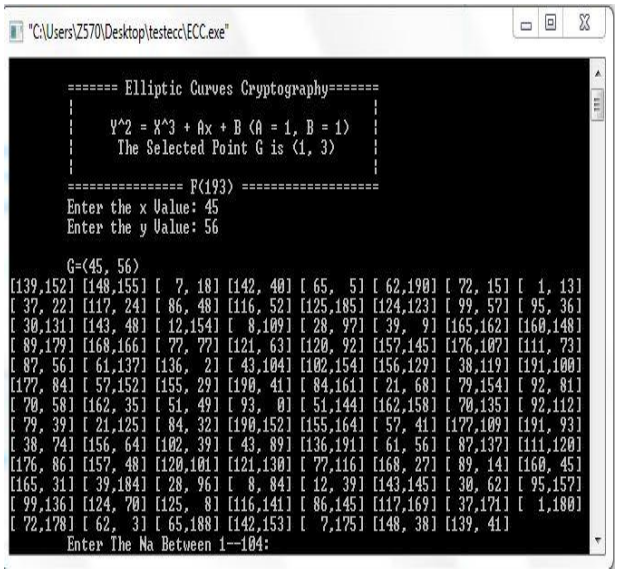

Fig. 21: Console output for ECC (2nd step)

The input for each parameter is taken by the user and the elliptical curve point is generated as shown in figure 21.

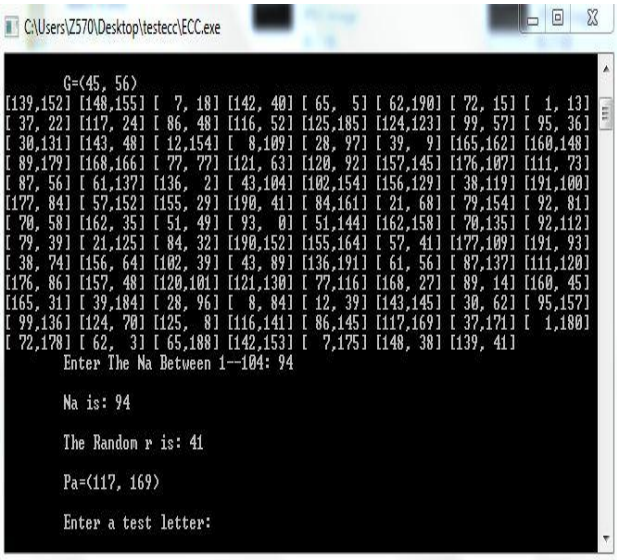

Fig. 22: Console output for ECC (3rd step)

After determining the curve point, one random r is chosen and the console is now ready for taking the data to be encrypted as shown in figure22.

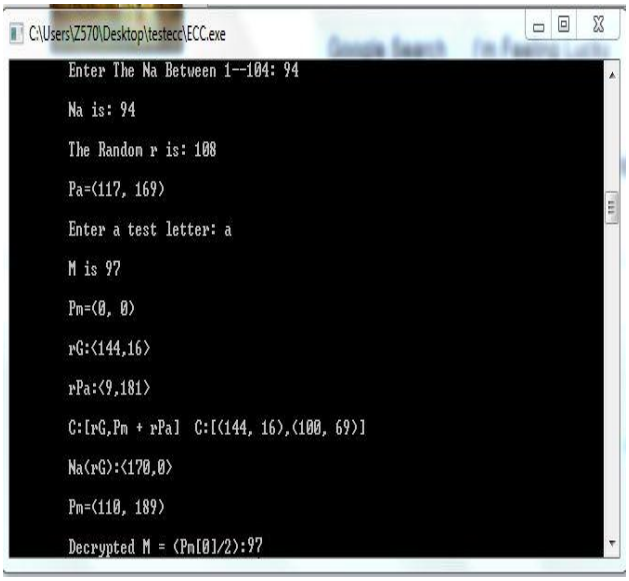

Fig. 23: Console output for ECC (final step)

The message to be encrypted is shown by M by converting the alphabets into integer to suite the processor functioning. Here encrypted message is shown by C. The decrypted message is shown by decrypted m.

Design of ECC allows sharing the arbitrated memory to store the arguments and results of operations. This both helps to save the resource in the extremely compact implementations and simplify the data transfer. The time required for memory loading is smaller than 7-8% Dedicated memory can be also be used.

VI. PERFORMANCE COMPARISON

The performance of individual cryptographic algorithms and the multi-layered cryptography algorithm can be done on the basis of encryption execution timing, memory required, output byte and finally throughput. The comparison is shown in Table 1.

Table 1: Performance Comparison

	Encryption Time (Sec)	Memory (KB)	Output Byte (KB)	Throughput (Mb/sec)
Advanced Encryption Standard (AES)	4.4	18,765	132,072	24.60
Data Encryption Standard (DES)	5.1	20,256	130,032	19.80
Rivest Shamir Adleman (RSA)	16.2	25,239	70,656	20.10
Layered Cryptography Processor	2.4	16,465	63,356	145.50

Encryption execution timing is the total time taken for encryption to the throughput and it should be as low as possible. Memory requirement is the use of memory space while performing the encryption or decryption which should be low for better performance. Output byte shows the length of encrypted text as it should be higher to avoid any hindrances to data. Finally a high throughput shows high speed and low power consumption by the system.

VII. CONCLUSION

The design of cryptographic processor presents a secure key generation internally in the architecture without knowledge of the user also.

This paper also implements a combinational architecture of Private-Public-Private key algorithm. This universal architecture has no bound of data size, so reduces the requirement of different cryptographic processor for different data size and other specifications. This paper can also be implemented with other private or public key algorithms to increase the efficiency even more.ECC usages can decrease the storage requirements for the execution of the protocols and so use of ECC for the future developments with regard to the network security is beneficial.

REFERENCES

[1] YadollahEslami, Ali Sheikholeslami, P.GlennGulak, Shoichi Masui, Kenji Mukaida"An Area-Efficient Universal Cryptography Processor for Smart Cards" IEEE transactions on very large scale integration (VLSI) systems, journal Vol. 14, NO.1, January2009.

[2] BehrouzA. Forouzan, "Cryptography and Network Security", Tata McGraw Hill, Special Indian 2007.

[3] Bruce Schneier, "Applied Cryptography", 2nd Edition. 1996.

[4] National Institute of Standards and Technology "Federal Information Processing Standards Publication 197", 2001.

[5] Joan Daemen and Rijmen, "The Design of Rijndael: AES - The Advanced Encryption Standard." Springer, 2002.

[6] Christ of Paar, Jan Pelzl, "The Advanced Encryption Standard", Springer, 2009.

[7] Biham, Eli: A Fast New DES Implementation in Software.

[8] Biham, Eli and Shamir, Adi (1991). "Differential Cryptanalysis of DES-like Cryptosystems". Journal of Cryptology 4 (1): 3–72.

[9] Diffie, Whitfield and Martin Hellman, "Exhaustive Cryptanalysis of the NBS Data Encryption Standard" IEEE Computer 10 (6), June 1977, pp74–84

[10] Vishwanath Patel, R. C. Joshi, A. K. Saxena "FPGA Implementation Of DES Using Pipelining Concept With Skew Core Key-Scheduling" JATIT Pub 2005 - 2009

[11] Rivest, R.; A. Shamir; L. Adleman (1978). "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems". Communications of the ACM 21 (2): 120–126.

[12] Modular Arithmetic for RSA Cryptography. [Courtesy:http://gtk.hopto.org:8089/MODULARRSA.PDF]

[13] RSA Encryption [Courtesy:http://www.geometer.org/mathcircles/RSA.PDF]

[14] U.S. Dept of Commerce/NIST, "Digital Signature Standard (DSS)", FIPS PUB 186-2, Jan.2000.

[15] N. Koblitz, "Elliptic curve cryptosystems", Mathematics of Computation, 48:203- 209, 1987.

[16] V. Miller, "Uses of elliptic curves in cryptography", Crypto 1985, LNCS218:Advances in Cryptology, Springer-Verlag, 1986.

[17] N. Smart, "How secure are elliptic curves over composite extension fields?", EUROCRYPT 2001, LNCS 2045 Springer-Verlag, pp. 30- 39, 2001.

[18] ANSI X9.63-199x: "Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport Using Elliptic Curve Cryptography. October, 1999. Working Draft.

[19] A. Lenstra and E. Verheul, "Selecting Cryptographic Key Sizes", Journal to Cryptology 14 (2001) pp. 255– 293, http:/www.cryptosavvy.com

[20] Alfred J. Menezes, Paul C. VanOorschot and Scott A.Vanstone, "Handbook of Applied Cryptography", CRC Press, 1996.

AUTHORS

First Author – Pushp Lata, M.tech Student, E & C Department,Dayananda Sagar College of Engineering, Bangalore, India, Email: pushplata@outlook.com, anithavijaya@gmail.com

Second Author – V. Anitha, Associate Professor, Dayananda Sagar College of Engineering, Bangalore, India