Analysis and Synthesis of RSA Algorithm using VHDL and Matlab

Ashwini Araballi, Zuhi Subedar

Assistant Professor& Electronics and Communication Department & VTU
Jain College of Engineering, Belagavi, Karnataka

***Abstract* —** *This paper represents the simulation and synthesis of RSA algorithm using MATLAB and VHDL. RSA is considered as one of the hardest algorithms as it is difficult to crack the standards used in the encryption for securely storing and transmitting the information. The analysis of this algorithm is carried out for the parameters like overall execution time, operating frequency and resource consumed during simulation. The result analysis depicts the comparison between RSA implementation using Matlab and VHDL.*

Keywords — *Cryptography, Encryption, Decryption, RSA, Security, Matlab, VHDL.*

I. INTRODUCTION

The amount and means of data transmitted over the transmission channel has changed a lot since the last few decades. Especially the information transmitted over the media electronically has grown fast and even growing exponentially [1][2]. As a result of this, it becomes very essential to find new ways for guaranteeing the information security during communication. An art of keeping the information secure is called cryptography and the basic model for the same is depicted in Fig. 1. This scheme helps the people to carry out their business routines electronically such as emails, e-commerce, ATM machines etc without the fear of deception and deceit. It not only assures message integrity but also ensures the authenticity of the sender [3].

In order to provide high performance security, it becomes necessary to use encryption schemes such as symmetric with private key encryption algorithms and asymmetric with public key encryption algorithms. A symmetric encryption algorithm makes use of one common key for both encryption and decryption at both sender and receiver sides. On the contrary, public key cryptosystem makes use of two keys namely private key and public key; one of which is used for encryption and other for decryption i.e two pairs of keys are generated for both the users.

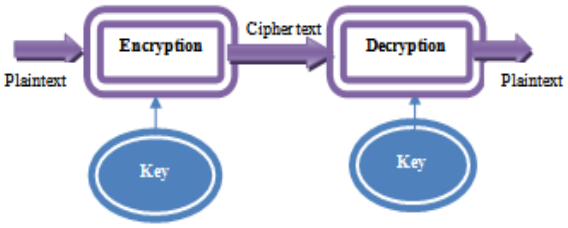

Fig 1: Cryptosystem

For example, consider two users Bob and Alice. Let [PR_a, PU_a] and [PR_b, PU_b] be the private keys and public keys of Alice and Bob respectively. There are two ways in which the Public Key Cryptography can be represented namely encryption with private key and encryption with public key. If Bob uses Alice's public key to encrypt the data to be transmitted, then at the receiver end, Alice can decrypt the message using its own private key and the scenario is depicted in Fig. 2. If Bob uses its private key to encrypt the message then Alice can decrypt the message using Bob's public key and the scenario is depicted in Fig. 3.

RSA algorithm is one of the public key cryptographic algorithms invented by Rivest, Adi Shamir and Leonard Adleman [5]. It is considered as a highly secured algorithm when sufficiently long keys are used such as 1024 bits and above [4]. The basic motto of this paper is to implement RSA algorithm in MATLAB and VHDL and analyse the performance of the same in both the cases using softwares/platforms.

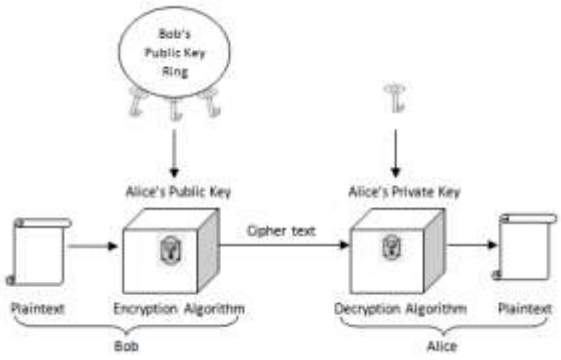

Fig 2: Encryption with Public Key

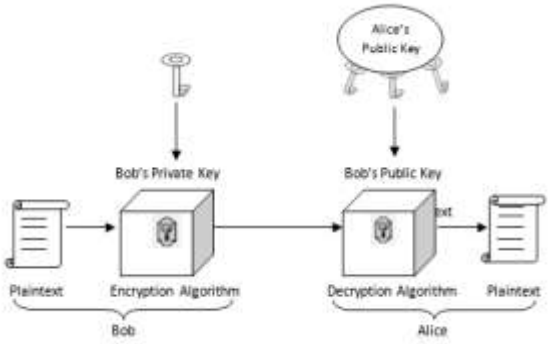

Fig 3: Encryption with Private Key

II. METHODOLOGY

A. *RSA Algorithm*

In 1977 Ron Rivest, Adi Shamir and Adleman proposed this method based on the Diffie Hellman key exchange algorithm. The difficulty in factorizing the large integers along with the use of sufficient length of the key assures highest security to this algorithm. Also, difficulty in guessing the prime numbers and even choosing large prime numbers makes factorization very difficult[6][7].
RSA structure combines three following elements:

- Key generation
- Encryption
- Decryption

a) Key Generation Unit

This module is used for the generation of the private key and public key that are to be used in Public Key C cryptographic algorithms. The prime numbers, denoted by p and q are multiplied by the multiplier and mod of the product is calculated. Then subtractor and the second multiplier block calculate Euler's totient function. GCD calculator checks whether e and ∅(n) are prime or not relatively. Another calculator, called multiplicative inverse is used to calculate d, which is nothing but private key [8].

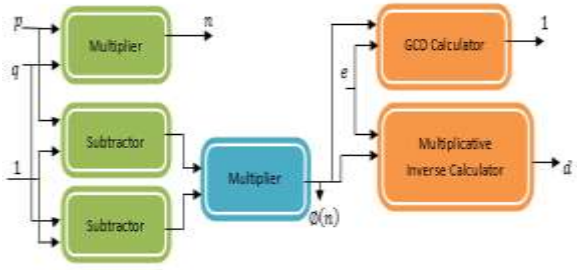

Fig 4: Key Generation

b) Encryption Unit

This unit makes use of modular exponentiation to perform encryption as shown in Figure 5. Modular exponentiation performs the mathematical operation to generate ciphertext. It incorporates encryption with public key cryptographic scheme [9][11].

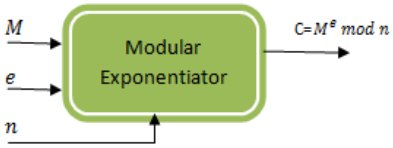

Fig 5: Encryption

c) Decryption Unit

This unit again is same as encryption unit as shown in Figure 6. The block does same mathematical operation as encryption unit i.e. modular exponentiation. It generates the plaintext back at the receiver end.

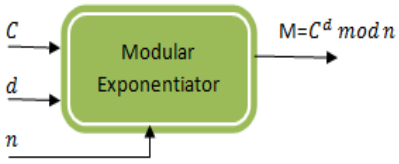

Fig 6: Decryption

TABLE I: STEPS OF RSA ALGORITHM

1	Select two prime numbers p and q
2	Calculate n=p * q
3	Calculate ∅(n)=(p-1)*(q-1)
4	Let (Public Key) e so that gcd (∅(n),e)=1
5	Calculate d*e=1 mod ∅(n) we get (private key)d
6	Compute ciphertext C=M ^{e} mod n for plaintext M
7	Compute plaintext M=C ^{d} mod n using ciphertext C

The above Table 1 depicts the overall steps included in RSA algorithm.

III. IMPLEMENTATION

The RSA algorithm is implemented in Matlab coding and in VHDL coding by using Matlab tool and Xilinx ISE 13.2.The parameters such as encryption decryption time i.e, Overall execution time, operating frequency and memory (CPU time) are taken for the analysis in both the cases.

Case i: MATLAB Implementation of RSA

In this case the RSA algorithm is analysed and simulated using Matlab. The parameters like, overall execution time which includes encryption and decryption time, the operating frequency and total memory used are analysed and the readings are tabulated as shown in Table 2. For some fixed length of prime numbers the factorization is a little simple process. But for random / larger lengths of primes the factorization becomes complex and consumes relatively more time compared to fixed ones. In-short, larger the prime number selected, the complex the factorization becomes.

TABLE II: PARAMETRIC RECORDS USING MATLAB FOR FIXED KEY LENGTH

Message Length (Bytes)	Total Execution Time (sec)		Operating Frequency (Hz)	Total Memory Usage (MBytes)
	Fixed Key length	Variable Key length		
2	0.0676	0.7489	14.7928	758
4	0.1149	0.8065	08.7032	758
8	0.2159	0.9298	04.6317	758
16	0.3125	1.1572	03.2000	759
32	0.3718	2.5313	02.6896	759

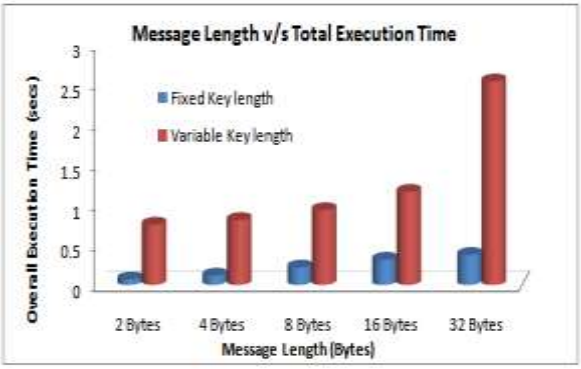

Figure 6: Plot of Message length v/s Execution time with fixed and variable key sizes

As the length of message increases, the time required for the encryption and decryption also increases as illustrated in Figure 6. However, frequency of operation decreases and memory consumption increases.

Case ii: Xilinx ISE 13.2

In this case, the RSA algorithm is analysed and synthesized using Xilinx ISE 13.2[10]. The parameters like overall execution time, operating frequency and total memory used are observed and the readings are represented in Table 3. As the size of message increases the time required for the encryption and decryption increases as illustrated in Figure 7, frequency of operation decreases and memory consumption increases. Here the execution time increases because of the Modular Exponentiator and multipliers during encryption and decryption processes. It follows radix -2 Montgomery multiplications.

TABLE III: PARAMETRIC RECORDS USING XILINX ISE (VHDL)

Message Length (Bytes)	Synthesis Report			
	Total Execution Time (sec)		Operating Frequency (Hz)	Total Memory Usage (Kbytes)
	Total Real Time	Total CPU Time		
2	11	10.95	0.0909	150000
4	11	11.01	0.0909	149872
8	11	10.90	0.0909	150768
16	12	11.40	0.0833	150001
32	10	10.08	0.1000	150448

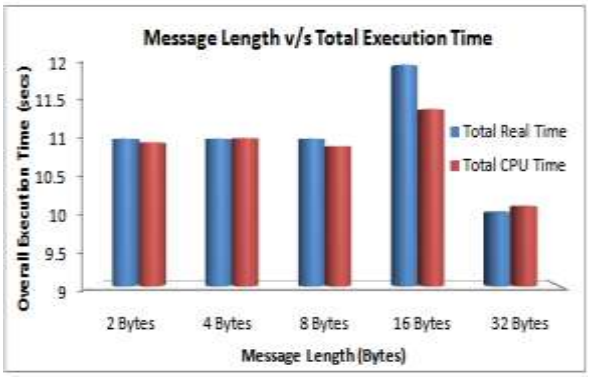

Figure 7: Plot for message length v/s Total execution time

From the above results it is concluded that the execution time taken for Matlab is very less as compared to VHDL and the analysis plot is represented by Figure 8. Similarly it is known to us that the operating frequency is inversely proportional to the execution time i.e, as the execution time decreases operating frequency increases. Also from Figure 9 it is concluded that the memory consumed in Matlab is very high as compared to VHDL synthesis report. Figure 10 depicts the simulation output of the algorithm with some initialization values.

Figure 8: Plot for Total execution time analysis

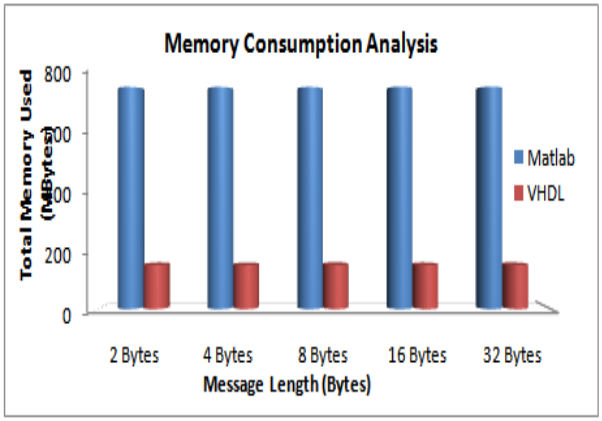

Figure 9: Plot for Memory used analysis

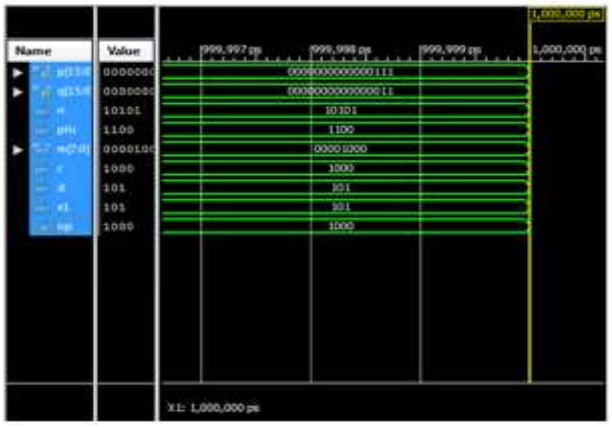

Figure 10: Snapshot of VHDL simulation output

IV. CONCLUSION

The Public key cryptographic algorithm RSA is synthesized and simulated successfully in MATLAB and XILINX ISE 13.2 using VHDL. From the analysis and synthesis reports it is concluded that Matlab execution of RSA takes very less time as compared to Xilinx because VHDL Xilinx makes use of modular exponentiation and multiplication blocks during encryption and decryption. Also the operating frequency is high as compared to Xilinx and the memory consumed by Matlab is high compared to Xilinx because the number of processes followed in the implementation of RSA is very lengthy. Almost 83% of the CPU space is consumed by Matlab and 17% is consumed by Xilinx. The speed of implementation of RSA in MATLAB is 97% and in case of Xilinx is only 3%.

REFERENCES

[1] Sushanta Kumar Sahu and Manoranjan Pradhan, "FPGA Implementation of RSA Encryption system", International Journal of Computer Applications, vol. 19, Issue 9, 2011.

[2] Ari Shawkat Tahir, "Design and Implementation of RSA Algorithm using FPGA", Research Gate, 2015.

[3] Sandeep Singh, Parminder Singh Jassal, "Synthesis and Analysis of 32-Bit RSA Algorithm Using VHDL", International Journal of Engineering Sciences, vol. 1, Jan. 2016.

[4] Ankit A. and Pushkar P, "Implementation of RSA Algorithm on FPGA", International Journal of Engineering Research & Technology (IJERT), ISSN: 2278-0181, vol. 1, Issue 5, 2012.

[5] Amit Thobbi, Shriniwas Dhage, Pritesh Jadhav and Akshay Chandrachood,"Implementation of RSA Encryption Algorithm on FPGA", American Journal of Engineering Research (AJER), vol. 4, Issue 6, pp-144-151,2015.

[6] Khaled Shehata, Hanady Hussien, Sara Yehia, "FPGA Implementation of RSA Encryption Algorithm for E-Passport Application", World Academy of Science, Engineering and Technology International Journal of Computer and Information Engineering, vol. 8, 2014.

[7] Gurpreet K. and Vishal A, "An Efficient Implementation of RSA Algorithm using FPGA and Big Prime Digit", International Journal of Computer & Communication Engineering Research (IJCCER), Volume 1 - Issue 4, 2013.

[8] M. A. Smadi, Qasem A. and Abdullah A, "Efficient FPGA Implementation of RSA Coprocessor Using Scalable Modules", International Symposium on Emerging Inter-networks, Communication and Mobility, Procedia Computer Science 34, pp. 647 – 654, 2014.

[9] Vibhor G, Aruna c, "Architectural analysis of RSA cryptosystem on FPGA ", International Journal of Computer Applications, vol. 26, 2011.

[10] Chiranth E, Chakravarthy H.V.A, Nagamohanareddy P, Umesh T.H, Chethan Kumar M,"Implementation of RSA Cryptosystem Using Verilog", International Journal of Scientific & Engineering Research, vol. 2, Issue 5, pp.1-7,2011.

[11] Amer, K.M, Sharif, S.M, Ashur, A.S., "Enhancement of hardware modular multiplier radix-4 algorithm for fast RSA cryptosystem," International Conference on Computing, Electrical and Electronics Engineering, vol. 1, pp. 692-696, Aug. 2013.

[12] S.Syamkumar Ch.Umasankar, "A Novel methodology for Implementation RSA algorithm using FFT in Galois Field", International Journal of Engineering Trends and Technology (IJETT), vol. 30, December 2015.

[13] Fathima Nizar, Fathima Latheef, Alfina Jamal, "RSA Based Encrypted Data Embedding Using APPM", International Journal of Engineering Trends and Technology (IJETT), vol. 9, Mar 2014.