Sandeep Singh[1,a], Parminder Singh Jassal[2,b]

Synthesis and Analysis of 32-Bit RSA Algorithm Using VHDL

Sandeep Singh[1,a], Parminder Singh Jassal[2,b]

[1]M.Tech Student, ECE section, Yadavindra collage of engineering, Talwandi Sabo, India
[2]Assistant professor, ECE section, Yadavindra collage of engineering, Talwandi Sabo, India
E-mail: [a]sandeepmaan294@email.com, [b]pammi_jassal@yahoo.co.in

Abstract. This paper presents the implementation of RSA algorithm design using VHDL. The Xilinx ISE 14.1 is used with device Spartan-3. In [1], the RSA encryption technique is implemented by using right to left binary radix-2 montmgomery multiplier. This paper presents the implementation of encryption technique by using left to right radix-2 montmgomery multiplier. The modular exponentiation is used for encryption and decryption of RSA algorithm. The device utilization is improved by 14%. The delay is improved by 2%. The frequency of the implementation is 79.546 MHz and is increased by 4.5%. Hence, the presented work is area and delay efficient than previous work.

Keywords: Encryption, RSA, FPGA, VHDL, Delay.

1. Introduction

Cryptography is playing a major role in data protection in applications running in a network environment. It allows people to do business electronically without worries of deceit and deception in addition to ensuring the integrity of the message and authenticity of the sender. It has become more critical to our day-to-day life because thousands of people interact electronically every day; through e-mail, e-commerce, ATM machines, cellular phones, etc. This geometric increase of information transmitted electronically has made increased reliance on cryptography and authentication by users.

RSA is one of the commonly used public key algorithms invented in 1977 by Ron Rivest, Adi Shamir and Leonard Adleman [1]. It supports encryption and digital signature. Its security is based on integer factorization problem of large numbers. It is commonly used in security protocols such as IP data security, transport data security, email security, terminal connection security, conferencing service security, etc.

The major time consuming arithmetic operations in RSA are modular computation and exponential computation. Programmable hardware structures, especially FPGA devices are useful for implementing the prototype because offer high performance hardware at a reduced cost in comparison with Application specific Integrated Circuits (ASIC). Currently, FPGA devices are composed from complex functional blocks, offering a variety of functions as multipliers, distributed RAM memory; block RAM, digital clock manager (DCM) and some structures contain even processor cores immersed [7]. In encryption applications, in offer good processing performance data that can change the algorithm in terms of hardware and the ability to be connected to high speed as peripheral devices with special function tot systems with microprocessors that don't have these features.

2. Related Work

There are some previous works related with the RSA algorithm:

In [1], the paper presented the RSA encryption technique implemented in FPGA of Spartan-3 type XC3S400. The RSA design showed that the algorithm was implemented with very small area requirements (practically 1211 number of slices, from reprogrammable structure). Compared with previous techniques (e.g. cellular automata ones), this method does not used any special features of the FPGA and thus can be implemented on any FPGA devices. As was presented, physical resources of FPGA used to implement RSA encryption system were very small, compared with the FPGA capacity, allowing further development of algorithms with higher encryption capacity.

In [2], the aim of the project was to development of high performance RSA encryption system. The proposed multiplier architecture achieved a significant improvement in performance. The encoded multiplier having only shift registers and adder circuits reduced the complexity, cost, power consumption and delay. The RSA Encryption algorithm using various multipliers was taken for comparison and the implementation using encoder multiplier was found to be 1.26 times faster than Vedic multiplier, 6.5 times faster than Booth multiplier and 8.2 times faster than Array multiplier. The usage of slice LUTs is also found to be more efficient with encoder multiplier which used 2% lesser slice LUTs compared to Vedic multiplier, 33% lesser slice LUTs compared to Booth multiplier and 42% lesser slice LUTs compared to Array multiplier.

In [3], this paper presented the modified algorithm for RSA with enhanced security. The security feature here was the elimination of n from the original RSA algorithm. Instead, the newly generated replacement for n can be used in both the keys. The RSA algorithm was prone to mathematical factorization attacks. The algorithm that was presented in this paper eliminates this issue making the algorithm more secure with a slight increase of time complexity.

3. Design of Algorithm

RSA algorithm is an asymmetric cipher which consists two parts.

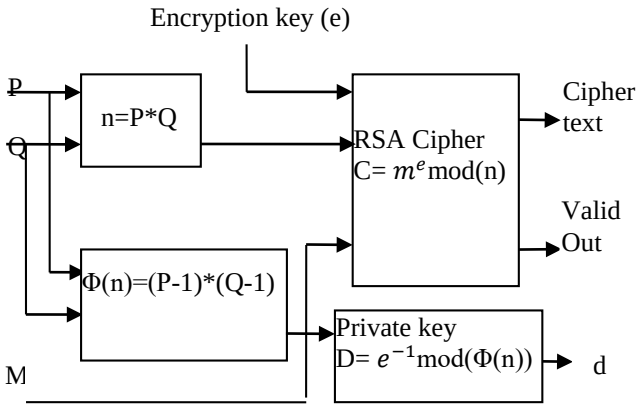

Fig. 1. Block diagram of RSA 32 bit encryption.

3.1. Key Expansion

- Select two random prime numbers p and q.
- Solve n=p*q.
- Solve Φ(n)=(p-1)*(q-1).
- Selecting at random the encryption key e,Where, 1<e<Φ(n) and gcd(e, Φ(n))=1.
- Solve d= e^{-1} mod Φ(n) and 0<d<n to find the decryption key d.
- Public encryption key: KU={e,n}.
- Private decryption key: KR={d,p*q}.

3.2. Data Encryption

To encrypt a message M, the sender should obtain public key of recipient KU={e,n}. Compute C=M^e mod n where 0<M<n.

3.3. Data Decryption

To decrypt the cipher test C, the private key KR={d,p,q}is used by the owner. Compute M=C^d mod n.

4. Previous Work

The flowchart for Right to left binary method is as follows:
The right to left binary method with modular exponential using Radix-2 Montmgomery product is used to implement the encryption technique of RSA algorithm.

$$C=M^e \bmod n \quad (1)$$

where e, n are the public keys, M is the message and C is the Cipher text.

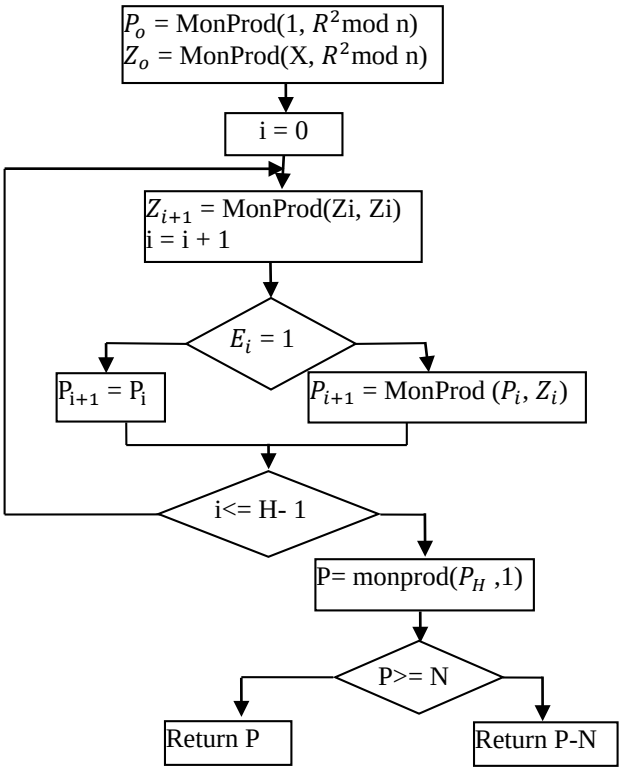

Fig. 2. Flowchart of right to left montgomary exponentiation [1]

In [1], RSA was implemented on FPGA spartan-3 XC3S400-FG456 using VHDL. This module was implemented and optimized for used area and not for the speed processing. The sequence of 1024bits with an exponent on 1024 bits is 212.99ms with a 50 MHz clock system, which means approximated of 208Kbps.

Table 1. Performance parameters.

Parameters	Values
Minimum Period	16.592 ns
Maximum Frequency	60.270 MHz
Minimum input arrival time before clock	7.363 ns

Maximum output required time before clock	13.609 ns

5. Presented Work

In this paper, the implementation of RSA algorithm using left to right binary method exponentiation with radix-2 montogmery multiplication is presented.
The ways to improve the square-and-multiply algorithm are to reduce the number of operations required. The number of squaring is fixed; the number of multiplicationscan be reduced. The square-and-multiply algorithm (left to right binary exponentiation) requires fewer multiplications than the Right to left binary exponentiation.

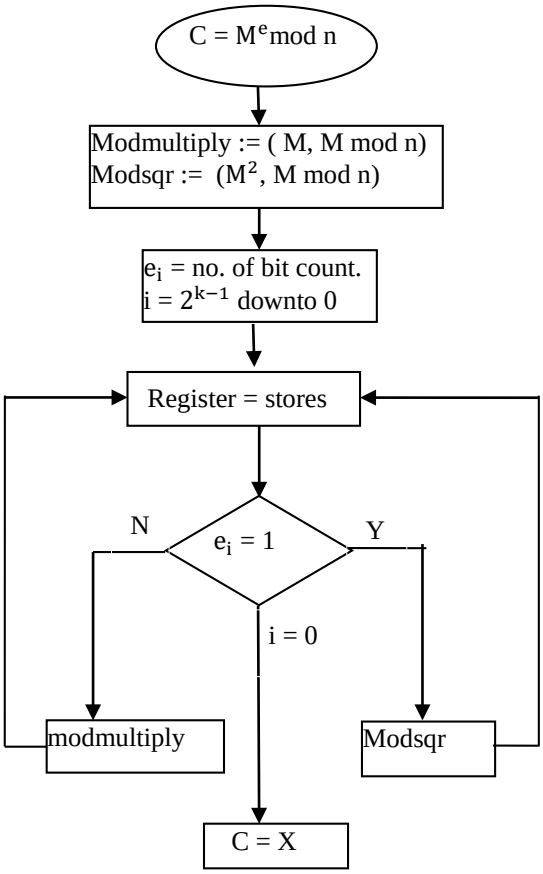

Fig. 3. Flowchart of Left to right binary method for exponentiation.

The Spartan-3 XC3S400-FG456 board is used for implementation of algorithm. Also, we improve the device utilization and delay.

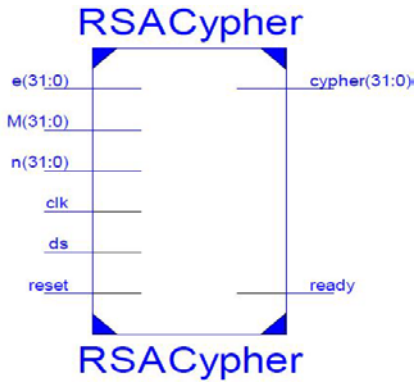

Fig. 4. RTL schematic for RSA Cypher.

Table 2. Device Utilization for 32 bit RSA algorithm.

Logic used	Iana et. al[1]		Our Work		
	Used	Percentage	Used	Percentage	Available
Slices	1211	33%	503	13%	3584
Slices Flip-flop	236	3%	459	3%	7168
4 input LUTs	2376	33%	936	14%	7168
Bounded IOB	135	51%	135	51%	264
FIFO 16	1	12%	1	12%	8
Frequency	60.270 MHz		79.546 MHz		

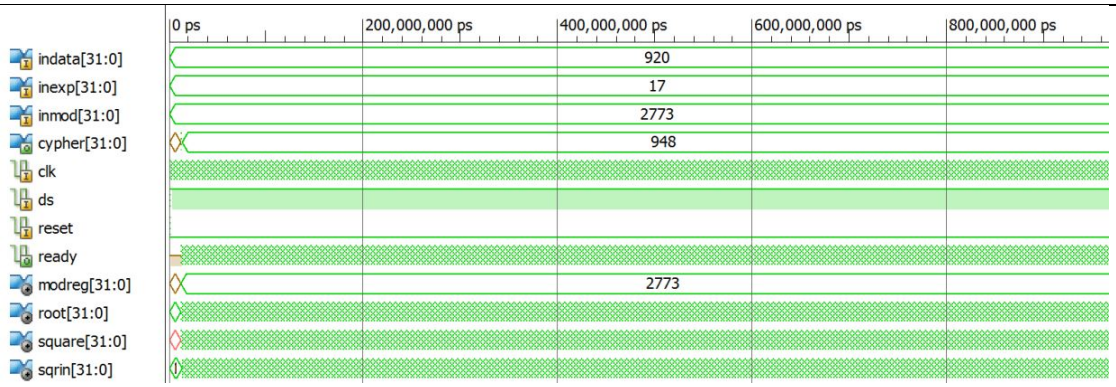

Fig. 5. Simulation Waveform of 32 bit RSA encryption

The simulation waveforms for decryption are shown in figure below:

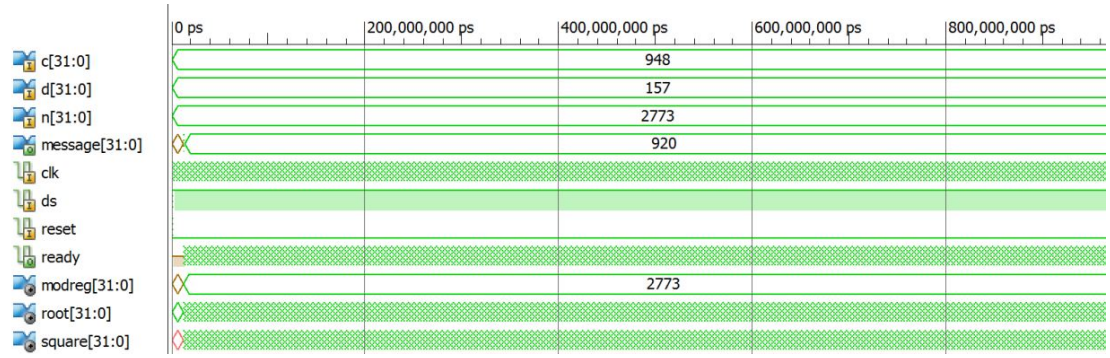

Fig. 6. Simulation waveform of 32 bit RSAdecryption.

6. Conclusion

RSA algorithm is synthesized and simulated successfully. The implementation of RSA is performed on Xilinx ISE 14.1 with device Spartan 3 using VHDL. The implemented results shows that RSA encryption using squaring and multiplication algorithm binary method left to right is more efficient than right to left binary method because it takes one less variable for operation. The left to right binary exponentiation method is faster than right to left binary method exponentiation method for combined square and multiply approach. The results are simulated and verified by using ISim simulator. The device utilization is improved and is compatible with the used device Spartan-3 XC3S400.
For future, it is recommended to use k-ary method rather than radix-2 which may be faster and more efficient processing with VHDL and better device can be used whose performance is better than Spartan-3 and maximum frequency will be obtained.

References

[1] Iana, G.V.; Anghelescu, P.; Serban, G., "RSA encryption algorithm implemented on FPGA," *International Conference on Applied Electronics*, vol. no. 1, pp.1-4, 7-8 Sept. 2013.
[2] George, D.; Bonifus, P.L., "RSA encryption system using encoded multiplier and vedic mathematics," *International Conference on Advanced Computing and Communication Systems*, vol. no 3, pp.1-4, 19-21 Dec. 2013.
[3] Minni, R.; Sultania, K.; Mishra, S.; Vincent, D.R., "An algorithm to enhance security in RSA," *International Conference on Computing, Communications and Networking Technologies*, vol. no. 4, pp.1-4, 4-6 July 2013.
[4] Oksuzoglu, E.; Savas, E., "Parametric, Secure and Compact Implementation of RSA on FPGA," *International Conference on Reconfigurable Computing and FPGAs*, vol. no. 1, pp.391-396, 3-5 Dec. 2008.
[5] R. L. Rivest. A. Shamir., L. Adleman. "A method for obtaining digital signatures and public-key cryptosystems" *Communications of the ACM*, vol. 21, no. 2, 12&126. 1978.
[6] Nibouche, O.; Nibouche, M.; Bouridane, A.; Belatreche, A., "Fast architectures for FPGA-based implementation of RSA encryption algorithm," *IEEE International Conference on Field-Programmable Technology*, vol. no. 1, pp.271-278, 6-8 Dec. 2004.
[7] Nibouche, O.; Nibouche, M.; Bouridane, A., "High speed FPGA implementation of RSA encryption algorithm," *IEEE International Conference on Electronics, Circuits and Systems*, vol.1, no. 10, pp.204-207, 14-17 Dec. 2003.
[8] Rahman, M.; Rokon, I.R.; Rahman, M., "Efficient hardware implementation of RSA cryptography," *International Conference on Anti-counterfeiting, Security, and Identification in Communication*, vol., no. 3, pp.316-319, 20-22 Aug. 2009.
[9] Sushanta Kumar, SahuManoranjan Pradhan, "FPGA Implementation of RSA Encryption System", *International Journal of Computer Applications*, Vol. 19, no. 9, pp 10-12, 2011.
[10] 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., no. 1, pp.692-696, 26-28 Aug. 2013.