ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 2, Issue 7, July 2013

RSA algorithm realization on FPGA

A.R.Landge[1], A.H. Ansari[2]
[1] P.D.V.V.P. CoE, Ahmednagar, Maharashtra,India
[2]P.R.E.C, Loni, Ahmednagar, Maharashtra,India.

ABSTRACT
The RSA algorithm is a secure, high quality, public key algorithm..The paper presents the architecture and modeling of RSA public key encryption/decryption systems. It is been observed that it is difficult to implement RSA algorithm on FPGA, as resources required are more than processors resource. This paper studies encryption & decryption module as individual module & investigates possibility of implementing individual module , in this case decryption on FPGA.

Key words
Decryption, FPGA , Public Key, RSA ,VHDL

1. INTRODUCTION

Cryptography is the study of methods for sending messages in secret (namely, in enciphered or disguised form) so that only the intended recipient can remove the disguise and read the message (or decipher it, thereby providing confidentiality). It is the art of using mathematics to address the issue of information security. Cryptography has, as its etymology *kryptos* from the Greek, meaning *hidden*, and *graphy*, meaning to *write.*

The original message is called the 'Plaintext' and the disguised message is called the 'Cipher text'. The final message, encapsulated and sent, is called a 'Cryptogram'. The process of transforming plaintext into cipher text is called 'Encryption' or 'Enciphering'. The reverse process of turning cipher text into plaintext, which is accomplished by the recipient who has the knowledge to remove the disguise, is called 'Decryption' or 'Deciphering'

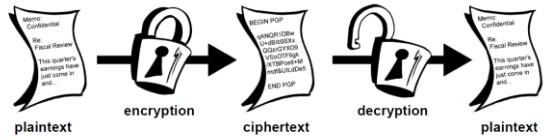

Figure 1 Encryption And Decryption

***Amol R .Landge[1]**, Department of Electronics and Telecommunication PDVVPCOE, Ahmednagar, Maharashtra, India*
***A. H. Ansari[2]**, Department of Electronics and Telecommunication, PREC Loni, Maharashtra, India*

Two types of cryptography are **private/secret/single key** cryptography & Public **key** cryptography.RSA is public key algorithm.
The RSA algorithm was publicly described in 1978 by Ron Rivest, Adi Shamir, and Leonard Adleman at MIT , MIT was granted U.S. Patent [4] for the algorithm in 1983. From the DWPI's abstract of the patent, The system includes a communications channel coupled to at least one terminal having an encoding device and to at least one terminal having a decoding device.

A message-to-be-transferred is enciphered to cipher text at the encoding terminal by encoding the message as a number M in a predetermined set. That number is then raised to a first predetermined power (associated with the intended receiver) and finally computed. The remainder or residue, C, is.computed when the exponentiated number is divided by the product of two predetermined prime numbers (associated with the intended receiver).

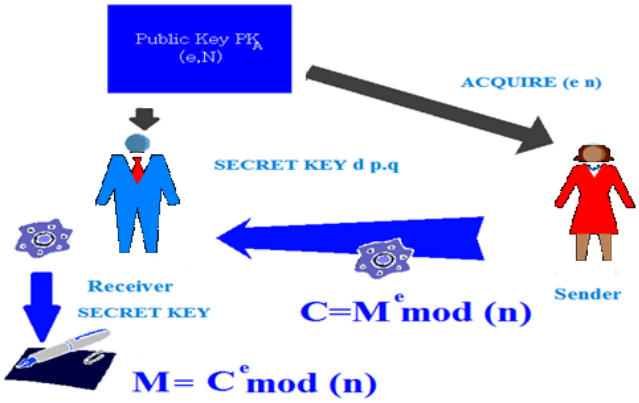

Figure 2 Basic working of RSA System

The RSA algorithm is a secure, high quality, public key algorithm. The RSA algorithm is a secure, high quality, public key algorithm. A hardware implementation of RSA.encryption scheme has been proposed by Deng Yuliang & Mao Zhigang. in [2], where they use Montgomery algorithm for modular multiplication. A similar approach has been taken by C. N. Zhang & Y. Xu. in [3]. J. Fry, and M. Langhammer [14] proposed method for low cost FPGA implementation o RSA.

This design scheme focuses on the implementation of a RSA cryptographic processor using Bit-Serial Systolic Algorithm. Sushanta Kumar Sahu & Manoranjan Pradhan[11] have used multiple key sizes for implementing on fpga using shift &add algorithm.

II RSA ALGORITHM – A OVERVIEW

The RSA algorithm is a secure, high quality, public key algorithm. Fig shows steps involved in key generation , encryption & decryption of system.

A) RSA Encryption

PLAIN TEXT : M < n

CIPHER TEXT : $C = \mathbf{M^e \bmod (n)}$

B) RSA Decryption

CIPHER TEXT: C

PLAIN TEXT: $M= \mathbf{C^d \bmod (n)}$

Figure 3 Encryption/Decryption Equation of RSA

Key Generation

Choose 2 large prime numbers, p & q

Compute n=p*q

Compute $\Phi(n) = (p-1) *(q-1)$.

Choose e, relatively prime to $\Phi(n)$.

Find d, such that e*d=1 mod $\Phi(n)$.

(e*d mod $\Phi(n)$ = 1 i.e.[(e*d) / $\Phi(n)$] reminder =1)

The *Public* key is *(n, e).*

The Private key is *(n, d),*

Figure 4 Key Generation steps of RSA Algorithm

RSA encryption and decryption [6] are mutual inverses and commutative due to symmetry in modular arithmetic as shown in fig. 3.

The process of transforming plaintext into cipher text is called 'Encryption' or 'Enciphering'. The reverse process of turning cipher text into plaintext, which is accomplished by the recipient who has the knowledge to remove the disguise, is called 'Decryption' or 'Deciphering'.

The RSA encryption/decryption is just a modular exponentiation operation. This mathematical operation is represented as C=Me mod n [5], where C is cipher text, M is plain text, e is [10] the public key exponent, and n is the modulus.

This operation has involved a few modular operations: modular [8], multiplication, modular addition, and subtraction.

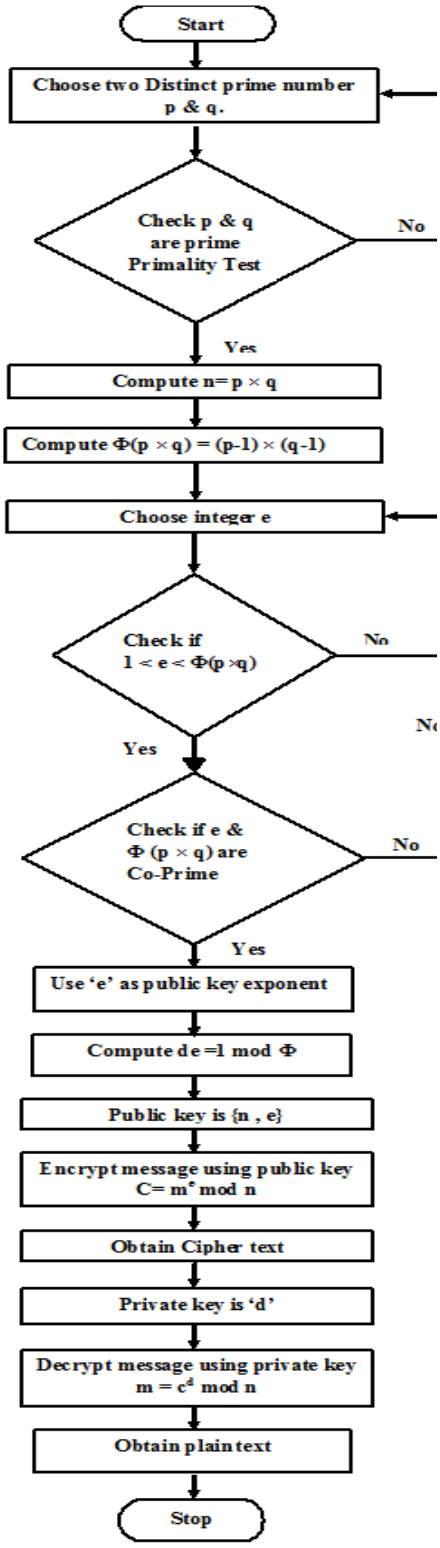

Figure 5 Flow chart of RSA Algorithm.

III. MODULUS MODULAR ADDITION

The modular addition[13]problem is defined as the computation of $S = A + B \pmod n$ given the integers A, B, and n **[7]**. It is usually assumed that A and B are positive [9] integers with $0 \leq A, B < n$. The most common method of computing S is as follows:

1. Compute $S = A + B$.
2. Then $S = S - n$.
3. If $S \geq 0$, then repeat step 2, else $S = S$.

Note that modular addition involves subtraction operation..Let's start with one of the simplest ciphers: general Caesar cipher. Its encryption and decryption operation can be represented using the following mathematical functions for ceasers cipher (uses alphabets & replaces alphabets from alphabets in other position so mod 26)

$$C = (P + K) \bmod 26$$
$$P = (C - K) \bmod 26$$

Table 1 *Addition Modulo 10*

P \ K	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	0
2	2	3	4	5	6	7	8	9	0	1
3	3	4	5	6	7	8	9	0	1	2
4	4	5	6	7	8	9	0	1	2	3
5	5	6	7	8	9	0	1	2	3	4
6	6	7	8	9	0	1	2	3	4	5
7	7	8	9	0	1	2	3	4	5	6
8	8	9	0	1	2	3	4	5	6	7
9	9	0	1	2	3	4	5	6	7	8

For simplicity, we replace 26 with 10, and show the general Caesar cipher, which is also the modular addition operation. the decryption operation, which subtracts the secret key K from cipher text C modulo 10, can also be done by adding K′, which is the additive inverse of K modulo 10.An additive modular inverse of K is the number which is added to K to get 0 after modular operation. For example, 4's inverse (modulo 10) is 6, because (4+6) mod 10 = 0. If the secret key [12] were 4, then to encrypt in general Caesar cipher, 4 is added to the plaintext; and to decrypt, 6 is added to the cipher text. Formally, we have

$$C = (P + K) \bmod 26$$
$$P = (C + K') \bmod 26$$

Where $K + K' \bmod 10 = 0$.

IV. MODULUS MODULAR MULTIPLICATION

The modular multiplication[13]problem is defined as the computation of $P = A \times B \pmod n$, given the integers *A*, *B*, and *n* [7]. It is usually assumed that *A* and *B* [9] are positive integers. With $0 \leq A, B < n$.The modulus multiplication operation is needed after the separation of exponentiation into a number of squaring and multiplication. For decryption, we can look for multiplicative inverse, and undo the multiplication by multiplying the cipher text by the multiplicative inverse of the key. Multiplicative inverse [13] of K, denoted by K−1, is the number by which you'd multiply K to get 1 in mod n. Formally, the cryptosystem[7] [9]can be represented as follows.

$$C = (P \cdot K) \bmod n$$

$$P = (C \cdot K{-}1) \bmod n$$

Where $K \cdot K{-}1 \bmod n = 1$

V. MODULUS MODULAR EXPONENTIATION

Now let's consider encryption and decryption using modular exponentiation operation.

$$C = (PK) \bmod n$$

$$P = (CK'') \bmod n$$

Where K″ is the exponentiative inverse of K.

Exponentiation is achieved by performing a number of squaring and multiplications. Given the integers *M*, *e*, and *n*, the *e* has to be changed to binary in order to start the algorithm to compute *Me*. There are two variations which depend on the direction by which the bits of *e* are scanned: Left-to-Right (LR) and Right-to- Left (RL). The LR binary method is more widely known which has been listed in pseudo codes .

Left-to-Right Method

Output $C = M^e$
(e contains h-bits)
1. if $e_{h-1} = 1$, *then C:=M else C:=1*
2. for i = h-2 down to 0
2a. C:= C x C
2b. if $e_i = 1$, *then C:= C x M*
3. return C

Figure 6 Algorithm for Modulus Exponentiation Operation

Let $e = 43 = 101011$. So the $h = 6$ (*e* contains 6 bits). Using Left-to-Right method, as $e5 = 1$, $C = M$ algorithm starts as the following table

Table 2.LR Method Of Computing Exponentiation

i	e_i	Step 2a	Step 2b
4	0	M^2	M^2
3	1	$(M^2)^2$	$M^4 \times M$
2	0	$(M^5)^2$	M^{10}
1	1	$(M^{10})^2$	$M^{20} \times M$
0	1	$(M^{21})^2$	$M^{42} \times M = M^{43}$

V. RESULT & DISCUSSIONS

The RTL schematic diagram for 32 bit decryption engine is shown in figure. The synthesis report for 32 bit decryption is given in Table .

Figure 7. RTL schematic for RSA decryption engine with 32 bits.

Table 3 Device utilization summary

	Available	Used	% of use
Device name	XC3S400		
Number of Slices	3584	518	14
Number of Slice Flip Flops	7168	459	6
Number of 4 input LUTS	7168	936	13
Number of IOS	132		
Number of bonded IOBS	141	132	93
Number of GCLKS	8	1	12

Table 4 HDL Synthesis Report (Timing Summary)

Speed Grade	-4
Minimum period	14.538 ns
Maximum Frequency	68.57 MHZ
Minimum input arrival time before clock	7.188 ns
Maximum output required time after clock	9.008 ns

VI. CONCLUSIONS

The VHDL code for RSA Decryption algorithm is developed block wise. Optimized and synthesized. VHDLcode for each block synthesized using Xilinx ISE 9.2 and verified for functionality. The maximum clock frequency is found to ne 68.57 MHz. As the device require less than 100% resources, decryption engine can be implemented in FPGA..

VII. REFERENCES

[1] SCHNEIER, B., 1996. Applied Cryptography: Protocols, Algorithms, and Source Code in C, John Wiley & Sons.
[2] Deng Y., Mao Z., and Ye Y.,. 1998. Implementation of RSA Crypto-Processor Based on Montgomery Algorithm.
[3] Zhang. C.N, Xu. Y and Wu. C., 1997. A Bit-Serial Systolic Algorithm and VLSI Implementation for RSA.
[4] Ron Rivest, Adi Shamir , Leonard Adleman "Cryptographic communications system and method", MIT , U.S. Patent 4,405,829 A, published on Sep 20, 1983.
[5] Rivest, R., Shamir, A., and Adleman, L, 1978. A Method for Obtaining Digital Signatures and Public Key Cryptosystems. Communications of the ACM.
[6] Selva Kumar M., Thamarai P., Arulselvi S Network Data Security Using FPGA. International Journal of Scientific Engineering and Technology (ISSN : 2277-1581) Volume 2 Issue 5, pp : 454-457 1 May 2013
[7] Jüri Põldre Cryptoprocessor PLD001, Department of Computer science Tallinn Technical University June 1998
[8] O. Prasanthi, M. Subba Reddy, RSA Algorithm Modular Multiplication *International Journal of Computer Applications in Engineering Sciences* ISSN: 2231-4946**]** *VOL II, ISSUE II, JUNE 2012]*
[9] C etin Kaya Koc , RSA Hardware Implementation , RSA Laboratories, RSA Data Security, Inc. Copyright c RSA Laboratories Version 1.0 August 1995
.
[10] Sushanta Kumar Sahu &Manoranjan Pradhan ,Implementation of Modular multiplication for RSA Algorithm,2011 International Conference on Communication Systems and Network Technologies, 978-0-7695-4437-3/11 © 2011 IEEE DOI 10.1109/CSNT.2011.30.
[11] Sushanta Kumar Sahu &Manoranjan Pradhan , 2011. FPGA Implementation of RSA Encryption System , International Journal of Computer Applications (0975 – 8887)Volume 19– No.9, April 2011
[12] Yuan Xue , Overview of Public-Key Cryptography and RSA
[13] Yuan Xue , RSA Algorithm
[14] J. Fry, and M. Langhammer, "FPGAs Lower cost for RSA Cryptography."

Amol R.Landge
BE(Electronics) from University of Pune , pursuing ME (VLSI & Embedded Systems) form PREC, loni, University of Pune, Presently working as an assistant professor at PDVVPCOE, Ahmednagar, Maharashtra, India His field of interest is VLSI & Embedded Systems.

Abdul .H.Ansari [2]
BE and ME from S.S.G.MCE , Shegaon, Amravati University has 16 years of teaching experience, presently working as Associate professor at PREC ,Loni. His field of interest is Wireless Comm. And Cognitive Radio.