Implementation of Multinomial Standard Product for RSA State Identify Algorithm

S. Surya Prasad Guttula, M.Tech student, KIET, kiet2ece@gmail.com
B. Ratna Raju, Assoc.Prof, KIET, ratna_ril@yahoo.com
Dr. G. N. Swamy, Professor, ECE, V.R. Siddhartha Engineering College, gavini_s@yahoo.com
Dr. K. Padma Raju, Professor, ECE, University College of Engg, Kakinada, padmaraju_k@yahoo.com

Abstract—This paper presents architecture and modeling of public key RSA encryption/decryption systems. The RSA (rivest-shamir-adleman) algorithm is a secure, high quality public key algorithm. Public key supports confidential and authentication methods. It supports multiple key sizes like 128 bits, 256 bits, 512 bits. RSA gets its security from factorization problem. The RSA encryption/decryption uses polynomial modular multiplication, it makes the processing time faster and used comparatively smaller amount of space in the FPGA due to its reusability each block is coded with verilog high description language. Verilog code is synthesized and simulated using Xilinx-ISE 12.2.

Keywords—RSA (encryption/decryption), Verilog, polynomial multiplication, FPGA.

I. INTRODUCTION
The art of keeping messages secure is Cryptography. Cryptography plays an important role in the security of data. It enables us to store sensitive information or transmit it across insecure networks so that unauthorized persons cannot read it. The urgency for secure exchange of digital data resulted in large quantities of different encryption algorithms (with private key algorithms) and asymmetric key algorithms (with public key algorithms) [1]. The asymmetric key algorithm requires two different key, one for encryption and another for decryption as shown in figure 1.

The RSA algorithm is a secure, high quality, public key algorithm. It can be used as a method of exchanging secret information such as keys and producing digital signatures. However, the RSA algorithm is very computationally intensive, operating on very large integers.

A vast numbers and wide varieties of works have been done on this particular field of hardware implementation of RSA encryption algorithm [2]. A hardware implementation of RSA encryption scheme has been proposed by Deng Yuliang & Mao zhigang. This design scheme focuses on the implementation of a RSA cryptographic processor using bit-serial systolic algorithm.

II. OVERVIEW OF RSA ALGORITHM
The most widely used public key cryptosystem is RSA.

The Plaintext is encrypted in blocks, with each block having a binary value less than some number n. That is, the block size must be less than or equal to \( \log_2(n) \); then block size is \( i \) bits, where \( 2^i < n < 2^{i+1} \).

Encryption and decryption are of the following form, for some plaintext block \( M \) and ciphertext block \( C \):

\[
\text{Encryption/Decryption}
\]

\[
\text{Plaintext block } M \text{ is encrypted to a ciphertext block } C \text{ by:}
C = M^e \mod n 
\]

\[
\text{The plaintext block is recovered by:}
M = C^d \mod n
\]
A public key encryption scheme has six ingredients
1. plaintext: this is the readable message or data that is
fed into the algorithm as input. 2. Encryption
algorithm: the encryption algorithm performs various
transformations on plaintext 3. public and private keys: this is a pair of keys that have been selected so
that if one is used for encryption, the other is used for
decryption. The exact transformations performed by
the algorithm depend on the public or private key that
is provided as input. 5. Cipher text: this is the
scrambled message produced as output. It depends on
the plaintext and the key. For a given message, two
different keys will produce two different cipher texts.
6. Decryption algorithm: This algorithm accepts the
cipher text and the matching key and produces the
original plaintext.

RSA encryption and decryption are mutual inverses
and commutative as shown in equation (1) and (2),
due to symmetry in modular arithmetic.

The mathematics involved in modular arithmetic
is as follows:

The integers ‘A’ and ‘B’ are congruent modulo m
if and only if  A-B is divisible by m. this congruence
is written as:

\[ a \equiv b \pmod{m} \]

Where m is a positive integer is called modulus.
When ‘A’ and ‘B’ are divided by m, the same
remainder is obtained.
The sign \( \equiv \) indicates congruence.

For examples: 14 \( \equiv \) 2mod 12.

III. CONFIDENTIALLY AND AUTHENTICATION

Suppose source A, if wanting to communicate
confidentially with dest. B, can encrypt a message
using B’s publicly available key. Such a

\[ C = M^e \pmod{n} \]

where \( e \) is chosen such that gcd(\( e, \phi(n) \)) = 1 for

certain modulus n = \( pq \) where \( \phi(n) = (p-1)(q-1) \)

\( k \) is the binary equivalent of \( e \) \( \phi(n) \) can
be calculated by using the algorithm as shown in fig 3.

Fig 3. Square and multiply algorithm

Let us take an example exponent (e) = (21)\(_{10}\)
Cipher text (C) = Message (M\(^{21}\))
Hence binary equivalent of (21)\(_{10}\) is 10101 (k=5)
Where k is number of bits of exponent
V. POLYNOMIAL MODULAR MULTIPLICATION

Multiplication results in a polynomial of degree greater than \( n - 1 \), then the polynomial is reduced modulo some irreducible polynomial \( m(x) \) of degree \( n \). That is, we divide by \( m(x) \) and keep the remainder. For a polynomial \( f(x) \), the remainder is expressed as \( r(x) = f(x) \mod m(x) \).

Computational considerations from [7]

A polynomial \( f(x) \) in \( \text{GF}(2^n) \)

\[ f(x) = a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \ldots + a_1x + a_0 = \sum_{i=0}^{n-1} a_i x^i \]

can be uniquely represented by its \( n \) binary coefficients \( (a_{n-1}, a_{n-2}, \ldots, a_0) \). Thus, every polynomial in \( \text{GF}(2^n) \) can be represented by an \( n \)-bit number.

Computing with polynomials in \( \text{GF}(q^n) \)

- Have seen arithmetic modulo a prime number \( \text{GF}(p) \).
- Also can do arithmetic modulo \( q \) over polynomials of degree \( n \) which also form a Galois Field \( \text{GF}(q^n) \). Its elements are polynomials of degree \( (n-1) \) or lower.

\[ a(x) = a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \ldots + a_1x + a_0 \]

- Have residues for polynomials just as four integers \( p(x) = q(x)d(x) + r(x) \). And this unique if \( \text{deg}(r(x)) < \text{deg}(d(x)) \). If \( r(x) = 0 \), then \( d(x) \) divides \( p(x) \), or is a factor of \( p(x) \) addition in \( \text{GF}(q^n) \) just involves summing equivalent terms in the polynomial modulo \( q \) (XOR if \( q = 2 \)).

\[ a(x) + b(x) = (a_{n-1} + b_{n-1})x^{n-1} + \ldots + (a_1 + b_1)x + (a_0 + b_0) \]

Multiplication with polynomials in \( \text{GF}(q^n) \).

- Multiplication in \( \text{GF}(q^n) \) involves multiplying the two polynomials together (cf longhand multiplication; here use shift & XORs if \( q = 2 \)).

Then finding the residue modulo a given irreducible polynomial of degree \( n \).

An irreducible polynomial \( d(x) \) is a prime polynomial; it has no polynomials divisors other than itself and 1.

Modulo reduction of \( p(x) \) consists of finding some \( r(x) \) st: \( p(x) = q(x)d(x) + r(x) \), nb. In \( \text{GF}(2^n) \) with \( d(x) = x^3 + x + 1 \) can do simply by replacing \( x^i \) with \( x^{i+1} \) e.g. in \( \text{GF}(2^2) \) there are 8 elements

\[ 0, 1, x, x+1, x^2, x^2+1, x^2+x, x^2+x+1 \]

VI. RESULTS & DISCUSSIONS

The RTL schematic diagram for 672 bit engine is shown in figure. The synthesis report for encryption/decryption is given in table by changing the generic parameter the RSA encryption module of different key size may be obtained.
Table 2: Device Utilization Summary

<table>
<thead>
<tr>
<th>Component</th>
<th>No's</th>
</tr>
</thead>
<tbody>
<tr>
<td>#multipliers</td>
<td>16</td>
</tr>
<tr>
<td>4×3-bit multiplier</td>
<td>16</td>
</tr>
<tr>
<td>#address/subtractors</td>
<td>464</td>
</tr>
<tr>
<td>42-bit adder</td>
<td>16</td>
</tr>
<tr>
<td>6-bit adder</td>
<td>336</td>
</tr>
<tr>
<td>6-bit subtractor</td>
<td>112</td>
</tr>
<tr>
<td>Counters</td>
<td>16</td>
</tr>
<tr>
<td>4-bit up counter</td>
<td>16</td>
</tr>
<tr>
<td>#registers</td>
<td>1376</td>
</tr>
<tr>
<td>Flip flops</td>
<td>1376</td>
</tr>
<tr>
<td>Multiplexers</td>
<td>16</td>
</tr>
<tr>
<td>42-bit 16 to 1 multiplexer</td>
<td>16</td>
</tr>
<tr>
<td>Logic shifters</td>
<td>16</td>
</tr>
<tr>
<td>42-bit shift logic left</td>
<td>16</td>
</tr>
</tbody>
</table>

Table 3: Timing Summary

<table>
<thead>
<tr>
<th>Speed grade</th>
<th>-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum period</td>
<td>26.611 ns</td>
</tr>
<tr>
<td>Maximum frequency</td>
<td>37.578 MHz</td>
</tr>
<tr>
<td>Minimum I/p arrival time before clock</td>
<td>27.462 ns</td>
</tr>
<tr>
<td>Maximum O/p required time after clock</td>
<td>5.129 ns</td>
</tr>
</tbody>
</table>

CONCLUSIONS

The Verilog code for RSA encryption/decryption algorithm is developed block wise. Optimized and synthesizable verilog code for each block synthesized using Xilinx ISE 12.2 and verified that functionally correct. The maximum clock frequency is found to be 37.578 MHz. Since the device require more than 100% resources, it is difficult to implement in FPGA.

REFERENCES