

# A Systolic Hardware Architecture for Montgomery Modular Multiplication for Secure Data Communication

V.Hendri Rajesh<sup>1</sup> M.Ravikumar<sup>2</sup>

<sup>1</sup>Student <sup>2</sup>Guide

<sup>1,2</sup>Department of ECE

<sup>1,2</sup>Mahendra Engineering College

**Abstract**— In modern communication systems, security is a main concern which is offered by cryptosystems Modular arithmetic is core operation in most of the cryptography systems. Most cryptosystems including RSA and DSA and ECC systems need modular multiplication for private key generation. That uses modular exponentiation function of large numbers to encrypt data, which is a slow process due to repeated modular multiplications. The efficiency of cryptography systems practically depends on how fast the multiplication is done. This is the base of computation. Most hardware and software designs for faster modular multiplication have been proposed, the Montgomery Multiplication Algorithm is recognized as the most efficient as others. This project that presents a 32 bit implementation of a Faster Montgomery algorithm for performing modular multiplication. The algorithm is based on the method proposed Montgomery for modular multiplication. The Simulation results shows that our design performs faster in terms of clock frequency while it requires lower Area and power.

area aware cryptosystem due to its limitation of battery power. For such case applications it is need to develop effective hardware architectures to carry out fast modular multiplications process with low energy consumption. And other the division operation in modular reduction is time consuming, Montgomery method proposed a new algorithm where division is skipped. A famous method to implement an architecture for modular multiplication is based on the Montgomery multiplication algorithm (MMA). It is efficient method for modular multiplication with an arbitrary modulus. The proposed algorithm uses simple divisions by a power of two instead of divisions by M, which is used in a conventional modular operation. Low-radix designs are usually more attractive for hardware implementation. so it's very useful for IoT system.

## A. Cryptography

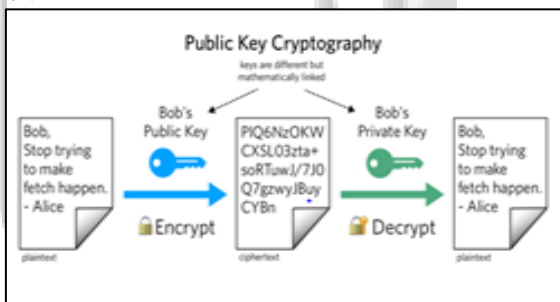
The word cryptography comes from the Greek term kryptos, meaning is hidden, and graphia, meaning is writing. Cryptography- literally means the art of secret writing. The art of hiding information therefore is not as modern as one might predict but is known to be some more than thousand years old. Cryptography gives, amongst others, means of hiding/recovering information called encryption schemes. In general an encryption scheme consists of a pair of encryption and decryption operations each associated with a key, which is to be kept secret. the relation between the two keys, an encryption scheme can be related to either main category of encryption: symmetric or public key encryption

## B. Symmetric encryption

An encryption technique is related to symmetric cryptography when it is easy to find the second key, knowing one of these. In most of time the two keys will be same which is said by the word symmetric and that the shared key is called to as secret key .the advantage of technique related to this category is their performance in terms of efficiency, as these schemes mostly uses Boolean operations, permutations and shift operations on bit or byte. The disadvantage of the system is that all subscribers involved in communication process have to share a common key. That's implicates more difficulties. First of all it means, that one can only communicate securely with another party, if the two have agreed or shared a symmetric key before. The actual act of sharing or agreeing on a key might be difficult, if you consider both parties on different for example, having in mind, that the key must not be disclosed to others during transfer.

## C. Public Key Encryption

An encryption method is said to be public key encryption, when it is impossible to compute the second key. In this encryption operation, using the encryption key, can be



**Keyword:** Systolic Hardware, Architecture, Montgomery Modular Multiplication

## I. INTRODUCTION

The rapid growth of information technology and internet of thinks such as e-commerce, basic security requirements for protecting data during digital transmission have become an important matter. Many of the systems rely on public-key cryptography to provide security services. Modular multiplication and modular division with big modulus are the basic operations in processing many public key cryptography. RSA (Rivest, Shamir, and Adleman) is one of the widely accepted public key algorithms at present. Rivest, Shamir, and Adleman (RSA) requires repeated modular multiplications to accomplish the computation of modular exponentiation. Processing these cryptosystems needs a large amount of computation and there is, therefore, a great demand for developing dedicated hardware to speed the computation speed. Additionally, security requirements are increasingly important for private data transmission through mobile devices with Internet access, such as smart phones and notebook computers, which require an energy efficient and

regarded as a trapdoor one way function, with the decryption key being the trapdoor that allows easy to message recovery. The Message recovery without knowledge of the decryption key is computationally impossible. An advantage of a scheme belonging to this category is the fact, that one cannot compute the decryption key knowing only the encryption key. In order to keep the decryption key secure, even though the encryption key is available in public, the encryption scheme needs to be more complex than a symmetric one. This delivers that the operations being performed become much complex and more time consuming. To get an idea of Public Key Encryption method one can imagine a simple mailbox. Anyone can put a letter into the mailbox (public encryption key), but only the owner of the mailbox's key can get the letters out of it (private decryption key).

#### D. RSA

RSA (Rivest–Shamir–Adleman) is an algorithm used by computers to encrypt and decrypt the messages. It is an asymmetric type cryptographic algorithm. Asymmetric means that have two different keys. This is called public key cryptography, because one of the keys can be given to anyone and other key must be keep in private. The algorithm is based on the finding the factors of a large composite number is difficult: when the factors are prime numbers, the problem is prime factorization. It is also a key pair of public and private key generator. The advantages include; RSA (Rivest–Shamir–Adleman) algorithm is safe and secure for its users through the use of complex mathematics. RSA (Rivest–Shamir–Adleman) algorithm is hard to crack since it involves factorization of prime numbers which are difficult to factorize. The disadvantages include; RSA algorithm can be very slow in cases where large data needs to be encrypted by the same computer. It requires a third party to verify the reliability of public keys.

### II. PROPOSED ARCHITECTURE DESIGN

#### A. Modular Arithmetic Used In (Rivest–Shamir–Adleman) RSA Cryptography

Cryptography is an essential for secure communication. It is practice and study the techniques for secure communication in the presence of third parties called adversaries and attackers. The rising growth of data communication and electronic transactions over the internet has made security to become the most important issue over the network. The Cryptography not only protects data from theft or alteration, but can also provide Privacy and confidentiality, user authentication, there are Two types of cryptographic schemes are used It includes secret key or symmetric cryptography (DES), public-key or asymmetric cryptography Rivest–Shamir–Adleman (RSA). The asymmetric key algorithm requires two different keys, one for encryption and other for decryption. The RSA algorithm is a highly secure, high quality, public key algorithm. It is used as a method of exchanging secret Information such as keys and producing digital signatures. It uses modular exponentiation of large numbers to encrypt data, which is a slow process due to it is used a repeated modular multiplications method. the Encryption is the process of converting a plain text into a (cipher) format which is not easily readable and is called as

cipher. The conversion from plain text to cipher text involved only mathematical operation.

#### B. Modular Multiplication Technique

Modular multiplication problem is defined as the computation of  $P = A \times B \pmod{m}$ , given the integers  $A$ ,  $B$  and  $m$ . It is usually assumed that  $A$  and  $B$  are positive integers with  $0 \leq A, B < m$ . in the square or multiplication operation is just a simple multiplication. There are many way to perform multiplication such as multiply then divide, interleaving multiplication and reduction, normally modular multiplication is done by repeated subtraction of modulus from the multiplicand until the result is smaller than modulus. This technique is time consuming when the value of modulus is too large. Modular Multiplication can be also performed by division of the modulus. This method take time since division is a time consuming task. Modular exponentiation ( $a \pmod{m}$ ) and its key constituent operation, modular multiplication ( $a \cdot b \pmod{m}$ ), are the fundamental process underlying cryptographic algorithms. Modular multiplications account for most of the time spent for encryption and decryption, their optimization is nice.

This is achieved either by reducing the number of modular multiplications or by reducing the latency of each modular multiplication. Modular exponentiation operation can further simplified in to series of modular multiplication and squaring operation.

### III. FASTER MONTGOMERY MUTLIPLIER

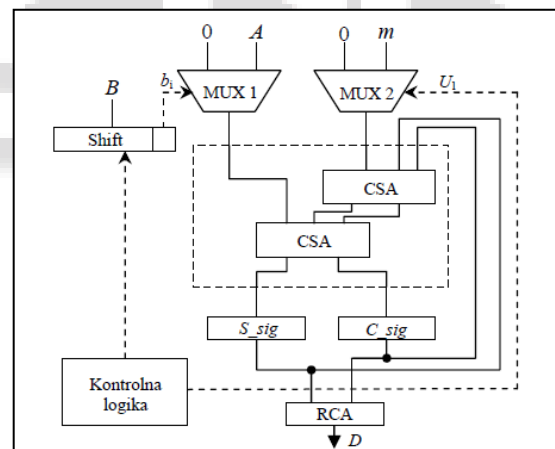


Fig. 1: Architecture of Montgomery modular multiplier with one bit scanning

```

Require:  $N = \sum_{i=0}^{m-1} (2^k)^i n_i, n_i \in \{0, 1, \dots, 2^k - 1\}$ 
 $B = \sum_{i=0}^{m-1} (2^k)^i b_i, b_i \in \{0, 1, \dots, 2^k - 1\}$ 
 $A = \sum_{i=0}^{m-1} (2^k)^i a_i, a_i \in \{0, 1, \dots, 2^k - 1\}, R = (2^k)^m$ 
 $A, B < 2N; N < R = 2^{km}; N' = -N^{-1} \pmod{2^k}.$ 
return  $S_{i+1} = ABR^{-1} \pmod{N}$ 
 $S_0 = 0$ 
For  $i = 0$  to  $m - 1$  do
 $q_i = ((S_0 + a_i \times b_0)N') \pmod{2^k}$ 
 $S_{i+1} = (S_i + q_i \times N + a_i \times B) / 2^k$ 
End for
    
```

Algorithm 1: Montgomery modular multiplication.

The proposed systolic architecture is based on the arithmetic operations of the Montgomery Algorithm, which are performed in a numerical base. The Montgomery Algorithm has additions and multiplications involving large integers that make use of multiple precision arithmetic operations. The new architecture is composed of Processing Elements distributed in a 1D. For ex: for a 1024bits modular multiplication with radix-32, the operands are split in 32 words of 32 bits which results in a 1D array of 32 Processing Elements. Between the Processing Elements, there is a propagation of carry signals which are the most significant bits of the arithmetic processes in each PE. The carry signals are processed by input parameters by the Processing Elements that receive them. In the new architecture, the Processing Elements are arranged by finite state machines.

#### A. Field Programmable Gate Arrays

Field programmable gate arrays (FPGAs) have a flexible, regular, programmable architecture of configurable logic blocks, programmable input/output blocks. These may be other configurable blocks such as delay locked loops, synchronous block Random Access Memorys. These functional elements are interconnected via routing channels. The FPGA inner parts are slices that contain programmable look up tables (LUT) and flip flops. Tables are used for combinational logic and flip flops are used for sequential logic. Most FPGAs are customized by loading configuration data into internal static memory cells. Endless reprogramming cycles are possible with this method. The Stored values of these cells determine the logic functions and interconnections implemented in the Field programmable gate arrays (FPGAs).

### III. RESULTS

The result shows the FPGA synthesis results of proposed modular multiplication architectures. The designs were described in hardware description languages (VHDL) and synthesized for Virtex-5 Xilinx FPGAs. All results are post implementation, and no area or speed optimizations were set for the synthesis. The results shown in this paper are improvements when compared with our previous work .The modified multiplexed architecture is implemented with a reduced number of slices registers. The synthesis for the systolic architecture presented high clock frequencies. Since the modular exponentiation is performed by successive modular multiplication executions, the left-to-right (MSB) binary square and multiply algorithm was managed in the modular exponentiation. The results show that, considering the no of clock cycles for a modular multiplication execution, the multiplexed architecture is very faster than the systolic implementation. On the other hand, the systolic architecture has a clock frequency higher than the clock frequency presented by the multiplexed architecture. Various techniques are available for performing modular multiplication was compared. The total time consuming division operation has been removed. It is seen that a Montgomery multiplier performs very fast modular multiplication. Also the power consumption of the circuit is lower in a Montgomery multiplier. The new method has a

simple structure and requires a small amount of pre computation and storage. It reduces the number of necessary additions. The possible values are stored in a lookup-table (LUT). The new Montgomery multiplier consumes less area and also less power.

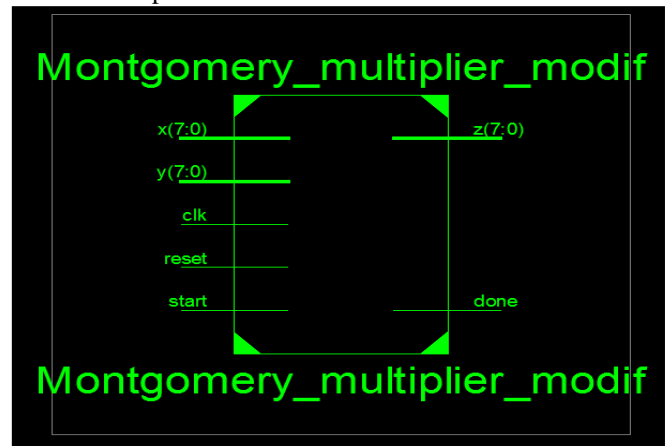


Fig. 2: RTL View

Multiplier	Area	Delay
Normal Montgomery multiplier	Slice-69/LUT-270/ IOB-195	47.702ns
Faster Montgomery multiplier	Slice-81/LUT-227/ IOB-131	32.55

Table 1 : Area & Delay Comparison

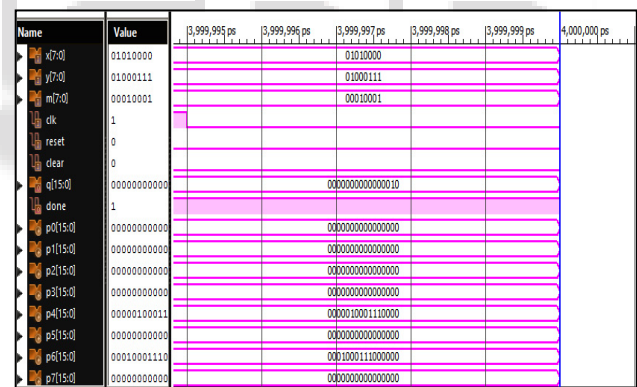


Fig. 3: simulation output waveform.

### IV. CONCLUSION

The estimated total circuit area and the critical path delay of the modular multiplier based Montgomery algorithm show that it can be implemented in smaller hardware than that necessary to implement multiplier and divider separately. I conclude that the various algorithms proposed in literature for calculating modular multiplication, the Montgomery modular multiplication algorithm seems to be the suitable one to be combined. This project presented an efficient algorithm to reduce the energy consumption and enhance the throughput of Montgomery modular multipliers simultaneously. The work can be further developed to modular exponentiation and squaring. A reversible architecture can be implemented for lower power consumption.

#### REFERENCES

- [1] Kuang, Shiann-Rong, et al. "Energy-efficient high-throughput Montgomery modular multipliers for RSA cryptosystems." *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on* 21.11 (2013): 1999-2009.
- [2] Knezevic, Miroslav, Frederik Vercauteren, and Ingrid Verbauwhede. "Faster interleaved modular multiplication based on Barrett and Montgomery reduction methods." *Computers, IEEE Transactions on* 59.12 (2010): 1715-1721.
- [3] Cho, Koon-Shik, Je-Hyuk Ryu, and Jun-Dong Cho. "High-speed modular multiplication algorithm for RSA cryptosystem." *Industrial Electronics Society, 2001. IECON'01. The 27th Annual Conference of the IEEE*. Vol. 1. IEEE, 2001.
- [4] Shieh, Ming-Der, et al. "A new algorithm for high-speed modular multiplication design." *Circuits and Systems I: Regular Papers, IEEE Transactions on* 56.9 (2009). 12) A. Royo, J. Moran, J. C. Lopez. Design and implementation of a coprocessor for cryptography applications. In European Design and Test Conference, pages 213–217, Paris, France, March 1–20 1997.
- [5] L. Adleman, R.L. Rivest, A. Shamir "A method for obtaining digital signature and public-key cryptosystems," *Comm. ACM*, vol. 21, no. 2, pages. 120–126, February 1978.
- [6] C. Paar, T. Blum, "Montgomery modular exponentiation on reconfigurable hardware" in *IEEE 14th Symposium on Computer Arithmetic*, pages. 70–77, IEEE Computer Society Press, Los Alamitos, CA, 1999.

