A Review on Implementation of RSA Cryptosystem Using Ancient Indian Vedic Mathematics

Shahina M. Salim
M.E. Student, Department of Electronics and Tele-communication,
G. H. Raisoni C.O.E.T, S.G.B. Amravati University,
Amravati(Maharashtra State),India.
shahina.salim786@gmail.com

S. A. Lakhotiya
Professor, Department of Electronics and Tele-communication,
Amravati(Maharashtra State),India
sonal.lakhotiya@raisoni.net

***Abstract*—** RSA is one of the most safest standard algorithm based on public key, for providing security in network. The hierarchical overlay multiplier is used in RSA circuitry for multiplication operation. The most significant aspect is the development of division architecture based on Ancient Indian Vedic Mathematics and embedding it in RSA encryption/decryption circuitry for improved efficiency. Typically, modular-multiplication algorithm is used since no trial division is necessary, and the carry-save addition (CSA) is employed to reduce the critical path. The implementation of RSA encryption/decryption algorithm using the algorithm of Ancient Indian Vedic Mathematics that have been modified to improve performance. RSA circuitry implemented using vedic multiplication is efficient in terms of area, speed compared to its implementation using conventional multiplication.

***Keywords*—** *RSA Cryptosystem, Modular Multiplication,, Modular exponentiation, Vedic Mathematics, FPGA, VHDL.*

*****

I. INTRODUCTION

The word 'Vedic' is derived from the word 'veda' which means the store-house of all knowledge. Vedic mathematics is mainly based on 16 Sutras (or aphorisms) dealing with various branches of mathematics like arithmetic, algebra, geometry etc.

The standard techniques for providing privacy and security in data networks include encryption/decryption algorithms such as Advanced Encryption System (AES) (private-key) and RSA (public- key). Rivest–Shamir–Adleman (RSA) is one of the most widely preferred algorithms used in public-key cryptography systems. RSA is one of the safest standard algorithms, based on public-key, for providing security in networks. RSA has a very slow ciphering rate if used in software. Security has become an increasingly important feature with the growth of electronic communication. The development of public-key cryptography (PKC) is the greatest and perhaps the only true revolution in the entire history of cryptography . Many PKC algorithms such as Rivest–Shamir–Adleman (RSA) algorithm and Diffie–Hellman algorithm have been proposed. PKC is asymmetric involving the use of two separate keys, in contrast to symmetric conventional encryption, which uses only a single key. The use of two keys provides solution to key management and user authentication in a cryptosystem.

RSA algorithm is the best known, the most versatile and widely used public key algorithm today RSA depends on the modular exponentiation of long integers, which is the critical operation for a variety of the most widely accepted cryptosystems . Therefore, fast modular multiplication becomes the key to real-time encryption and decryption since a high throughput is needed in data communication. The most widely used algorithm for efficient modular multiplication is Montgomery's algorithm. The binary Montgomery's modular-multiplication algorithm employs only simple addition, subtraction, and shift operation to avoid trial division, a critical and time-consuming operation in conventional modular multiplication. The modular exponentiation is usually accomplished by performing repeated modular multiplications.

II. LITERATURE REVIEW

Sriraman, L, Kumar K.S, Prabakar, T.N [1] presented Vedic mathematics is one of the ancient Indian mathematics which contains sixteen sutras. These sutras can be used to solve problems in any branch of Mathematics in a faster way. The proposed squarer is based on sutra called Ekadhikena Purvena. It means that "one more than the previous". This sutra is used for finding the square of decimal numbers ending with `5'. In this paper this sutra is generalized and used for squaring of binary numbers.

Gustavo D. Sutter, Jean-Pierre Deschamps, and José Luis Imaña [2] presented Modular exponentiation with large modulus and exponent, which is usually accomplished by repeated modular multiplications, has been widely used in public key cryptosystems. Typically, the Montgomery's modular-multiplication algorithm is used since no trial division is necessary, and the carry–save addition (CSA) is employed to reduce the critical path. In this paper, we optimize the Montgomery's multiplication and propose architectures to perform the least significant bit first and the most significant bit first algorithm [3],[4],[5].

 ISSN: 2321-8169
162 - 164

Xiaoming Tang [6] presented A certain range of real number presented by ASCII code is converted to single precision floating-point by pipeline processing with VHDL language. Through functional simulation and download verification, the conversion time is about 10 us when the clock is 50 MHz .

Huddar, S.R. , Rupanagudi, S.R. , Kalpana, M. , Mohan, S. [7] presented With the advent of new technology in the fields of VLSI and communication, there is also an ever growing demand for high speed processing and low area design. It is also a well known fact that the multiplier unit forms an integral part of processor design. Due to this regard, high speed multiplier architectures become the need of the day. In this paper, we introduce a novel architecture to perform high speed multiplication using ancient Vedic maths techniques.

Jaina, D, Sethi, K. , Panda, R. [8] presented Real-time signal processing requires high speed and high throughput Multiplier-Accumulator (MAC) unit that consumes low power, which is always a key to achieve a high performance digital signal processing system. In this paper, design of MAC unit is proposed. The multiplier used inside the MAC unit is based on the Sutra "Urdhva Tiryagbhyam" (Vertically and Cross wise) which is one of the Sutras of Vedic mathematics. Vedic mathematics is mainly based on sixteen Sutras and was rediscovered in early twentieth century. In ancient India, this Sutra was traditionally used for decimal number multiplications within less time. The same concept is applied for multiplication of binary numbers to make it useful in the digital hardware.

G.P. Saggese , L. Romano[9] presented An accelerator which can effectively improve the security and the performance of virtually any RSA cryptographic application. The accelerator integrates two crucial security- and performance enhancing facilities: an RSA processor and an RSA key-store. An RSA processor is a dedicated hardware block which executes the RSA algorithm. An RSA key-store is a dedicated device for securely storing RSA key-pairs. We chose RSA since it is by far the most widely adopted standard in public key cryptography[10].

III. PROPOSED WORK

The RSA Algorithm is based on the mathematical fact that it is easy to find and multiply the large prime numbers together, but it is extremely difficult to factor their product. The public and private keys in RSA are based on very large prime numbers[10]. The algorithm is simple but the complexity lies in the selection and generation of public and private keys. The algorithm steps are as follows:

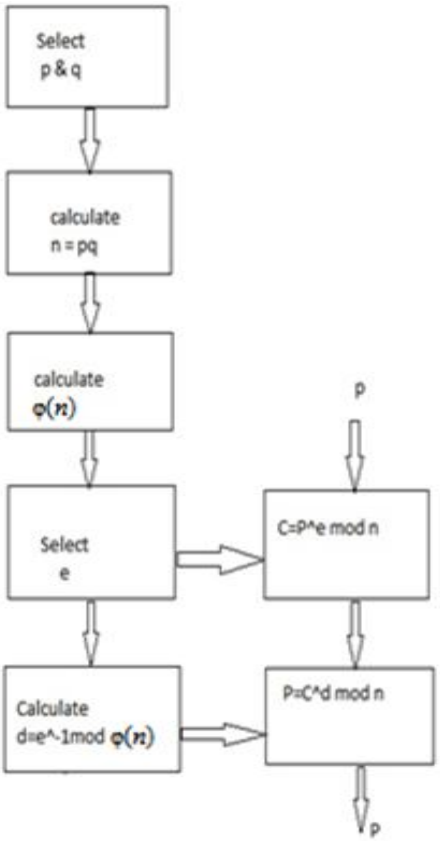

Fig: Flow chart of RSA algorithm

The RSA algorithm involves three steps: key generation, encryption and decryption.

1. *Key generation :*

RSA involves a public key and a private key. The public key can be known by everyone and is used for encrypting messages. Messages encrypted with the public key can only be decrypted in a reasonable amount of time using the private key[11]. The keys for the RSA algorithm are generated the following way:

1. Choose two distinct prime numbers p and q.
- For security purposes, the integers p and q should be chosen at random, and should be of similar bit-length. Prime integers can be efficiently found using a primality test.
2. Compute n = pq.
- n is used as the modulus for both the public and private keys. Its length, usually expressed in bits, is the key length.
3. Compute $\varphi(n) = \varphi(p)\varphi(q) = (p-1)(q-1) = n - (p + q -1)$, where φ is Euler's totient function.
4. Choose an integer e such that $1 < e < \varphi(n)$ and $\gcd(e, \varphi(n)) = 1$; i.e., e and φ(n) are coprime.
- e is released as the public key exponent.
- e having a short bit-length and small Hamming weight results in more efficient encryption – most commonly $2^{16} + 1 = 65,537$. However, much smaller

values of e (such as 3) have been shown to be less secure in some settings.

5. Determine d as $d \equiv e^{-1} \pmod{\varphi(n)}$; i.e., d is the multiplicative inverse of e (modulo φ(n)
- This is more clearly stated as: solve for d given $d \cdot e \equiv 1 \pmod{\varphi(n)}$
- This is often computed using the extended Euclidean algorithm. Using the pseudocode in the Modular integers section, inputs a and n correspond to e and φ(n), respectively.
- d is kept as the private key exponent.

The public key consists of the modulus n and the public (or encryption) exponent e. The private key consists of the modulus n and the private (or decryption) exponent d, which must be kept secret. p, q, and φ(n) must also be kept secret because they can be used to calculate d.

2. *Encryption :*

A transmits its public key (n, e) to B and keeps the private key d secret. B then wishes to send message P to A , then computes the ciphertext C corresponding to

$$C = P^e \pmod{n}$$

This can be done efficiently, even for 500-bit numbers, using Modular exponentiation. B then transmits C to A[10].

3. *Decryption :*

A can recover P from C by using its private key exponent d via computing

$$P = C^d \pmod{n}$$

Thus we get the original message.

IV. CONCLUSION

The RSA encryption/decryption system is implemented using the Vedic Mathematics algorithm to increase its computation speed. The advantage of the Vedic multiplier is that it calculates the partial products in one single step and there are no shift operations which saves the time and the hardware. As the number of message bits increases the gate delay as well as the area increase slowly. Hence it can be used effectively in all the cryptographic applications. It is found that this design is quite efficient in terms of silicon area and speed and should result in substantial savings of resources in hardware when used for crypto and security applications.

REFERENCES

[1] Sriraman, L. Dept. of Electron. & Commun. Eng., Oxford Eng. Coll., Trichy, India ; Kumar, K.S. ; Prabakar, T.N ,“ Design and FPGA implementation of binary squarer using Vedic mathematics” *IEEE Trans. Ind. Electron.*, July 2013.

[2] Gustavo D. Sutter, *Member, IEEE*, Jean-Pierre Deschamps, and José Luis Imaña ,“ Modular Multiplication and Exponentiation Architectures for Fast RSA Cryptosystem Based on Digit Serial Computation” *IEEE Trans. Ind. Electron.*, vol. 57, no. 10, pp. 3308–3316, Oct. 2010.

[3] E. Monmasson and M. N. Cirstea, “FPGA design methodology for industrial control systems—A review,” *IEEE Trans. Ind. Electron.*, vol. 54, no. 4, pp. 1824–1842, Aug. 2007.

[4] J. J. Rodriguez-Andina, M. J. Moure, and M. D. Valdes, “Features, design tools, and application domains of FPGAs,” *IEEE Trans. Ind. Electron.*, vol. 54, no. 4, pp. 1810–1823, Aug. 2007.

[5] R. L. Rivest, A. Shamir, and L. Adleman, “A method for obtaining digital signatures and public-key cryptosystems,” Commun. *ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978.

[6] Xiaoming Tang ; Res. Inst. of Inf. Fusion, Naval Aeronaut. & Astronaut. Univ., Yantai, China; Tao Zhang ; Zhenjie Wang ; Wenliang Yuan, “A novel data format conversion method based on FPGA” *IEEE Trans. Ind. Electron* ,July 2011.

[7] Huddar, S.R. ; WorldServe Educ., Bangalore, India ; Rupanagudi, S.R. ; Kalpana, M. ; Mohan, S. , “Novel high speed vedic mathematics multiplier using compressors” *IEEE Trans. Ind. Electron* ,March 2013.

[8] Jaina, D. ; Dept. of Electron. & Telecommun. Eng., VSS Univ. of Technol., Burla, India ; Sethi, K. ; Panda, R, “Vedic Mathematics Based Multiply Accumulate Unit” *IEEE Trans. Ind. Electron* ,Oct. 2011.

[9] G.P. Saggese a, L. Romano a,*, N. Mazzocca b, A. Mazzeo, “A tamper resistant hardware accelerator for RSA cryptographic applications, Journal of Systems Architecture 50 (2004) 711–727 ”.

[10] William Stallings, “ Cryptography and Nework Security”, Third Edition, Pearson Education, 2003

[11] S.E. Eldridge, C.D. Walter, Hardware Implementation of Montgomery’s modular multiplication algorithm, IEEE Trans. Comput. 42 (6) (1993) 693–699.

[12] A.Z. Alkar, R. So¨ nmez, An ASIC Implementation of the RSA Algorithm 18th MUG International Conference, February 2002.

[13] Sumit Vaidya, Deepak Dandekar, “Delay-Power Performance Comparison of multipliers in VLSI circuit design”, *International Journal of Computer Networks & Communications* (IJCNC), Vol.2, No.4, July 2010.