IMPLEMENTATION OF MONTGOMERY MODULAR MULTIPLICATION USING HIGH SPEED MULTIPLIER

Ankush Yete[1], Ananya Kajava P[2], Hazel Melita Rodrigues[3], Namratha P[4], Kiran Kumar V.G[5]
[1,2,3,4]UG Student, [5]Associate Professor
Department of E & C, Sahyadri College of engineering and Management, India

Abstract

Many algorithms for Public Key Cryptography, such as RSA, Elliptic Curve Cryptography and Diffie-Hellman are used for secure communication. These algorithms mainly use modular exponentiation as their basic operation. Modular exponentiation involves modular multiplication to be done repeatedly which is very costly for computations as there are large operands used. Therefore, time taken for computing is very large. Therefore, this computation time can be reduced immensely by Montgomery multiplication algorithm which is the most efficient modular multiplication algorithm available. It replaces the division by the modulus with a series of additions and divisions by numbers which are a power of 2 which is easier to compute. Montgomery multiplication algorithm involves three basic steps. 1. Conversion of operands to Montgomery domain. 2. Multiplication of operands. 3. Conversion of operands back to integer domain. The speed of the system mainly depends on multipliers and adder used in the system. Therefore, for multiplier architecture we have used Vedic multiplication which is faster than the other methods such as booth algorithm and array multiplier.

Index terms- Cryptography, Montgomery modular multiplication, Vedic multipliers, Euclid's algorithm.

I. INTRODUCTION

Cryptography is the study of ciphers. Ciphers are the encrypted plain text. Different aspects in information security such as data integrity, data confidentiality, non-repudiation and authentication are most important to modern cryptography. Modern cryptography includes the concept of computer science, mathematics and electrical engineering. Cryptography has many applications in military communications, electronic commerce, secure information transmission and computer passwords. Symmetric encryption and asymmetric encryptions are the two major types of encryption.

In asymmetric ciphers two different keys are used for encryption and decryption of data. The keys are named as public and private key. Asymmetric encryption transforms plaintext into cipher text using one of two keys and an encryption algorithm. Using decryption algorithm and the shared key, the plaintext is obtained from the cipher text. The most widely used public-key cryptosystem is RSA. Modular exponentiation is a primary operation in RSA public-key cryptography. exponentiation, can be converted into Montgomery form with simple computations and in short period of time. Montgomery multiplication is faster than the available alternatives. Many known cryptosystems like RSA and Diffie–Hellman key exchange are based on modulo arithmetic operations on a large number, and for these cryptosystems, to increase the computation performances of this operation the Montgomery algorithm is used, which replaces the division and mod function. Montgomery multiplication is a method for computing a x b mod p for positive integers a, b and p. It replaces the division operation by the modulus operation with a number of additions and divisions by a power of 2. Therefore, it is suitable for hardware implementations. This is the reason why this

algorithm forms the basis for many of the RSA based hardware architectures.
There are number of techniques proposed to avoid carry propagation in the addition stages of the algorithm as they are a very important in determining the performance of the algorithm. An alternate approach is presented by Blum and Paar which shows the use of FPGA systolic array multiplier and another approach by is to break these additions into x-bit stages by Elbirt and Paar, each of which had their own disadvantages. In [3] a new, generic Montgomery multiplier and a RSA processor architecture, which can be quickly reconfigured to accommodate different operand sizes for implementation with a variety of technology (FPGA, PLD, ASIC) options is presented.
In the proposed method of Montgomery multiplication a and b are represented into a form known as Montgomery form. 'a' is represented as a x r mod N. If (a x r) mod N and (b x r) mod N are the Montgomery forms of a and b, then their Montgomery product is (a x b x r) mod N. This method is an efficient way to compute the Montgomery product. Transforming the result from this Montgomery form gives us the product a x b mod N. Multiplier forms the essential component and so use of a fast and efficient multiplier is necessary. The [4] paper presents a high speed 16 × 16 multiplier designed by using Urdhawa Tiryagbhayam sutra. This sutra helps to generate the partial products and sums in a single step which helps in reduction of design architecture in the processor.

II. METHODOLOGY

2.1 Steps
1. Converting integer domain to Montgomery domain
$g(a)=a\times r\%N$ $g(b)=b\times r\%N$
2. Multiplication in transformed domain
$z=(a)\times g(b)\times r-1\%\ N$
3. Transform back to integer domain
$w=g-1(z)$ $w=z\times r-1\%N$

2.2 R computation
R is a integer whose value is greater than the prime number p and is the form of 2^k. Initially consider a variable temp. Shift left the contents of prime till its most significant bit becomes one. Simultaneously shift the contents of temp to the right. And all this operation is put inside a loop. At the end of the loop when most significant bit of the prime becomes one, temp holds the value of r=2k, illustrated in Fig.

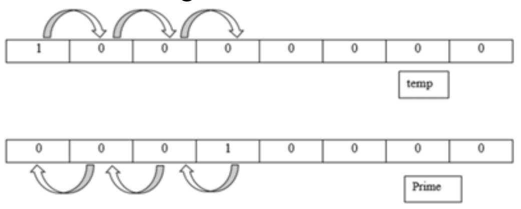


Fig 5.1: r computation

2.3. Inverse calculation Inverse can be calculated using Extended Euclid's algorithm. In the expression u modulo m u-1 is called the multiplicative inverse element of u.
where u x u^{-1} = 1 mod m.

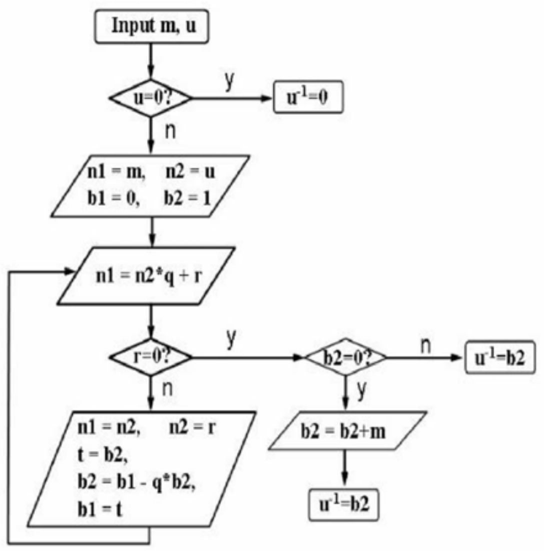


Fig 2: Flow chart for Multiplicative inverse

2.4 Vedic multiplier
Multiplier is the most important component for most of the applications such as DSP's, encryption and decryption algorithm and in other logical calculations. Speed of the multiplier determines the performance of the system. The most common multipliers are array multipliers and booth multiplier. Many studies have been going on to improvise speed of multipliers. One of the oldest methods is Vedic multiplication which is easy to understand and simple.

2.5 Division by subtract and shift method
The division is carried out using this method by following steps below
1. Initially set quotient to zero.

2. Align the left most digits of both the dividend and divisor
3. Repeat the below process to number that is half the number of bits in the numerator
4. If left portion of dividend is greater than or equal to divisor
i. Subtract divisor from that portion of dividend.
ii. Add 1 to right end of dividend.
5. Shift dividend one place right.
6. Left part gives you quotient and right remainder.

III SIMULATION ESULTS

The r computed for given prime p. Given a prime p, we need to compute r = 2k, such that 2k-1 < p < 2k labelled p[15:0] is a 16 bit prime number and labelled r[15:0] is the 16 bit r computed. Simulation result of r computation is shown in Fig.

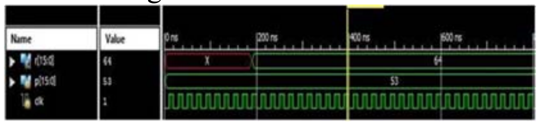


Fig 3: r computation.

Results are shown for a 16 bit by 16 bit multiplier using vedic multiplier. The multiplicand and multiplier are both 32 bits and are labelled as a[15:0] and b[15:0]. The resultant product labelled c[31:0] is 32 bits. Simulation results of multiplier is shown in Fig.

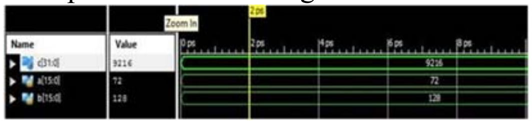


Fig 4: Multiplier.

For any prime number and a positive integer r, there exists two integers $r-1$ and z such that, $r\times r-1-p\times z=1$. Results are shown below in Fig.6.3 for multiplicative inverse of r.

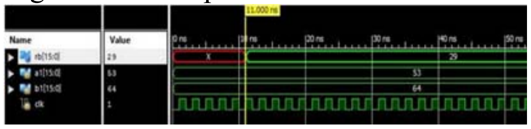


Fig 5: Inverse.

Results are shown for a 32 by 16-bit divider. Labelled n[31:0] is the 32 bit numerator and d[31:0] is the 32 bit denominator. Results of the division are stored in 16 bit quotient labelled as q[15:0] and remainder labelled as r[15:0] which is 16 bits. Fig.6.4 shows the simulation results of modulus calculations.

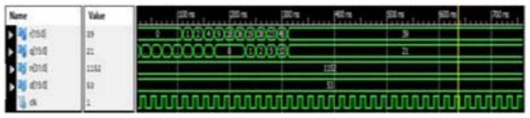


Fig 6: Modulus operation

Montgomery multiplication

For plain text u=51, key v=49 and prime number p=53 result after encryption is shown below in Figure.

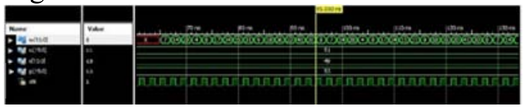


Fig 7: Final Result

The RTL schematic for Montgomery modular multiplication is as shown in below Fig.

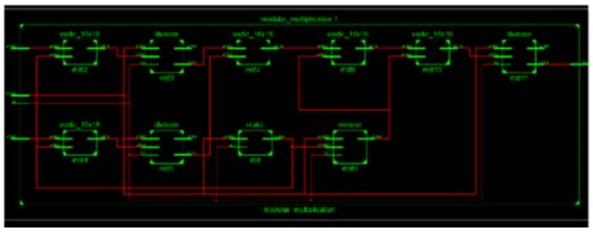


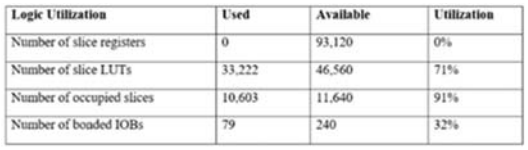
Fig 8: RTL schematic

IV.DEVICE TILIZATION

The synthesis analysis and result is carried out using FPGA board (virtex6) and is given as:

1. Device Utilization on virtex 6 for 16 bit for exponentiation using Montgomery modular multiplication.

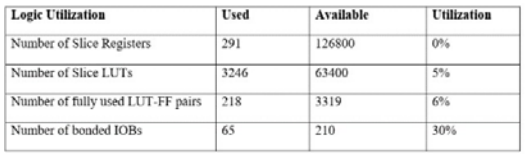
Table I: Utilization summary for virtex 6 for 16bits



Logic Utilization	Used	Available	Utilization
Number of slice registers	0	93,120	0%
Number of slice LUTs	33,222	46,560	71%
Number of occupied slices	10,603	11,640	91%
Number of bonded IOBs	79	240	32%

2. Device Utilization on virtex 6 for 16 bit for Montgomery modular multiplication.

Table II: Utilization summary for virtex 6 for 16bits



Logic Utilization	Used	Available	Utilization
Number of Slice Registers	291	126800	0%
Number of Slice LUTs	3246	63400	5%
Number of fully used LUT-FF pairs	218	3319	6%
Number of bonded IOBs	65	210	30%

V.CONCLUSION

As most encryption algorithm rely on modulus operation proposed method describes the design of circuit module applied to cryptography.

Montgomery multiplication computes a x b mod p for positive integers a, b and p. The multiplicative inverse is computed using extended Euclid’s algorithm. Technique used for division is shift and subtract operation and the multiplier block is implemented using Urdhawa Tiryagbhyam.Vedic multiplier can be replaced by encoder multiplier. Computation time can be reduced by Montgomery multiplication algorithm which is the most efficient modular multiplication algorithm available. It replaces trial division by the modulus with a number of additions and divisions by a power of 2.

REFERENCES

Journal Papers:

[1] “Implementation of modular exponentiation using Montgomery algorithms” by Manish Bansal, Amit Kumar, Faculty of Technology UTU, Dehradun, Uttarakhand. International journal of scientific & engineering research, volume 6, Isssue 11, November 2015.

[2] “FPGA implementation of Montgomery modular multiplier” by veliborSkobic, Branko Dokic International Journal of computer applications, volume 19-No.9,april 2011.

[3] “Fast Montgomery modular multiplication and RSA cryptographic processor architectures”, Ciaran Mclovr, Maire McLoone, John V McCanny, The institute of electronics, communications and Information Technology school of Electrical and electronic Engineering, Alan Daly, William marnane Ireland .

[4] “A High Speed 16×16 Multiplier Based On Urdhva Tiryakbhyam Sutra”, B.Ratna Raju, International Journal of Science Engineering and Advance Technology, IJSEAT, Vol 1, Issue 5, October – 2013.

[5] “Modulo Multiplicative Inverse Circuit Design”, Xiaoying Li, Fuming Sun, Ehua Wu Department of Computer and Information Science, FST, University of Macau, Macao, China