MODIFIED MODULAR MULTIPLIER FOR RSA CRYPTOSYSTEM USING MONTGOMERY ALGORITHM

Ebin Geo Johnson[1], Neethu K.N[2]
[1]Department of Electronics and Communication Engineering, REVA Institute of Technology, Bangalore, India, 560064, [2]Asst. Prof in REVA University, Bangalore, India

***Abstract:** The paper uses the Montgomery algorithm and modifies it to increase the efficiency of large binary operand multiplication. The new architecture reduces the hardware complexity compared to the Montgomery algorithm. The use of carry save adders (CSA) keeps the intermediate values in the carry save format thus hardware for conversion is avoided in intermediate architecture stages, finally at the end of multiplication the format is converted. In the paper a modified carry propagation adder (CPA) is proposed which is less complex in hardware compared to conventional CPAs. The input output interfacing are done effectively and the architecture is implemented in FPGA using verilog HDL*

***Keywords:** Public Key Cryptosystem, RSA Cryptosystem, Carry save adders, Montgomery modular multiplier.*

I. INTRODUCTION

The advancement in communication and increase in Internet transaction like online commerce started demanding more and more security. Thus public key cryptosystem [1] is used to provide the required security out of which RSA scheme is widely used. The main operation in the RSA scheme is to calculate the modular exponentiation by doing the modular multiplication repeatedly. But, the large binary operands (e.g. 1024bit) modular operation leads to a slow throughput and the hardware become complex. Fig. 1 shows structure of Montgomery algorithm by RSA scheme.

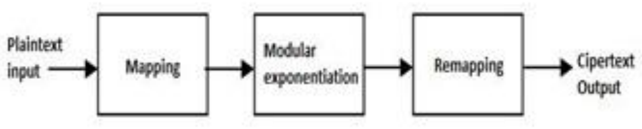

Fig. 1. RSA System

The issue is solved by adopting the Montgomery modular reduction algorithm [2] for the multiplication and thereby making the process fast . There are various architectures where modifications are done in algorithm and hardware structure of basic design out of which this paper concentrates on reduction of operating time and area of chip for increased speed for hardware implementation. In section II , III, IV the RSA scheme is explained briefly, modified Montgomery algorithm and algorithm for exponentiation where the operation limits are removed by first bit of exponent is proposed. Thus in general a new architecture is proposed by using modified adder using shift registers to reduce the hardware complexity and enhanced considering the works [4], [6], [7]. In section V, the performance of proposed RSA processor is analysed in terms of execution time and cost of hardware.

II. MODULAR MULTIPLIER

In modular exponentiation the public key is derived from the formula $e \times d = \mathrm{mod}(p-1)(q-1)$, where e is the encryption key and d is the decryption key and the modulus term N is the product of large primes numbers p and q The encryption using the public key can shown as follows;

$C = E(M) = M^e \mathrm{mod} N$,

Where C is the cipertext and M is the plaintext that is $0 < M < N$. The cipertext can be decrypted using the private key d shown as follows;

$D(C) = C^d \mathrm{mod} N$

The RSA system is designed utilizing the principles of Montgomery algorithm, the main procedure can be listed as three steps as shown in Fig. 1. they are mapping, modular exponentiation and remapping. These each steps can be executed by using the Montgomery modular multiplication. For example consider two operands A and B, and an extra number R, where $R = 2^n$. The multiplication output can be show as

$S = A \times B \times R^{-1}$. The algorithm followed is shown below.

```
Algorithm 1 Montgomery Modular Multiplication
Inputs: A; B; N(mod)
Output: S[n]
S = 0;
for i=0 to n-1 ( $S = S + A_i \times B$;
$S = S + S_0 \times N$; $S = S/2$; )
if($S \geq N$)$S = S - N$;
return S;
```

The 6th step in the algorithm is to remove the oversized output. This process is done by a comparison followed by a subtraction step, which will ultimately increase the hardware complexity. To avoid this comparison and subtraction process the R is made two bits bigger to N [3]. This is taken from the paper proposed by Walter, thus the step 6 from the algorithm can be removed. Thus the multiplication output become $S = A \times B \times 2^{-(n+2)} \mathrm{mod} N$. The intermediate multiplication output are fed back as inputs to complete the whole multiplication.

III. MODIFIED MODULAR OPERATION

The multiplication process is carried out by continuous modular multiplication. Here in the RSA system based on Montgomery MM mainly two process are carried out to com-plete the multiplication process, first is conversion of inputs to Montgomery form that is M into $M \times R \mathrm{mod} N$ and the output is obtained as $M^e \times R \bmod N$. Second when the process is done it must be converted back to normal form by removing the R factor by an operation represented as MMM($M^e \times R \bmod N$, 1)and the required output is obtained. The algorithm for modified modular exponentiation is as shown below;

```
Algorithm 2 Modified Montgomery MM
Inputs: M (plaintext), e(exponent), $R = 2^{(n+2)} \bmod N$,
N (modulus)

Output: $S = M^e \bmod N$; $0 \le S < N$
$M^l$ = MMM(M; R); S = MMM(R; 1);
for i=0 to n-1 (S = MMM(S; S)
If ($e_i$ = 1) then ( (S = MMM(S; $M^l$))
else S = S )
S = MMM(S, 1)
return (S);
```

In the multiplication process the exponent first bit having limits create problems, this can be avoided by fixing 1 to first bit or by finding the initial non zero bit, all these done by making use of extra operation. The proposed algorithm is different other previous modifications, here both M^l and initial value of S are computed, thus limit issue is solved by using MMM function once. But this may increase the count of multiplication required and in the worst case it may go up to 2n+3. Since the n value taken is large this can be ignored.

IV. DESIGN OF HARDWARE

The design shown in the Fig.2. is done by the modified algorithm to form the RSA system. The architecture composed of MMM part and the control system. The MMM part calculates the Montgomery modular multiplication and the inputs outputs are controlled by the control system.

A. Design of Carry Propagation Adder

The architecture of CPA is elaborated in Fig. 3. The outputs in the MMM module from the carry save adder are added to get the final MM result. The format conversion from the carry save format is done by the help of CPA that increases the hardware complexity and becomes difficult for hardware implementation. Here the addition are done by full adder and each clock gives out one bit output from each iteration [5]. But these process require two 1024 bit register to store the operands and there by the hardware become complex. The demerits of the previous mentioned multiplier with normal CPA can be avoided by using a CPA with 32bit along with shift register. In the shift registers, show in Fig. 4., the inputs are provide which get shifted based on the clocks given.

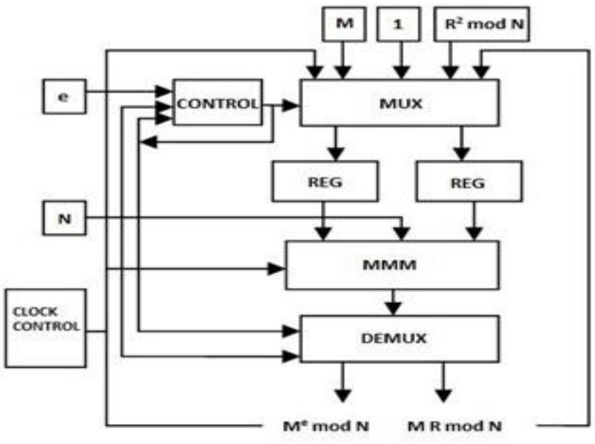

Fig. 2. Modified Architecture

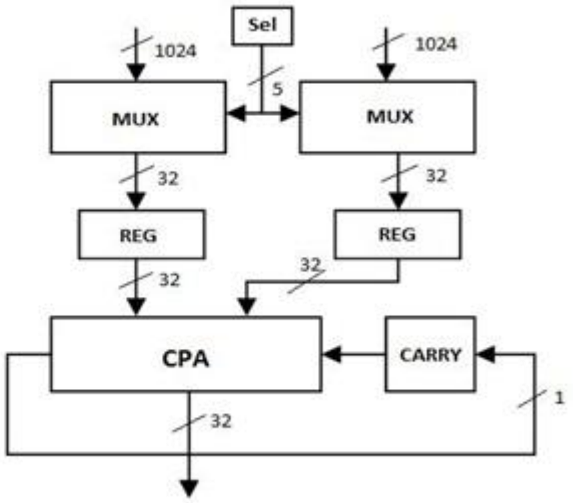

Fig. 3. Carry Propagation Adder

The input shifter register consist of 32 registers. The modified CPA required 32 extra clocks and at each clock a 32 bit output is generated. As a whole the modified CPA with shifter reduces the hardware complexity and cost.

B. MMM Module

The Montgomery Multiplication process is carried out in the MMM module. The structure of the MMM module is shown in Fig. 5., it is designed with two CSAs connected back to

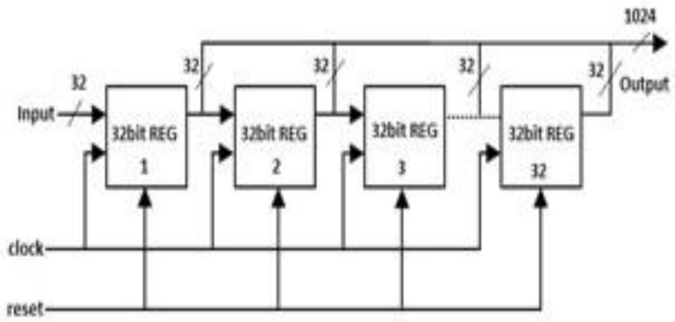

Fig. 4. Input Registers

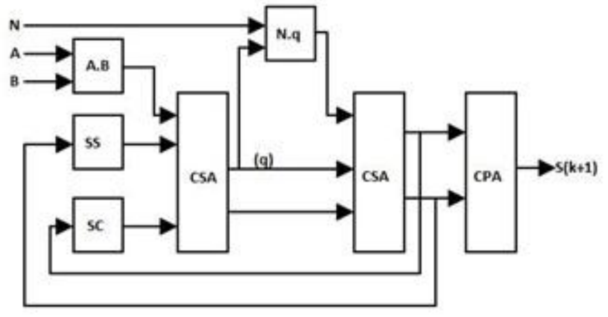

Fig. 5. 1024-BIT Montgomery MM

back. The design can be made with one CSA and a multiplexer but the hardware cost remain same so the two CSA design is continued. The module contain two separate registers to store carry and sum, two CSA and format converter from carry save format that is the CPA with shifter.

V. RESULT ANALYIS

A. Time for Operation

As mentioned before in the paper the worst case for modular multiplication is 2n+3 by using the previous designs and the modular multiplication using Montgomery algorithm gives an average of 1.5n+3 multiplications. The extra iteration step makes the multiplication count n+2, followed by the 32bit CPA add up the count to n+2+32. The increase in count is small compared to the total 1024 bit and only a 3

B. Comparison of Hardware

The paper has compared the RSA system structure which consist of the MMM module and the control system designed my multiplexers with the two CSA and the modified CPA design, where the hardware cost is reduced compared to the former A lot of hardware is saved by implementing the modified CPA compared to the previous CPA using full adders.

VI. CONCLUSION

The paper proposes modified modular multiplier for RSA cryptosystem using Montgomery algorithm. The modification makes the whole multiplication simple by one additional multiplication. The paper also proposes a new modified CPA for conversion of result from carry save format to normal format by minimized hardware. Thus the proposed system is an appreciation for VLSI implementation with low cost and less hardware complexity.

REFERENCES

[1] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signature and public-key cryptosystems," Comm. ACM, v01.2 1, pp. 120-1 26, 1978.

[2] P. L. Montgomery, "Modular multiplication without trial division," Math. Computation, vol. 44, pp.519-521,1985.

[3] T. Blum and C. Paar, "Montgomery modular exponentiation on recon-figurable hardware," in Proc. 14th IEEE Symp. on Computer Arithmetic, 1999, pp.

[4] S. E. Eldridge and C. D. Walter, "Hardware implementation of Mont-gomery's modular multiplication algorithm," IEEE Trans. Comput., vol. 42, pp. 693 -699, June 1993.

[5] C. C. Yang, T. S. Chang, and C. W. Jen, "A new RSA cryptosystem hardware design based on Montgomery's algorithm," IEEE Trans. Cir-cuits and Systems 11: Analog and Digital Signal Processing, vol. 45, pp. 70 -77. 908-913, July 1998.

[6] A. Cilardo, A. Mazzeo, L. Romano, and G. P. Saggese, "Exploring the design-space for FPGA-based implementation of RSA," Microprocessors Microsyst., vol. 28, no. 4, pp. 183–191, May 2004.

[7] J.-H. Hong and C.-W. Wu, "Cellular-array modular multiplier for fast RSA public-key cryptosystem based on modified Booth's algorithm," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 11, no. 3, pp. 474–484, Jun. 2003.