International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)
Volume 3, Issue 11, November 2014

High Speed Finite Field Multiplier GF(2^M) for Cryptographic Applications

M.Uma Maheswari
ME VLSI Design
Angel College of Engineering& Technology
Tirupur,Tamil Nadu, India

S.Baskar
Dept of ECE
Angel College of Engineering& Technology
Tirupur,Tamil Nadu, India

G.M.Keerthi
ME VLSI Design
Angel College of Engineering& Technology
Tirupur,Tamil Nadu, India

Abstract - **In this paper,we present a low latency finite Field multiplier over GF(2^M) that is used in Cryptographic applications for secure data encryption and decryption which deals with discrete structure and mathematical arithmetic. Since it uses modular arithmetic operation,It is found that it has the latency of m cycles. To overcome this problem, we proposed an efficient software implementation of high speed montgomery multiplier over GF(2^M) in FPGA for pentanomial which is based on residue number system and it uses the concept of parallel processing in which multiplication is decomposed into number of independent units and "pre-computed addition" techniques. The proposed design involves significantly less delay and power complexities compared to traditional multipliers.**

Index Terms— **Low latency, Montgomery multiplication, cryptography, finite field (or galois field), modular arithmetic.**

I. INTRODUCTION

Finite field has applications in information theory,number theory algebraic coding theory, error-control codes, and cryptography [1]. Particularly In public key crypto-system there are several aspects to be considered such as integrity, security, key lengths, speed and implementation issues. For security modular multiplication of finite field multipliers are used.however field multiplication consumes more amount of time which is the bottleneck of cryptographic algorithm..To overcome this problem, in 1985 Peter.L.Montgomery introduced an algorithm for finite field multiplier which avoids time consumption and can be implemented in both hardware and software. There are a few bases available for finite fields GF(2^M), namely, polynomial basis, normal basis, dual basis, weakly dual basis, triangular basis, and redundant basis. Dual and normal basis multipliers require a basis conversion, in which complexity heavily depends on the irreducible polynomial . In case of standard basis multipliers do not require a basis conversion; Polynomial basis multipliers are popularly used because they are relatively simple to design, and offer scalability for the fields of higher orders.

Different Montgomery multiplier was constructed using different implementation techniques,like systolic ,semi-systolic,non-systolic. Nonsystolic architecture has global signals. Therefore, if becomes large, propagation delay also increases. But systolic architecture consists of replicated basic cells and each basic cell is connected with its neighboring cells through pipelining, i.e., there are no global signals. Hence the systolic architecture is a better choice than the nonsystolic architecture for a high-speed VLSI implementation Montgomery multiplier can be implemented in three version: bit-serial,digit-serial,bit-parallel. Bit-serial multipliers which processes one bit of input data per clock cycle and is area-efficient and suitable for low-speed applications . In contrast, bit-parallel multipliers which processes one whole word of input data per clock cycle, and is ideal for high-speed applications .

In this paper, we proposed an efficient software implementation of high speed montgomery multiplier over GF(2^M) in FPGA For pentanomial . A transformation method is used to implement bit-parallel systolic Montgomery multipliers over GF(2^M) , which is defined by irreducible polynomial.the proposed algorithm uses the concept of parallel processing in which multiplication is decomposed into a number of independent units, and "pre-computed addition" (PCA) technique in which the latency of a multiplier can be reduced further. The proposed design involves significantly less delay and power complexities compared to traditional multipliers.

The remainder of the paper is organized as follows: The proposed Montgomery multiplication algorithm and for finite field multiplication over based on irreducible pentanomials are presented in Section II. In Section III, architecture of the proposed multiplier is depicted. In Section IV, we have discussed the estimation of hardware and time-complexities and compared those with traditional multiplier. Conclusion is presented in Section V.future work is in section VI.

II. MONTGOMERY MULTIPLICATION

Modular multiplication generally consist of two steps: one generates the product and the other reduces this product modulo M.

Modular multiplication (Xm) of two integers a and b are given as,

Xm (a, b) = a · b (mod p)

Montgomery multiplication (MM) computes

MM (a, b) = a · b · r $^{-1}$ (mod p)

where r is a special constant

In 1985 Peter L. Montgomery proposed a method for modular multiplication using Residue Number System (RNS) representation of integers. In this method all elements need to be transformed to Montgomery residue field. Montgomery residue field is used to represent the r–multiplied values of the operands, as a·r and b·r. The operands a and b are binary extension field elements .i.e. GF(2^M). It replaces the costly division operation which is used to perform modular reduction by simple shift operations by transform the operands into the RNS before the operation and re-transforming the result thereafter.It is considered to be the fastest algorithm to compute ab mod P in computers when the values of a, b, and N are large. This algorithm computes the product of two integers modulo a third one without performing division by P.

The pre-conditions of the Montgomery algorithm are as follows: The modulus P have to be relatively prime and odd value i.e. There exists no common divisor for P and the radix r; The multiplicand and the multiplicator have to be smaller than P.

Hence transformation operations must be applied to both of the operands a and b before the multiplication and to the intermediate result $\bar{c}$ In order to obtain the final result c. To transform an input operand to Montgomery residue field, a multiplication by a constant value is required. This constant value is r^2 mod p, where radix $r = 2^N > P$.

To multiply two numbers from the integer field, four Montgomery multiplications are needed. First a and b are transformed to their images in the Montgomery domain. Then multiplication is performed in the Montgomery domain, and finally multiplying the final result by 1, to transform the result c back to the integer domain.

The four step Montgomery multiplication operation are:

$\bar{a}$= MM (a, r^2) = a · r^2 · r $^{-1}$ (mod p) = a · r (mod p)

$\bar{b}$ = MM (b, r^2) = b · r^2 · r $^{-1}$ (mod p) = b · r (mod p)

$\bar{c}$= MM ($\bar{a}$, $\bar{b}$) = a · r · b · r · r $^{-1}$ (mod p)= a · b·r(modp)

c = MM ($\bar{c}$, 1) = a · b · r · r $^{-1}$ (mod p) = a · b (mod p)

RNS representations of integers are called M-residues and are usually denoted as the integer variable name with a bar above it.However, for Montgomery algorithm in which lot of modular multiplications need to be performed with respect to the same modulus, the performance gain is high due to the ratio between transformation overhead and actual modular arithmetic is much lower. In figure 2.1, Montgomery multiplication of two integers is illustrated.

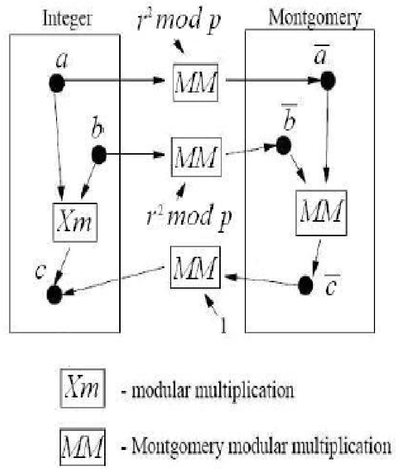

Figure 2.1 Montgomery Modular Multiplication

```
function
 reduce (x)
 begin q := (x mod r)n0 mod r;
  a := (x + qn)/r;
  if a > n then
  a := a − n;
  return a;
  end
```

Figure 2.2: Algorithm for the computation of a = xr^{-1} mod P.

III. ARCHITECTURE

The proposed systolic structure for field multiplication over GF(2^M) based on pentanomial is shown in Fig. 3.1.(a). It consists of four systolic arrays. The first systolic array consists of5 pes, while second, third, and fourth systolic arrays consists of four pes and a delay cell which is required to meet the data dependence requirement. Although one can use a systolic adder array consisting of four addition cells for the final addition of the four arrays, we can use a pipelined adder-tree consisting of three addition cells for a low latency implementation. The four arrays can function in parallel manner, such that after five cycles, the adder-tree receives its first input and yields its first output in two cycles.

The detailed design of the PEs of the proposed structure is shown in Fig. 3.1.(b) PE-1 of the arrays contains only one NMR cell (NMRC. The regular PE, as shown in Fig. 3.1(c) consists of one AND cell, one XOR cell, and one NMRC. The AND cell and XOR cell corresponds to the bit multiplication node and bit addition node Respectively. Each AND cell and XOR cell, respectively,Consists of number of AND gates and XOR gates working in parallel.Fig. 3.1.(f) gives an example of the design, Note that in PE-3 (PE-4 in the first array), as shown in Fig. 3.1.(d), the output of NMRC consists of bits. Moreover, we have also shown the design detail of the NMRC (PE-1 of the second array) in Fig. 3.1.(e).

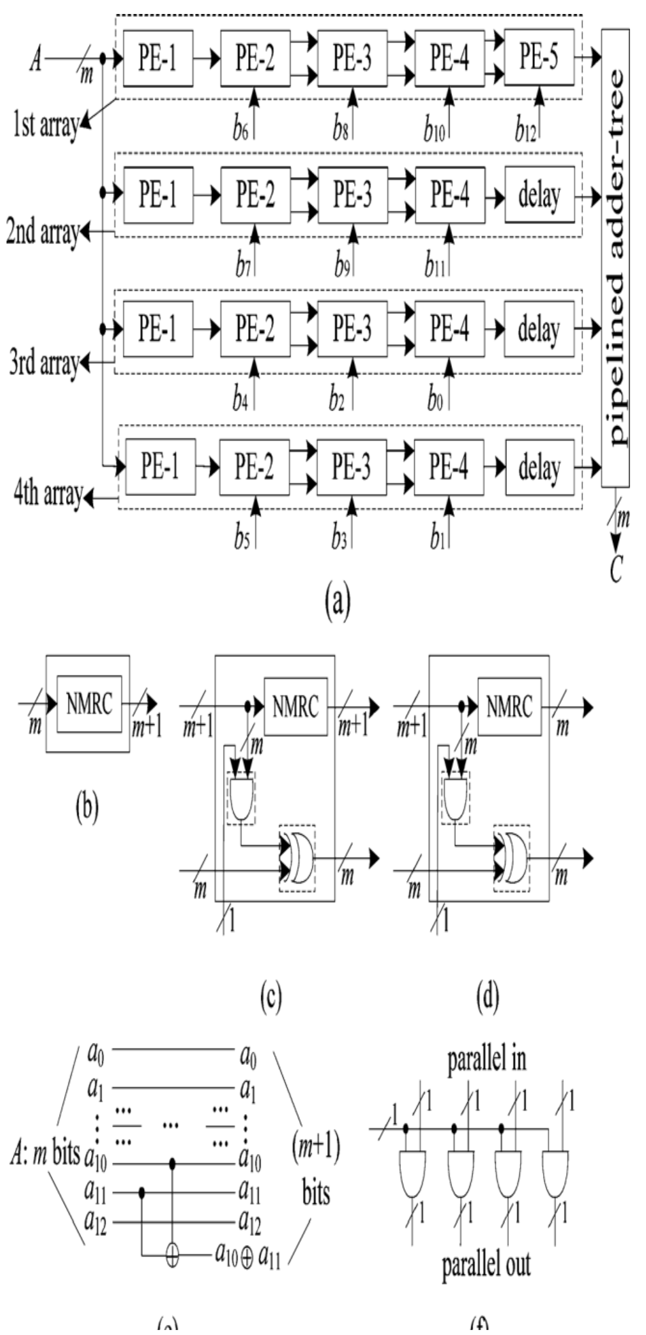

fig. 3.1. proposed systolic-like structure for m=13 . (a) proposed structure.(b) design of the pe-1. (c) design of a regular pe. (d) design of the second pe in the array (from right). (e) structure of nmrc in the pe-1 of the second array.(f) an example of structure of the and cell.

IV. RESULT AND DISCUSSION

the simulation results and device utilization summary of 8-bit montgomery multiplier as shown in fig. 4.1 & fig.4.2 and table .i.

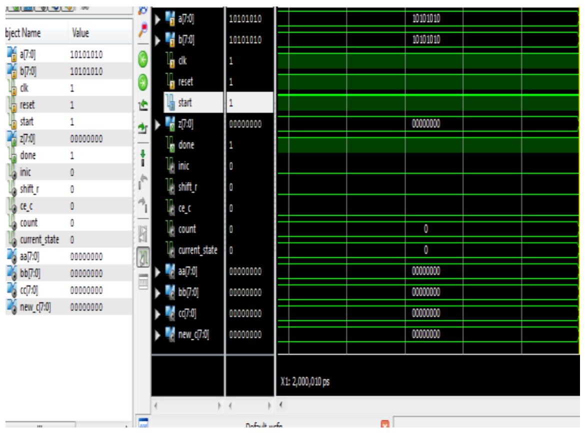

Fig. 4.1:simulation results of montgomery multiplier

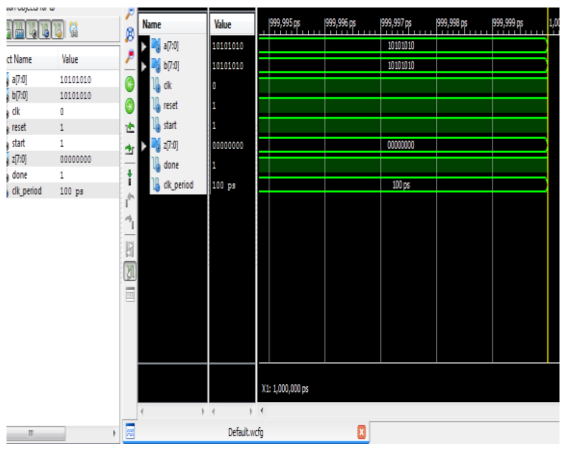

Fig. 4.2: simulation of Montgomery multiplier with testbench.

Table .I: Device Utilization Summary

Device Utilization Summary (estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	19	960	1%
Number of Slice Flip Flops	31	1920	1%
Number of 4 input LUTs	28	1920	1%
Number of bonded IOBs	28	66	42%
Number of GCLKs	1	24	4%

The RTL schematic of Montgomery multiplier as shown in Fig. 4.3.

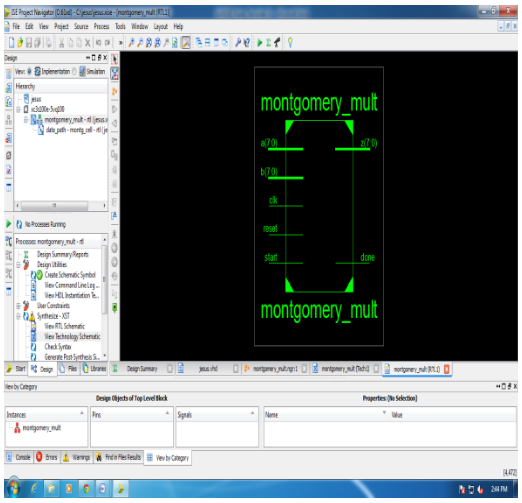

Fig.4.3. RTL schematic

Table.II:Comparision Table

Multiplier TYPE	**PARAMETER**	
	Dynamic power(watt)	**Speed(mHz)**
Array	0.357	23.680
Tree	0.295	30.662
Booth	0.288	22.158
Montgomery multiplier	0.206	55.795

V.CONCLUSION

Montgomery multiplication algorithm is a very efficient way of modular multiplication. The algorithm is implemented in a unified and dual–radix architecture. The new unified dual–radix multiplier architecture operates in binary extension fields. The architecture includes a precomputation block, thus decreasing the longest path delay significantly. Implementation of parallel processing into the Montgomery multiplier architecture gives better performance.

The area and speed characteristics of the montgomery multiplier is also investigated and its performance in terms of EXTRA-OVERHEAD and multiplication time is compared against traditional multipliers. The results show sthat The proposed system of Montgomery multiplier used to reduce the area and usage of time is less . Multiplication period changes in GF(2^M) mode of operation . The clock cycle count is halved in GF(2^M) mode as two multiplier bits are processed in every clock cycle. Conclusions of this study show that the algorithm can be extended to process more multiplier bits in each clock cycle.

VIFUTURE WORK

To implement the montgomery multiplier can be applied to n number of bits and the main application of montgomery multiplier is cryptography like rsa and ecc algorithms etc…. Further the design is implemented in many applications.

REFERENCES

[1] Elliptic Curves in Cryptography, ser. London Mathematical Society Lecture Note Series. Cambridge, U.K.: Cambridge Univ. Press, 1999.

[2]]H.Wu,"Bit-parallel polynomial basis multiplier for new classes of finitefields, "ieeetrans. Comput., vol.57, no.8, pp.1023–1031,Aug. 2008.

[3] P. K. Meher, "On efficient implementation of accumulation in finite field over and its applications," ieeetrans. Very large scale Integr. (VLSI) Syst., vol. 17, no. 4, pp. 541–550, Apr. 2009.

[4] C.-Y. Lee, J.-S. Horng, I.-C. Jou, and E.-H. Lu, "Low-complexity bit-parallel systolic montgomery multipliers for special classes of GF(2^M) ," IEEE Trans. Comput., vol. 54, no. 9, pp. 1061–1070, Sep. 2005.

[5]. P. Montgomery, "Bit-serial and bit-parallel Montgomery multiplication and squaring over ," IEEE Trans. Comput., vol. 58, no. 10, pp. 1332–1345, Oct. 2009.

[6] C.-Y. Lee, J.-S. Horng, I.-C. Jou, and E.-H. Lu, "A digit-serial multi- plier for finite field ," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 13, no. 4, pp. 476–483, Apr. 2005.