I J C T A, 10(5) 2017, pp. 959-965

VLSI Implementation of Bit/Digit Serial-Parallel Finite Field GF (2^m) Multiplier using Standard Basis

M. Manasa*, C. Piyanka* and A. Vamseekrishna**

Abstract: A Finite Field is a field with finitely number of elements. It is also known as Galois field in which the elements can take *q* different values is referred as GF (*q*). Finite fields are of great interest in applications like Elliptic Curve Cryptography (ECC), error control and coding. In this paper, we present a simple implementation of finite field GF (2^m) multiplier. Here the main aim is to enhance the vital factors such as area, time complexity, critical path and power. The proposed design is implemented with the finite field accumulator using *x*-or gates and T-flip-flops. The previous design is even done so that the comparison of the vital factors can be performed. The multiplier structures are implemented using polynomial basis, which is the standard representation and simulated using Xilinx ISE tool version-12.1 and the comparison is performed using Cadence (CAD) tool.

Keywords: Polynomial basis, finite field addition, Galois field, Finite field multiplication.

1. INTRODUCTION

Multiplication using finite field theory has got many applications which include Cryptography, Encoding, Digital Signal Processing and Cryptosystems. In Cryptography two parties communicate without third party hearing it. So the encoding and decoding process is performed using finite fields where multiplication is more preferred than addition because of its simplicity and advantageous. Here the addition of two single bits requires only a logical XOR operation. Finite fields are used in a variety of applications including classical coding theory in linear block codes such as Reed Solomon codes and in cryptographic algorithms. In this paper, we present the design of finite field accumulator with XOR gates and T flip-flops. The vital factors such as area, time-complexity, critical path, number of cells are reduced and compared with its previous implementation. The synthesis reports of each implementation are also obtained. Although it is trivial and simple the proposed FFA can result in significantly more efficient implementation of other field operations and point operations for ECC. There are three different bases of representation over GF (2^m). Those are: polynomial basis, normal basis, and dual basis. Out of the three, polynomial basis and normal basis are popular due to their higher practical relevance [2], [3]. Polynomial basis which is the standard basis of representation has superior performance in field multiplication. Performing multiplication and addition in Standard basis of representation would be useful for any efficient implementation of a given application.

2. ACCUMULATION IN FINITE FIELDS

Let the finite field GF (2^m) defined by an irreducible polynomial of degree$_m$, given by [7]

$$Q(x) = x^m + q_{m-1}\, x^{m-1} + \ldots\ldots\ldots + q_2\, x^2 + q_1\, x + 1$$

The polynomial basis in the Figure 1 in the form of polynomial of degree ($m - 1$). Addition is the simplest operation which is performed by Q(x) introduces a polynomial basis which is the standard basis of representation used to represent the field elements. Take two arbitrary elements A and B in Galois Field

* Assistant Professor, Department of ECE, K.L. University, Guntur, India

** Assistant Professor, Department of ECM, K.L. University, Guntur, India

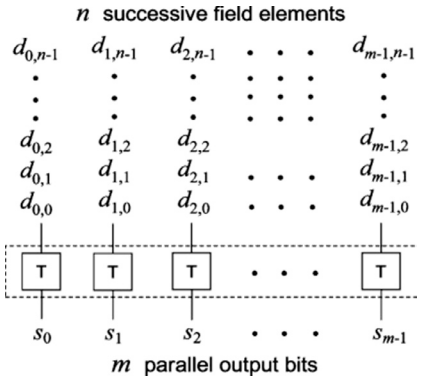

Figure 1: Finite Field Accumulator

represented bit by bit XOR operation of their operands so that the addition of any two finite field elements is given by

$$S = A + B.$$

Irrespective of the previous design which is implemented with D flip-flops and XOR gates the present design consists of T flip-flops where its complexity is approximately equal to the complexity of a D flip-flop. The input of a T flip-flop is however required to be fed along with the clock to a NAND gate followed by an inverter to derive the clock derivation of circuit. The states of all the T flip-flops of the finite field accumulator are reset at the beginning and also at successive field cycles. The elements to be accumulated are fed to the flip-flops in parallel. Since the state of a T flip-flop toggles on the arrival of each 1 as its input, the finite field accumulator performs the required accumulation where the input bits corresponding to all the elements are fed to the T flip-flops in successive cycles. [5]

3. SERIAL-PARALLEL MULTIPLIER OVER GF (2^M)

Design of efficient finite field arithmetic architectures is very important and of great practical concern. In terms of input, output structuring all the multipliers over GF (2^m) can be classified into three basic forms. They are parallel-in parallel-out architectures, serial-in serial-out architectures, and serial-in parallel-out architectures. Digit-serial/parallel architectures seen in Figure 2 are attractive both low energy and high performance applications. The bit-parallel designs are intended mainly for high-speed implementation of the multiplication over GF (2^m).

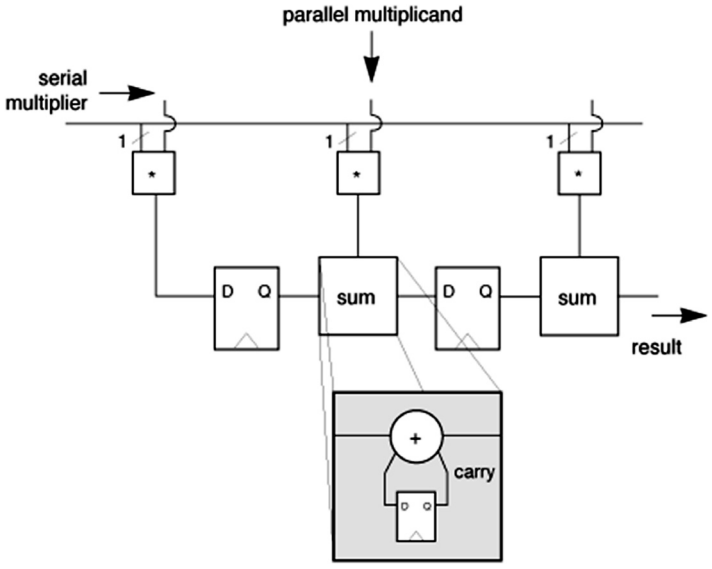

Figure 2: Serial parallel Multiplier

They have high throughput rate and are used in many applications like cryptography, digital signal processing etc. [2], the portable and embedded devices where ECC is currently targeted are heavily constrained in terms or cost, size, and power-consumption.

National Institute of Standards and Technology (NIST) [3] has recommended five binary finite fields for Cryptography implementation out of which two are generated by the trinomials which are

$$Q(x) = x_{233} + x_{73} + 1$$

and

$$Q(x) = x_{409} + x_{87}$$

Keeping these facts in view, using the proposed finite field accumulator in this section, we have derived a bit-serial/par-allel multiplier based on a general irreducible polynomial and a digit-serial/parallel multiplier for polynomial basis multiplications over GF (2^m) for trinomial.

A. Bit-Serial-Parallel Multiplication Over GF(2^m) Based On General Polynomials

The proposed structure consists of three units. Modular reduction unit (MRU), AND unit and finite field accumulator. The modular reduction unit consists of m number of D flip-flops and $(m - 1)$ number of reduction cells. During the first clock cycle the state of the D flip-flop is initialized by loading the operand word a in parallel. The reduction polynomial used was

$$X^8 + X^4 + X^3 + X + 1$$

It is observed that the structure and function seen in Figure 3 of the reduction cell depends on the value of the coefficient bit. If the coefficient bit q_i is 1 then the reduction cell performs X-or operation of its input from left with its input from top to produce an output to be fed to the D flip-flop in the right. If the coefficient bit q_i is 0 then the reduction cell is removed and directly the output of one D flip-flop is connected to the next D flip-flop on the right. [6]

This structure works exactly as a linear feedback shift register. It performs multiplication operation here where the number of delays is less. [7]

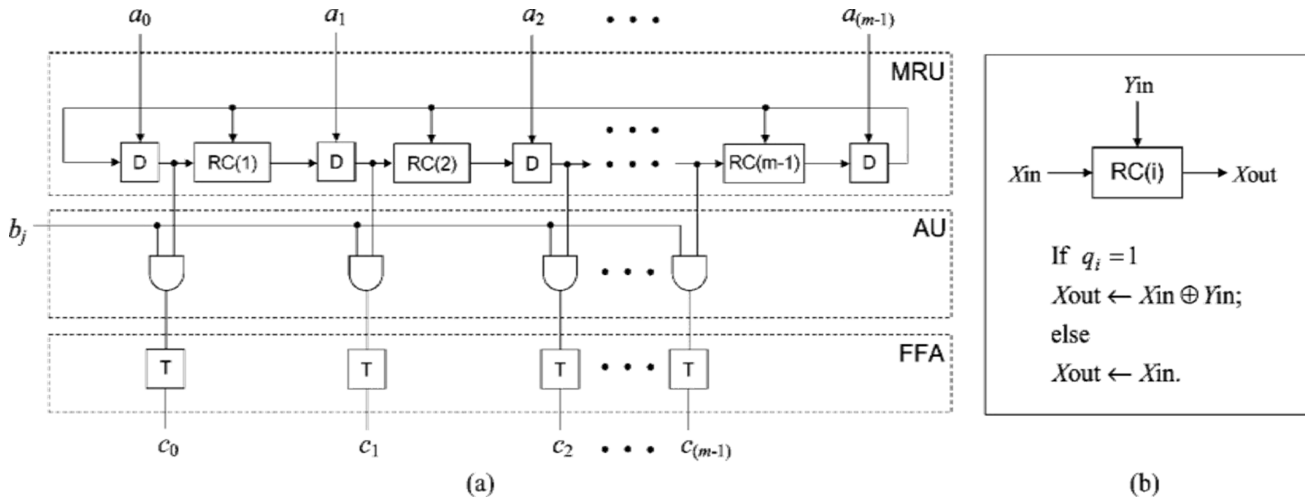

Figure 3: Bit-Serial-Parallel Multiplier

B. Digit-Serial-Parallel Multiplication over (2^m) Based On General Trinomials

Here the Digit-Serial-Parallel structure is implemented by using Polynomial basis with trinomials where the delay, critical path, area, power is well reduced. [4]

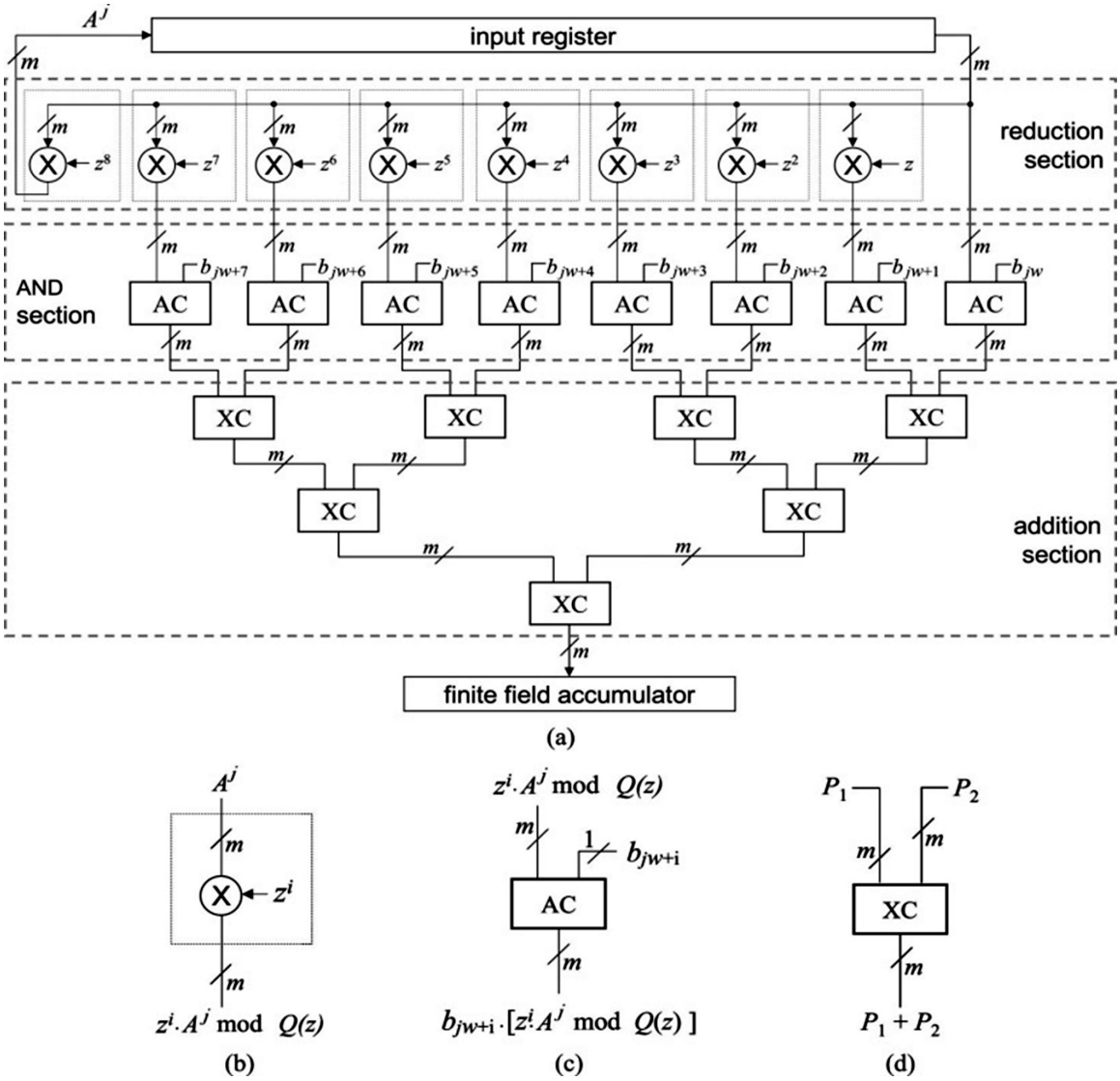

Figure 4: (a) Structure of PGCMR unit, (b) Modular reduction cell, (c) AND cell, (d) X-or cell [7]

The structure for the above multiplier is seen in Figure 3. It consists of Product generator cum modular reduction unit, which intern consists of AND section, reduction section and addition section. Also consists of finite field accumulator and input register. The AND section consists of AND cells. It performs AND operation which receives one input from the reduction section. Input register of the structure consisting of D flip-flops is initialized by one of the multiplicands A. The modular reduction unit performs bit by bit XOR operation for the given input. [7]

The output from the AND section is given as an input to the XOR cell where the addition operation is performed and the resultant multiplier output obtained is finally stored in a finite field accumulator. This entire operation is done only when reset is HIGH. Count gets incremented or decremented depending on the enable signal.

4. SIMULATION RESULTS

The multiplier structures using D flip-flops as well as T flip-flops are implemented and simulated using Xilinx ISE tool version 12.1. seen in Figure 6 and the results of each section are obtained individually along with the multiplier result. One example of the above results is:

Polynomial: $X^8 + X^4 + X^3 + X + 1$.

Let A = 8'h83 and B = 8'h57

$$57 \times 83 = C_1 = 01010111 \times 10000011 = 11000001$$

Step 1: From the logarithm table find L (57) and L (83).

$$L\ (57) = 62 \text{ and } L\ (83) = 50.$$

Step 2: Add them together (regular addition) to get B2.

$$\begin{array}{r} 01100010 \\ 01010000 \\ \hline 10110010\text{------}B_2 \end{array}$$

Step 3: Next, take the exponential of B2 to get the product C_1.

$$E(B_2) = C_1$$

Therefore $$57 \times 83 = C_1.$$

The synthesized report is also obtained which clearly shows the comparison of vital factors between the two multiplier structures.

Figure 5: Output Waveform of Galois Field GF (2^m)

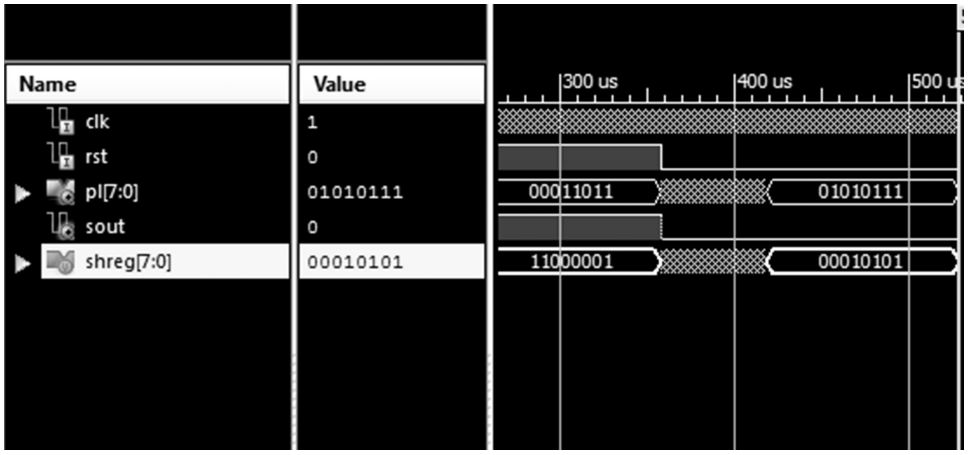

Figure 6: Output Waveform of Parallel in Serial out Shift Register

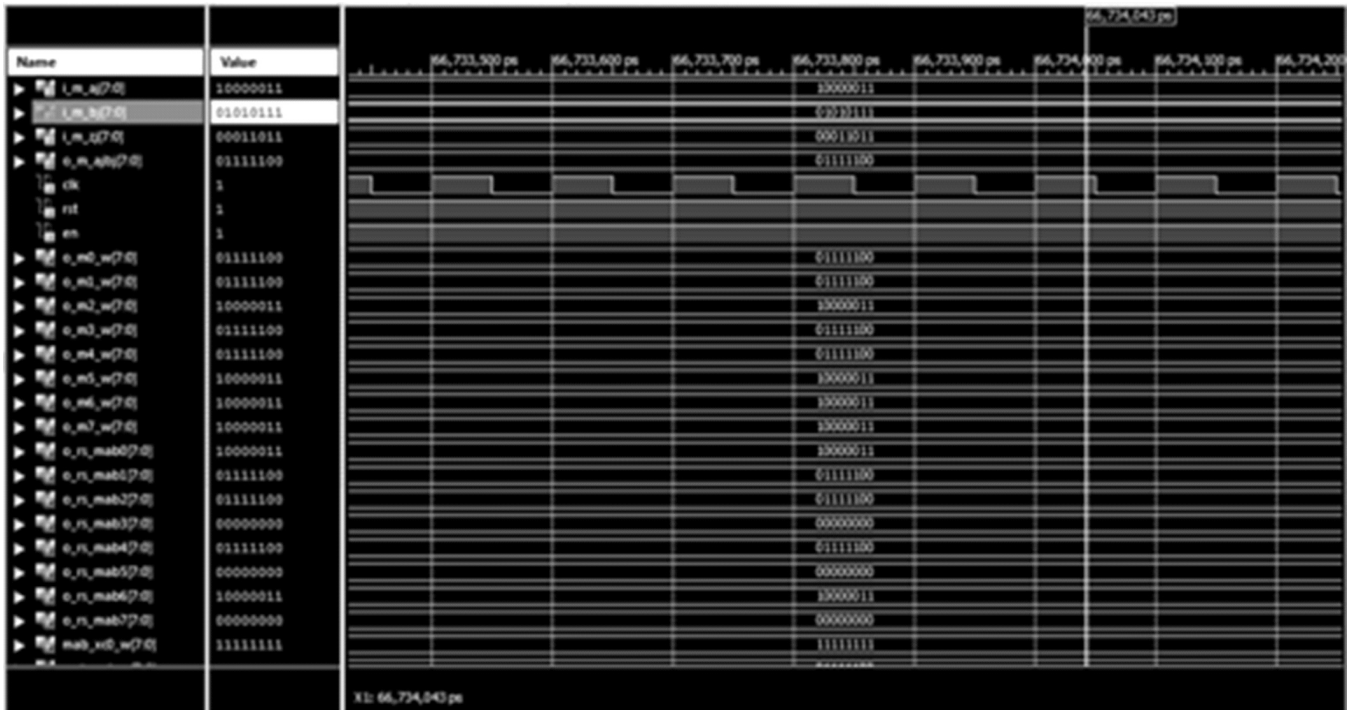

Figure 7: Simulation result of PGCMR Unit

The multiplier output is obtained after the individual sections are instantiated. The reset signal is given high only then the operation is performed. Depending on the enable signal whether it is zero or one the count gets started. If the enable is zero count doesn't gets incremented. If the enable signal is high then count gets incremented. When the reset signal is high the entire operation is performed and finally the output of the multiplier is stored in the Finite Field Accumulator.

Table 1
Comparison of Designs

Vital Factors	*Implementation using D flip-flops*	*Implementation using T flip-flops*
Number of Flip-Flops	24	12
Maximum Delay(ns)	2.084	2.013
Registers	24	12
Number of LUTs	43	74
Complexity	Easy	Medium

5. CONCLUSION

From this work we can decide that designing multipliers is fast and simple when finite fields are considered. As the entire operation is done using polynomial basis which is the standard representation it was found easy while implementing the multiplier structures. The present design which dealt with modular reduction unit consumes low power, occupies less area which is the most important part for any designer to design any sort of application and also takes less time and even the critical path is reduced. As the Theory of Encryption, Coding and Cryptography is based on mathematics of finite fields further studies can be made to find ways of improvement in the implementation of computations performed in Finite Fields. Arithmetic and computation in finite fields will certainly be an important area for ongoing research.

References

1. Blake, G. Seroussi, and N. P. Smart, "Elliptic curves in cryptography," in *London Mathematical Society Lecture Note Series*. Cam-bridge, U.K.: Cambridge University Press, 199.
2. J.-H. Guo and C.-L.Wang, "Digit-serial systolic multiplier for finite fields *m*," *IEE Proc.—Comput. Digit. Techn.*, Vol. 145, No. 2, pp. 143–148, Mar. 1998 on Systems in Medicine and Biology, December 16-18, 2010.
3. Lin Bai, Qinye Yin, "A modified nlms algorithm for adaptive noise cancellation*" IEEE on Intelligent Networks and Network Security, 2010.*
4. P.K. Meher, "Systolic formulation for low-complexity serial-parallel implementation of unified finite field multiplication over m ," in *Proc. 18th IEEE Int. Conf. Appl.-Specific Syst., Arch. Processors (ASAP)*, Montreal, QC, Canada, Jul. 2007, pp. 134-139.
5. *National Institute of Standards and Technology, FIPS 186-2, Digital Signature Standard (DSS)*, Federal Information Processing Standards Publication 186-2, 2000.
6. L. Song and K.K. Parhi, "Efficient finite field serial/parallel multiplica-tion," in *Proc. Int. Conf. Appl. Specific Syst., Arch. Processors (ASAP)*, Aug. 1996, pp. 72-82.
7. Meher, P.K. "On Efficient Implementation of Accumulation in Finite Field Over GF(2^m) and its Applications, "Very Large Scale Integration(VLSI) Systems, *IEEE Transactions on, Vol. 17.* April 2009, pp. 541-550.