Int. J. Electron. Commun. (AEÜ) 114 (2020) 153017

Contents lists available at ScienceDirect

International Journal of Electronics and Communications (AEÜ)

journal homepage: www.elsevier.com/locate/aeue

Regular paper

An area-efficient bit-serial sequential polynomial basis finite field GF (2^m) multiplier ☆

Siva Ramakrishna Pillutla *, Lakshmi Boppana

Department of Electronics and Communication Engineering, National Institute of Technology Warangal, Telangana, India

ARTICLE INFO

Article history:
Received 25 July 2019
Accepted 25 November 2019

Keywords:
Finite field arithmetic
Polynomial basis
Bit-serial multiplier
Elliptic curve cryptography
Internet of Things (IoT)

ABSTRACT

Many cryptographic and error control coding algorithms rely on finite field arithmetic. Hardware implementation of these algorithms requires an efficient realization of finite field GF(2^m) arithmetic operations. Finite field multiplication is complex among the basic arithmetic operations, and it is employed in field exponentiation and inversion operations. Various algorithms and architectures are proposed in the literature for hardware implementation of finite field multiplication to achieve a reduction in area and delay. In this paper, a modified interleaved modular reduction multiplication algorithm and its bit-serial sequential architecture are proposed. It is observed from the comparison of analytical results that the proposed architecture achieves the reduction in area and area-delay product compared to the existing multipliers. The proposed multiplier achieves an improvement of 39% in area and 17% in area-delay product estimations for field order of 409 when compared with the best sequential multiplier available in the literature. Application specific integrated circuit (ASIC) implementation of the proposed multiplier together with the two most comparable multipliers confirms that the proposed multiplier outperforms in terms of area and area-delay product. The proposed multiplier is suitable for implementation of security in Internet of Things (IoT) gateways and edge-devices.

1. Introduction

Internet of Things (IoT) is a recent communication technology which can extend network services for constrained environments also. The effective spreading of IoT into various heterogeneous environments depends on the customised security and privacy features employed in IoT devices [1]. Network security for IoT can be achieved by using Cryptography. Elliptic curve cryptography is a more suitable candidate for resource-constrained environments, especially for IoT applications, as it provides more security with shorter key sizes and involves less computational complexities [2]. Furthermore, cryptosystems such as elliptic curve cryptosystem and elgamal cryptosystem heavily depend on finite field arithmetic [3].

In addition to cryptography, many other applications such as error correcting codes (Reed-Solomon Coders) [4,5], computer algebra, and digital signal processing (convolution) make use of finite field arithmetic. A finite field GF(2^m) is an algebraic set structure of 2^m elements upon which various arithmetic operations such as addition, subtraction, multiplication, and division can be performed without leaving the set [6]. Finite field GF(2^m) addition is trivial, and it can be performed with bit-wise XORing two operands. Multiplication is a computationally complex operation, and it is used in more complex operations such as exponentiation and inversion. Various bases are proposed in the literature for GF (2^m) such as polynomial basis, normal basis, and redundant basis [3]. The efficiency of finite field arithmetic depends on the choice of basis. Normal basis multiplication is more complex and requires more hardware than polynomial basis multiplication [7]. Redundant basis eliminates the modulo reduction [8], however, it involves embedding GF(2^m) in a cyclotomic field of higher order which requires more bits to represent field elements, resulting in more hardware. Polynomial basis provides more simple and regular structures without any basis conversion requirements [7], and is a better choice of basis. Every GF(2^m) is characterized by its field defining m^{th} degree polynomial called irreducible polynomial. The performance of a finite field multiplier also depends on the type of selected field irreducible polynomial. Various classes of irreducible polynomials such as generic, trinomials, pentanomials, equally spaced polynomials (ESP), and all one polynomials (AOP) are proposed in the literature [9]. It is observed that generic polynomials require more power and more area compared to all other classes.

☆ Fully documented templates are available in the elsarticle package on CTAN.

* Corresponding author.
E-mail address: srk100p@student.nitw.ac.in (S.R. Pillutla).

https://doi.org/10.1016/j.aeue.2019.153017
1434-8411/

However, generic polynomials are preferred as other specific classes based implementations are not suitable for all applications. Finite field multiplication implementations with generic polynomials can be used for a wide range of applications especially for ECC based cryptography [10].

Various GF(2^m) multiplier design styles are proposed in the literature. Depending on the style of interfaces for applying the operands and taking the result, bit-serial [11,12], bit-parallel [13,14], and digit-level [8] structures are developed. Several architectures such as sequential, parallel, systolic, semi-systolic, and pipelined are developed depending on the organization of computation. However, bit-serial sequential multipliers are highly area-efficient and suitable for constrained applications.

Several algorithms and architectures for polynomial basis multiplication have been proposed in the literature to achieve better area and time complexities. Beth et al. [15] presented various bit-level serial-in parallel-out architectures. In these architectures, one input loaded in parallel and another one loaded serially one bit per clock cycle requiring a total of m clock cycles. Song et al. [16] presented a new polynomial basis bit-serial multiplier suitable for broadcast structures. A bit-serial systolic multiplier combined with squarer was presented in [17]. In [18], a serial multiplier was presented with an area-efficient FPGA implementation. In [19], a bit-level parallel-in serial-out polynomial basis multiplier was presented where two inputs are preloaded and output is generated one-bit per clock cycle. Deschamps et al. [9] presented an implementation of bit-serial multipliers using least-significant-bit first and most-significant-bit first algorithms. Imaña [20] proposed a new low latency parallel-in/parallel-out sequential polynomial basis multiplier over GF(2^m). It is a partially versatile multiplier and applicable to many irreducible polynomials. In [21], a low-power and high-speed bit-serial versatile multiplier was proposed which is flexible with field size as well as irreducible polynomial. Kim et al. [22] proposed a bit-serial systolic multiplier using Montgomery multiplication. Ho H [23] proposed a versatile sequential multiplier for a class of fields. El-Razouk et al. [24] presented a new bit-level serial GF(2^m) multiplier. It is a fully serial-in parallel-out multiplier which does not require preloading of multiplicands. Mathe et al. [25] presented a sequential polynomial basis multiplier for generic irreducible polynomials with a latency of m clock cycles. This architecture is designed to take one operand in parallel and another operand serially during computation. It is a versatile multiplier in the view that it is applicable to any irreducible polynomial over GF(2^m). In this paper, we propose a modified algorithm employing interleaved modular reduction multiplication approach [26] that is available in the literature. The modification involves formulating the algorithm employing more efficient logical relations and thereby achieving hardware efficiency as well as improved regularity. Derivation of these efficient logical relations is based on the fact that NAND gate has lower area and time complexities when compared to AND gate complexities, which can result in hardware efficiency and lower critical path delay [27,28]. An area-efficient sequential architecture is also developed for this proposed multiplication algorithm for GF(2^m) for generic irreducible polynomials. The performance comparisons based on area and area-delay estimations of the proposed multiplier with previous works is presented. In addition, ASIC implementation results of the proposed multiplier are presented.

The rest of the paper is organized as follows. A brief review of polynomial basis multiplication is presented in Section 2. Mathematical formulation of the proposed algorithm and the proposed architecture are presented in Section 3. Architectural complexities and comparisons of the proposed multiplier with existing multipliers are presented in Section 4. ASIC implementation results of the proposed multiplier are presented in Section 5. Finally, Section 6 presents the conclusions.

2. A brief review of Polynomial basis representation and multiplication in GF(2^m)

In this section, a brief review of polynomial basis representation in GF(2^m) and multiplication operation using the same basis are presented. Every GF(2^m) finite field contains 2^m elements. Each element of the field can be represented with a polynomial of degree less than or equal to $(m-1)$ over GF(2). Polynomials over GF(2) indicates that the coefficients of the polynomials come from the ground field GF(2) whose elements are 0 and 1. Every finite field contains at least one irreducible polynomial over GF(2) associated with it. The root of this irreducible polynomial is a field element whose individual powers ranging from 0 to $(m-1)$ form the polynomial basis with m elements. Multiplication of GF(2^m) elements involves addition and multiplication of coefficient elements coming from the base field GF(2). Multiplication in GF(2) is performed by the logical AND operation, and addition is performed by the logical XOR operation. For a given field order, m, the result of arithmetic operations depends on the irreducible polynomial selected for the field. The multiplication operation in GF(2^m) involves the usual multiplication of two polynomial elements followed by modular reduction using the selected irreducible polynomial. For a GF (2^m), the general form of the irreducible polynomial $R(x)$ is given by a monic polynomial of the form

$$R(x) = x^m + \sum_{j=m-1}^{1} r_j x^j + 1 \tag{1}$$

with at least one of r_j's to be non zero and all $r_j \in$ GF(2).

Let $\alpha \in$ GF(2^m) be a root of the field irreducible polynomial $R(x)$ over GF(2^m). Then, the polynomial basis is constituted by the following set of m elements given by $(1, \alpha, \alpha^2, \alpha^3, \ldots\ldots, \alpha^{m-1})$, and it follows $R(\alpha) = 0$. Let A and B be two arbitrary elements of GF(2^m) represented in polynomial basis as

$$A(\alpha) = \sum_{j=0}^{m-1} a_j \alpha^j = a_{m-1}\alpha^{m-1} + a_{m-2}\alpha^{m-2} + \ldots\ldots + a_1\alpha + a_0 \tag{2}$$

$$B(\alpha) = \sum_{j=0}^{m-1} b_j \alpha^j = b_{m-1}\alpha^{m-1} + b_{m-2}\alpha^{m-2} + \ldots\ldots + b_1\alpha + b_0 \tag{3}$$

where all $a_j, b_j \in$ GF(2). Then the product polynomial $P(\alpha)$ is given by

$$P(\alpha) = (A(\alpha) \times B(\alpha)) \bmod R(\alpha) \tag{4}$$

Rewriting Eq. (4) by using Eq. (2) and Eq. (3) gives

$$\begin{aligned} P(\alpha) = {} & \left(\left(a_{m-1}\alpha^{m-1} + a_{m-2}\alpha^{m-2} + \ldots\ldots + a_1\alpha + a_0\right)\right. \\ & \left.\times\left(b_{m-1}\alpha^{m-1} + b_{m-2}\alpha^{m-2} + \ldots \quad \ldots + b_1\alpha + b_0\right)\right) \bmod R(\alpha) \end{aligned} \tag{5}$$

Evaluation of Eq. (5) gives $P(\alpha)$ to be a $(m-1)^{th}$ degree polynomial as

$$P(\alpha) = p_{m-1}\alpha^{m-1} + p_{m-2}\alpha^{m-2} + \ldots\ldots + p_1\alpha + p_0 \tag{6}$$

where all $p_j \in$ GF(2).

Since GF(2^m) can also be viewed as an m-dimensional vector space over GF(2), the coordinate sets $(a_{m-1}, a_{m-2}, \ldots, a_0)$ and $(b_{m-1}, b_{m-2}, \ldots, b_0)$ corresponds to GF(2^m) elements A and B, respectively and irreducible polynomial $R(x)$ is denoted with the set $(r_{m-1}, r_{m-2}, \ldots, r_1, 1)$.