Computers and Electrical Engineering 46 (2015) 29–45

Contents lists available at ScienceDirect

Computers and Electrical Engineering

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

Versatile digit serial multipliers for binary extension fields [☆]

Bilal Uslu, Serdar Süer Erdem [*]

The Department of Electronics Engineering, Gebze Technical University, 41400 Gebze, Kocaeli, Turkey

A R T I C L E   I N F O

Article history:
Received 23 January 2015
Received in revised form 4 July 2015
Accepted 8 July 2015
Available online 23 July 2015

Keywords:
Binary extension fields
Digit serial multiplier
Polynomial basis
Elliptic curve cryptography

A B S T R A C T

This work investigates the digit serial polynomial basis multipliers performing multiplication in multiple binary extension fields $\mathbb{F}_{2^{m_1}}, \mathbb{F}_{2^{m_2}}, \ldots, \mathbb{F}_{2^{m_\lambda}}$. Designing such versatile multipliers encounters a number of difficulties. First of all, the element sizes of the supported fields are different from each other, and thus the elements are represented with different number of bits for each field. To deal with different sized elements, designs with left or right justified operands are investigated. Secondly, each field multiplication involves modular reduction with a different irreducible polynomial, and thus the complexity can increase rapidly with the number of supported fields λ. To prevent this, two methods are studied: Using sparse irreducible polynomials and unifying the modular reduction computation of the fields by choosing the irreducible polynomials suitably. Our work shows that multiple fields can be supported at the cost of an $\mathcal{O}(\lambda)$ increase in area and an $\mathcal{O}(\sqrt{\lambda})$ increase in time.

1. Introduction

The arithmetic of the binary extension field $\mathbb{F}_{2^m}$ is commonly used in cryptography and coding theory applications [1,2]. Important cryptographic applications such as elliptic curve cryptography need large number of field multiplications in $\mathbb{F}_{2^m}$ to perform cryptographic transformations [3,4]. Also, these applications use quite large fields. For example, the field size m is typically selected from the range $160 \leqslant m \leqslant 1000$ in elliptic curve cryptography. Thus, efficient hardware and software implementations of the field multiplication is crucial to reduce the cost of cryptographic systems.

In this work, we present digit serial $\mathbb{F}_{2^m}$ multiplier architectures supporting multiple field sizes m. The proposed architectures support a set of field sizes $m_1, m_2, \ldots, m_\lambda$, i.e. work in any one of the fields $\mathbb{F}_{2^{m_1}}, \mathbb{F}_{2^{m_2}}, \ldots, \mathbb{F}_{2^{m_\lambda}}$ according to the user selection. We also analyze the increase in the cost due to supporting multiple fields. Moreover, as a case study, we investigate the multipliers supporting the five NIST fields $\mathbb{F}_{2^{163}}$, $\mathbb{F}_{2^{233}}$, $\mathbb{F}_{2^{283}}$, $\mathbb{F}_{2^{409}}$, and $\mathbb{F}_{2^{571}}$, recommended for elliptic curve cryptography [5,6].

The proposed digit serial multipliers use polynomial basis. That is, $\mathbb{F}_{2^m}$ elements are represented by the binary polynomials of degree less than m $\{a(x) = \sum_{i=0}^{m-1} a_i x^i | a_i \in \mathbb{F}_2\}$ where x represents the root of some degree m irreducible polynomial $x^m + g(x)$ called generator. The product $f(x)$ of any two elements $a(x)$ and $b(x)$ is computed by the polynomial multiplication $a(x)b(x)$ modulo the generator polynomial $x^m + g(x)$. The digit serial multipliers keep the coefficients of each polynomial operand in an array of digits. One of the operands is multiplied by the digits of the other operand, one digit at a time. Each of these partial product computations is interleaved with a modular reduction step and an accumulation step.

☆ Reviews processed and approved for publication by the Editor-in-Chief.
* Corresponding author.
E-mail addresses: bilaluslu@hotmail.com (B. Uslu), serdem@gtu.edu.tr (S.S. Erdem).

http://dx.doi.org/10.1016/j.compeleceng.2015.07.006
0045-7906/© 2015 Elsevier Ltd. All rights reserved.

There are several works studying polynomial basis digit serial multipliers in the literature [7–13] but these multipliers have been developed to support a particular field size m. The work in [7] introduces efficient digit serial multipliers using polynomial basis. The work in [11] investigates optimum digit sizes and the effects of using multiple accumulators. The work in [12] proposes using T flip flops in the accumulators instead of D flip flops to reduce the area complexity.

The versatility is an important feature for hardware designs since the ASIC circuits cannot be altered after fabrication. Flexible implementations are possible with reconfigurable FPGA devices but FPGAs cannot compete with the ASIC in terms of performance, cost and power consumption. Naturally, customizable elliptic curve systems [14] and versatile multipliers [15–19] have been proposed in the literature but the proposed multipliers are all bit serial and there is not much work about versatile digit serial multipliers.

Let the supported field sizes be $m_1 < m_2 < \ldots < m_\lambda$. Then, the number of the bits used in the representation must be large enough to represent m_λ coefficients. When a field with smaller size $m_k < m_\lambda$ is selected, the elements $a(x) = \sum_{i=0}^{m_k-1} a_i x^i \in \mathbb{F}_{2^{m_k}}$ must be justified either to the right or to the left. When $a(x)$ is right justified, the leading $m_\lambda - m_k$ coefficients are set to zero as follows.

$$a(x) = 0x^{m_\lambda-1} + \cdots + 0x^{m_k} + a(x)$$

In left justified representation, $a(x)$ is shifted, then the trailing $m_\lambda - m_k$ coefficients are set to zero as follows.

$$\widehat{a}(x) = x^{m_\lambda-m_k} a(x) + 0x^{m_\lambda-m_k-1} + \cdots + 0x + 0$$

Also, the fields $\mathbb{F}_{2^{m_1}}, \mathbb{F}_{2^{m_2}}, \ldots, \mathbb{F}_{2^{m_\lambda}}$ supported by the multipliers use different generator polynomials $x^{m_k} + g^{(k)}(x)$ for $k = 1, 2, \ldots, \lambda$. Naturally, the modular reduction by each $x^{m_k} + g^{(k)}(x)$ can be performed with a separate circuit. Nevertheless, the reduction circuits can be unified when the generator polynomials $x^{m_k} + g^{(k)}(x)$ are appropriately chosen.

The paper is organized as follows. Section 2 introduces the basic polynomial basis digit serial multipliers supporting a single field size and presents a detailed analysis of their complexities. Section 3 proposes digit serial multiplier architectures supporting multiple field sizes. Some of these work with right justified operands and some of them work with left justified operands. Also, some of these employ separate reduction circuits and some of them employ unified reduction circuits. Section 4 studies the complexities of the proposed multipliers. Section 5 presents a discussion of our results.

2. Digit serial multipliers using polynomial basis

Naturally, m bits are sufficient to represent an element $a(x) = \sum_{i=0}^{m-1} a_i x^i$ in the field $\mathbb{F}_{2^m}$. This is because each binary valued coefficient of $a(x)$ can be stored in a bit. Let the hardware digit size be w bits. A digit serial multiplier divides one of the operands, say $b(x)$, into $\lceil m/w \rceil$ digits as follows.

$$b(x) = \sum_{i=0}^{m-1} b_i x^i = \sum_{j=0}^{\lceil m/w \rceil - 1} B_j x^{wj}$$

where $B_j = \sum_{i=0}^{w-1} b_{wj+i} x^i$, is the jth digit of $b(x)$ and holds its consecutive w coefficients. Then,

$$a(x)b(x) \bmod (x^m + g(x)) = \sum_{j=0}^{\lceil m/w \rceil - 1} a(x) B_j x^{wj} \bmod (x^m + g(x)) \tag{1}$$

gives the field product. The multiplier computes this product, accumulating the partial products of the digits B_j in $\lceil m/w \rceil$ iterations. Thus, the multiplier gets faster as the digit size w increases. However, its area also increases because the digits B_j, and thus their partial products get larger. In each iteration, one of the partial products $a(x)B_j$ is computed and added to an intermediate result $I(x)$ where $\deg(I(x)) < m + w$. The intermediate result $I(x)$ can be split into degree $w - 1$ and $m - 1$ polynomials as follows.

$$I(x) = \sum_{i=0}^{m+w-1} I_i x^i = q(x)x^m + r(x) \tag{2}$$

where $q(x) = \sum_{i=0}^{w-1} I_{m+i} x^i$ and $r(x) = \sum_{i=0}^{m-1} I_i x^i$. The generator $x^m + g(x)$ is usually chosen such that

$$\deg(g(x)) = \mu \leqslant m - w. \tag{3}$$

for fast modular reduction. Then, the intermediate result can be reduced without division as follows.

$$I(x) \bmod (x^m + g(x)) = q(x)x^m + r(x) \bmod (x^m + g(x)) = g(x)q(x) + r(x) \tag{4}$$

This can be done because $\deg(g(x)q(x)) < m$ for any degree $w - 1$ polynomial $q(x)$ when Eq. (3) holds.