Microprocessors and Microsystems 33 (2009) 106–116

Contents lists available at ScienceDirect

Microprocessors and Microsystems

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

Fast point multiplication on Koblitz curves: Parallelization method and implementations

Kimmo Järvinen *, Jorma Skyttä

Helsinki University of Technology, Department of Signal Processing and Acoustics, Otakaari 5A, FIN-02150 Espoo, Finland

ARTICLE INFO

Article history:
Available online 14 August 2008

Keywords:
Elliptic curve cryptography
Field programmable gate array
Koblitz curves
Parallelism

ABSTRACT

Point multiplication is required in every elliptic curve cryptosystem and its efficient implementation is essential. Koblitz curves are a family of curves defined over $\mathbb{F}_{2^m}$ allowing notably faster computation. We discuss implementation of point multiplication on Koblitz curves with parallel field multipliers. We present a novel parallelization method utilizing point operation interleaving. FPGA implementations are described showing the practical feasibility of our method. They compute point multiplications on average in 4.9 μs, 8.1 μs, and 12.1 μs on the standardized curves NIST K-163, K-233, and K-283, respectively, in an Altera Stratix II FPGA.

1. Introduction

Koblitz [1] and Miller [2] independently suggested using elliptic curves in public-key cryptography in 1985. Since then elliptic curve cryptography has attained considerable amount of interest in the cryptographic research community, and increasingly also in the industry. The main reason for the attractiveness of elliptic curves is that shorter keys can be used for attaining similar level of security than in traditional public-key cryptography schemes based on the difficulty of integer factorization or discrete logarithm. For example, elliptic curve cryptography achieves approximately the same level of security with 173 bits than RSA with 1024 bits [3]. Studies have shown that elliptic curve cryptography is superior to RSA also in terms of speed and area required in implementation [4–6].

Field programmable gate arrays (FPGAs) offer many advantages in implementing cryptographic algorithms because they provide fast performance and flexibility [7]. Hence, numerous studies on FPGA implementation of elliptic curve cryptography have been published including [6,8–20]. A comprehensive survey of the field is presented in [21].

The principal operation required in every elliptic curve cryptosystem is called point multiplication. Much effort has been allocated in developing methods for its efficient computation because it acts as the bottleneck in elliptic curve cryptosystems. A comprehensive review can be found in [22], for example. Point multiplication is commonly known to be an operation which is hard to parallelize because of data dependencies and much of the research has concentrated on studying effects of parallelization. Finite field multiplication dominates in the cost of point multiplication and, thus, reducing the number of field multiplications on the critical path is essential. The fastest method for computing point multiplication without precomputations, called Montgomery point multiplication, was presented by López and Dahab in [23]. Cheung et al. [13] and Rodríguez-Henríquez et al. [18] showed that Montgomery point multiplication can be efficiently computed with four parallel field multipliers.

In 1991, Koblitz [24] suggested using a special family of elliptic curves nowadays referred to as Koblitz curves. Point multiplication is considerably more efficient on them than on general curves. Koblitz curves have been widely studied in the academia and they have been included also in certain standards, such as [25–27]. Koblitz curves have attained some interest also in the FPGA community as they have been considered in [9,14–17].

The problem that we are addressing is that computations on Koblitz curves cannot be parallelized as efficiently as Montgomery point multiplication. We recently presented a study on parallelization of Koblitz curve computations in [14] but the methods presented are not as effective as those available for Montgomery point multiplication. Only three parallel field multipliers can be utilized in Koblitz curve point multiplication with the existing methods [14] whereas Montgomery point multiplication can efficiently use up to four multipliers [18,13]. Hence, the more multipliers are available the smaller is the benefit of using Koblitz curves. Koblitz curves are faster than general curves even if parallel field multipliers are available, but the difference becomes smaller which makes Koblitz curves less attractive.

The contributions of our paper are twofold:

(1) We present a simple and efficient method for speeding up Koblitz curve computations when parallel field multipliers

* Corresponding author. Tel.: +358 9 451 5176; fax: +358 9 452 3614.
E-mail addresses: kimmo.jarvinen@tkk.fi (K. Järvinen), jorma.skytta@tkk.fi (J. Skyttä).

0141-9331/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.micpro.2008.08.002

are available. The method is based on point operation interleaving and it achieves full field multiplier utilization with two or four field multipliers. Montgomery point multiplication can be implemented efficiently with two or four multipliers as well [18,13]. Hence, both Koblitz curves and general curves can achieve their full potential in the same hardware. Our method implies that Koblitz curves are approximately three times faster than general curves. Furthermore, our method can be applied also in point multiplication algorithms requiring precomputations whereas methods presented in [18,13] cannot which increases the difference even further.

(2) We describe a highly optimized architecture based on our method which achieves very fast point multiplication times. The feasibility of the method and the architecture is demonstrated by providing several implementations on an Altera Stratix II FPGA. The implementations achieve average point multiplication times of only 4.9 μs, 8.1 μs, and 12.1 μs on K-163, K-233, and K-283, respectively, which are curves recommended by National Institute of Standards and Technology (NIST) [25]. We also study the effects of field basis selection and conclude that polynomial basis provides faster results than normal basis.

The remainder of the paper is organized as follows. Section 2 presents the preliminaries of elliptic curve cryptography and discusses computation of point multiplication and Koblitz curves. We describe our method in Section 3. Implementations showing the practical feasibility of the method are presented in Section 4 and implementation results are given and compared to other published results in Section 5. We end with conclusions and suggestions for future research in Section 6.

2. Preliminaries

Elliptic curves defined over finite binary fields, denoted by $\mathbb{F}_{2^m}$, are commonly used in practical cryptosystems. These curves are called binary curves. We consider binary curves of the following form:

$$E : y^2 + xy = x^3 + ax^2 + b, \tag{1}$$

where $a, b \in \mathbb{F}_{2^m}$ with $b \neq 0$. Henceforth, curves of this form are called general curves.

Let $E(\mathbb{F}_{2^m})$ denote the set of points on E. A point (x, y) is in $E(\mathbb{F}_{2^m})$ if it satisfies Eq. (1). Also a point called the point at infinity, $\mathcal{O}$, is a point in $E(\mathbb{F}_{2^m})$. Points in $E(\mathbb{F}_{2^m})$ form an additive abelian group with $\mathcal{O}$ as an identity element. The additive operation of the group is referred to as point addition and it is defined as $P_3 = P_1 + P_2$ where $P_i \in E(\mathbb{F}_{2^m})$.

Point multiplication, which is a basic component of every elliptic curve cryptosystem, is defined by using point additions as follows:

$$Q = kP = \underbrace{P + P + \cdots + P}_{k \text{ times}} \tag{2}$$

where $Q, P \in E(\mathbb{F}_{2^m})$ and k is an integer. P is called the base point and Q is the result point. Security of elliptic curve cryptosystems is based on the difficulty of solving the inverse operation of point multiplication called elliptic curve discrete logarithm problem (ECDLP), i.e., the problem of finding k if P and Q are given.

Point multiplication decomposes into three levels of hierarchy from top to bottom as follows:

- Point multiplication,
- Point operations, and
- Finite field arithmetic.

These hierarchy levels are discussed in the following sections by concentrating on the subjects that are most relevant for this paper. Our contributions target to the two highest levels of the hierarchy, and they are considered in detail in Sections 2.1 and 2.2. The lowest level is also considered shortly in Section 2.3. Finally, Koblitz curves are discussed in Section 2.4.

2.1. Point multiplication

Point doubling is a special case of point addition, $P_3 = P_1 + P_2$, where $P_1 = P_2$ and, henceforth, point addition refers solely to the operation $P_3 = P_1 + P_2$ where $P_1 \neq P_2$. Point additions and point doublings can be used in computing Eq. (2) when the integer k is represented with binary expansion as

$$k = \sum_{i=0}^{\ell-1} k_i 2^i, \quad \text{where} \quad k_i \in \{0, 1\}. \tag{3}$$

The simplest point multiplication algorithm is called double-and-add algorithm and it scans the bits of k in order starting either from the least significant bit (lsb) or from the most significant bit (msb). Point doubling is performed for every bit, but point addition is required only if $k_i = 1$. The length of k is $\ell \approx m$ and, thus, the double-and-add algorithm requires on average m point doublings and $m/2$ point additions.

Because point addition is not needed if $k_i = 0$, it is of interest to reduce the number of nonzeros. A simple, but effective, option is to use a signed-bit representation, i.e., $k_i \in \{0, \pm 1\}$, called non-adjacent form (NAF). NAF has the property that adjacent bits are never both nonzeros, i.e., $k_i k_{i+1} = 0$ for all i. Every k has a unique NAF and it has the minimum number of nonzeros among all signed-bit representations. Let $H(k)$ denote the Hamming weight of k, i.e., the number of nonzeros in k. When k is in NAF, $H(k) \approx m/3$. NAF is especially useful in point multiplication because point subtraction, $P_3 = P_1 - P_2 = P_1 + (-P_2)$, has roughly the same cost as point addition. When integers are in NAF, a modification of the double-and-add algorithm is used called double-and-add-or-subtract algorithm. Otherwise it is similar to the double-and-add algorithm, but point subtraction is computed when $k_i = -1$. Thus, the average cost of point multiplication is m point doublings and $m/3$ point additions or point subtractions.

Further reductions in computational requirements can be achieved by allowing precomputations involving the base point P. Such methods include window methods and combings, for example, but they are not considered in depth in this paper. However, the proposed method can be used also for such methods as will be discussed in Section 3.

Montgomery's ladder [28] is a point multiplication algorithm which performs both point addition and point doubling in every iteration of the algorithm. Hence, it has the cost of m point doublings and m point additions. The efficiency of Montgomery's ladder arises from the fact that point multiplication can be computed without information of the y-coordinate. Thus, the main loop of the algorithm operates only on the x-coordinate and the y-coordinate of the result point is retrieved in the end. This leads to very efficient point addition and point doubling. An adaptation of Montgomery's idea for binary curves was presented by López and Dahab in [23] and, henceforth, Montgomery point multiplication refers to their algorithm.

2.2. Point operations

The traditional point representation with two coordinates as (x, y) is referred to as the affine coordinate representation, or $\mathcal{A}$ for short. If $P = (x, y)$, point negation is given by $-P = (x, x + y)$. Point doubling, point addition, and point subtraction all require an inver-