Microprocessors and Microsystems 39 (2015) 1139–1155

Contents lists available at ScienceDirect

Microprocessors and Microsystems

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

Comparing design approaches for elliptic curve point multiplication over $GF(2^k)$ with polynomial basis representation

Apostolos P. Fournaris [a,*], Ioannis Zafeirakis [b], Paris Kitsos [a], Odysseas Koufopavlou [b]

[a] Computer Informatics Engineering Dpt., Technological Educational Institute of Western Greece, Greece
[b] Electrical and Computer Engineering Dpt, University of Patras, Rio Campus 26500, Greece

ARTICLE INFO

Article history:
Available online 6 August 2015

Keywords:
Elliptic curve cryptography
Finite field computation
VLSI design

ABSTRACT

Point Multiplication (PM) is considered the most computationally complex and resource hungry Elliptic Curve Cryptography (ECC) mathematical operation. PM hardware accelerator design can follow several approaches that lead to a fast, small or flexible implementation, meeting related application specifications. However, each PM design decision has certain outcomes in utilized hardware resources and computation speed. Such a key design decision is related to the structure of the $GF(2^k)$ multipliers to be employed in the PM accelerator. In this paper, we highlight the $GF(2^k)$ multiplication role in the overall PM performance and investigate what are the trade-offs on a PM accelerator when using bit serial or bit parallel multiplication approach in terms of speed, chip covered area and flexibility. To achieve this goal, we estimate these tradeoffs for a single point operation and specify realistic design cases for bit serial and bit parallel multiplier based PM design approaches. To evaluate the theoretical modeling, a point operation design methodology based on the parallelism and rescheduling of $GF(2^k)$ operations is proposed. This design approach is adapted to two characteristic PM algorithm realizations, the traditional double & add algorithm and the side channel attack resistant Montgomery power ladder algorithm. Our goal is to assess the resulting PM accelerator overall performance so as to achieve high speed with an acceptable cost on chip covered area (hardware resources). Using this methodology, PM is performed in series of $GF(2^k)$ parallelism stages. To test the proposed methodology, 8 PM accelerator use cases are identified that can offer high speed, flexibility, side channel attack resistance or small chip covered area. To provide fair comparisons and results, a common PM architecture is devised and the use case PM accelerators are implemented in FPGA technology. Depending on the designers goal, the proposed architectures and 8 implementations can offer the benefit of either high speed (the proposed work is currently one of the fastest known $GF(2^k)$ bit parallel multiplier based PM realization) or flexibility with reasonable compromises in chip covered area.

1. Introduction

ECC due to its small key sizes and applicability to wide range of cryptographic schemes, constitutes an elegant solution for many security domains. However, the complex EC mathematics still torment cryptographic engineers trying to provide a computationally efficient EC cryptosystem. Point multiplication (PM), a key operation used in all EC cryptosystems like EC Diffie–Hellman (ECDH) key agreement protocol and EC Digital Signature Algorithm (ECDSA) scheme, is computationally complex and hardware resource hungry. Thus, many researchers have invested considerable effort in designing efficient PM hardware accelerator architectures in terms of resources (memory, chip covered area) and speed (time delay) both for $GF(2^k)$ or $GF(p)$ based ECs. In most $GF(2^k)$ approaches [1–3], various optimizations are proposed in the associated $GF(2^k)$ operations (especially $GF(2^k)$ multiplication) of a PM architecture, for the various $GF(2^k)$ field element representations (e.g. polynomial or normal basis representation).

In the most widely adopted representation of $GF(2^k)$ elements (polynomial representation) the field is defined over a specific monic, irreducible polynomial $f(x)$ and each such element is expressed as a k-degree polynomial with coefficient of either 1 or 0. $GF(2^k)$ addition is carry-free and can be realized as a XOR operation between two $GF(2^k)$ elements coefficients. $GF(2^k)$ inver-

* Corresponding author.
E-mail addresses: apofour@ieee.org (A.P. Fournaris), izafeirakis@di.uoa.gr (I. Zafeirakis), pkitsos@teimes.gr (P. Kitsos), odysseas@ece.upatras.gr (O. Koufopavlou).

http://dx.doi.org/10.1016/j.micpro.2015.07.009
0141-9331/

sion/division can be bypassed by transforming the EC point coordinates from the affine to the projective coordinate plane. $GF(2^k)$ multiplication between $GF(2^k)$ elements a(x) and b(x) is a moduli operation $a(x) \cdot b(x) \ mod \ f(x)$, that has crucial performance overhead on EC PM and cannot be replaced by any other operation. This operation can be realized in bit-serial, digit-serial or bit parallel fashion, leading to a wide variety of EC PM cryptosystem implementations [3,1]. Each multiplier design type has distinct space (required gate number) and time complexity (critical path delay, time to complete 1 multiplication) and along with the employed PM algorithm, dominates the performance overhead of the overall PM implementation. As recently pointed out in [4], determining the extent of influence/impact that the $GF(2^k)$ multiplier type has in various PM algorithm implementations can help PM designers choose an implementation approach that best fits a PM accelerator functional and non functional specifications (flexibility, high speed, low chip covered area, etc.).

Researchers have proposed bit serial $GF(2^k)$ multiplier based PM accelerators (BSPM) using the Most or Least Significant bit (MSB or LSB) [5–7] or $GF(2^k)$ Montgomery multiplication (MM) [8] algorithms in an effort to reduce area size. Other PM works use the Most or Least significant digit (MSD or LSD) algorithm as a tradeoff between speed and hardware resources overhead [9,2]. However, most researchers seem to agree that subquadratic bit parallel $GF(2^k)$ multiplier based PM accelerators (BPPM), following the Karatsuba–Offman (KO) algorithm and its extensions [10], are the most suitable for achieving high speed PM computation [11–13]. This high speed benefit can be gained by considerably restricting the designed multipliers flexibility. Efficient bit parallel multipliers are designed for a single $GF(2^k)$ field defined over a specific trinomial or pentanomial irreducible polynomial $f(x)$. Therefore, BPPM accelerator calculations are compatible only to a single EC defined over a specific $GF(2^k)$ field. On the other hand, BSPM accelerators using bit serial $GF(2^k)$ multipliers, although slower, offer high flexibility. The bit serial $GF(2^k)$ multiplier structure can be used for $GF(2^k)$ fields defined over any at most k-degree irreducible polynomials [8].

The BPPM overhead on chip covered area in the past has been considered too high for an efficient PM accelerator, however, recent works [11–14] have shown that high speed BPPM implementations, achieving PM time delays of a few μs (10.7 μs for $GF(2^{163})$ at best in Xilinx Virtex 4 FPGAs [13]), are possible with acceptable number of utilized hardware resources. Pipelining, parallelism, point operation scheduling and intricate FPGA LUT placement and routing are some techniques used in the above works. However, BPPMs, as mentioned above, still lack in flexibility since the EC specifications (including the underlined $GF(2^k)$ field) cannot be altered at run time [2]. The fact that BSPMs do not suffer from the above problem along with the bit serial $GF(2^k)$ multiplier small hardware resource requirements make BSPM beneficial in applications where flexibility and low hardware resources are required. Such BSPM designs however should not introduce excessive computation time delays.

In view of the above State of the Art PM accelerator research field, it is evident that most works aim at proposing the most efficient implementation for a specific PM accelerator type in comparison to other similar type of implementations. To achieve that, they employ techniques that are intrinsic to a specific PM type (BSPM or BPPM), a specific PM algorithm or a specific hardware technology (e.g. FPGA technology BRAMs, DSP processors etc., hard LUTs). However, efficiency in terms of speed or hardware resources is not always the only criteria for choosing a PM accelerator for a security application. Each application environment can have unique constrains (e.g. flexibility or side channel attack resistance) that may prohibit the use of the most efficient PM accelerator implementation. So, the design decision for such environment could be a complex tradeoff between various design parameters in order to meet such environment's constrains. This issue highlights the need for a design strategy that will take into account such parameters in order to create an adapted-to-the-needs PM accelerator implementation. Such design approach, is different in principle to other State of The Art works since it aims at answering the question of which PM accelerator type, and under which design parameters, is the best for a specific set of application specifications and not how can a specific PM type should be made more efficient in general. Based on the previous paragraphs' discussion, one of the most influential parameters to a PM accelerator implementation is the employed $GF(2^k)$ multiplier (type and/or number) as well as the utilized PM algorithm.

In this paper the trade-off between BSPM and BPPM accelerators for various PM algorithms and use case scenarios is investigated in an effort to propose a design strategy that is technology agnostic and can be applicable to diverse ECC application environments (with various constrains). Our research is an extension of our work done in [4] where possible trade-offs between the two accelerator types were identified only in the single scenario of the traditional (double & add) PM algorithm. However, in this paper, the impact of the underlined $GF(2^k)$ multiplier as a parameter of the PM accelerator performance is studied more extensively and not only for the double & add but also for side channel attack resistant (Montgomery power ladder) PM algorithms. To provide fair results, a design strategy based on $GF(2^k)$ operation parallelism and scheduling is proposed that is applicable to all the design cases and can benefit both type of approaches. In the proposed strategy, each point operation is decomposed to its basic $GF(2^k)$ operations. Data dependencies between $GF(2^k)$ operations are identified even if such operations belong to a different point operation (that can be performed in parallel or as a consecutive point operation) and are categorized in parallelism stages. All $GF(2^k)$ operations in each stage are performed in parallel. The series of the parallel $GF(2^k)$ operations are scheduled following the PM algorithm in relation to the adopted $GF(2^k)$ multiplier type so as to offer an optimal trade-off between chip covered area and speed. Outcome of this approach is the proposal and implementation of several different PM accelerator architectures using bit serial or bit parallel $GF(2^k)$ multipliers that when compared with similar works prove to be among the fastest implementations currently available.

The rest of the paper is structured as follows. In Section 2, a brief mathematical background on EC arithmetic is provided. In Section 3 an overview of $GF(2^k)$ multipliers is made and in Section 4 the proposed design methodology employed in our work is analyzed. In Section 5 the hardware architecture is presented and implementation performance results in comparison with other works are discussed in Section 6. Section 7 concludes the paper.

2. Elliptic curve cryptography

A non-super singular EC defined over a finite field F is the set of solutions (x, y) where $x, y \in F$, of an EC equation E. Focusing on ECs defined over binary extension fields $GF(2^k)$, the most widely accepted, EC equation form is the short Weierstrass equation $E : y^2 + xy = x^3 + ax^2 + b$, where $a, b \in GF(2^k)$ and $b \neq 0$. Three point operations are defined over the additive EC Group. Point addition (PA) is the operation of adding an EC point P_0 to an EC point P_1 to obtain a third EC point $P_2 (P_2 = P_0 + P_1)$. Point Doubling (PD) is the operation adding an EC point P_1 to itself to obtain the EC point $P_3, (P_3 = 2P_1)$. Point multiplication (PM) is the operation of