Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

A low-power and low-energy flexible GF(p) elliptic-curve cryptography processor*#

Hamid Reza AHMADI, Ali AFZALI-KUSHA

(*School of Electrical and Computer Engineering, University of Tehran, P.O. Box 14395-515, Tehran, Iran*)

E-mail: {hrahmadi, afzali}@ut.ac.ir

Received Oct. 30, 2009; Revision accepted Mar. 27, 2010; Crosschecked Aug. 2, 2010

Abstract: We investigate the use of two integer inversion algorithms, a modified Montgomery modulo inverse and a Fermat's Little Theorem based inversion, in a prime-field affine-coordinate elliptic-curve crypto-processor. To perform this, we present a low-power/energy GF(p) affine-coordinate elliptic-curve cryptography (ECC) processor design with a simplified architecture and complete flexibility in terms of the field and curve parameters. The design can use either of the inversion algorithms. Based on the implementations of this design for 168-, 192-, and 224-bit prime fields using a standard 0.13 μm CMOS technology, we compare the efficiency of the algorithms in terms of power/energy consumption, area, and calculation time. The results show that while the Fermat's theorem approach is not appropriate for the affine-coordinate ECC processors due to its long computation time, the Montgomery modulo inverse algorithm is a good candidate for low-energy implementations. The results also show that the 168-bit ECC processor based on the Montgomery modulo inverse completes one scalar multiplication in only 0.4 s at a 1 MHz clock frequency consuming only 12.92 μJ, which is lower than the reported values for similar designs.

Key words: Elliptic-curve cryptography (ECC), Prime field, Montgomery multiplication, Montgomery inverse, Low-energy
doi:10.1631/jzus.C0910660 **Document code:** A **CLC number:** TN4

1 Introduction

Standard public-key cryptography (PKC) algorithms and protocols have been used as the basis for providing security in many real-life applications (Stamp, 2006). The implementation of these algorithms for use in applications based on energy/area constrained devices, like radio-frequency identification (RFID) tags and wireless sensors, has been the subject of many recent research activities (Kaps, 2006; Kumar, 2006). These implementations are difficult due to the limitations of the power/energy consumption and calculation time and also due to the complex and calculation-intensive nature of the algorithms (Kaps, 2006). Amongst standard PKC algorithms, elliptic-curve cryptography (ECC) algorithms have the advantage of providing an equal level of security using smaller numbers (Hankerson *et al.*, 2004). This makes ECC algorithms more suitable for use in applications with limitations in power/energy and timing (Kaps, 2006). Recently, ECC hardware implementations have been shown to be able to meet power/energy and timing limitations of these devices and applications (Öztürk *et al.*, 2004; Gaubatz *et al.*, 2005; Wolkerstorfer, 2005; de Dormale *et al.*, 2006; Batina *et al.*, 2007; Feldhofer and Wolkerstorfer, 2007; Fürbass and Wolkerstorfer, 2007).

Different approaches have been taken by researchers to design low-power/energy ECC processors. In many of these approaches, whilst the main goal has been to reach lower power consumption levels, researchers have concentrated on minimizing

* Project supported in part by the Iran Telecommunication Research Center (ITRC) and the Research Council of University of Tehran

This paper is the extension on the papers "Very low-power flexible GF(p) elliptic-curve crypto-processor for non-time-critical applications", which appeared in the Proceedings of the International Symposium on Circuits and Systems, Taipei, Taiwan, May 24–27, 2009, and "Low-power low-energy prime-field ECC processor based on Montgomery modular inverse algorithm", which appeared in the Proceedings of EUROMICRO Conference on Digital System Design, Architectures, Methods and Tools, Patras, Greece, Aug. 27–29, 2009

the total calculation time at the same time to make their implementations applicable to a wider range of applications (Öztürk *et al.*, 2004; Gaubatz *et al.*, 2005; Wolkerstorfer, 2005; de Dormale *et al.*, 2006; Batina *et al.*, 2007; Feldhofer and Wolkerstorfer, 2007; Fürbass and Wolkerstorfer, 2007). In a hardware ECC processor, the total calculation time may be reduced either by using a higher clock frequency or by reducing the total number of clock cycles necessary to complete the calculations. The use of a higher clock frequency directly translates to more power consumption and complicates the design process (Öztürk *et al.*, 2004). Hence, to minimize the calculation time while maintaining low power consumption, the method of choice is to reduce the number of necessary clock cycles, which can be achieved by selecting faster algorithms. The most time-consuming operation in a prime field ECC processor is the field inversion, which can be calculated with different algorithms (Hankerson *et al.*, 2004). Therefore, the choice of the algorithm for the field inversion operation will have a great impact on the calculation time. Obviously, this will also affect the power and energy consumption of the ECC processor.

In this paper, we present two low-power ECC processor hardware designs which may be considered for wireless sensor network and RFID tag applications. The designs, which are flexible prime-field ECC processors, differ only in the chosen field inversion algorithm. One design takes the Fermat's theorem approach for the implementation of the field inversion while the other uses the Montgomery modulo inverse algorithm. The designs are compared in terms of energy consumption, area, and calculation time.

2 Mathematical background of elliptic-curve cryptography

In cryptography, elliptic curves can be defined over prime number fields and over binary extension fields (Hankerson *et al.*, 2004). In this work, we consider only the elliptic curves over prime fields. The reasons for this choice will be provided in the following sections. A prime field, denoted by GF(p), is defined as the set of all integers modulo a prime number p (i.e., {0, 1, 2, ..., $p-1$}) together with the addition and multiplication operations on integers modulo p (Hankerson *et al.*, 2004). In the cryptography context, an elliptic curve E over GF(p) is defined as the set of points $P=(x, y)$ with $x, y\in$GF(p), which satisfy the simplified Weierstrass equation (Hankerson *et al.*, 2004) as

$$y^2 \equiv x^3 + a\,x + b \pmod{p}. \tag{1}$$

Here, $a, b\in$GF(p) and $4a^3+27b^2\neq 0$. Another point O at infinity is also included in the curve (Hankerson *et al.*, 2004). The field parameter is p whose length in bits is the field size, while a and b are the curve parameters. Note that the field size is the bit-length of the numbers corresponding to the size of the registers in the hardware implementation.

Over any curve E, a 'point addition' operation is defined where for every two points $P_1, P_2\in E$, there exists a point $P_3\in E$, $P_3=P_1+P_2$. The set of the points on the curve E together with the point addition operation forms an abelian group, with the point at infinity serving as the identity member. This abelian group is the basis for elliptic-curve cryptographic systems (Hankerson *et al.*, 2004). Based on the geometric properties of elliptic curves, the point addition operation is defined as follows:

$$\begin{aligned}&\forall\, P_1=(x_1,y_1),\ P_2=(x_2,y_2)\in E,\\ &\exists\, P_3=(x_3,y_3)\in E,\ \ni P_3=P_1+P_2,\\ &x_3=\Delta^2-x_1-x_2,\ y_3=\Delta(x_1-x_3)-y_1,\end{aligned} \tag{2}$$

$$\Delta=\begin{cases}\dfrac{y_2-y_1}{x_2-x_1}, & P_1\neq P_2,\\[2ex] \dfrac{3x_1^2+a}{2y_1}, & P_1=P_2.\end{cases} \tag{3}$$

Using repeated point additions, another operation called 'scalar point multiplication' is defined over the curve E, where for every point $P\in E$ and every integer $k\in$GF(p), there exists a point $Q\in E$, $Q=kP$.

$$Q=kP=\underbrace{P+P+P+\ldots+P}_{\text{add } k-1 \text{ times}}. \tag{4}$$

Elliptic curves defined over binary extension fields follow the same definitions but have slightly

different equations. Further details of the mathematics of the ECC can be found in Hankerson *et al.* (2004).

The cryptography using elliptic curves is based on the hardness of finding the unknown value k from known points P and Q when it is known that $Q=kP$. This is called the 'elliptic-curve discrete logarithm problem' (ECDLP). The cryptographic protocols based on elliptic curves are mathematical procedures that manipulate and hide the information (i.e., numbers or messages) using the hardness of ECDLP. The point addition and scalar point multiplication operations are performed by ECC processors, which are the building blocks of the systems based on ECC.

As deduced from Eqs. (2)–(4), to perform the scalar point multiplication, one needs to do addition (and subtraction), multiplication, and division (i.e., inversion) operations on integer numbers. All of these operations must be performed modulo the prime number p; i.e., the results of the calculations must be reduced modulo p, hence becoming a member of GF(p) (Hankerson *et al.*, 2004). The modulo addition may be easily done by the addition operation, followed by a conditional subtraction of the modulus p. The modulo multiplication and modulo inversion, however, are much more complex operations, and different methods exist for their calculation (Hankerson *et al.*, 2004).

2.1 Modulo multiplication algorithms

The modulo multiplication can be accomplished by first multiplying the numbers as ordinary integers and then reducing the result modulo p. In general, this reduction step requires an integer division, which is an expensive operation in terms of hardware cost and timing (Hankerson *et al.*, 2004). One way to avoid this cost is to limit the application to using only specially chosen modulus values that enable us to use fast reduction algorithms. Examples of such p values and the corresponding fast reduction algorithms can be found in the NIST recommendations (NIST, 2000). If these limitations cannot be accepted in a system, other relatively fast reduction methods like the Barrett reduction or the Montgomery multiplication can be used (Hankerson *et al.*, 2004).

The Montgomery multiplication (Montgomery, 1985) is a method for performing the modulo multiplication using only additions and binary shifts and hence is suitable for low-power implementations. For this to be possible, the ordinary integers must first be mapped into the so-called 'Montgomery domain'. Any number of modulo additions and multiplications can then be performed on the numbers, before mapping back to the ordinary integer domain. Each mapping of the numbers into and out of the Montgomery domain requires one Montgomery multiplication, which can be performed using the same hardware. The timing overhead of the mapping operations will be negligible when many modulo multiplications are to be calculated, as is the case in the ECC calculations. Therefore, to handle the modulo multiplication with general p values in ECC, the Montgomery multiplication is a reasonable choice.

3 Modulo inverse algorithms

In a prime-field ECC processor, the modulo inverse operation has the greatest effect on the timing of the calculations, especially for the implementations based on the affine coordinates. In this section, we briefly discuss different algorithms for performing the modulo inverse operation in prime number fields.

3.1 Algorithm based on Fermat's Little Theorem

The most favorable modulo inverse algorithm for use in low-area/low-power ECC hardware is based on Fermat's Little Theorem (Hankerson *et al.*, 2004). This algorithm performs the inversion by repeating the modulo multiplication operation using

$$\frac{1}{X} \equiv X^{-1} \equiv X^{p-2} \pmod{p}. \tag{5}$$

The modulo multiplier used for this operation is an essential part of the ECC processor hardware (Hankerson *et al.*, 2004), and hence the inversion operation may be implemented with negligible hardware overhead and a very small increase in power consumption (Hankerson *et al.*, 2004). Note, however, that this algorithm increases the number of clock cycles necessary for the ECC calculations (especially when the affine coordinates are used), and is suitable only for the applications where the calculation time is not a design concern (Hankerson *et al.*, 2004). As a result of the longer calculation time, the total energy consumption for each ECC operation will also be

higher. In this paper, we quantitatively evaluate the effect of using this algorithm on the calculation time of affine-coordinate processors.

3.2 Binary extended Euclidean algorithm

The extended Euclidean algorithm (EEA) determines the inverse of an integer in a number field using repeated division operations (Hankerson *et al.*, 2004). Since performing the division with hardware is very expensive, the binary version of the EEA replaces the division with shifts and subtractions, which can be implemented efficiently in hardware (Hankerson *et al.*, 2004). In comparison to the Fermat- based algorithm, the binary EEA needs additional hardware resources, but performs the inversion operation in many fewer clock cycles (Hankerson *et al.*, 2004). While the power consumption will be higher, the calculation time and the total energy consumption per ECC operation will be lower. Variants of the binary EEA have also been implemented for use in the ECC hardware (Wolkerstorfer, 2005; Fürbass and Wolkerstorfer, 2007).

3.3 Montgomery modulo inverse algorithm

This algorithm, which has been introduced by Kaliski (1995) and later completed by Savaş and Koç (2000), is a variant of the binary EEA, adjusted to manipulate the numbers in the Montgomery domain. The algorithm performs the inversion in the Montgomery domain, and is therefore more suitable than the EEA when the ECC design is based on the Montgomery multiplication (Hankerson *et al.*, 2004). It also works faster than the binary EEA for numbers in the Montgomery domain (Kaliski, 1995; Hankerson *et al.*, 2004).

4 Related works

An obvious design method to obtain a low-area (and hence low-power) hardware for calculation-intensive mathematics applications such as ECC, is to simplify the underlying equations. Different research groups have taken various approaches to this simplification.

Some researchers have confined their designs to the elliptic curves over the binary extension fields (de Dormale *et al.*, 2006; Batina *et al.*, 2007). Working on these fields, one can take advantage of the more simple carry-free binary field operations. In addition to using binary extension fields, the flexibility of the design parameters has been further reduced to meet the power limitations. Confining the design to binary fields and limiting the flexibility of the design parameters restrict the number of the applications that can use the final device, because any change in the security parameters will require that all of the tags/sensors be replaced. A more important drawback of these designs is that they cannot be used for higher level standard EC-based protocols like the elliptic curve digital signature algorithm (ECDSA) (Feldhofer and Wolkerstorfer, 2007). These higher level protocols, in addition to the ECC operations, are based on ordinary integer modulo operations (Hankerson *et al.*, 2004). For the binary field designs, as stated in Feldhofer and Wolkerstorfer (2007), these modulo operations must be undertaken either in a separate processor (which is not present in many of the RFID tags) or in separate hardware (which will consume additional power).

The same approach of limiting the flexibility of the design parameters for reducing the power consumption has also been used in designs based on elliptic curves over the prime fields. In Öztürk *et al.* (2004), Gaubatz *et al.* (2005), and Kaps (2006), special prime numbers were chosen to simplify the arithmetic implementations. However, these GF(p)-based designs will also need separate hardware for ordinary integer operations (Fürbass and Wolkerstorfer, 2007), imposing restrictions on the application.

Many other designs, which consider the case of the ECC over prime fields without limiting the flexibility, can be found in the literature (Daly *et al.*, 2004; Byrne *et al.*, 2007). These designs support GF(p) ECC calculations with any desired field or curve parameter values. Here, we focus only on those that have power consumption levels suitable for RFID tags and/or wireless sensors (Wolkerstorfer, 2005; Feldhofer and Wolkerstorfer, 2007; Fürbass and Wolkerstorfer, 2007). These designs have minimized the power consumption using, e.g., an optimum choice of the curve point coordinates, different mathematical methods, and design techniques such as bit-serialization.

A flexible GF(p) ECC processor for use in ECDSA calculations on RFID tags was presented in

Fürbass and Wolkerstorfer (2007). For the field multiplication, a radix-4 Montgomery multiplier with Booth recoding was used. The radix-4 was used to reduce the area-cycles product of the processor. To perform the field inversion, a variant of the EEA was used. Also, two separate units were designed for multiplication and inversion operations using a total of nine registers for the whole ECC processor. Finally, to minimize the critical paths, carry-save adders (CSAs) have been employed.

A more efficient design has been presented in Feldhofer and Wolkerstorfer (2007), a dual-field ECC processor performing both the prime and binary field operations. To reduce the power consumption, the authors made use of techniques such as clock-gating and glitch-elimination and used a bit-serial implementation of the Montgomery multiplier. Since their RFID tag application is not time-critical, the design was operated at a frequency of 100 kHz to further reduce the power consumption. Another similar dual-field ECC processor for RFID tag applications has been presented in Wolkerstorfer (2005). CSAs and redundant number representation were used to minimize the critical path delay for obtaining a higher maximum clock frequency. This design also uses the EEA for field inversion and the Montgomery method for field multiplication, and achieves low-power consumption by lowering the clock frequency to 175 kHz.

5 Target applications and design decisions

Several types of wireless sensors and RFID tags have been introduced, having a vast range of capabilities (Kaps, 2006). The main problem of these devices is the limited amount of power/energy available for the operation of the hardware. Also, many applications of these devices do not impose high clock frequency requirements.

Based on our discussion of the previously reported low-power ECC processors and the specifications of the target applications mentioned above, we made two design decisions. As discussed in Section 4, most of the designs used the approach of limiting the flexibility of the field type and/or curve parameters to simplify the implementation for lowering the power consumption of the hardware. Based on our explanations, we decided to design a flexible prime-field ECC processor with the additional ability to perform ordinary integer modulo operations with any required modulus value. The second decision was motivated by the fact that some of the reported designs attempted to reduce the critical path to obtain a higher maximum clock frequency. The designs, however, used a much lower clock frequency to reach the desired low-power specifications. Note that, most often, reducing the critical path is accompanied by an increase in the area and power consumption. Since high clock frequencies are not necessary for our target applications, we decided to focus mainly on lowering the power/energy consumption as our primary design concern.

In this paper, two ECC processor designs based on Fermat's theorem and Montgomery modulo inversion algorithms are investigated. The architectures of both designs are very similar enabling us to evaluate the efficiency of the algorithms.

6 Design of proposed elliptic-curve cryptography processors

In this section, we describe two designs of a flexible prime-field ECC processor suitable for use in power/energy-constrained devices like RFID tags and wireless sensors.

6.1 Features of the designs

Here we explain the features of the prime-field ECC processor implementations. Each of the processors can perform all the ECC cryptography operations by itself without requiring any additional hardware. Additionally, the processors can perform the integer modulo operations required for the implementation of the cryptography protocols (e.g., for protocols like ECDSA), mentioned in Section 4. It was decided that the designs would be completely flexible in terms of the field parameter (i.e., the modulus) and curve parameters without using any special hardware optimization for a specific prime field or curve. When synthesized for a specific field size, the processors can perform calculations (both the ECC and the integer modulo operations) for any chosen modulus with a length not larger than the designed field size. All of the field and elliptic curve parameters are given as inputs to the ECC processors. This guarantees the reusability of the hardware design

when a change in the security parameters is needed. As explained in Section 2, to have flexibility, the best way to perform modulo multiplication is the Montgomery multiplication. Therefore, all of the operations are performed on the numbers in the Montgomery domain.

The designs are based on the affine representation of the curve points. This simplifies the design, reduces the necessary storage elements to the minimum of two registers for a curve point, and leads to lower power consumption (Fürbass and Wolkerstorfer, 2007). Also, to further minimize the area and power consumption, we avoided introducing complexity throughout the design as much as possible. For example, instead of using a random access memory (RAM) block to store the intermediate variables (Wolkerstorfer, 2005; Batina *et al.*, 2007), we have used ordinary flip-flops as the storage elements. This, in addition to lowering the power consumption, simplified the controllers.

6.2 Architecture of the proposed ECC processors

The general architecture of both designs is shown in Fig. 1. The architecture, which is kept as simple as possible, consists of only two calculation units as the central part of the design. The two units, an adder/subtractor with the modulo reduction (Fig. 2a) and a Montgomery multiplier (Fig. 2b), are very similar to those used in Daly *et al.* (2004) and Byrne *et al.* (2007), with the difference that we have not merged them into one larger arithmetic logic unit (ALU). Our study shows that merging the two units, as done in Daly *et al.* (2004) and Byrne *et al.* (2007), increases the number of multiplexers, giving rise to an increase in the power consumption. Similar to what was performed in Byrne *et al.* (2007) to eliminate a conditional subtraction step between the consecutive Montgomery multiplications, the width of the registers is taken three bits longer than the field size, and each Montgomery multiplication is completed in n+2 clock cycles, where n is the field size. Further details of the functionalities of the units may be found in Byrne *et al.* (2007). A dedicated controller consisting of three inter-related finite state machines (FSMs) controls the adder/subtractor and multiplier units to perform the elliptic-curve point addition and doubling operations. In addition to the two dedicated registers of the Montgomery multiplier (B and M in Fig. 2b), this controller uses three registers to store the temporary values. These temporary registers are also three bits longer than the field size.

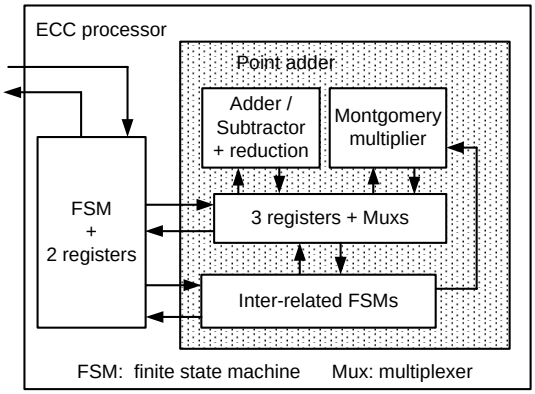

Fig. 1 Architecture of the elliptic-curve cryptography (ECC) processor
Redrawn from Ahmadi and Afzali-Kusha (2009b)

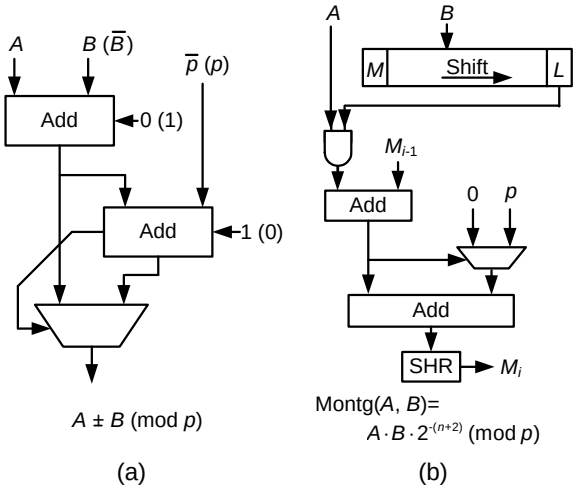

Fig. 2 Adder/Subtractor with modulo reduction (a) and Montgomery multiplier (b)
Redrawn from Ahmadi and Afzali-Kusha (2009a)

The controller (three inter-related FSMs) together with the three temporary registers, the necessary multiplexers, and the two calculation units, are called the 'point-adder' (Fig. 3). At the top level (Fig. 1), another controller uses the point-adder to perform the scalar point multiplication with the double-and-add method (Hankerson *et al.*, 2004). At this level, only two more registers are needed that will finally contain the coordinates of the resulting curve point. The value of the 'scalar' is shifted in bit-by-bit by the controller. This is the best choice to reduce the area and power consumption of the hardware.

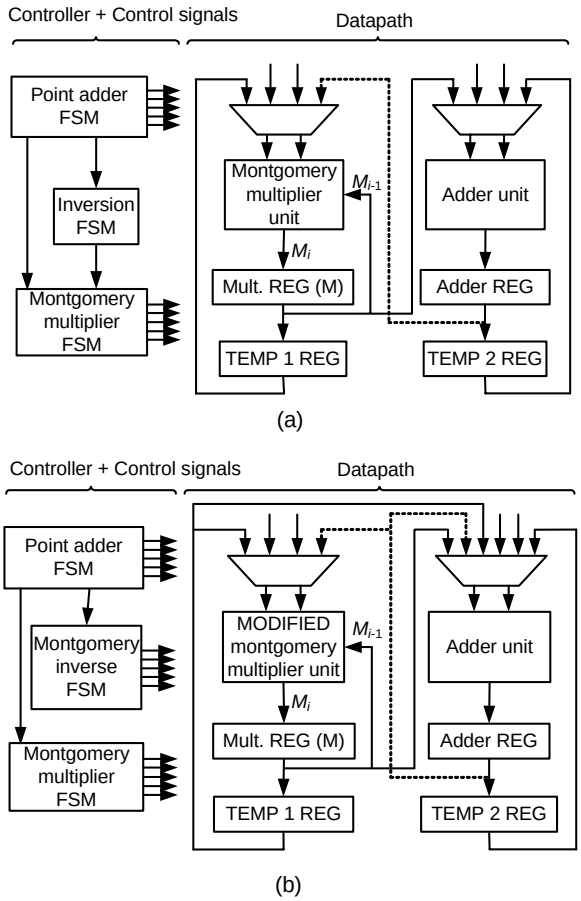

Fig. 3 Details of the 'point adder' block in the first design (a) and the second design (Montgomery inversion) (b)

In the first design, to achieve a very small hardware size and lower power consumption, we use the same adder/subtractor and multiplier units to perform the field inversion. To do this, Fermat's Little Theorem is used. As mentioned in Section 3, in this algorithm a prime-field inversion is transformed into multiple modulo multiplications, which can be performed using the same Montgomery multiplier, with no additional hardware units. The 'Inversion FSM' uses the 'Montgomery multiplier FSM' to perform the inversion (Fig. 3a). Since the FSM for multiplication is available, the inversion controller will be very small including only a four-state FSM and will not add any new controlling signals to the hardware units (Fig. 3a). Although the decision to use Eq. (5) for the inversion operation reduces the hardware cost to only a four-state FSM in the controller, it causes the calculation time to become very long, leading to higher energy consumption.

In the second design, we use a fast inversion algorithm to shorten the calculation time and achieve lower energy consumption. Since our ECC processor design is based on the Montgomery-domain numbers and uses the Montgomery multiplication, the best algorithm to perform field inversion is the Montgomery modulo inverse (MMI), as discussed in Section 3. Only three adders and four registers are needed to implement the Montgomery modulo inverse algorithm (Daly *et al.*, 2004). Therefore, we are able to re-use the existing units after some small modifications inside the Montgomery multiplier and adding some extra multiplexers. This way adding a separate unit for the inversion operation is avoided. In this design, a 'Montgomery inverse FSM' directly controls the hardware units to perform the Montgomery modulo inversion (Fig. 3b). The details of the implemented algorithm follow.

6.3 Modified Montgomery modulo inversion

The MMI algorithm, initially introduced in Kaliski (1995), consisted of a calculation phase and a halving phase. In both phases, only repeated additions (or subtractions) and binary shifts were used, making it suitable for hardware implementation. However, it could not be used in its original form in Montgomery ECC hardware, because when input with an integer mapped into the Montgomery domain, the algorithm automatically maps the result back to the ordinary integer domain (Savaş and Koç, 2000). Savaş and Koç (2000) showed that adding one extra Montgomery multiplication operation, either before or after the calculation phase in the algorithm of Kaliski (1995), can correct the result such that it could be used in Montgomery-domain ECC designs. Our study shows that adding the extra multiplication before the calculation phase requires no additional resources, leading to lower power consumption. In our design, we used the correction proposed in Savaş and Koç (2000) before the calculation phase of Kaliski (1995). Details may be found in Kaliski (1995) and Savaş and Koç (2000).

The halving phase of the algorithm of Kaliski (1995) should be modified further before being used in our Montgomery-domain ECC hardware. This modification is explained here. The halving phase of this algorithm for n-bit numbers is shown in Fig. 4. As mentioned above, each Montgomery multiplication in

our design takes n+2 cycles, where n is the field size. The halving phase in the algorithm of Kaliski (1995) must be corrected for these two extra cycles in our design. The corrected code is given in Fig. 5. Note that only two conditional shift operations are added, which has only a small effect on the hardware and power consumption.

6.4 Security of the ECC processors against SPA

Security against simple power analysis (SPA) is a key design criterion for every public-domain cryptography hardware (Hankerson *et al.*, 2004). The architecture of our ECC processors may be secured against SPA, as explained here. In the top level of our processor architecture, we have used a double-and-add method to perform scalar multiplication. In this method, the point addition operations are performed only when the bits of the 'scalar' are set. In addition, because of the difference between the point doubling and point addition operations, these operations can be distinguished from their timing behavior and/or power consumption. Therefore, an analysis of the power consumption of the hardware can reveal the value of the bits of the 'scalar' (Hankerson *et al.*, 2004).

```
for i=1 to k−n do
    if r is even then r ← r/2
    else r ← (r+p)/2
return p−r
```

Fig. 4 Halving phase of the inversion algorithm from Kaliski (1995)
Redrawn from Ahmadi and Afzali-Kusha (2009a)

```
L = k − (n+2)
if L = −2 then r ← r >> 2
else if L = −1 then r ← r >>1
else for i=1 to L do
     if r is even then r ← r/2
       else r ← (r+p)/2
return p−r
```

Fig. 5 Corrected halving phase of the inversion algorithm
Redrawn from Ahmadi and Afzali-Kusha (2009a)

One step toward securing the hardware against SPA is to change the 'point-adder' block so that it performs both the point addition and point doubling operations in exactly the same flow (Hankerson *et al.*, 2004). This will cause the timing and power consumption traces for each of these operations to be indistinguishable during the operation of the ECC processor (Hankerson *et al.*, 2004). As will be explained shortly, this may be easily achieved in our proposed ECC processors as we used a unified circuit to perform the point addition and doubling operations. Considering Eqs. (2) and (3), the only difference between the point addition and doubling is in the calculation of Δ in Eq. (3). In the implementation of the 'point-adder' controller, we have only one extra multiplication and one extra addition in the point doubling operation. If we perform exactly the same calculations during the point addition operation (obviously, without actually using the results), the timing and power traces of all point operations will be the same and the operations cannot be distinguished. In the first design, since the calculation time is large, this change will have a negligible effect, while in the second design it will introduce a 3.6% increase in the calculation time (and energy consumption) of the whole scalar multiplication. In terms of the area and power consumptions, the overhead will be negligible.

Note that, even if the addition and doubling operations are exactly equal in timing and power behavior, because of the double-and-add method, the attacker will still be able to extract the number of 1's in the 'scalar' by counting the total number of point operations. Therefore, the above changes cannot be regarded as a way to completely secure the hardware against SPA. Complete security can be reached with the always-double-and-add method in our ECC processors. A very simple change in the controller can force the hardware to perform a point addition for every bit of the 'scalar' regardless of its value. Therefore, our proposed ECC processors can be completely secured against SPA with no area or power consumption overhead. However, the calculation time of the scalar multiplication will increase 25% on average, due to the additional operations.

6.5 Low power design of the two ECC processors

The application of our ECC processors in RFID tag and wireless sensor network allows the use of very low clock frequencies. We have used this property throughout the design process. This has allowed us to use smaller adder blocks with longer critical paths inside the two calculation units contrary to other

works (Wolkerstorfer, 2005; Fürbass and Wolkerstorfer, 2007), which used larger adder blocks to have shorter critical paths. In addition, special care was taken during the synthesis of the design to use only the smallest-size version of each technology standard cell, which led to some extra power savings.

As a more effective technique to reduce the power consumption, clock-gating is also used. Each one of our designs uses a total of seven registers, each with the same width as the field size (168 bits or more), which adds up to more than 1000 flip-flops. Therefore, an efficient application of the clock-gating technique may significantly reduce the power consumption. Since the performance of the clock-gating in reducing the power consumption of a register is dependent on the design, the synthesis parameters, and the CMOS technology characteristics, the exact effect of the clock-gating on each of the registers should be checked by simulation. In the first design, the simulations showed that, the two registers of the Montgomery multiplier had high enough usage and therefore the clock-gating did not reduce their power consumption. The other five registers benefited from the clock-gating, especially the two registers at the top level, which had relatively low value changes. Our results showed that the application of this technique in the case of a 192-bit ECC processor led to a 39.4% reduction in the total power consumption. For the second design, simulations indicated that all seven registers could benefit from clock-gating. In this design, for the case of a 192-bit ECC processor, a 35.2% reduction in the total power consumption was achieved.

7 Results and discussion

In this section, first we compare the area, power/energy consumptions, and calculation time of the two ECC processors. Then, the designs are compared against the similar designs previously reported in the literature.

7.1 Results for our ECC processors

The two prime-field ECC processor designs were synthesized in a standard 0.13 μm CMOS technology for three standard-recommended field sizes, i.e., 168, 192, and 224 bits (NIST, 2000). The hardware specifications and the results of the hardware synthesis are shown in Table 1. The reported areas, which are the results of placement and routing (PAR), show that the processors are small enough to be embedded in RFID and wireless sensor network applications (Wolkerstorfer, 2005). For the target applications, such as RFID item management and electronic product code (EPC), a maximum frequency of higher than 13.56 MHz will be enough for the ECC processors (Chawla and Ha, 2007). As the results show, both ECC processors meet the 13.56 MHz limit. In practice, for lowering the power consumption, much lower clock frequencies will be applied to the ECC processors. The examination of the circuit timing shows that the maximum frequency is limited by the long carry propagation path inside the adders. This is the reason why the two designs have similar timing specifications and why the maximum frequency is inversely proportional to the field size.

Table 2 shows the power/energy consumption and calculation time of the two ECC processors at 1 MHz clock frequency. The average power consumption is derived from the simulations with many different random field parameters, curve points, and

Table 1 Results of synthesis at different field sizes (in bit) for the two elliptic-curve cryptography (ECC) processor designs*

Design	Placement and routing area (mm^2)			Maximum frequency (MHz)		
	168	192	224	168	192	224
First	<0.13	<0.15	<0.18	>21	>18	>15.7
Second	≈0.14	≈0.17	<0.20	≈22.3	≈18.5	≈15.2

* Both prime-field and flexible-parameter designs are implemented in 0.13 μm CMOS technology

Table 2 Power, energy, and timing at three different field sizes for the two designs at 1 MHz clock frequency*

Design	Average power (μW)			Total energy (μJ)**			Total time (ms)**		
	168 bits	192 bits	224 bits	168 bits	192 bits	224 bits	168 bits	192 bits	224 bits
First design	23.1	26.3	30.7	225.5	410.1	782.6	~9800	~15525	~25565
Second design	32.3	39.3	48.6	12.92	20.63	35.23	~400	~525	~725

* Both prime-field and flexible-parameter designs are implemented in 0.13 μm CMOS technology. ** For one scalar multiplication

scalar numbers. The post-synthesis netlist was simulated with a large amount of random input data and then the power consumption was calculated using exactly the same approach as explained in Lee *et al.* (2008).

Table 3 shows the details of area consumptions of different parts in the 192-bit version of both ECC processors in gate equivalents (GE). It can be seen that using the MMI algorithm in the second design results in a much larger controller and also some increase in the data-path area due to the larger multiplexers (Fig. 3a). Also, the results show that the storage elements consume nearly 30% of the total area.

Table 3 Area consumptions of different parts for the two elliptic-curve cryptography (ECC) processor designs at a field size of 192 bits

Type	Area	
	First design	Second design
Total	25900 GE	30200 GE
Storage	8600 GE (33.2%)	8600 GE (28.5%)
Datapath	10000 GE (38.6%)	10600 GE (35.1%)
Controller	7300 GE (28.2%)	11000 GE (36.4%)

GE: gate equivalent. Value in brackets is the ratio to the total area

7.2 Comparison of modulo inverse algorithms

In this part, we compare the efficiency of the modulo inverse algorithms in terms of power, energy, and timing results.

Table 4 compares the calculation time and clock cycles of the two designs for one scalar multiplication. For a field size of n bits, the inversion algorithm based on Fermat's theorem takes $O(n^2)$ cycles due to the serial implementation of the modulo multiplication while the MMI takes $O(n)$ cycles for the inversion operation. Since our designs are based on the affine coordinates, a complete scalar multiplication will take $O(n^3)$ cycles in the first design and $O(n^2)$ cycles in the second design. This explains the larger number of clock cycles in the first design.

Table 5 compares the power consumptions of both designs at 1 MHz clock frequency. The increase in the power consumption in the second design (MMI) is mostly due to the multiplexers added to the circuits to enable the re-use of the same calculation units for performing the MMI. The growth in the complexity of the multiplexers will be higher for larger field sizes, and hence, the increase in the power consumption becomes larger with the field size.

Table 6 compares the total energy consumption of the designs for one scalar multiplication. The MMI algorithm shows a great improvement which is nearly proportional to the field size for our specific implementation.

Table 4 Calculation time at three different field sizes for the two elliptic-curve cryptography (ECC) processor designs at 1 MHz clock frequency

Design	Calculation time (ms)		
	168 bits	192 bits	224 bits
First design	~9800	~15525	~25565
Second design	~400	~525	~725
Design	**Number of clock cycles**		
	168 bits	192 bits	224 bits
First design	$\sim 9.8\times10^6$	$\sim 15.5\times10^6$	$\sim 25.5\times10^6$
Second design	$\sim 400\times10^3$	$\sim 525\times10^3$	$\sim 725\times10^3$
Improvement ratio	24.5%	29.6%	35.3%

Table 5 Power consumption at three different field sizes for the two elliptic-curve cryptography (ECC) processor designs at 1 MHz clock frequency

Design	Power (μW)		
	168 bits	192 bits	224 bits
First design	23.1	26.3	30.7
Second design	32.3	39.3	48.6
Improvement ratio	39.8%	49.4%	58.3%

Table 6 Total energy for the two elliptic-curve cryptography (ECC) processor designs

Design	Energy (μJ)		
	168 bits	192 bits	224 bits
First design	225.5	410.1	782.6
Second design	12.92	20.63	35.23
Improvement ratio	17.45%	19.88%	22.21%

7.3 Comparison to previous works

Table 7 compares the results of the ECC processors used to evaluate the efficiency of the designs presented in this work. The designs selected for this comparison have used the same technology as in this work. In addition, all have been optimized for low-power/energy applications. Note that, although the designs in Daly *et al.* (2004) and Byrne *et al.* (2007) are similar to our design, they were implemented in field-programmable gate arrays (FPGAs) and did not report any values for power/energy consumption, and hence, were not included in the comparison. The

Table 7 Comparison of elliptic-curve cryptography (ECC) processor designs

Design	Field size & type	Total energy (μJ)	Reported average power (μW) @ clock frequency	Calculation time (ms)	Total number of cycles
Öztürk *et al.* (2004)	168-bit, special prime	31.58	990 @ 20 MHz	31.9	545440
Gaubatz *et al.* (2005)	102-bit, special prime	161.88	394.4 @ 500 kHz	410.54	205270
de Dormale *et al.* (2006)	193-bit, binary	8.10	428 @ 10 MHz	~19	190000
Batina *et al.* (2007)*	134-bit, binary	2.73	<13 @ 200 kHz	210	42000
Lee *et al.* (2008)	163-bit, binary	8.94	36.63 @ 1.13 MHz	244.08	275816
First design	168-bit, flexible prime	225.50	23.1 @ 1 MHz	9800	$\sim 9.8\times10^6$
Second design	168-bit, flexible prime	12.92	32.3 @ 1 MHz	400	~400000
Second design	192-bit, flexible prime	20.63	39.3 @ 1 MHz	525	~525000

* The power/energy values did not include the RAM block

results reveal that our first design cannot compete with other works, since it has a very long calculation time. The second design, however, outperforms the other designs in terms of power consumption. Note that Batina *et al.* (2007) did not take into account the power consumption of its RAM blocks. The total energy consumption of the second design is also lower than that of the prime-field designs, even when our design supports larger fields. Only the binary-field designs have somewhat lower total energy consumption than our second design, which was foreseeable, since the calculations are faster in binary fields.

Table 8 compares our ECC processors with some other prime-field processors in terms of area and total cycles. The designs selected for this comparison are all prime-field and have the same sizes as those of our processors. The areas for both ECC processors are in the same order as those of other similar designs. For the second design, the total number of cycles for one scalar multiplication is also in the same order.

Table 8 Area and the total number of cycles of the elliptic-curve cryptography (ECC) processors

Design	Field size & type	Total number of cycles	Area (GE)
Öztürk *et al.* (2004)	168-bit, SP	545440	30333
Fürbass and Wolkerstorfer (2007)	192-bit, FP	502000	23656
Wolkerstorfer (2005)	192-bit, FP	677500	23800
First design	192-bit, FP	15.5×10^6	25900
Second design	168-bit, FP	400000	25200
Second design	192-bit, FP	525000	30200

GE: gate equivalent. SP: special prime; FP: flexible prime

7.4 Further improvements

The comparison results discussed above reveal that the second design is a good candidate for the target low-power/energy applications. This design, however, may be further improved. Here, we discuss two possible modifications for the improvement of the second design. One of the ways to reduce the calculation time is to speed up the modulo multiplication operation. This could be done using high-radix serial multiplication, in which two or more bits of the multiplication operand are processed in each clock cycle. Using this method will improve the speed at the price of increasing the area and power consumptions, while the net effect on the energy consumption will be dependent on the implementation. As an example, we implemented the second ECC processor with a radix-4 Montgomery multiplication (two rounds of the multiplication in one clock cycle). The results showed a 23% decrease in the calculation time, while the power and area consumptions were increased by 17% and 10%, respectively. The total energy consumption decreased by 9%. Since high-radix multiplication can be implemented in different ways, the design space to be explored for this optimization is very large.

To improve the area and power consumption, latches (instead of flip-flops) could be used for the storage elements as suggested in de Dormale *et al.* (2006) and Bock *et al.* (2008). Note that, since the flip-flops are edge-triggered and the latches are level-triggered, for using latches in place of flip-flops, special pulse-generation circuits must be employed (Nedovic and Oklobdzija, 2005). The use of pulse-generation circuits requires special attention, making it more suitable for custom-designed hardware blocks.

To see the impact of this optimization on our ECC processor design, we compared the standard-cell flip-flops with standard-cell latches in the 0.13 μm technology. We have assumed that the overheads for the pulse-generation circuits for latches and the clock-gating circuits for flop-flops are the same. The results presented in Tables 9 and 10 indicate that there is not much improvement in the area or power consumption of our design if we use standard-cell latches instead of flip-flops in our 0.13 μm technology. In custom-designed hardware, however, one can build custom low-area storage elements, which greatly reduces the storage area (and power), as explained in de Dormale *et al.* (2006).

Table 9 Comparison of latches vs. flip-flops

Cell	Total power (μW/MHz)*	Cell area (μm^2)
Flip-flop	0.0196	30.55
Latch	0.0167	25.46

* Sum of power for D(in), CK/G(in), and Q(out)

Table 10 Improvements in power and area

Power	Percentage
Power reduction (latch vs. FF)	15%
Percentage of FF power in our design	25% (of total)
Total power reduction in our design	3.75%
Area	**Percentage**
Area reduction (latch vs. FF)	17%
Percentage of FF area in our design	33% (of total)
Total area reduction in our design	5.66%

FF: flip-flop

8 Conclusions

In this work, we studied two designs for a flexible prime-field elliptic-curve crypto-processor. The designs for both ECC processors were the same except for the modulo inverse algorithm. The first processor used the Fermat's Little Theorem approach for the inversion and the second one was based on the Montgomery modulo inverse algorithm. To evaluate the efficiency of the designs in terms of power/energy consumption and timing, we synthesized and simulated the processors in a 0.13 μm CMOS technology. The results showed that both processors had low area and low power consumptions while the design based on the Montgomery modulo inverse had much shorter calculation time and lower energy consumption. In addition, we compared our processors with some previously reported processors. The results showed that our second design outperforms similar reported designs in terms of power and energy consumption while having a similar area. Finally, the use of high-radix multiplication and using latches as the storage elements to further improve the efficiency of the second ECC processor were considered.

References

Ahmadi, H.R., Afzali-Kusha, A., 2009a. Low-Power Low-Energy Prime-Field ECC Processor Based on Montgomery Modular Inverse Algorithm. 12th Euromicro Conf. on Digital System Design, Architectures, Methods and Tools, p.817-822. [doi:10.1109/DSD.2009.140]

Ahmadi, H.R., Afzali-Kusha, A., 2009b. Very Low-Power Flexible GF(*p*) Elliptic-Curve Crypto-Processor for Non-Time-Critical Applications. IEEE Int. Symp. on Circuits and Systems, p.904-907. [doi:10.1109/ISCAS.2009.5117903]

Batina, L., Mentens, N., Sakiyama, K., Preneel, B., Verbauwhede, I., 2007. Public-Key Cryptography on the Top of a Needle. IEEE Int. Symp. on Circuits and Systems, p.1831-1834. [doi:10.1109/ISCAS.2007.378270]

Bock, H., Braun, M., Dichtl, M., Hess, E., Heyszl, J., Kargl, W., Koroschetz, H., Meyer, B., Seuschek, H., 2008. A Milestone towards RFID Products Offering Asymmetric Authentication Based on Elliptic Curve Cryptography. Workshop on RFID Security.

Byrne, A., Meloni, N., Crowe, F., Marnane, W.P., Tisserand, A., Popovici, E.M., 2007. SPA Resistant Elliptic Curve Cryptosystem Using Addition Chains. 4th IEEE Int. Conf. on Information Technology, p.995-1000. [doi:10.1109/ITNG.2007.185]

Chawla, V., Ha, D.S., 2007. An overview of passive RFID. *IEEE Commun. Mag.*, **45**(9):11-17. [doi:10.1109/MCOM.2007.4342873]

Daly, A., Marnane, W., Kerins, T., Popovici, E., 2004. An FPGA implementation of a GF(*p*) ALU for encryption processors. *Microprocess. & Microsyst.*, **28**(5-6):253-260. [doi:10.1016/j.micpro.2004.03.006]

de Dormale, G.M., Ambroise, R., Bol, D., Quisquater, J.J., Legat, J.D., 2006. Low-Cost Elliptic Curve Digital Signature Coprocessor for Smart Cards. IEEE 17th Int. Conf. on Application-Specific Systems, Architectures and Processors, p.347-353. [doi:10.1109/ASAP.2006.44]

Feldhofer, M., Wolkerstorfer, J., 2007. Strong Crypto for RFID Tags: a Comparison of Low-Power Hardware Implementations. IEEE Int. Symp. on Circuits and Systems, p.1839-1842. [doi:10.1109/ISCAS.2007.378272]

Fürbass, F., Wolkerstorfer, J., 2007. ECC Processor with Low Die Size for RFID Applications. IEEE Int. Symp. on Circuits and Systems, p.1835-1838. [doi:10.1109/ISCAS.2007.378271]

Gaubatz, G., Kaps, J.P., Öztürk, E., Sunar, B., 2005. State of the Art in Ultra-Low Power Public Key Cryptography for Wireless Sensor Networks. Third IEEE Int. Conf. on Pervasive Computing and Communications Workshops, p.146-150. [doi:10.1109/PERCOMW.2005.76]

Hankerson, D., Menezes, A.J., Vanstone, S., 2004. Guide to Elliptic Curve Cryptography. Springer-Verlag New York Inc., New York, USA.

Kaliski, B.S.Jr., 1995. The Montgomery inverse and its applications. *IEEE Trans. Comput.*, **44**(8):1064-1065. [doi:10.1109/12.403725]

Kaps, J.P., 2006. Cryptography for Ultra-Low Power Devices. PhD Thesis, ECE Department, Worcester Polytechnic Institute, Worcester, Massachusetts, USA.

Kumar, S.S., 2006. Elliptic Curve Cryptography for Constrained Devices. PhD Thesis, EE and IT Department, Ruhr University, Bochum, Germany.

Lee, Y.K., Sakiyama, K., Batina, L., Verbauwhede, I., 2008. Elliptic-curve-based security processor for RFID. *IEEE Trans. Comput.*, **57**(11):1514-1527. [doi:10.1109/TC.2008.148]

Montgomery, P.L., 1985. Modular multiplication without trial division. *Math. Comput.*, **44**(170):519-521. [doi:10.2307/2007970]

National Institute of Standards and Technology (NIST), 2000. Digital Signature Standard (DSS), FIPS PUB 186-2. Federal Information Processing Standards Publication. National Institute of Standards and Technology, USA.

Nedovic, N., Oklobdzija, V.G., 2005. Dual-edge triggered storage elements and clocking strategy for low-power systems. *IEEE Trans. VLSI Syst.*, **13**(5):577-590. [doi:10.1109/TVLSI.2005.844302]

Öztürk, E., Sunar, B., Savaş, E., 2004. Low-power elliptic curve cryptography using scaled modular arithmetic. *LNCS*, **3156**:92-106. [doi:10.1007/978-3-540-28632-5_7]

Savaş, E., Koç, Ç.K., 2000. The Montgomery modular inverse-revisited. *IEEE Trans. Comput.*, **49**(7):763-766. [doi:10.1109/12.863048]

Stamp, M., 2006. Information Security: Principles and Practice. John Wiley & Sons Inc., New Jersey, USA.

Wolkerstorfer, J., 2005. Scaling ECC Hardware to a Minimum. ECRYPT Workshop: Cryptographic Advances in Secure Hardware. Invited Talk.