A power-optimized low-energy elliptic-curve crypto-processor

Hamid Reza Ahmadi,[1] Ali Afzali-Kusha,[1] and Massoud Pedram[2]

[1]*School of Electrical and Computer Engineering, University College of Engineering, University of Tehran*

[2]*Department of Electrical Engineering, University of Southern California*

Abstract: This paper presents a low-energy prime-field elliptic-curve cryptography (ECC) hardware processor, suitable for low-power and/or energy-efficient applications and systems. The ECC processor is obtained by power-optimizing a previously reported design. The optimization is performed by making the power consumption profile of the processor as uniform as possible, in an attempt to increase the secondary battery life between discharge and recharge cycles and to create resistance against simple power attacks (SPA) to the cryptosystem by analyzing the power dissipation trace of the hardware. The optimized ECC processor performs a single 192-bit scalar multiplication in 652 ms consuming only 22.3 μJ at a clock frequency of 1MHz. This indicates, in addition to the more uniform power consumption, a 13% reduction in the energy and power consumption compared to the previously reported design.

Keywords: Elliptic-curve cryptography (ECC), Prime field, Flexible crypto-processor, Low-energy Low-power hardware.
Classification: Integrated Circuits

References

1. M. Stamp, *Information Security - Principles and Practice*, John Wiley & Sons Inc., New Jersey, USA, 2006.
2. J. P. Kaps, "Cryptography for Ultra-Low Power Devices," Ph.D. dissertation, ECE Department, Worcester Polytechnic Institute, Worcester, Massachusetts, USA, May 2006.
3. D. Hankerson, A. J. Menezes, and S. Vanstone, *Guide to Elliptic Curve Cryptography*, Springer-Verlag New York Inc., New York, USA, 2004.
4. S. S. Kumar. "Elliptic Curve Cryptography For Constrained Devices," Ph.D. dissertation, Electrical Engineering and Information Technology Department, Ruhr-University, Bochum, Germany, June 2006.
5. Y.K. Lee, K. Sakiyama, L. Batina, and I. Verbauwhede, "Elliptic-Curve-Based Security Processor for RFID," *IEEE Transactions on Computers*, vol. 57, no. 11, pp. 1514-1527, Nov 2008.
6. D. Hein, J. Wolkerstorfer, and N. Felber, "ECC Is Ready for RFID – A Proof in

[DOI: **.****/**********]
Received September **, 2008;
Accepted September **, 2008;
Published in October **, 2008 issue

Silicon," SAC 2008, Lecture Notes in Computer Science, vol.5381, pp. 401–413. Springer, Heidelberg (2009).

7. H.R. Ahmadi and A. Afzali-Kusha, "A Low-Power and Low-Energy Flexible GF(p) ECC Processor," *Journal of Zhejiang University - Science C*, vol. 11, no. 9, pp. 724-736, Sep 2010.
8. M. Feldhofer and J. Wolkerstorfer, "Strong Crypto for RFID Tags – A Comparison of Low-Power Hardware Implementations," In Proceedings of the IEEE International Symposium on Circuits and Systems, ISCAS'07, pp. 1839-1842, May 2007.
9. E. Savaş and Ç. K. Koç, "The Montgomery Modular Inverse – Revisited," *IEEE Transactions on Computers*, vol. 49, no. 7, pp. 763–766, Jul 2000.
10. K. Lahiri, A. Raghunathan, and S. Dey, "Efficient Power Profiling for Battery-Driven Embedded System Design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, no. 6, pp. 919–932, Jun 2004.
11. P. Rong and M. Pedram. "An analytical model for predicting the remaining battery capacity of Lithium-ion batteries," *IEEE Transactions on VLSI systems*, vol. 14, no. 5, pp. 441-451, May 2006.

1. Introduction

Data and communication security is an essential requirement in every computing environment [1]. Providing security in pervasive computing environments, which rely on low-power and/or energy efficient devices such as wireless sensor network (WSN) motes and radio frequency identification (RFID) tags, is very challenging because of limited available resources and high processing demand of cryptography calculations [1-3]. As a solution to this challenge, light-weight algorithms and protocols have been developed which are more easily integrated in such devices [2]. However, standardized public-key cryptography algorithms such as the elliptic-curve cryptography (ECC) offer more robust and reliable security [2-4]. Therefore, several designs have been reported which are able to integrate the ECC functionality in low-power/energy devices [4-8]. While most of these designs (e.g., [4-6]) have used a binary-field ECC system to lower the hardware cost, some others (e.g., [7-8]) have invoked prime-field ECC processors which are more advantageous when integrated in resource-limited applications [7].

In this work, we present a power/energy-optimized prime-field elliptic-curve cryptography hardware processor with fairly uniform power consumption. Section 2 describes the architecture of the initial ECC processor while Section 3 discusses the power optimization techniques performed on the initial design and the results of the optimizations. Finally, Section 4 concludes the paper.

2. Initial low-energy ECC processor architecture

We perform optimization techniques on a low-power energy-efficient hardware implementation of a flexible prime-field ECC processor designed and reported by our group in [7]. The benefits of the prime-field over the binary-field ECC processors have been described in [7-8]. In particular the benefits of flexibility (i.e.,

[DOI: **.****/**********]
Received September **, 2008;
Accepted September **, 2008;
Published in October **, 2008 issue

the ability of hardware to calculate different curves without any hardware changes) have been discussed in [7]. The data-path of the processor along with the details of the calculation units are shown in Fig. 1. In this paper we describe the architecture only briefly. More details about the overall architecture of the ECC processor and the controllers may be found in [7].

The top-level controller uses the double-and-always-add method to perform a scalar multiplication with security against simple power attacks (SPAs.) The lower-level controllers use the data-path shown in Fig. 1(a) to perform either a point addition or a point doubling, using the formulas given below:

$$\Delta = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & P1 \neq P2 \\ \dfrac{3x_1^2 + a}{2y_1} & P1 = P2 \end{cases} \tag{1}$$

$$x_3 = \Delta^2 - x_1 - x_2 \,,\; y_3 = \Delta(x_1 - x_3) - y_1 \tag{2}$$

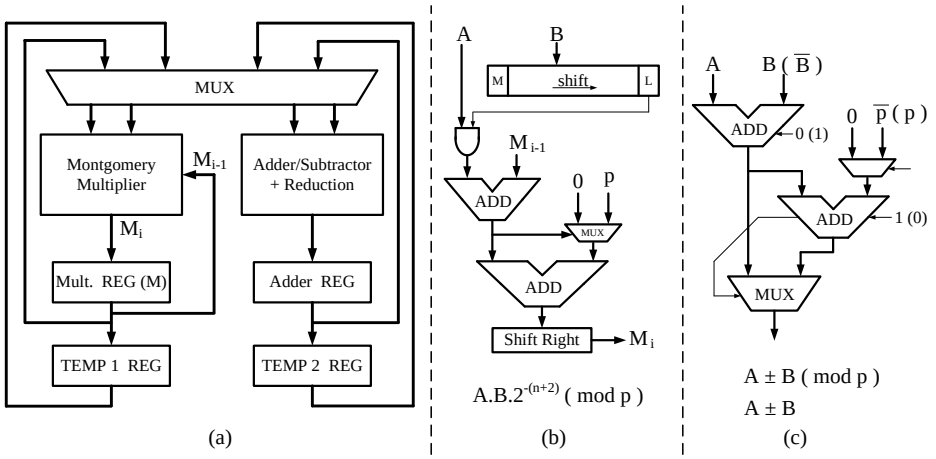


Fig. 1. Initial ECC processor architecture [7] (a) Data-path of the ECC processor. (b) Montgomery multiplier. (c) Adder/subtractor with reduction.

In either case, first, the denominator of the quotient in Eq. (1) is calculated using the adder unit shown in Fig. 1(c). Next the modulo inverse of the result is derived. The modulo inverse operation uses a modified version of the Montgomery modulo inverse algorithm presented in [7] and [9]. The inverse calculation uses both the adder unit and the multiplier unit along with all the registers shown in Fig. 1(b). After calculating the inverse, the remaining operations of Eq. (1) and (2) are performed using the arithmetic units.

Table I shows the specifications and the power/energy consumption values of the initial ECC processor, implemented in a 0.13µm standard CMOS technology for a field size of 192 bits. The clock cycles and the timing of the ECC processor are given for the SPA-secured version which uses the double-and-always-add method [7]. Table I also shows three partial values for the average power consumption, each of them measured for a different part of the calculations. The inverse calculation has two phases, a '*calculation*' phase and a '*halving*' phase. Other parts of the ECC calculations are grouped under '*the rest*' phase. The ratio of

[DOI: **.****/**********]
Received September **, 2008;
Accepted September **, 2008;
Published in October **, 2008 issue

Table I. Initial ECC processor design – specifications, power and energy consumption

Technology	CMOS 0.13 μm	Percentage of total calculation time "inverse – calculation"	41.1 %
Field Size	192 bits	Partial Average Power "inverse – calculation"	44.1 μW
Total Energy (one scalar mult.)	25.5 μJ double & always-add	Percentage of total calculation time "inverse – halving"	7.5 %
Total Average Power **@ f = 1 MHz**	39.3 μW	Partial Average Power "inverse – halving"	63.3 μW
Total Clock Cycles (One scalar mult.)	650,000 double & always-add	Percentage of total calculation time "the rest"	51.4 %
Total Time **@ f = 1 MHz** (One scalar mult.)	650 ms double & always-add	Partial Average Power "the rest"	32.1 μW

the time duration of each phase to the total calculation time is also reported as a percentage. The basis for distinguishing and separating these three phases is the changes in the data-path configuration. In each phase of the calculations, the data-path is in a different configuration, which affects the average power consumption. The difference in the average power dissipation among the phases of the calculations is the base of our optimizations, as will be described next.

3. Optimization of the ECC processor power consumption

In this section, we explain motivations for the power optimization strategy used for the ECC processor along with the techniques employed for the optimization of the processor. The results are also discussed in this section.

3.1 Optimization strategy

To perform different parts of the ECC operations, the data-path is placed in different configurations. As shown in Table I, this causes the average power consumption to be non-uniform during different parts of the calculations leading to large deviations from the total average power consumption in some time intervals. A low-power/energy ECC processor with uniform average power consumption has the following benefits:

1. In practice, the energy source of the hardware (*i.e.*, a battery) must be able to supply as much current as the highest value of the power consumption. Our ECC processor, for example, must be supplied with an amount of 63.3 μW of power during the '*halving*' phase which accounts for only 7.5 percent of the calculation time. With a uniform average power consumption, the highest value needed is close to the total average value, hence, reducing the required battery energy delivery capability. In addition to enabling the use of smaller battery due to lowering of the peak current demand, a uniform '*power profile*' [10] tends to increase the service time of rechargeable batteries such as the Li-Ion batteries due to the rate capacity effect [11]. Finally, with a uniform average power consumption, the battery management will be easier, because it will be easier to predict the '*power profile*' [10] of the hardware.
2. Security attacks based on analyzing the 'power trace' of the cryptography hardware (so-called simple power attacks), attempt to extract secret data

[DOI: **.****/**********]
Received September **, 2008;
Accepted September **, 2008;
Published in October **, 2008 issue

from the shape of the power trace [3]. Uniform average power consumption causes the power trace to be nearly flat, which makes it harder to distinguish different parts of the power trace from each other. This, in turn, makes extracting the secret data harder for the attacker and makes the hardware more secure against SPAs.

These factors motivated us to optimize the initial ECC processor design in such a way that its power consumption becomes as uniform as possible. Based on the results provided in Table I, the '*halving*' phase of the inverse calculation has the *highest* power consumption value while taking a short time. Hence, special attention should be paid to this part of the calculations.

3.2 Optimization techniques

As described above, to make the average power consumption of the ECC processor uniform, we must lower the power consumption of the '*halving*' phase. Detailed block-by-block measurements of the power consumption show that the adder/subtractor unit is the main power consuming block in the '*halving*' phase (as well as in the '*calculation*' phase.) Based on the inversion algorithm (described in more detail in [7]) and the hardware architecture of the initial design, two different techniques for lowering the power consumption of the adder/subtractor unit are invoked. The techniques include operation isolation and hardware reduction as explained next.

3.2.1 Operand isolation

As shown in Fig. 1(c), the adder/subtractor unit is capable of performing both ordinary and modulo operations. The modulo operation of this unit is used for additions and subtractions of the ECC algorithm and ordinary operation is used during the inversion algorithm. During the design of the ECC processor, '*operand isolation*' was applied to the inputs of the second adder block inside this unit– *i.e.*, the bottom adder in Fig 1(c) – during both phases of the inversion (using AND logic gates), in order to lower its power consumption. However, the complexity of the '*calculation*' phase of the inversion algorithm caused an increase in the total power consumption and therefore we decided to remove the operand isolation gates.

Since the power consumption during the '*halving*' phase is much higher than that during the '*calculation*' phase, and our goal is to move toward a more uniform average power consumption, we now apply the operand isolation only during the '*halving*' phase. For this purpose, the controller isolates the inputs of the second adder only during the '*halving*' phase. This is much simpler in terms of hardware complexity and can reduce the power consumption in the '*halving*' phase, as can be seen in the upper section of Table II for the 'Operand Isolation' results. Note that the partial average power values show a more uniform average power consumption. The addition of the isolation gates, however, causes a small increase in the power consumption of the other phases of the calculations. As a result, only the power consumption of the '*halving*' phase is reduced, and the total average power and total energy remained nearly at their original levels.

[DOI: **.****/**********]
Received September **, 2008;
Accepted September **, 2008;
Published in October **, 2008 issue

Table II. Results for the ECC processor after optimizations

Operand Isolation during "halving" phase		Percentage of total time "inverse – calculation"	41.1 %
		Partial Average Power "inverse – calculation"	46.8 μW
Total Energy (one scalar mult.)	25.3 μJ double & always-add	Percentage of total time "inverse – halving"	7.5 %
Total Average Power **@ f = 1 MHz**	38.9 μW	Partial Average Power "inverse – halving"	38.8 μW
Total Clock Cycles (one scalar mult.)	650,000	Percentage of total time "the rest"	51.4 %
Total Time **@ f = 1 MHz** (one scalar mult.)	650 ms	Partial Average Power "the rest"	32.7 μW
Hardware Reduction		Percentage of total time "inverse – calculation"	40.9 %
		Partial Average Power "inverse – calculation"	34.1 μW
Total Energy (one scalar mult.)	22.3 μJ double & always-add	Percentage of total time "inverse – halving"	7.4 %
Total Average Power **@ f = 1 MHz**	34.2 μW	Partial Average Power "inverse – halving"	40.4 μW
Total Clock Cycles (one scalar mult.)	652,000	Percentage of total time "the rest"	51.7 %
Total Time **@ f = 1 MHz** (one scalar mult.)	652 ms	Partial Average Power "the rest"	33.5 μW

3.2.2 Hardware reduction

In the design of the initial ECC processor, to complete every modulo addition/subtraction in one clock cycle, two adders are included in the adder/subtractor unit (cf. Fig. 1(c).) One way to reduce the power consumption of the adder/subtractor unit is to reduce it to one adder only. Using this configuration, every modulo addition/subtraction requires two clock cycles to be completed. Since these modulo operations constitute a small percentage of the ECC calculations, the impact on the total calculation time will be small.

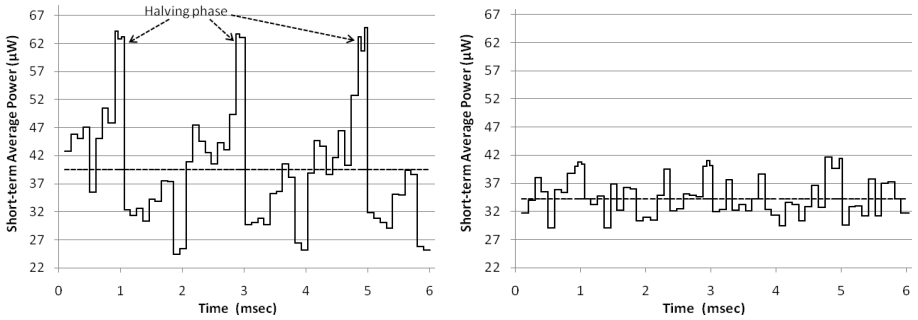
The power/energy results for the ECC processor with only one adder in the adder/subtractor unit are shown in the lower part of Table II. It can be seen that the partial power consumption is reduced in both the '*calculation*' and '*halving*' phases, and therefore, we have gained a total decrease of nearly 13 percent both in total average power consumption and in total energy. More importantly, it can also be seen that this optimization has resulted in a much more uniform average power consumption profile, which was the main goal of the optimization.

3.3 Discussion

A comparison of the partial average power values given in Table II shows that the "Hardware Reduction" technique has performed better, both in lowering the power consumption and in making the average power more uniform. To compare the average power obtained by the "Hardware Reduction" with that of the original ECC processor, the "short term" average power is shown in Fig. 2. The diagrams were obtained by averaging the power consumption over short time intervals where each diagram includes three consecutive ECC point operations.

In Fig. 2(a), it can be observed that the large changes in the power consumption may help an attacker in distinguishing different parts of the calculations. In the

[DOI: **.****/**********]
Received September **, 2008;
Accepted September **, 2008;
Published in October **, 2008 issue



(a) (b)

Fig. 2. Average power consumption measured for short time intervals (a) Original ECC processor. (b) The result after the 'hardware reduction'.

more uniform diagram of Fig 2(b), however, it is much more difficult to distinguish and separate the details of the calculations. The more uniformity, therefore, can bring more security against attacks based on analyzing power traces.

Regarding the battery capacity and peak current, it should be noted that the diagrams of Fig. 2 resemble instantaneous power measurements of the hardware because of the 'short term' averaging of power. In the optimized design, the average power in the 'halving' phase is lowered from about 63μW to about 40μW. Since the average power in this phase more or less translates to the peak power/current of the hardware, an increase in the secondary battery life between discharge and recharge cycles will be achieved.

Finally, note that comparing the results of 'Hardware Reduction' presented in Table II with those of [5-7], shows that the optimized ECC processor performs better in terms of the average power and total energy consumption.

4. Conclusion

In this paper, for the first time a low-power/energy prime-field ECC processor with a nearly uniform average power consumption was presented. The proposed design was achieved by applying optimization techniques to a previously reported design. Making the average power consumption of the ECC processor as uniform as possible brings more security against simple power attacks and enables more effective battery management. The results for the 192-bit prime-field ECC processor, implemented in a 0.13 μm CMOS technology, showed that one scalar multiplication could be completed in 652 ms, using only 22.3 μJ at a clock frequency of 1 MHz. The optimized design had a much more uniform power profile. In addition, it exhibited nearly 13% improvement in terms of energy consumption, with a negligible increase in calculation time when compared to the initial design. Similar optimization techniques could be applied to other cryptography hardware processors to obtain a more uniform power consumption.

Acknowledgments

HRA and AAK acknowledge the financial support of this work by the Iran Telecommunication Research Center (ITRC).

[DOI: **.****/**********]
Received September **, 2008;
Accepted September **, 2008;
Published in October **, 2008 issue