An Energy-Efficient Dual-Field Elliptic Curve Cryptography Processor for Internet of Things Applications

Ling-Yu Yeh, Po-Jen Chen, Chen-Chun Pai, and Tsung-Te Liu, *Member, IEEE*

***Abstract*—This brief presents an energy-efficient elliptic curve cryptography (ECC) processor for Internet of Things (IoT) security applications. The proposed processor supports dual-field computations, and employs various design techniques across the algorithm, architecture, and arithmetic circuit levels to minimize power and energy consumption. The proposed elliptic curve point multiplication (ECPM) algorithm employs signed binary representation (SBR) with the m-ary method to reduce both area and energy consumption, while avoiding attack from simple power analysis (SPA). In addition, the proposed hybrid modular arithmetic architecture effectively increases the hardware utilization to reduce both area and energy cost. Finally, the proposed processor uses an energy-efficient data flow to further minimize memory overhead for group operations. The proposed ECC processor achieves 51.6% and 50.5% lower energy consumption for each $GF(p)$ and $GF(2^m)$ ECPM operation, respectively, when compared to state-of-the-art ECC designs.**

***Index Terms*—Elliptic curve cryptography (ECC), hardware security, low power, energy-efficient, dual field, side-channel attacks, Internet of Things (IoT).**

I. Introduction

ASYMMETRIC cryptography, or public-key cryptography (PKC), is developed to avoid the key distribution issues of traditional symmetric cryptography. Compared to other popular PKCs such as RSA and Diffie-Hellman key exchange, elliptic curve cryptography (ECC) offers a relatively higher security level with a shorter key size [1]. As a result, ECC has been employed to secure the Internet of Things (IoT) devices for various kinds of applications such as NFC [2], WAVE [3], and DSA [4]. Since the available energy in an IoT device is highly limited, hardware implementation of ECC is a more practical approach than software-based solutions.

Conventional ECC hardware implementations focus mainly on the improvement of throughput performance [5], [6]. ECC implementations specifically designed over the binary extension field exhibit substantially lower hardware complexity than their prime field counterparts, but their application space is substantially limited [5], [7]. Most of the design techniques aim for the optimization of modular arithmetic operations to speed up the performance [5], [6], [8]. For example, [6] and [8] minimized the cost of expensive modular division (MD) with alternative computation flows. Instead of performing a full modular reduction, the partial modular reduction method can be employed to lower the complexity of modular multiplication (MM) [6]. Additional performance improvement can be achieved with parallel and/or pipelined processing architecture [7]–[9]. However, neither do these designs realize an energy-efficient ECC implementation for IoT applications, nor do they consider the whole optimization space across the algorithm, architecture, and arithmetic circuit levels for further performance enhancement.

On the other hand, the private data stored in an unprotected IoT device could be extracted through side-channel attacks (SCA) during ECC computation [10], [11]. Among all kinds of SCA schemes, simple power analysis (SPA) is the simplest SCA method that directly observes the power profiles of devices to obtain secret key information. Since the standard double-and-add algorithm used in elliptic curve point multiplication (ECPM) involves non-identical group operations based on the private key, the resulting power consumption has a distinct profile that matches the actual key value. As a result, the attacker can perform SPA to easily obtain the key information. To prevent the SPA attack, double-and-add-always (DAA) algorithm is usually employed by inserting dummy elliptic curve point addition (ECPA) during ECPM operations [12]. While DAA can effectively eliminate the power variation and the corresponding information leak via SPA, the additional dummy operations severely degrade both the computation performance and energy efficiency. Another popular SPA-resistant scheme, the m-ary method, avoids using redundant operations while eliminating the power variation by storing the pre-computed points [13]. However, the associated implementation complexity is significant and causes additional power and energy consumption.

In this brief, an energy-efficient, SPA-resistant ECC processor that supports dual-field computations for IoT security applications is presented. The proposed processor exploits

Manuscript received March 31, 2020; revised June 9, 2020; accepted June 27, 2020. Date of publication July 28, 2020; date of current version September 3, 2020. This work was supported in part by the Ministry of Science and Technology, Taiwan, and in part by the Intelligent and Sustainable Medical Electronics Research Fund in National Taiwan University. This brief was recommended by Associate Editor N. Maghari. *(Ling-Yu Yeh and Po-Jen Chen contributed equally to this work.) (Corresponding author: Tsung-Te Liu.)*

Ling-Yu Yeh, Po-Jen Chen, and Tsung-Te Liu are with the Graduate Institute of Electronics Engineering, National Taiwan University, Taipei 10617, Taiwan (e-mail: lyyeh@eecs.ee.ntu.edu.tw; pjchen@eecs.ee.ntu.edu.tw; ttliu@ntu.edu.tw).

Chen-Chun Pai was with the Graduate Institute of Electronics Engineering, National Taiwan University, Taipei 10617, Taiwan. He is now with the Department of Silicon Products Development, MediaTek Inc., Hsinchu 300, Taiwan (e-mail: tom820501@gmail.com).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSII.2020.3012448

various design approaches across the algorithm, architecture, and arithmetic circuit levels to maximize energy efficiency. The proposed ECPM algorithm realizes SPA resistance with high area and energy efficiency by combining signed binary representation (SBR) with the m-ary method. Moreover, the proposed hybrid modular arithmetic architecture in which each arithmetic unit cooperatively performs modular operations significantly enhances the hardware utilization and energy efficiency. Finally, the processor employs an energy-efficient data flow that effectively minimizes memory overhead during group operations. Compared to the previous designs, the proposed ECC processor has a substantially lower area, and lower power and energy consumption with comparable speed performance. This makes it a promising candidate for resource-constrained IoT security applications.

The remainder of this brief is organized as follows. Section II introduces the proposed energy-efficient ECPM algorithm with SPA resistance. The hardware architecture of the proposed ECC processor is presented in Section III. Section IV gives FPGA and ASIC implementation results of the proposed ECC processor. Section V concludes this brief.

II. Proposed ECPM Algorithm

The traditional m-ary method with a window size w decomposes an ECPM computation into two operation phases: pre-computation phase and evaluation phase. During the pre-computation phase, multiples of the base point P, $2P, 3P, \ldots, (2^w - 1)P$, are first calculated and stored. This requires $(2^w - 2)$ ECPA and one elliptic curve point double (ECPD) operations. During evaluation, the complete ECPM is then performed based on the binary representation of the private key from the most significant window to the least significant window, with w ECPD operations followed by one ECPA operation in each window. Because of this operation regularity, the m-ary method is immune from SPA attack. Moreover, it realizes fewer ECPA operations and thus higher performance than the DAA algorithm. However, the associated memory overhead and hardware complexity with m-ary method is substantial. In addition to the storage of the pre-computed points, extra computations and memory are required to maintain SPA resistance specifically for the cases when the zero bit length of the most significant sequence in the private key is longer than the window size, or when the window value is equal to zero. This severely increases the implementation cost and overall energy consumption.

Algorithm 1 Signed Binary Representation m-Ary Method

```
Input: a point $P$, an integer $K = (K_{l-1}, \ldots, K_1, K_0)$ and a window $w \geq 2$
Output: $KP$
Key-encoding Phase:
  $K' \leftarrow (1, K_{l-1}, \ldots, K_2, K_1)_2$
  Return $K' = K + 1 = \sum_{i=0}^{d-1} a_i 2^{iw}$ where $a_i \in \{\pm 1, \ldots, \pm(2^w - 1)\}$
Pre-computation Phase:
  $P_1 = P$, $P_{2^w-1} = 2P$
  for $i = 3$ to $2^w - 1$ by 2 do $P_i = P_{i-2} + P_{2^w-1}$
  $Q' = O$
Evaluation Phase:
  for $i = d - 1$ to 0 by $-1$ do
      $Q' = 2^w Q'$, which requires $w$ ECPD
      $Q' = Q' + P_{a_i}$
  $Q_0 \leftarrow Q' - P$
  $Q_1 \leftarrow Q'$
  $Q \leftarrow Q_{K_0}$
  Return $Q$
```

TABLE I
Comparisons of ECPM Operation Cycle and Storage Between Different ECPM Algorithms

ECPM	DAA	m-ary	SBR m-ary
Pre -compute	0	$[1]PD$ $+[2^w - 2]PA$	$[1]PD$ $+[2^{w-1} - 1]PA$
Eva- luate	$[l-1]PD$ $+[l-1]PA$	$[(d-1)w]PD$ $+[d-1]PA$	$[(d-1)w]PD$ $+[d-1]PA + [1]PA$
Total Cycle	$[l-1]PD$ $+[l-1]PA$	$[wd - w + 1]PD$ $+[2^w + d - 3]PA$	$[wd - w + 1]PD$ $+[2^{w-1} + d - 1]PA$
Storage	0	$2l(2^w - 1)$	$2l(2^{w-1})$

In order to reduce the implementation cost and energy consumption while achieving SPA resistance, we propose to introduce an additional key-encoding phase to the m-ary method during ECPM computation. Algorithm 1 shows the proposed ECPM algorithm using the m-ary method encoded with signed binary representation (SBR), SBR m-ary algorithm. Instead of using the original key with 1 and 0 representation for pre-computation in the traditional m-ary method, the proposed SBR m-ary first transforms the private key into SBR with 1 and -1 before pre-computation. Since every SBR-encoded window consists of only odd numbers, the proposed SBR m-ary needs to compute and store only odd points during pre-computation. Moreover, because SBR ensures that the most significant bit of the encoded private key K' and the window value are never equal to zero, the additional computations and memory required to maintain SPA resistance specifically for the critical cases mentioned above with the traditional m-ary method can be avoided. On the other hand, since the implementation of SBR encoder requires only a shifter circuit, and the point inversion is trivial in ECC, the proposed SBR m-ary algorithm introduces nearly zero additional implementation overhead when compared to traditional DAA and the m-ary method. Table I summarizes and compares the operation cycle and the required storage size between DAA, the m-ary method, and the proposed SBR m-ary algorithm. The proposed SBR m-ary realizes twice the reduction in both memory size and operation time at the pre-computation phase when compared to the traditional m-ary method. To demonstrate the advantages of power and energy efficiency of the proposed approach, Fig. 1 shows the simulated performance results of different ECPM algorithms for a 192-bit ECC implementation with a window size $w = 4$. The ECC design based on the proposed SBR m-ary algorithm demonstrates 10.9% and 13.2% lower power and energy consumption than other approaches.

III. Proposed ECC Hardware Architecture

Traditional ECC hardware implementations for high-speed applications usually avoid the use of the affine coordinate

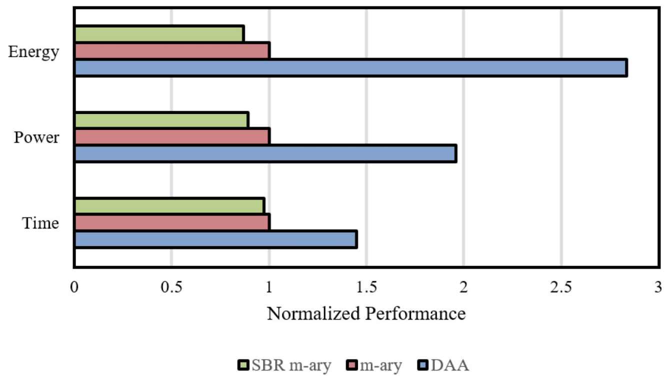

Fig. 1. Simulated performance results of different ECPM algorithms, where each result is normalized by the result of the m-ary method.

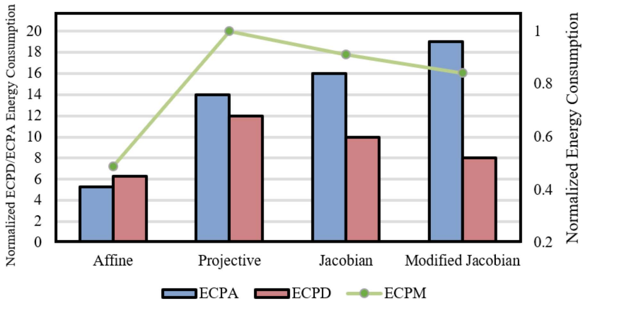

Fig. 2. Performance comparisons of ECC designs using different coordinate systems with $l = 192$ and $w = 4$. The right axis represents the total energy consumption normalized by the result of the projective system, while the left axis represents the normalized energy consumption contributed by ECPA and ECPD operations, respectively.

system due to the associated divider circuit cost [5], [7]. For energy-constrained IoT applications, however, the affine coordinate system emerges as a better solution to energy-efficient ECC implementation. Fig. 2 shows the performance comparisons of ECC designs based on different coordinate systems. The simulation results show that compared to other coordinate systems [14], the affine coordinate system consumes substantially less energy contributed by ECPA and ECPD operations. As a result, the ECC design based on the affine coordinate system realizes 42% lower energy per ECPM operation despite having expensive division operations.

Fig. 3 shows the proposed energy-efficient dual-field ECC architecture employing the proposed SBR m-ary algorithm and affine coordinate system. It consists of a key encoder, a Galois field arithmetic unit (GFAU), and an ECC controller. The key encoder converts the private key to SBR as shown in Algorithm 1. The GFAU performs modular arithmetic operations in both prime and binary fields. The ECC controller coordinates different modules, schedules data flow, and executes the entire ECC scheme.

A. Galoid Field Arithmetic Unit (GFAU)

The proposed GFAU consists of several modular arithmetic units to support dual-field computations. Instead of using the conventional design approach that optimizes each individual arithmetic unit independently, the proposed GFAU employs an optimization approach that jointly optimizes arithmetic units together to maximize the overall hardware utilization and energy efficiency. This leads to a hybrid modular arithmetic architecture in which each arithmetic unit cooperatively performs efficient arithmetic operations with a delicate resource sharing scheme. As illustrated in Fig. 4, the proposed GFAU employs only five l-bit registers for all modular arithmetic operations over both prime and binary fields. Fig. 4 also illustrates different arithmetic operations and the corresponding data flows with the proposed hybrid modular arithmetic architecture.

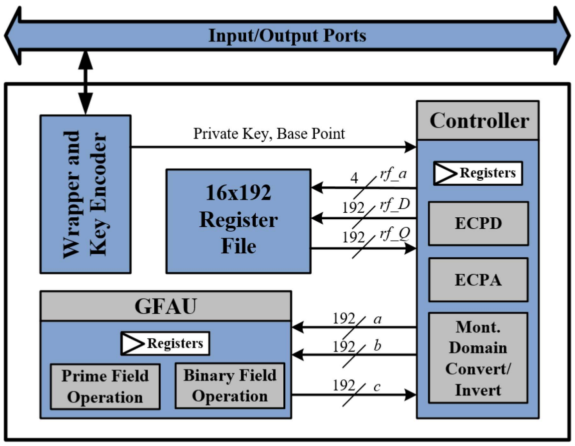

Fig. 3. Proposed energy-efficient dual-field ECC architecture.

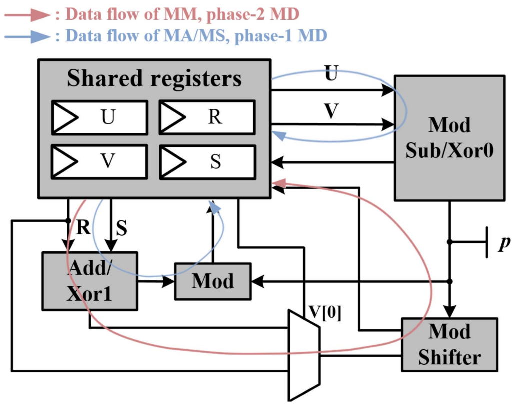

Fig. 4. Proposed hybrid modular arithmetic datapaths.

1) Modular Addition (MA) and Modular Subtraction (MS): The inputs a and b are first stored in the registers R and S (U and V), and then processed through the modular adder (subtractor) for the computation of $a + b \pmod{p} (a - b \pmod{p})$.

2) Modular Multiplication (MM): The inputs a and b are stored in the registers S and V instead. The adder originally used for MA and the modular shifter together perform the computation $(a + b[i])/2 \pmod{p}$ with multiple folded steps to complete an MM operation.

3) Modular Division (MD): The MD in the proposed GFAU is based on the Montgomery division algorithm [15] in which the inputs a and b are stored in the registers S and V, while the registers U and R are initialized with the prime p and 0. The modular adder and subtractor for MA and MS

▨ : Register is in use, - : Register is idle

State	Operation	R1	R2	R3	R4
ECPD: $\mathbf{P_3 = 2P_1}$					
0	*Init*	▨	▨	-	-
1	P_{1x}^2	▨	▨	▨	-
2	$3P_{1x}^2$	▨	▨	▨	▨
3	$3P_{1x}^2 + a$	▨	▨	-	▨
4	$2P_{1y}$	▨	▨	▨	▨
5	$m = (3P_{1x}^2 + a)/2P_{1y}$	▨	▨	-	▨
6	m^2	▨	▨	▨	▨
7	$P_{3x} = m^2 - 2P_{1x}$	▨	▨	▨	▨
8	$P_{1x} - P_{3x}$	▨	▨	▨	▨
9	$m(P_{1x} - P_{3x})$	-	▨	▨	▨
10	$P_{3y} = m(P_{1x} - P_{3x}) - P_{1y}$	-	-	▨	▨
ECPA: $\mathbf{P_4 = P_2 + P_3}$					
11	$P_{2y} - P_{3y}$	-	▨	▨	▨
12	$rf\_Q - P_{3x}$	▨	▨	▨	▨
13	$m = (P_{2y} - P_{3y})/(rf\_Q - P_{3x})$	▨	-	▨	▨
14	m^2	▨	▨	▨	▨
15	$m^2 - rf\_Q$	▨	▨	▨	▨
16	$P_{4x} = m^2 - rf\_Q - P_{3x}$	▨	▨	-	▨
17	$P_{3x} - P_{4x}$	▨	▨	▨	▨
18	$m(P_{3x} - P_{4x})$	▨	▨	▨	▨
19	$P_{4y} = m(P_{3x} - P_{4x}) - P_{3y}$	-	-	▨	▨

Fig. 5. Proposed data flow of the ECC controller.

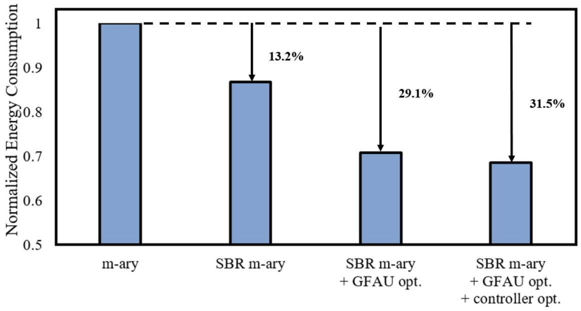

Fig. 6. Comparison of energy consumption with different optimization steps.

are again re-used to perform the extended Binary GCD algorithm during the Phase-1 MD step with four registers U, V, R, and S. The modular reduction operation during the Phase-2 MD can be efficiently completed by re-utilizing the modular shifter originally dedicated only to MM with the registers S and V initialized with zero value. As a result, the modular reduction circuit traditionally required for MD operation can be completely eliminated.

By employing the proposed hybrid modular arithmetic architecture, the cost of MM and MD operations are substantially reduced without the need for dedicated multiplier and divider modules. Therefore, the proposed GFAU with hybrid modular arithmetic architecture achieves an 29.7% reduction in energy consumption when compared to the conventional GFAU design, which is realized by implementing Montgomery multiplication algorithm and division algorithm [15] with dedicated modular multiplier and divider units.

B. ECC Controller

The proposed ECC controller is composed of a Montgomery domain converting unit, an ECPD, and an ECPA processing unit. The domain converting unit first transforms the base point $P(P_x, P_y)$ and the related elliptic curve parameters into Montgomery domain, and also inverts the final computation result KP, back to the normal integer domain. The proposed ECC controller architecture employs a data flow and a computation scheme that optimize both the memory access and storage for maximum utilization rate, as shown in Fig. 5. The proposed ECC controller requires only four l-bit registers with almost full utilization rate. With the proposed data flow and computation scheme, the ECC controller achieves 11.9% lower energy consumption when compared to traditional methods.

TABLE II
FPGA IMPLEMENTATION RESULTS

Design	Area(ALUTs)	Area(Registers)	f(MHz)	Hardware Opt.
m-ary	13109	8895	50	No
SBR m-ary	12934	5629	50	No
SBR m-ary	10802	5297	50	GFAU
SBR m-ary	10420	5105	50	GFAU + controller

C. Summary

By applying all optimization techniques described above, we can reduce the energy consumption of the ECC processor by up to 31.5%. As shown in Fig. 6, the energy of the proposed ECC processor is reduced by 13.2% using the proposed SBR m-ary ECPM algorithm. Together with the proposed hybrid modular arithmetic architecture, the total energy reduction reaches 29.1%. Finally, when the optimized data flow is further utilized in the controller, 31.5% of the total energy consumption can be reduced.

IV. IMPLEMENTATION RESULTS

A 192-bit dual-field ECC prototype with the proposed design approaches was synthesized using Quartus Prime 18.1, and implemented and verified on Altera Stratix IV EP4SGX230C2 FPGA platform. Table II summarizes the FPGA implementation results of ECC designs in which different optimization methods have been employed. The areas of ALUTs and registers are both reduced accordingly with different optimization techniques, which is consistent with the simulation results shown in Fig. 6. The total implementation cost has been significantly reduced by 27.0% with the proposed SBR m-ary ECPM algorithm and the optimization techniques used in GFAU and ECC controller.

Since it is difficult to compare the power and energy results of ECC designs implemented on different FPGA platforms, the proposed ECC processor was also implemented in ASIC with TSMC 90-nm CMOS technology for further performance characterization and comparison. The power and energy results are obtained using Synopsys Prime-Time EDA tool. Table III summarizes the ASIC implementation results and performance comparisons with state-of-the-art designs. The proposed dual-field ECC processor realizes a more compact design with comparable speed performance, while consuming the lowest amount of energy and power. In terms of energy-area product, which is a useful performance metric to evaluate the hardware efficiency for resource-constrained IoT applications, the proposed design substantially outperforms other ECC implementations by a factor of 2.39 and 2.69 over the prime and

TABLE III
ASIC IMPLEMENTATION RESULTS AND COMPARISONS

	This Work ‡		VLSI'14 [9] ‡		TCAS-II'18 [6]†	TCAS-I'19 [8]†
Technology	90		90		180	180
Gate count (kGates)	83		122		145	466
Operation Field	$G(p)$	$G(2^m)$	$G(p)$	$G(2^m)$	$G(p)$	$G(p)$
	192	192	192	192	256	256
fmax (MHz)	250	250	263	263	185	360
Op. Time (ms)	0.85	0.80	0.36	0.32	0.20	0.04
Power (mW)	14.0	11.3	67.8	56.9	N/A	456
Energy (mJ/Op.)	11.8	9.0	24.4	18.2	N/A	19.9
Area Energy Product ◇	100	100	275	269	N/A	239⋆

‡: Design supports dual-field and reports post-layout simulation results.
†: Design supports only prime field and reports only pre-layout simulation results.
◇: Area energy product = $(\text{gate count}\times\text{energy}\times 100)/\lambda$, where $\lambda = (\text{gate count}\times\text{energy})$ of this work.
⋆: Technology and application scaling factor is 0.1157: $(90\text{-nm}/180\text{-nm}) \times (1.0\text{V}/1.8\text{V})^2 \times (192\text{-bit}/256\text{-bit})$; normalized with pre-layout results.

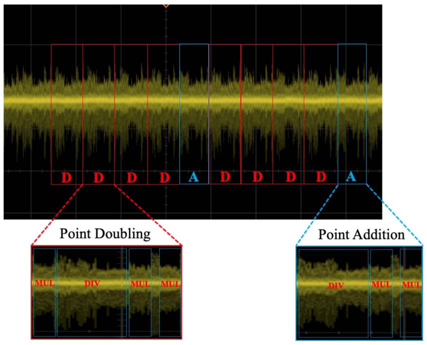

Fig. 7. Measured power trace of the proposed ECC processor. Two sets of operations, each consisting of four ECPD (D) and one ECPA (A) operations, are highlighted in boxes.

the binary fields, respectively. To demonstrate the SPA resistance, Fig. 7 shows the measured power trace of the proposed ECC processor. In each iteration, four ECPD and one ECPA operations are regularly performed irrespective of the private key. The proposed ECC processor demonstrates low power and energy consumption with SPA resistance, which makes it a promising solution to securing the IoT devices.

V. CONCLUSION

An ECC hardware implementation for energy-constrained IoT applications must consume low power and energy with high reliability. In this brief, an energy-efficient ECC processor is presented to support dual-field computations with SPA resistance. The proposed design uses the SBR m-ary algorithm to achieve both efficient and secured ECPM computations. The proposed hybrid modular arithmetic architecture significantly enhances the hardware utilization, which reduces both area and energy costs. The processor further employs an energy-efficient data flow to minimize the memory overhead. The ECC hardware implementation using the proposed design techniques achieves energy reduction of 51.6% and 50.5% for each $GF(p)$ and $GF(2^m)$ ECPM operation, respectively, when compared to state-of-the-art ECC designs.

ACKNOWLEDGMENT

The authors would like to thank Taiwan Semiconductor Research Institute, Taiwan for their technical support. The authors would also like to thank Prof. J.-P. Chen for his technical support in FPGA measurement.

REFERENCES

[1] K. Lauter, "The advantages of elliptic curve cryptography for wireless security," *IEEE Wireless Commun.*, vol. 11, no. 1, pp. 62–67, Feb. 2004.

[2] P. Urien and S. Piramuthu, "Elliptic curve-based RFID/NFC authentication with temperature sensor input for relay attacks," *Decis. Support Syst.*, vol. 59, pp. 28–36, Mar. 2014.

[3] O. Hauck, A. Katoch, and S. A. Huss, "VLSI system design using asynchronous wave pipelines: A 0.35 /spl mu/m CMOS 1.5 GHz elliptic curve public key cryptosystem chip," in *Proc. 6th Int. Symp. Adv. Res. Asynchronous Circuits Syst. (ASYNC) (Cat. No. PR00586)*, Eilat, Israel, Apr. 2000, pp. 188–197.

[4] D. Johnson, A. Menezes, and S. Vanstone, "The elliptic curve digital signature algorithm (ECDSA)," *Int. J. Inf. Security*, vol. 1, no. 1, pp. 36–63, 2001.

[5] Z. Khan and M. Benaissa, "Throughput/area-efficient ECC processor using montgomery point multiplication on FPGA," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 62, no. 11, pp. 1078–1082, Nov. 2015.

[6] P. Choi, M. Lee, J. Kim, and D. K. Kim, "Low-complexity elliptic curve cryptography processor based on configurable partial modular reduction over NIST prime fields," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 65, no. 11, pp. 1703–1707, Nov. 2018.

[7] Z. U. A. Khan and M. Benaissa, "High-speed and low-latency ECC processor implementation over GF(2^m) on FPGA," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 1, pp. 165–176, Jan. 2017.

[8] J. Ding, S. Li, and Z. Gu, "High-speed ECC processor over NIST prime fields applied with toom–cook multiplication," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 3, pp. 1003–1016, Mar. 2019.

[9] J. Lee, S. Chung, H. Chang, and C. Lee, "Efficient power-analysis-resistant dual-field elliptic curve cryptographic processor using heterogeneous dual-processing-element architecture," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 22, no. 1, pp. 49–61, Jan. 2014.

[10] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Advances in Cryptology—CRYPTO'99* (Lecture Notes in Computer Science). Heidelberg, Germany: Springer, 1999, pp. 388–397.

[11] P. C. Kocher, "Timing attacks on implementations of Diffie–Hellman, RSA, DSS, and other systems," in *Advances in Cryptology—CRYPTO'96* (Lecture Notes in Computer Science). Heidelberg, Germany: Springer, 1996, pp. 104–113.

[12] J.-S. Coron, "Resistance against differential power analysis for elliptic curve cryptosystems," in *Cryptographic Hardware and Embedded Systems* (Lecture Notes in Computer Science). Heidelberg, Germany: Springer, 1999, pp. 292–302.

[13] I. F. Blake, *Elliptic Curves in Cryptography*. Cambridge, U.K.: Cambridge Univ. Press, 1999.

[14] H. Cohen, A. Miyaji, and T. Ono, "Efficient elliptic curve exponentiation using mixed coordinates," in *Advances in Cryptology—ASIACRYPT'98*, K. Ohta and D. Pei, Eds. Heidelberg, Germany: Springer, 1998, pp. 51–65.

[15] B. S. Kaliski, "The montgomery inverse and its applications," *IEEE Trans. Comput.*, vol. 44, no. 8, pp. 1064–1065, Aug. 1995.