Dual-Basis Superserial Multipliers for Secure Applications and Lightweight Cryptographic Architectures

Siavash Bayat-Sarmadi, *Member, IEEE*, Mehran Mozaffari Kermani, *Member, IEEE*, Reza Azarderakhsh, and Chiou-Yng Lee, *Senior Member, IEEE*

***Abstract*—Cryptographic algorithms utilize finite-field arithmetic operations in their computations. Due to the constraints of the nodes which benefit from the security and privacy advantages of these algorithms in sensitive applications, these algorithms need to be lightweight. One of the well-known bases used in sensitive computations is dual basis (DB). In this brief, we present low-complexity superserial architectures for the DB multiplication over $\mathrm{GF}(2^m)$. To the best of our knowledge, this is the first time that such a multiplier is proposed in the open literature. We have performed complexity analysis for the proposed lightweight architectures, and the results show that the hardware complexity of the proposed superserial multiplier is reduced compared with that of regular serial multipliers. This has been also confirmed through our application-specific integrated circuit hardware- and time-equivalent estimations. The proposed superserial architecture is a step forward toward efficient and lightweight cryptographic algorithms and is suitable for constrained implementations of cryptographic primitives in applications such as smart cards, handheld devices, life-critical wearable and implantable medical devices, and constrained nodes in the blooming notion of Internet of nano-Things.**

***Index Terms*—Crypto-systems, finite-field multiplication, lightweight cryptographic algorithms, security, superserial.**

I. Introduction

LIGHTWEIGHT and resource-constrained applications require low-area and power-efficient hardware implementations. In cryptography, this requirement is elevated considering that sensitive applications might be deployed in remote areas for which low battery power is available, e.g., remote sensitive habitat monitoring. More importantly, replacing the discharged batteries in power-inefficient architectures may not be possible in some applications; for instance, in implanted medical devices, not only may surgery be needed for removing the drained batteries but this could also be in many cases life threatening [1]. Aside from these important applications, there are many other sensitive area-constrained applications such as handheld devices, smart cards, and active near-field communication tags [2] for which lightweight cryptography is essential.

Cryptographic algorithms use finite-field arithmetic operations for their primitive computations [3]–[16]. Among the multiplication schemes used for these algorithms, there have been many polynomial-basis (PB) multipliers presented to date (see, e.g., [17]–[24]), which include bit-parallel, bit-serial, and digit-serial implementations. Nonsystolic and nonsubquadratic high-performance bit-parallel architectures require the space complexity of typically $O(m^2)$, and their latencies are typically of $O(m)$, where m is the field size. The hardware complexity for these multipliers can be decreased by using digit-serial multipliers at the expense of slower multiplication. Moreover, bit-serial multipliers require only typically $O(m)$ space complexity, typically suitable for area-constrained applications. Furthermore, a PB superserial multiplier has been presented in [25].

Dual-basis (DB) multipliers have received much attention in the literature (see, e.g., [26]–[29]). Berlekamp [30] first developed the DB of the PB for implementing bit-serial multiplication, which can be efficiently used in Reed–Solomon encoders. Moreover, the research work presented in [27] has derived a bit-serial DB multiplier to reach low space and time complexity. Furthermore, it is shown in [28] that the DB can be represented by the corresponding PB. In this brief, we present a superserial multiplication scheme for the DB as an extension to the PB superserial multiplier presented in [25]. We also present complexity analysis for regular bit-serial and DB superserial multipliers along with a PB superserial multiplier [25]. Our experimental results show that this multiplier is more lightweight compared with the regular bit-serial multipliers and the PB superserial multiplier. However, due to the overhead of the basis conversion, it is more efficient to use the presented multipliers in a series of multiplication.

This brief is organized as follows. Section II presents the preliminaries regarding the DB multiplication over $\mathrm{GF}(2^m)$. In Section III, our new DB superserial multiplier is presented and its complexities are also presented. Moreover, complexity analysis is presented in this section. Finally, we conclude the brief in Section IV.

Manuscript received July 25, 2013; accepted October 31, 2013. Date of publication December 13, 2013; date of current version February 21, 2014. This brief was recommended by Associate Editor M. M. Mansour.

S. Bayat-Sarmadi is with the Department of Computer Engineering, Sharif University of Technology, Tehran 11365-9363, Iran (e-mail: bayatsr@gmail.com).

M. Mozaffari Kermani is with the Department of Electrical and Microelectronic Engineering, Rochester Institute of Technology, Rochester, NY 14623 USA (e-mail: m.mozaffari@rit.edu).

R. Azarderakhsh is with the Department of Computer Engineering, Rochester Institute of Technology, Rochester, NY 14623 USA (e-mail: azarderakhsh@gmail.com).

C.-Y. Lee is with the Department of Computer Information and Network Engineering, Lunghwa University of Science and Technology, Taoyuan 33306, Taiwan (e-mail: pp010@mail.lhu.edu.tw).

Digital Object Identifier 10.1109/TCSII.2013.2291075

1549-7747 © 2013 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

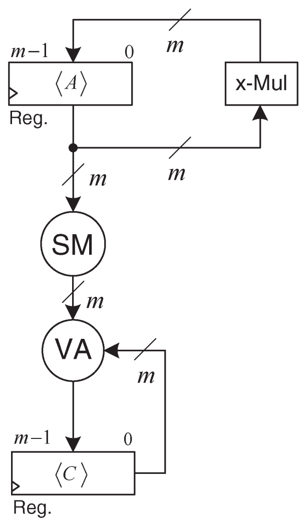

Fig. 1. MSB-first bit-serial multiplier over $\mathrm{GF}(2^m)$.

II. Preliminaries

A. DB Representation of Binary Extension Fields

A finite field of characteristic two or binary extension field $\mathrm{GF}(2^m)$ is a finite field that contains 2^m different elements (size of the field is 2^m) [31]. This finite field is an extension of GF(2), which contains 0 and 1. The $\mathrm{GF}(2^m)$ is associated with an irreducible polynomial of degree m (m is the extension degree over the binary field) as follows:

$$P(z) = p_m z^m + p_{m-1} z^{m-1} + \cdots + p_1 z + p_0 \tag{1}$$

where $p_i \in \mathrm{GF}(2)$. The field elements in $\mathrm{GF}(2^m)$ can be represented using different representation bases such as normal basis, PB, or DB. Let x be a root of the irreducible polynomial $P(z)$, i.e., $P(x) = 0$. Then, the set $\{1, x, x^2, \ldots, x^{m-1}\}$ is known as the PB, and an element $A \in \mathrm{GF}(2^m)$ can be represented as linear combinations of this set with a polynomial of degree $m-1$ over GF(2), as $A = \sum_{i=0}^{m-1} a_i x^i$, where $a_i \in \mathrm{GF}(2)$. For simplicity, a bit-vector representation is commonly used so that $A = (a_{m-1}, a_{m-2}, \ldots, a_1, a_0)$, where a_{m-1} and a_0 are the most significant bit (MSB) and the least significant bit (LSB), respectively.

Suppose that the DB (also known as complementary basis) of the PB defined earlier is $\{y_0, y_1, \ldots, y_{m-1}\}$, where $y_i \in \mathrm{GF}(2^m)$. Then, one obtains

$$\mathrm{Tr}(y_i x^j) = \begin{cases} 0, & \text{if } i \neq j \\ 1, & \text{otherwise} \end{cases}$$

where $\mathrm{Tr}(z \in \mathrm{GF}(2^m)) = \sum_{i=0}^{m-1} z^{2^i}$ is the trace function defined over $\mathrm{GF}(2^m) \to \mathrm{GF}(2)$.

Let us represent the element $A \in \mathrm{GF}(2^m)$ in DB and the element $B \in \mathrm{GF}(2^m)$ in PB. Then, their product, i.e., $C = A \times B$, is represented in DB as follows (note that the multiplication is low to high or LSB first):

$$\begin{aligned} C_{\mathrm{DB}} &= A_{\mathrm{DB}} B_{\mathrm{PB}} \mathrm{mod} P(x) = A \sum_{i=0}^{m-1} b_i x^i \\ &= A b_{m-1} x^{m-1} \oplus A b_{m-2} x^{m-2} \oplus \cdots \oplus A b_1 x \oplus A b_0. \end{aligned} \tag{2}$$

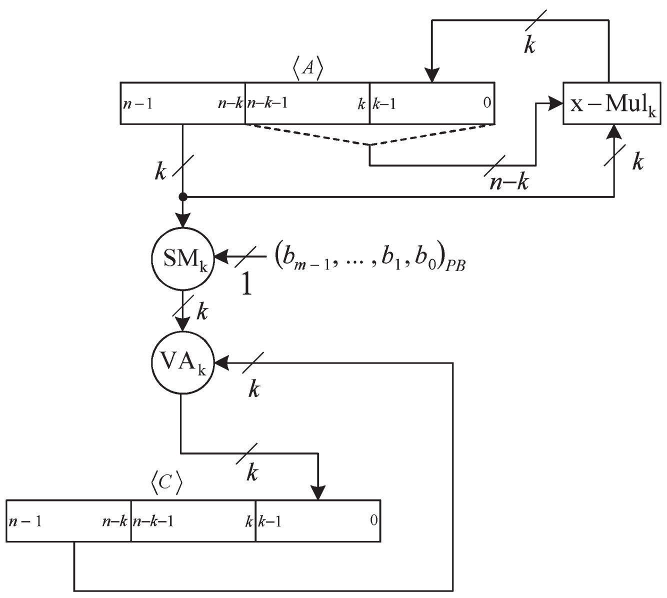

Fig. 2. Proposed DB superserial multiplier.

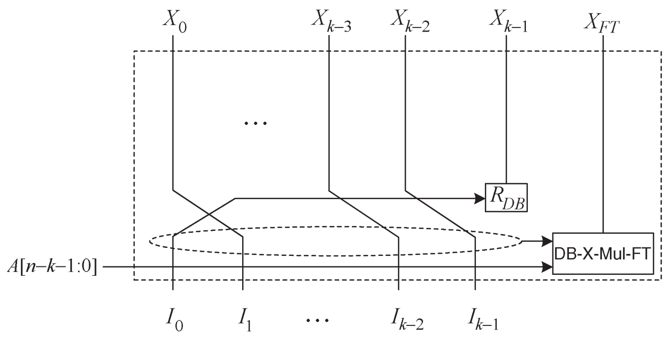

Fig. 3. The partial k-bit x-Mul (x-Mul$_k$) module.

There are three main arithmetic operations in (2) as follows: 1) m-bit modulo-2 addition or bitwise XOR operation referred to as VA; 2) bitwise AND operation (for m-bit) referred to as SM; and 3) multiplication by x referred to as x-Mul. As such, Fig. 1 depicts the DB bit-serial (or simply serial) multiplier.

III. Proposed Superserial Multiplier Using DB

In this section, we present the DB superserial multiplier. The DB usually requires a basis conversion from and/or to PB; however, we would like to note that for a series of multiplication, two basis conversions at the very beginning and/or at the very end are needed [32]. This can achieve lower overhead impact for the basis conversion if the field is generated by trinomials or by pentanomials. We also should note that DB superserial architecture is more regular in the sense that all the field defining polynomial addition (reduction process) happen on the final results. Then, the architecture is not very much dependent on the distribution of the nonzero coefficients in the irreducible polynomial [will be shown in (3) and (4)]. Thus, this means that the controller complexity in the DB multiplier is lower than the PB one. Furthermore, the space complexity of the DB superserial multiplier is less than the PB one in general. This can further compensate for the overhead of the basis conversion.

TABLE I
COMPLEXITIES OF THE PREVIOUS WORKS AND THE PRESENTED DB SUPERSERIAL MULTIPLIER

Architecture	XOR (#)	AND (#)	FF and/or memory bit (#)	Latency (# of clock cycles)	Critical path delay
Regular bit-serial DB	$m+\omega-1$	m	$3m$	m	Max $(T_A+T_X,\ (\omega-1)T_X)$
DB superserial (proposed)	$k+\omega-1$	k	$m+2n+1$	qn	Max $(T_A+T_X,\ (\omega-1)T_X)$
PB superserial [19]	$k+\omega-1$	$k+\omega-1$	$2m+(2n+1+p')q+1$	qn	T_A+2T_X
DB KOM [23]	k^2+5k+1	k^2+2	$2(k^2+2k+n)$	q^2	$(k+2)\times(T_A+T_X)+T_A+2T_X$
Scalable DB [20]	$4(k^2+k)$	$4k^2$	$2n+8k^2+4k$	q^2+4k-1	$2k(T_A+T_X)$

m : field size, w : hamming weight of $P(x)$, n : size of each vector after zero padding, k : bit length, q : number of k-bit parts.
T_A : delay of an AND gate, T_X : delay of an XOR gate, p': the number of slices needing to have feedback from last term.
- We Consider the area of a three-input XOR gate as 1.5 times of that of a two-input one.

As mentioned in Section II, one of the main components for the DB multiplier is x-Mul. We perform the x-Mul calculation considering [27] as follows:

$$xA=x\sum_{i=0}^{m-1}a'_iy_i=\sum_{i=0}^{m-2}(a'_{i+1}y_i)+\sum_{i=0}^{m-1}a'_ip_iy_{m-1}=\sum_{i=0}^{m-1}a'_{i+1}y_i \tag{3}$$

where p_i are the coefficients of the field defining the polynomial $P(x)$ and

$$a'_m=\sum_{i=0}^{m-1}a'_ip_i. \tag{4}$$

Let us assume that ω is the hamming weight of $P(x)$. Then, the number of modulo-2 addition operations (XOR) in (3) is $\omega-1$.

To have a subword product, we need to divide each vector to, e.g., k bits $(k<m)$. Let $m=q'k+r$; if $r\neq 0$, one needs to zero-pad the vector. Let us assume $n=qk$ be the size of each vector after zero padding. Clearly, $n\geq m$; then, we have

$$xA=\sum_{i=0}^{n-1}a'_{i+1}y_i \tag{5}$$

where $a'_i=0$ for $m<i\leq n$.

Now, we can divide (5) as follows:

$$xA=\sum_{j=0}^{q-1}\sum_{i=0}^{k-1}a'_{jk+i+1}y_{jk+i}. \tag{6}$$

A. Architecture

Here, the hardware architecture for the DB superserial multiplier is presented, as depicted in Fig. 2.

Register A is divided into q parts of k-bit length. The most significant k bits of A are used for both x-Mul and SM partial modules. The partial k-bit x-Mul, SM, and VA modules are referred to as x-Mul$_k$, SM$_k$, and VA$_k$ modules, respectively.

It is worth mentioning that, according to (3) and (4), only for the final term in x-Mul, there is a need for addition (XOR) of other coefficients (with regard to $P(x)$) and for the rest, it is basically a shift. Therefore, even for x-Mul$_k$, where the coefficients are not in their original location while we calculate the final term, we only need to know where they are moved to. Fig. 3 depicts the x-Mul$_k$ module. This module simply implements 1-bit shift toward the LSB, storing the LSB in a register (R_{DB}) for the next computation cycle. We note that the initial value for R_{DB} is zero.

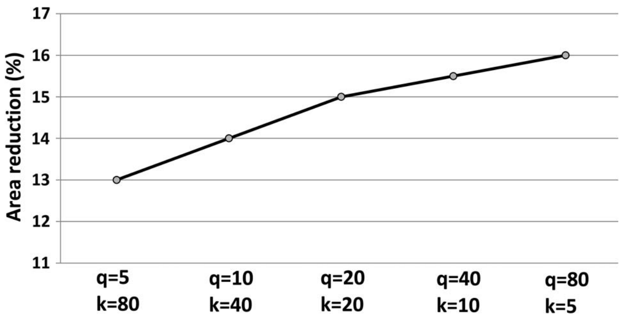

Fig. 4. Area reduction for the proposed superserial multiplier compared with the regular architecture.

The final term, i.e., the MSB, of x-Mul that should be computed using (4) is computed at the beginning of each full x-Mul round, i.e., $CC\%q=1$, where CC is the clock cycle count. The module performing such calculation is referred to as DB-x-Mul-FT in Fig. 3, which is basically an XOR tree, and is as follows:

$$X_{\text{FT}}=\sum_{i=0}^{n-k-1}a'_ip_i+\sum_{i=n-k}^{n-1}I_{i-n+k}p_i. \tag{7}$$

As discussed earlier, (7) requires $\omega-1$ modulo-2 addition. Hence, the DB-x-Mul-FT module needs $\omega-1$ XOR gates.

Assuming that $X[k-1:0]$ is the output of x-Mul$_k$, register A in Fig. 2 is updated as follows.

- if $CC=0$ (at reset)
  - $A[n-1:0]\Leftarrow$ input A;
- else if $CC\%q=1$
  - $A[n-1:0]\Leftarrow\{A[n-k-1:m-k],X_{\text{FT}},A[m-k-2:0],X[k-1:0]\}$
  - $*$ Clearly if $m=n$, we have:
  - $*$ $A[m-1:0]\Leftarrow\{X_{\text{FT}},A[m-k-2:0],X[k-1:0]\}$
- otherwise
  - $A[n-1:0]\Leftarrow\{A[n-k-1:0],X[k-1:0]\}$.

SM$_k$ and VA$_k$ modules contain k two-input AND and XOR gates, respectively. Moreover, input B is shifted into the

TABLE II
COMPLEXITY REDUCTIONS (PERCENT REDUCTION) OF THE PRESENTED ARCHITECTURE COMPARED TO THE PREVIOUS WORKS ($m = 409$, $n = qk$, DIFFERENT VALUES FOR q AND k)

Parameters		Reductions compared to [19]		Reductions compared to [23]		Reductions compared to [20]	
q	k	Area (%)	Time×area (%)	Area (%)	Time×area (%)	Area (%)	Time×area (%)
28	15	94	91	17	5	61	51
21	20	93	89	37	6	75	72
17	25	91	86	51	21	82	83
14	30	89	83	62	38	87	90
12	35	88	81	70	51	91	94

SM_k module one bit at a time from the LSB to the MSB after every q clock cycles.

Finally, after each clock cycle, register C in Fig. 2 is updated as follows (assuming that $\mathrm{VA}[k-1:0]$ is the output of VA_k):

- if $CC = 0$ (at reset)
  - $C[n-1:0] \Leftarrow 0$
- else
  - $C[n-1:0] \Leftarrow \{C[n-k-1:0], \mathrm{VA}[k-1:0]\}$.

B. Complexity Analysis and Comparisons

Table I presents the space complexity and time complexity of the previous and the proposed DB superserial multipliers (given that FF stands for flip-flop).

In the following, we justify the numbers presented in the table for the DB superserial multiplier.

- VA_k and DB-x-Mul-FT modules require k and $\omega - 1$ XOR gates, respectively.
- SM_k also requires k AND gates.
- The registers A, C, and R_{DB} consist of n FFs, n FFs, and one FF, respectively. In addition, shift register B requires n FFs.
- The multiplier multiplies each bit of the input B by the input A in q clock cycles.
- The critical path is the slower path among the path from A to C or the path from A to A (shift k-bit most significant digit register) through the DB-x-Mul-FT module.

We note that the complexity of the controller for none of the given multipliers is considered in Table I. However, the complexity of a PB superserial multiplier is higher than the DB one. The reason is that in the PB superserial multiplier, depending on the location of the nonzero coefficients in the field-defining polynomial, it might need a number of partial modulo-2 addition operations (partial reduction) in several cycles; however, in the DB multiplier, all modulo-2 addition operations (XOR) happen only for the final term.

For comparing the time and area overheads, we use the NanGate standard-cell library[1] [33]. We use the typical corner ($V_{dd} = 1.10$ V and $T_j = 25^\circ$ C.) and the drive strength of one for all the utilized primitives here (implying similar input transition and load capacitance for the primitives).

[1]The library was generated using NanGate's Library Creator and the 45-nm FreePDK Base Kit from North Carolina State University and characterization was done using the Predictive Technology Model from Arizona State University [33].

According to Table I (see first two rows), we have shown the area overhead reductions of the proposed architecture compared with to the regular architecture in Fig. 4. As shown in this figure, considering $m = n = 400$, $w = 5$, and different values for q and k, the area reductions achieved through utilizing the proposed superserial scheme in this brief are depicted. As shown in Fig. 4, the area saving starts from 13% going up to more than 16%; however, as this saving goes up, the latency (the values for qn) is increased. Therefore, according to the performance requirements and area reductions needed, one can use the selected architectures based on the savings depicted in Fig. 4 to construct the proposed superserial multipliers.

In addition, we have compared the area and time × area of the proposed DB superserial multiplier with those of the previous multipliers presented in [25] (considering $p' = 1$ which results in the lowest area), [29], and [26], the results of which are reported in Table II. As shown in this table, the area reductions and the time × area savings are presented for different values of q and k. Lightweight and efficient applications are typically characterized through lower area and time × area metrics. Based on this table, it is shown that the proposed architecture is more lightweight and efficient compared with these multipliers.

IV. CONCLUSION

In this brief, we have presented a superserial multiplication scheme for the DB. The merit in proposing such multipliers is their lightweight architectures, which are useful for constrained applications in cryptography and security. Our analysis results show the area savings achieved by our architecture and its efficiency benefits. Our experimental results show that this multiplier is more lightweight compared with both regular bit-serial multiplier and the PB superserial multiplier. Specifically, we achieve the area reductions of 13%–16% for the utilized parameters of q and k, at the expense of increased latency compared with the architecture of the regular bit-serial multiplier. The fact that, in [32], it is shown that, if the field is constructed by trinomials, the basis conversion from the DB to the PB is without hardware cost is also another very useful characteristic, motivating the use of DB. The proposed architecture is suitable for cryptographic applications and secure environments, applicable to five binary fields recommended by the National Institute of Standards and Technology. Based on the requirements in terms of performance, area, and power consumption, one can fine-tune and utilize the proposed DB superserial scheme for various secure, constrained, and lightweight applications.

Acknowledgment

The authors would like to thank the reviewers for their careful reading and giving constructive comments. R. Azarderakhsh would like to thank A. Ava-Azarderakhsh for the kind support throughout the research performed for writing the paper.

References

[1] M. Mozaffari Kermani, M. Zhang, A. Raghunathan, and N. K. Jha, "Emerging frontiers in embedded security," in *Proc. VLSI Design*, Jan. 2013, pp. 203–208.

[2] NFC. [Online]. Available: http://www.nfc-forum.org/

[3] J. Adikari, V. S. Dimitrov, and R. J. Cintra, "A new algorithm for double scalar multiplication over Koblitz curves," in *Proc. IEEE ISCAS*, 2011, pp. 709–712.

[4] M. Mozaffari Kermani and A. Reyhani-Masoleh, "Parity-based fault detection architecture of S-box for Advanced Encryption Standard," in *Proc. IEEE Int. Symp. Defect Fault-Tolerance VLSI Syst.*, 2006, pp. 572–580.

[5] M. Mozaffari Kermani and A. Reyhani-Masoleh, "Concurrent structure-independent fault detection schemes for the Advanced Encryption Standard," *IEEE Trans. Comput.*, vol. 59, no. 5, pp. 608–622, May 2010.

[6] K. Järvinen and J. Skytta, "On parallelization of high-speed processors for elliptic curve cryptography," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 16, no. 9, pp. 1162–1175, Sep. 2008.

[7] V. S. Dimitrov, K. U. Järvinen, M. J. Jacobson, W. F. Chan, and Z. Huang, "Provably sublinear point multiplication on Koblitz curves and its hardware implementation," *IEEE Trans. Comput.*, vol. 57, no. 11, pp. 1469–1481, Nov. 2008.

[8] M. Mozaffari Kermani and R. Azarderakhsh, "Efficient fault diagnosis schemes for reliable lightweight cryptographic ISO/IEC standard CLEFIA benchmarked on ASIC and FPGA," *IEEE Trans. Ind. Electron.*, vol. 60, no. 12, pp. 5925–5932, Dec. 2013.

[9] J. Pan, R. Azarderakhsh, M. Mozaffari Kermani, C.-Y. Lee, W. Lee, C. Chiou, and J. Lin, "Low-latency digit-serial systolic double basis multiplier over GF(2^m) using subquadratic Toeplitz matrix-vector product approach," *IEEE Trans. Comput.*, to be published.

[10] M. Mozaffari Kermani, R. Azarderakhsh, C.-Y. Lee, and S. Bayat-Sarmadi, "Reliable concurrent error detection architectures for extended Euclidean-based division over GF(2^m)," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, to be published.

[11] R. Azarderakhsh, K. Järvinen, and M. Mozaffari Kermani, "Efficient algorithm and architecture for elliptic curve cryptography for extremely constrained secure applications," *IEEE Trans. Circuits Syst. I, Reg. Papers*, to be published.

[12] M. Mozaffari Kermani and A. Reyhani-Masoleh, "A high-performance fault diagnosis approach for the AES SubBytes utilizing mixed bases," in *Proc. Workshop FDTC*, 2011, pp. 80–87.

[13] R. Azarderakhsh and A. Reyhani-Masoleh, "Efficient FPGA implementations of point multiplication on binary Edwards and generalized Hessian curves using Gaussian normal basis," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 8, pp. 1453–1466, Aug. 2012.

[14] M. Mozaffari Kermani and A. Reyhani-Masoleh, "Reliable hardware architectures for the third-round SHA-3 finalist Grostl benchmarked on FPGA platform," in *Proc. IEEE Int. Conf. DFT VLSI Nanotechnol. Syst.*, 2011, pp. 325–331.

[15] R. Azarderakhsh and K. Karabina, "A new double point multiplication algorithm and its application to binary elliptic curves with Endomorphisms," *IEEE Trans. Comput.*, to be published.

[16] M. Mozaffari Kermani and A. Reyhani-Masoleh, "Efficient and high-performance parallel hardware architectures for the AES-GCM," *IEEE Trans. Comput.*, vol. 61, no. 8, pp. 1165–1178, Aug. 2012.

[17] S. Bayat-Sarmadi and M. Hasan, "On concurrent detection of errors in polynomial basis multiplication," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 15, no. 4, pp. 413–426, Apr. 2007.

[18] A. Hariri and A. Reyhani-Masoleh, "Digit-serial structures for the shifted polynomial basis multiplication over binary extension fields," in *Proc. LNCS Int. WAIFI*, vol. 5130, *LNCS*, 2008, pp. 103–116.

[19] P. Meher, "Systolic formulation for low-complexity serial-parallel implementation of unified finite field multiplication over $\mathrm{GF}(2^m)$," in *Proc. IEEE Intl Conf. Appl.-Spec. Syst., Architect. Processors*, 2007, pp. 134–139.

[20] P. Meher, "Systolic and super-systolic multipliers for finite field $\mathrm{GF}(2^m)$ based on irreducible trinomials," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 55, no. 4, pp. 1031–1040, May 2008.

[21] É. Schost and A. Hariri, "Subquadratic polynomial multiplication over $\mathrm{GF}(2^m)$ using trinomial bases and chinese remaindering," in *Selected Areas in Cryptography*. Berlin, Germany: Springer-Verlag, 2008, pp. 361–372.

[22] P. Meher, "Systolic and non-systolic scalable modular designs of finite field multipliers for Reed-Solomon codec," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 17, no. 6, pp. 747–757, Jun. 2009.

[23] J. Xie, P. Meher, and J. He, "Low-complexity multiplier for $\mathrm{GF}(2^m)$ based on all-one polynomials," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 21, no. 1, pp. 168–173, Jan. 2013.

[24] J. Imana, "Low latency $\mathrm{GF}(2^m)$ polynomial basis multiplier," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 58, no. 5, pp. 935–946, May 2011.

[25] G. Orlando and C. Paar, "A super-serial galois fields multiplier for FPGAs and its application to public-key algorithms," in *Proc. IEEE Symp. Field-Programm. Custom Comput. Mach.*, 1999, pp. 232–239.

[26] L. H. Chen, P. L. Chang, C.-Y. Lee, and Y. K. Yang, "Scalable and systolic dual basis multiplier over $\mathrm{GF}(2^m)$," *Int. J. Innov. Comput., Inf. Control*, vol. 7, no. 3, pp. 1193–1208, Mar. 2011.

[27] M. Wang and I. Blake, "Bit serial multiplication in finite fields," *SIAM J. Discr. Math.*, vol. 3, no. 1, pp. 140–148, Feb. 1990.

[28] S. Fenn, M. Benaissa, and D. Taylor, "$\mathrm{GF}(2^m)$ multiplication and division over the dual basis," *IEEE Trans. Comput.*, vol. 45, no. 3, pp. 319–327, Mar. 1996.

[29] Y. Hua, J. Lin, C. Chiou, C.-Y. Lee, and Y. H. Liu, "A novel digit-serial dual basis systolic Karatsuba multiplier over $\mathrm{GF}(2^m)$," *J. Comput.*, vol. 23, no. 2, pp. 80–94, Jul. 2012.

[30] E. Berlekamp, "Bit-serial Reed-Solomon encoder," *IEEE Trans. Inf. Theory*, vol. IT-28, no. 6, pp. 869–874, Nov. 1982.

[31] R. Lidl and H. Niederreiter, *Introduction to Finite Fields and Their Applications*. New York, NY, USA: Cambridge Univ. Press, 1986.

[32] R. Furness, M. Benaissa, and S. Fenn, "$\mathrm{GF}(2^m)$ multiplication over triangular basis for design of Reed-Solomon codes," *Proc. Inst. Elect. Eng.—Comput. Digit. Tech.*, vol. 145, no. 6, pp. 437–443, Nov. 1998.

[33] *Nangate Standard Cell Library*. [Online]. Available: http://www.si2.org