World Academy of Science, Engineering and Technology
International Journal of Electronics and Communication Engineering
Vol:3, No:4, 2009

Efficient Large Numbers Karatsuba-Ofman Multiplier Designs for Embedded Systems

M.Machhout, M.Zeghid, W.El hadj youssef, B.Bouallegue, A.Baganne, and R.Tourki

***Abstract*—**Long number multiplications (n ≥ 128-bit) are a primitive in most cryptosystems. They can be performed better by using Karatsuba-Ofman technique. This algorithm is easy to parallelize on workstation network and on distributed memory, and it's known as the practical method of choice. Multiplying long numbers using Karatsuba-Ofman algorithm is fast but is highly recursive. In this paper, we propose different designs of implementing Karatsuba-Ofman multiplier. A mixture of sequential and combinational system design techniques involving pipelining is applied to our proposed designs. Multiplying large numbers can be adapted flexibly to time, area and power criteria. Computationally and occupation constrained in embedded systems such as: smart cards, mobile phones…, multiplication of finite field elements can be achieved more efficiently. The proposed designs are compared to other existing techniques. Mathematical models (Area (n), Delay (n)) of our proposed designs are also elaborated and evaluated on different FPGAs devices.

***Keywords*—**finite field, Karatsuba-Ofman, long numbers, multiplication, mathematical model, recursivity.

I. Introduction

FINITE field multiplication in GF (2^n) is one of the most important operations in cryptographic protocols (RSA, Diffie-Hellman key exchange, DSS, ECC …) [1, 8, 21]. That means the optimization on multiplication is critical for overall performance of cryptographic implementations. It is very costly in terms of Area and Delay performances. A lot of researches have been performed in designing performant multipliers (low Area occupation and high-speed computation). Several designs have been reported for multiplication on fields of characteristic two [16]. Efficient bit-parallel multipliers for both canonical and normal basis representation as well as hybrid multiplication have been proposed in literature [15]. All these algorithms exhibit a space complexity of O (n^2). However, there are some asymptotically faster methods for finite field multiplications, such as the Recursive Karatsuba-Ofman Algorithm [18]. Published in 1962, it was the first algorithm to accomplish polynomial multiplication in O ($n^{1,58}$) operations [20]. The Karatsuba-Ofman Algorithm (KOA) can successfully be applied to polynomial multiplication step. The fundamental Karatsuba-Ofman multiplication (KM) for polynomial in GF (2^n) is based on the idea of divide-and-conquer, since the operands are divided into two segments [3]. Compared to the well-known Schoolbook method, the KOA saves multiplications of the partial products at the cost of extra additions. Further work of hardware and software implementations of Karatsuba-Ofman multipliers in literature, was done to improve the KOA and to find bounds of the complexity. Several works detailed information on the usage of KOA in order to multiply with the least cost are provided [2, 22]. Multiplying long numbers (n ≥ 128-bit) using Karatsuba-Ofman algorithm is fast but the algorithm is highly recursive. Our work is related to Karatsuba-Ofman multiplier for large numbers. In this paper, we proposed and developed different Karatsuba-Ofman multiplier designs in GF (2^n), intended to perform the design of the cryptographic protocols in embedded system such as smart card and mobile phone. Our design constraints are: the latency, the energy consumption and the area occupation.

The remainder of this paper is organized as follows: The illustration of the efficiency of the multiplication arithmetic operator throughout the applications of cryptographic protocols for embedded systems is described in Section 2. Section 3 describes the Recursive Karatsuba-Ofman multiplication. In Section 4, we present new designs of adapted KOA, so that it can be implemented efficiently. Many approaches to Sequential and Parallel Recursive Karatsuba-Ofman Multipliers (RKM) to optimize the designs are developed in this section. Experimental results of FPGA implementation (area, latency, power) are presented in Section 5. Mathematical models (Area (n), Delay (n)) of our proposed Sequential/Parallel designs are also elaborated in Section 6. Section 7 concludes the paper.

II. Efficient Cryptographic Protocols for Embedded System

Embedded systems (such as : smart cards, mobile phones, Personal Digital Assistants, etc.), which ubiquitously are used

M. Machhout, W.El hadj youssef, B.Bouallegue, and R.Tourki are with the Electronics and Micro-Electronic Laboratory (LEME), University of Sciences, Monastir, Tunisia. (e-mail: machhout@yahoo.fr).
A.Baganne is with the Laboratoire des Sciences et Techniques de l'Information, de la Communication et de la Connaissance (Lab-STICC), University of South Brittany, Lorient, France. . (e-mail: adel.baganne@univ-ubs.fr).
corresponding author :
M.Zeghid is with the Laboratoire des Sciences et Techniques de l'Information, de la Communication et de la Connaissance (Lab-STICC) and Electronics and Micro-Electronic Laboratory (LEME), Monastir, Tunisia. (phone +216 73 501 785, e-mail: medien.zeghid@fsm.rnu.tn)
Manuscript received March 11, 2008.

to capture, store, manipulate, and access data of a sensitive nature, pose several unique and interesting security challenges [12]. Therefore, providing security and trust in embedded systems raises important technological challenges. The cryptographic primitives and security protocols are hard to realize in embedded systems.

Embedded systems are extremely resource constrained devices in terms of computing and communication capabilities, energy, power, chip and memory area, etc. Moreover, they generally have to work in harsh, uncontrolled, and even hostile surrounding conditions. The security is not only the addition of features, such as specific cryptographic algorithms and security protocols, to the system but also the elaboration of novel design principles, methods, algorithms, designs and techniques in order to efficiently and securely realize cryptographic primitives and security protocols for embedded systems [23], which are the building blocks for security, privacy, and trust. The implementation of cryptographic systems presents several requirements and challenges. First, the performance of the algorithms is often crucial. One needs encryption algorithms to run at the transmission rates of the communication links. Slow running cryptographic algorithms translate into consumer dissatisfaction and inconvenience. On the other hand, fast running encryption might mean high product costs since traditionally, higher speeds were achieved through custom hardware devices. Recently, ECCs are especially suited to smart cards because of the limited memory and computational power available on these devices. Another benefit of ECCs is that they can use a much shorter key length than other public key cryptosystems such as RSA to provide an equivalent level of security [5, 17]. For ECC systems, the most important operation is the scalar multiplication which is based on the finite field multiplication. For large integer, the efficiency of multiplication dominates the overall performance of ECC implementation [9, 10]. It was shown that as much as 85% of execution time is spent on multiplication for a typical point multiplication in ECC. (see figure 1).

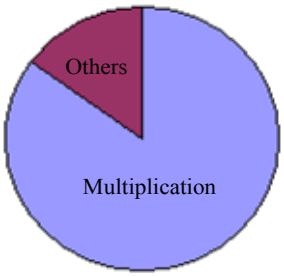


Fig. 1 Distribution of area occupation in ECC point multiplication arithmetic

Several techniques of multiplication were introduced in literature. The KOA is recommended for many cryptosystems. Therefore, the performance is primarily determinated by the efficient implementation of the arithmetic multiplication. In this context, new designs of Karatsuba-Ofman multiplication are necessary to achieve the following two goals: one goal is to design a high speed multiplier, the second goal is to reduce the area and the power consumption for large numbers (n ≥ 64-bit) on embedded processors and constrained platforms.

Hence, in order to improve time, area and power requirements of the ECC (on GF (2^n) with n ≥ 163-bit), it is essential to have a variety of Karatsuba-Ofman multiplication designs with consideration to trade-offs between the application and platform requirements.

III. Recursive Karatsuba-Ofman Multiplication (RKM)

In this section, we introduce the fundamental KOA which can successfully be applied to polynomial multiplication. The fundamental Karatsuba-Ofman multiplication for polynomial in GF (2^n) is a recursive divide-and-conquer technique. It is considered as one of the fastest way to multiply long numbers [11]. For polynomial multiplication with original Karatsuba method both operands have to be divided into two equal parts. If the length of operands is odd, they have to be padded with leading '0' [4]. Therefore, the KOA becomes recursive. A straightforward application of the KOA requires log2 (n) iteration steps for polynomials of degree (n-1). Let A = (a0, a1,…, an-1) and B = (b0, b1,…,bn-1), the binary representation of two long integers. The operands A and B can be decomposed into two equal-size parts A1A0 and B1B0 respectively, which represent the n/2 higher and lower order bits of A and B. We can split them into two parts as follows:

$$A(x)=\sum_{i=0}^{n-1}a_i x^i=\sum_{i=\frac{n}{2}}^{n-1}a_i x^i+\sum_{i=0}^{\frac{n}{2}-1}a_i x^i$$

$$=x^{\frac{n}{2}}\sum_{i=0}^{\frac{n}{2}-1}a_{i+\frac{n}{2}}x^i+\sum_{i=0}^{\frac{n}{2}-1}a_i x^i=x^{\frac{n}{2}}A_1+A_0 \quad (1)$$

$$B(x)=\sum_{i=0}^{n-1}b_i x^i=\sum_{i=\frac{n}{2}}^{n-1}b_i x^i+\sum_{i=0}^{\frac{n}{2}-1}b_i x^i$$

$$=x^{\frac{n}{2}}\sum_{i=0}^{\frac{n}{2}-1}b_{i+\frac{n}{2}}x^i+\sum_{i=0}^{\frac{n}{2}-1}b_i x^i=x^{\frac{n}{2}}B_1+B_0 \quad (2)$$

So the product P(x) = A(x)*B(x) can be computed as follows:

$$A(x)B(x)=A_0B_0+A_1B_1x^n+(A_1B_0+A_0B_1)x^{n/2} \quad (3)$$

Equation 3 needs four (n/2-bits) multiplications to compute the product P(x). To improve the computation of P(x), Equation 3 can be modified to Equation 5 as follows:

$$A_1B_0+A_0B_1=(A_1+A_0)(B_1+B_0)-A_1B_1-A_0B_0$$

$$A(x)B(x)=A_0B_0+A_1B_1x^n+(A_0B_0+A_1B_1+(A_1+A_0)(B_1+B_0))x^{n/2} \quad (5)$$

The result is:

$$A(x)B(x)=C_0+C_1x^n \quad (6)$$

Field multiplication can be performed into two steps. Firstly, we perform an ordinary polynomial multiplication of

two field elements as shown in Equation 5. Secondly, a reduction operation with an irreducible polynomial is needed to be performed in order to obtain the (n - 1) degree polynomial. It is noticed that once the irreducible polynomial $F(x) = x^n + x^k + 1$ has been selected, the reduction step can be accomplished by using XOR gates only [3]. However, we want to focus on an efficient method based on KOA to calculate the polynomial multiplication. Figure 2 shows the procedure of KOA where A and B are two n-bit length polynomials.

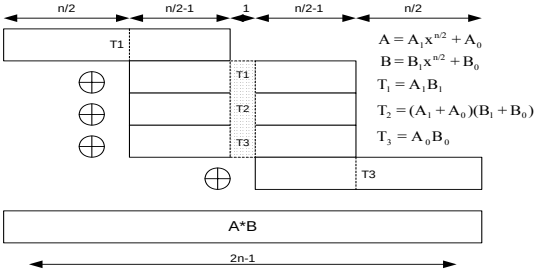
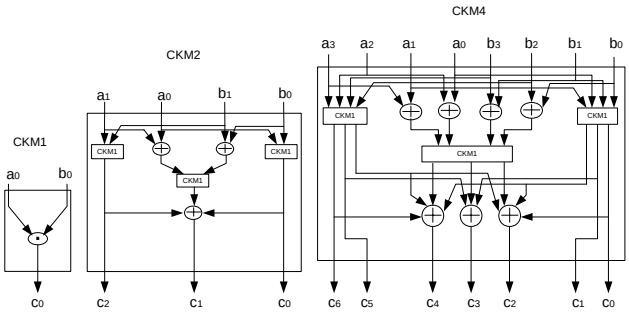


Fig. 2 Karatsuba-Ofman's multiplication

The multiplication over GF (2^n) is computed by a single AND operation. After completion of these polynomial multiplications, the final value of the lower half of C0 as well as the upper half of C1 are determined. The KOA can be applied recursively to the three polynomial multiplications. Hence, we can postpone the computations of the polynomial products A0B0, A1B1 and (A1+A0)(B1+B0); and instead we can split again each one of these three factors into three polynomial products. By applying recursively this strategy, each polynomial multiplication is transformed into three polynomial multiplications with their degrees reduced to about half of its previous value [7]. This leads immediately to a recursive construction process of Karatsuba-Ofman Multiplication (RKM), which builds Combinational Karatsuba Multiplier (CKMs) of width $n = 2^i$ for arbitrary $i \in N$ [13, 14]. (see figure 3).



(a) CKM1 : 1 bit (b) CKM2: 2 bits (c) CKM4 : 4bits

Fig. 3 Recursive Karatsuba-Ofman multiplication (RKM)

As can be concluded, parallel multipliers perform partial products in minimum clock cycle but require more area to be implemented. The area occupation metric becomes more and more important for $n \geq 64$-bit. Therefore, we required new RKM designs to adjust their performances: delays; and especially area occupation.

IV. Proposed RKM Designs

Multiplication can be implemented either in serially or parallel. In one side, serial multipliers require small area, less complex structure and high latency. In the other side, parallel multipliers perform the total operation in a minimum clock cycle but require more space to be implemented. When the operands are long ($n \geq 64$-bit), the performances of the algorithm especially the area occupation, is deteriorated as will be shown in section 5.1. This problem is not noticed for little numbers ($n < 64$-bit). Hence, to ameliorate the performances of RKM with $n \geq 64$-bit, we proposed new hybrid RKM designs that can be adapted to various performances as known: delay, area and power constraints for large finite field's elements.

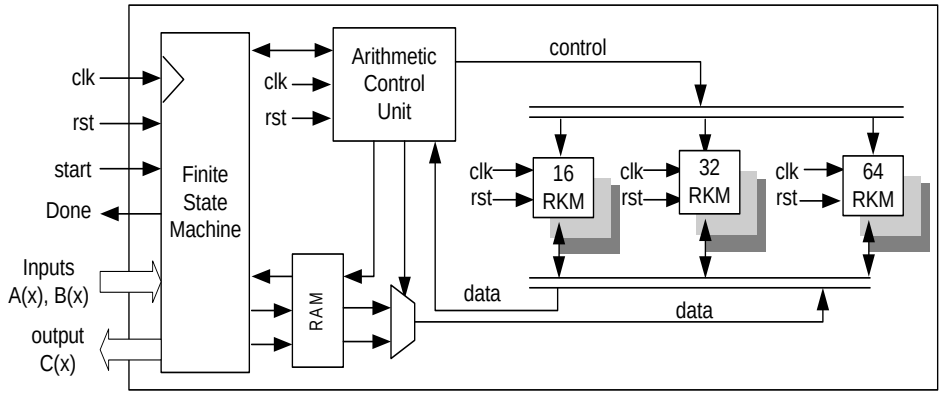


Fig. 4 Design of Recursive 128-bit Karatsuba-Ofman's multiplication (RKM)

A block diagram of 128-bit RKM's design is developed as shown in Figure 4.

The design includes the following units:

- Architecture Unit (AU): Constituted by 64-bit, 32-bit and 16-bit RKM multipliers. It takes care of reading operands from a port operand memory.
- Memory (RAM): is used to load initial operands.
- Arithmetic Control Unit (ACU): generates control

signal for all other units and the memory.

- Finite State Machine (FSM): determinates when enable, reset the multiplication process, accept input data and register output results in the memory. It decided when the result obtained is to be used or ignored.

In order to optimize this design, we have developed and tested five new 64-bit RKM designs as shown in Figure 5 and 6. All these designs are based on Parallel and Sequential applications of RKM. The major purpose of these approaches is to obtain a reduced area consumption and power of the hardware designs. The second purpose is to keep the high speed performance of RKM hardware implementations.

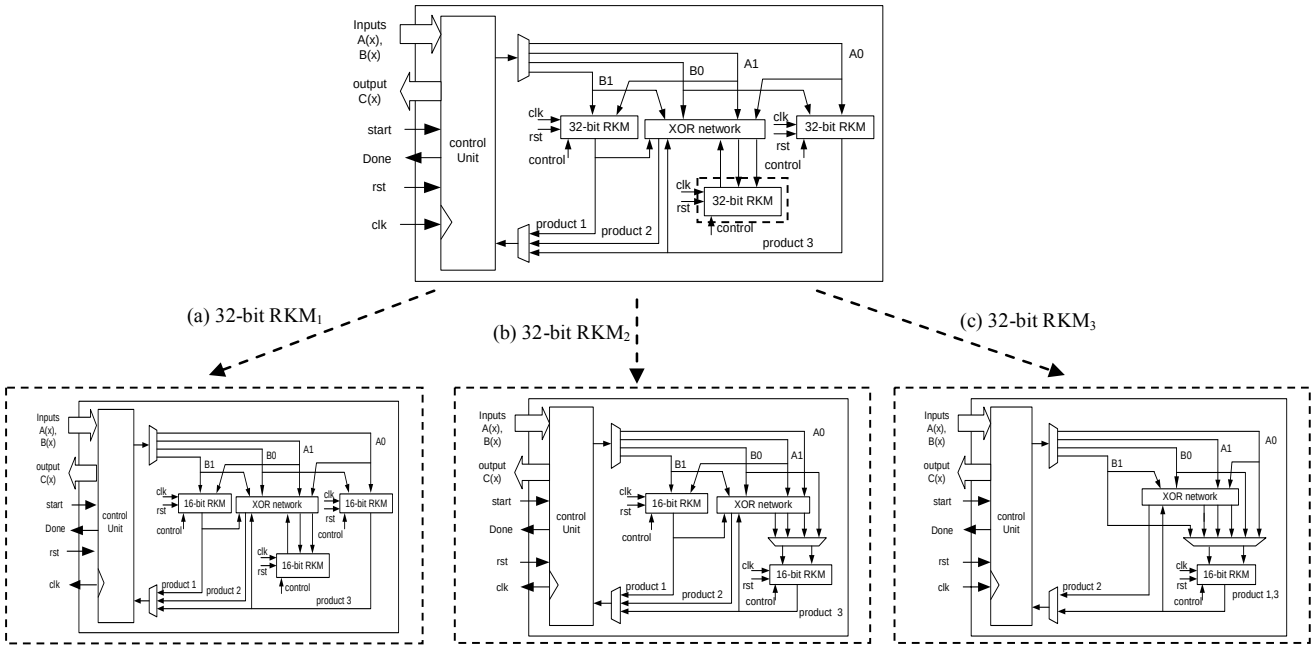


Fig. 5 A proposed 64-bit RKMj designs using Three parallel 32-bit RKM with (a) Three parallel 16-bit RKM; (b) Two parallel and one sequential 16-bit RKM; (c) Three sequential 16-bit RKM

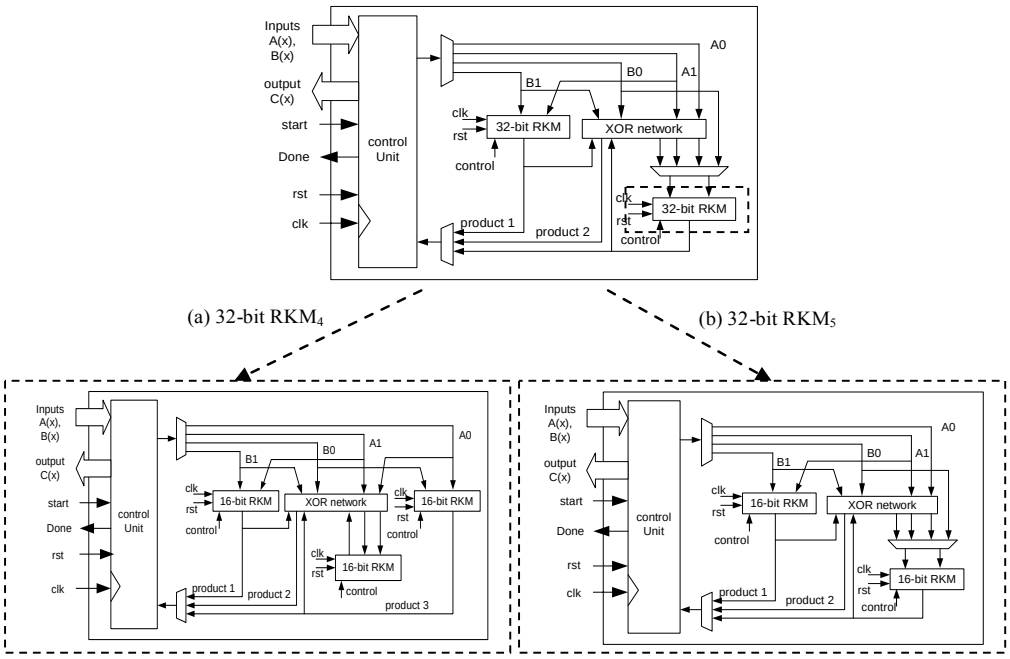


Fig. 6 A proposed 64-bit RKMj designs using Two parallel, one sequential 32-bit RKM with (a) Three parallel 16-bit RKM; (b) Two parallel and one sequential 16-bit RKM

The proposed Parallel and Sequential 64-bit RKMj progress cycles are shown in Table 1. For two proposed designs, multitasks sequences functions are detailed. Table 1 shows all steps describing the Architecture Units activation and deactivation. For example, to perform 64-bit RKM using RKM1, three 16-bit multipliers are performed in parallel; all

16-bit AUs are active "ON". Therefore, the three 32-bit RKMs required are consequently active "ON". However, when applying RKM2, the two 16-bit AUs are done in parallel (active: "ON"). Next sub-step, 16-bit AU is performed sequentially; the other unit is deactivated ("OFF") as shown in Table 2. Consequently, the three 32-bit RKMs are activated in parallel.

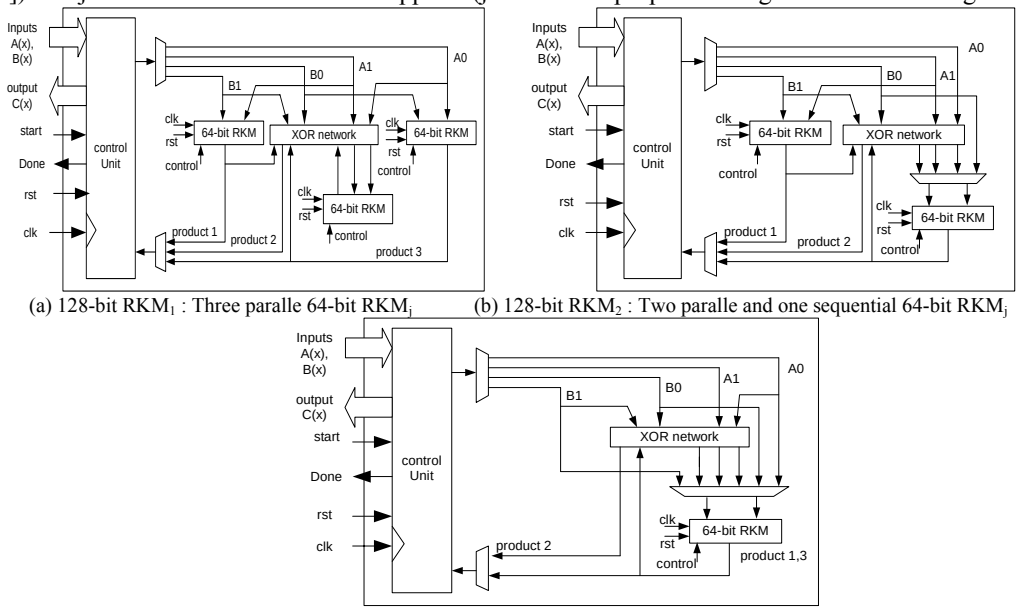
TABLE I
CYCLE PROGRESS OF 64-BIT RKM1 DESIGN

64-bit RKM1		32-bit RKM			32-bit RKM			32-bit RKM		
		16-bit RKM	16-bit RKM	16-bit RKM	16-bit RKM	16-bit RKM	16-bit RKM	16-bit RKM	16-bit RKM	16-bit RKM
Step1	1.1	ON	ON	ON	ON	ON	ON	ON	ON	ON
		ON			ON			ON		
Step2		Result Ready								

TABLE II
CYCLE PROGRESS OF 64-BIT RKM2 DESIGN

64-bit RKM2		32-bit RKM		32-bit RKM		32-bit RKM	
		16-bit RKM	16-bit RKM	16-bit RKM	16-bit RKM	16-bit RKM	16-bit RKM
Step 1	1.1	ON	ON	ON	ON	ON	ON
	1.2	ON	OFF	ON	OFF	ON	OFF
		ON		ON		ON	
Step 2		Result Ready					

In the following, we note Aij with i the number of 128-bit RKMs (i Є [1, 3]) and j the number of 64-bit RKM applied (j Є [1, 5]). So, we have fourteen 128-bit RKM designs. The detailed proposed designs are shown in Figure 7.



(a) 128-bit RKM_1 : Three paralle 64-bit RKM_j (b) 128-bit RKM_2 : Two paralle and one sequential 64-bit RKM_j

(c) 128-bit RKM_3 : Three sequential 64-bit RKM_j

Fig.7 Three proposed 128-bit RKM designs

Table 3 presents the cycle number of 128-bit RKM designs.

TABLE III
CYCLE REQUIREMENTS OF 128-BIT RKM'S DESIGN

Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Design	A_{11}	A_{21}	A_{31}	A_{12}	A_{32}	A_{14}	A_{24}	A_{34}	A_{15}	A_{25}	A_{35}	A_{13}	A_{23}	A_{33}
Cycle number	13	28	42	25	78	26	52	78	48	96	144	34	70	105

As can be seen, the design A11 needs 13 cycles to perform a full 128-bit multiplication. So, it's the fastest proposed design.

V. IMPLEMENTATION AND COMPARISONS RESULTS

In this section we present the results obtained from fourteen proposed 128-bit RKM's designs and we give some recommendations for RKM designs with focus on the cell-based design techniques. The different proposed designs are implemented in the Very High Speed Integrated Circuit Description Language -VHDL by using the Model Technology's ModelSim Simulator. The VHDL codes of the different design are synthesized placed and routed using three target devices: Xilinx Virtex II 2v2000ff896-6, Xilinx Virtex E XCV2000Efg860-6 and Spartan 3s2000fg900-5 FPGAs. The designs were simulated for verification of the correct functionality.

According to the required performances design, performance metrics such as area occupation (slices), speed (ns) and power consumption (mW) are used. As can be seen, the 32-bit RKM design as know A1 (three 16-bit RKM are

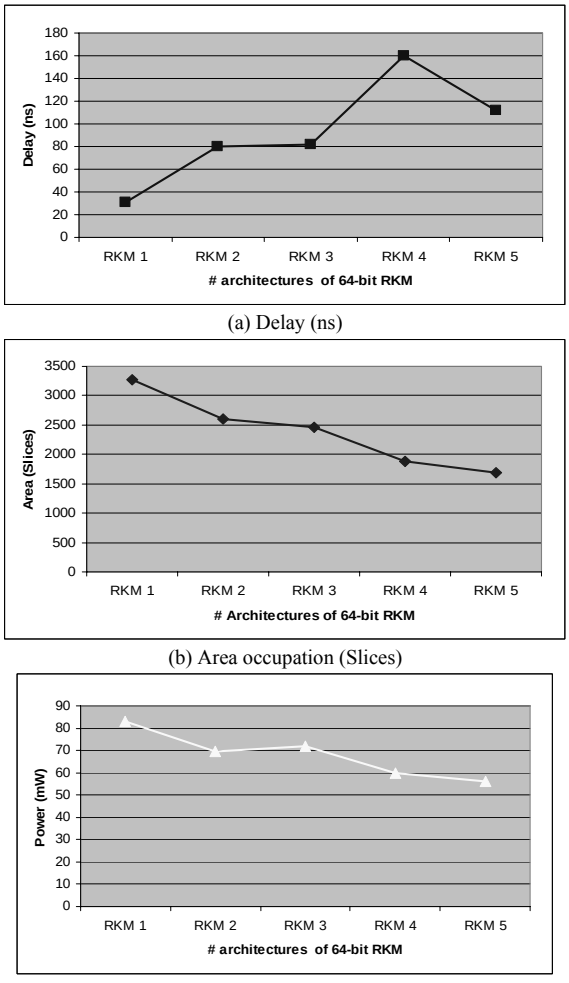
applied in parallel) is four time faster (154,163 ns) than the 32-bit RKM design as know A2 (three 16-bit RKM are applied sequentially) on two platforms. As concluded, when applying (A2), we obtain a profit of (26%) in speed with a little waste in area occupation by only (3%). Furthermore, the design (A1) takes (5 %) of the total number of CLB slices available in the same platform. Table 4 presents detailed results for 32 bit-RKM design. Still, the latency is the main metric of RKM algorithm when n ≤32-bit.

TABLE IV
DELAY AND AREA PERFORMANCES OF 32-BIT A1 AND A2 DESIGNS ON TWO PLATFORMS

Designs		A1 (Parallel)	A2 (Sequential)
Delay (ns)	Spartan 3	40,7	154,163
	Virtex 2	24,31	113,264
Area (Slices%)	Spartan 3	5 %	2 %
	Virtex 2	6 %	4 %

A. 64-bit RKMj Implementation Results

The implementation results of the five 64-bit RKMj designs developed in the previous section are shown in Figure 8.



(a) Delay (ns)

(b) Area occupation (Slices)

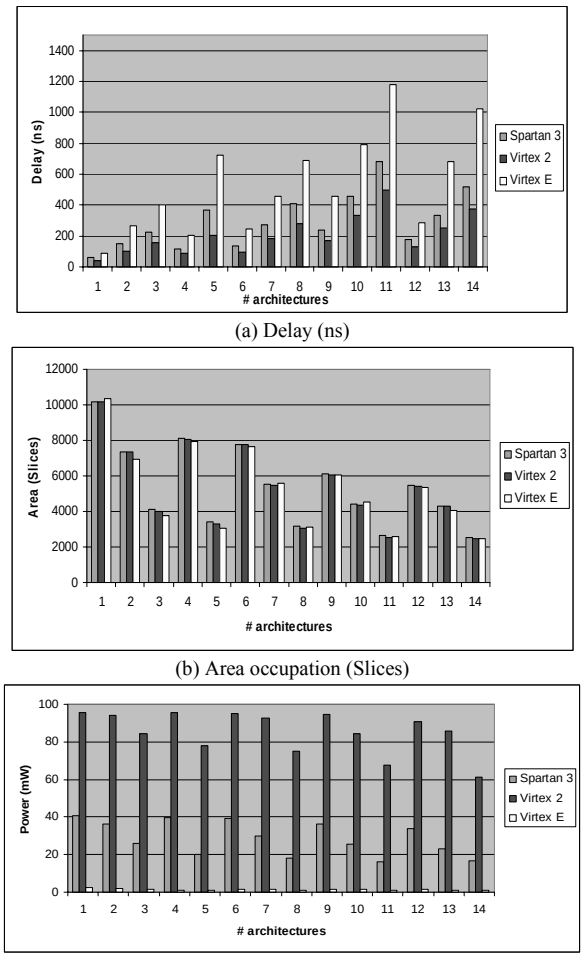
(c) Power consumption (mW)

Fig. 8 Sensitivity tests: (a) Delay (ns); (b) Area occupation (Slices); (c) Power consumption (mW) for five 64-bit RKMij's designs on Virtex 2

As can be noticed the 64-bit RKM1 design presents the best results in speed (30, 88 ns), however the area occupation is about 3263 slices and the power consumption 83,05mW. We can divide the five proposed 64-bit RKMj design in three groups: the high latency group (RKM1) and the low area occupation group (RKM5). The third group is composed by medium latency and medium occupation (RKMj with j Є [2, 4]). The 64-bit RKM3 design can be adopted as the optimal solution. The important point is to conserve the speed of the RKM, which is much more important than the waste in the area occupation. In the subsection below, we present the efficiency of the 128-bit RKM based on 64-bit RKMj designs.

B. 128-bit RKMij Implementation Results

Figure 9, shows the computation time (ns), area occupation (slices) and power consumption (mW) required by our proposed 128-bit RKMs designs.



(a) Delay (ns)

(b) Area occupation (Slices)

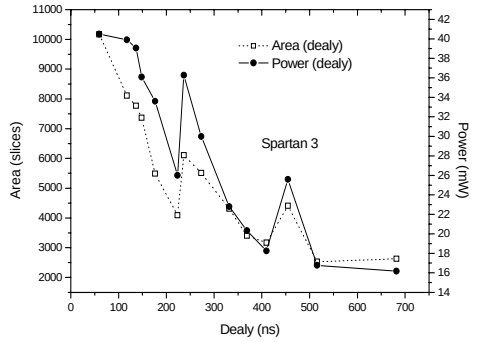
(c) Power consumption (mW)

Fig. 9 Sensitivity tests: (a) Delay (ns); (b) Area occupation (Slices); (c) Power consumption (mW) for fourteen 128-bit RKMij's designs on Spartan 3, Virtex 2 and Virtex E

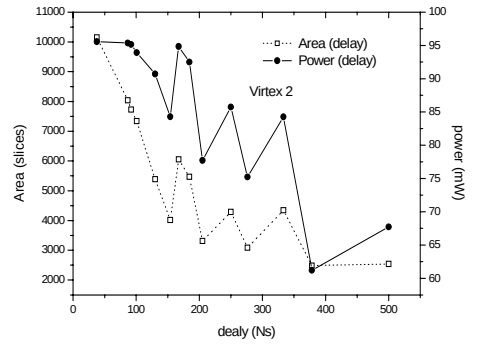
From figure 9.a, we remark that each design have a different computing time on three FPGAs devices. However, the design number 1 (A11) provides the least computing time to perform a 128-bit multiplication in all platforms. In figure

9.b, area performances are given in terms of total numbers of slices necessary for the implementation. The design (A11) has the worst area occupation on the three platforms. The design number 14 (A33) introduces the best area performances. As can be noticed, the proposed design number 14 (A33) requires the least power consumption on all targeted devices compared to the other proposed designs.

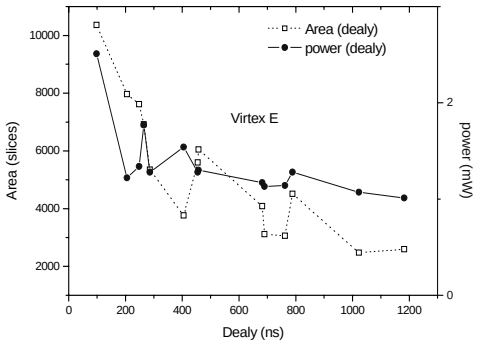
We conclude from figure 9.c, that independently of targeted FPGA devices, the 128-bit RKM's designs (A11) and (A33) can be adapted respectively to obtain a minimum delay and least area occupation. The different RKM's designs and circuits have to be adapted in order to take full advantage of the corresponding logic block resources of FPGAs. The choice of the adequate RKM design depends on the platform targeted and the goal aimed application.



(a)



(b)



(c)

Fig. 10 Delay, Area and Power criteria for fourteen 128-bit RKMij's designs on: (a) Spartan 3; (b) Virtex 2 (c); Virtex E

These figures can be adapted for user as a reference to choose the ideal RKM's design for a targeted FPGA platform. A study is also made on the adapted FPGA for all proposed RKM's designs in order to give to the designers the flexibility to make trade-offs between speed, area and power criteria. So to implement an RKM in a constrained platform, one need to consider trade-offs between these parameters and the user can get all advantages of his FPGA as depicted in Figure 10.

C. Comparisons results

Table 5 shows the performance results obtained by the Xilinx project synthesizer of our proposed RKM designs. A comparison performance result for the hardware implementations against each other is not straight forward. This is because, in literature large numbers (for n≥128-bit) karatsuba multiplication are not treated. Furthermore, different proposed multiplier designs performances (delay (ns) and area occupation (Slices)) deduced from this study are compared with our previous work on multipliers as know booth and hybrid multiplier with 128-bit length [6]. Furthermore, Table 5 compares our designs with other works recently reported in literature [19] in terms of 240-bit karatsuba multipliers, using the Virtex II reconfigurable hardware platform.

TABLE V
COMPARISON RESULTS OF MULTIPLIER DESIGNS ON VIRTEX-II OVER GF (2^{128}) AND GF (2^{240})

Ref.	Design	Delay (us)	Area (Slices)
Our previous work [6] GF(2^{128})	Booth	2,183	4%
	Hybrid multiplier	0,280	92%
[19] GF(2^{240})	Hybrid karatsuba	0,378	14%
	classical karatsuba	0,523	15%
This work GF(2^{128})	A31	0,153	37%
	A32	0,205	30%
	A23	0,250	39%
	A33	0,378	23%
This work GF(2^{240})	RKM	0.29	45%

In order to have a fair evaluation, we choose the (A31), (A32), (A23) and (A33) RKM designs because of optimal performances. We can see from Table 5 that the (A31) design is able to operate at a delay of 153, 93 ns, which is 14,267-times faster than the booth method. However, it occupied (33%) slices besides. Compared to hybrid multiplier, the (A23) design runs approximately at the same delay. In other side, it consumes (53%) slices less of the total platform area. The focus of [19] was to implement karatsuba in classical and hybrid karatsuba method for 240-bit length using the Virtex II as a target. We notice that our proposed RKM multiplier occupied more area occupation (+3332 and +3225 slices) compared to the hybrid and classical karatsuba methods developed in [19]. In the other side, our proposed RKM design is respectively 0.09us and 0.233us faster.

VI. RKM DESIGNS MODELING

The objective of this section is to present mathematical models (Area (n), Delay (n)) of the proposed 128-bit RKM designs. They enabled us to establish the basis of a mathematical formalism of the RKM designs.

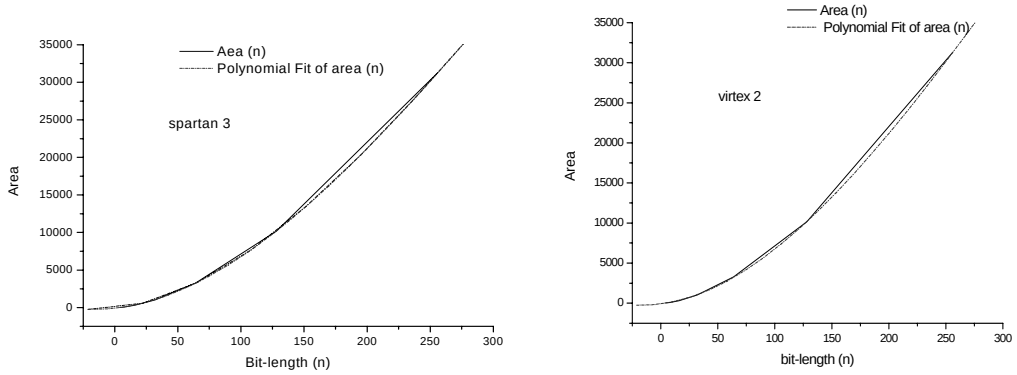
These models are available to designers before hardware and software embedded system implementation. So, a clear idea is available for FPGA designers about the delay and the area occupation of the RKM multiplier for different bit rang (n=64, 128,256...).

Mathematical performances models of all proposed designs are elaborated on two platforms. In the following, we presented mathematical modeling of two designs as know (A11) and (A33) ones.

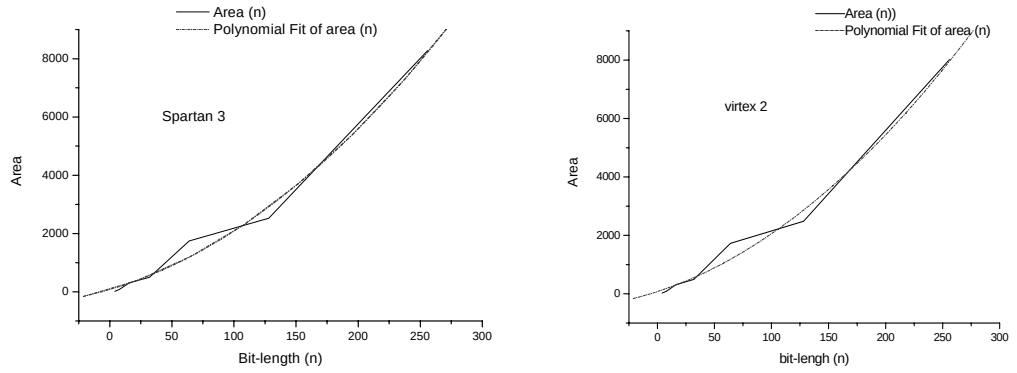
Figure 11 shows that the two Area variation's curves of RKM designs have approximately a polynomial pace. However, in this figure they have no ideal pace because of peaks. To perform a polynomial fit on the active data plot, a Polynomial Fit Analysis is needed. In this operation, a number of points are drawn in the fit curve. The fit equation model of the K^{th} order is stated:

$$Y = A + B_1 X + B_2 X^2 + B_3 X^3 + \ldots + B_K X^K \quad (7)$$

With: A and Bi are the parameters.



(a) A_{11} design.



(b) A_{33} design.

Fig. 11 Mathematical model (Area (n)) of RKM multiplier based on:(a) A_{11} design; (b) A_{33} design

TABLE VI
AREA (N) MODEL PARAMETERS OF RKM A11 AND A33 DSIGNS

Parameters		Mode Equation
For RKM-A11 architecture	Spartan 3	$A + B_1 n + B_2 n^2 + B_3 n^3$ $B_1 \in [16.84, 20.78]$, $B_2 \in [0.53, 0.57]$, $B_3 \in [-6.49\text{ E-}4, -5.29\text{ E-}4]$
	Virtex 2	$A + B_1 n + B_2 n^2 + B_3 n^3$ $B_1 \in [16.88, 20.83]$, $B_2 \in [0.53, 0.57]$, $B_3 \in [-6.39\text{ E-}4, -5.19\text{ E-}4]$
For RKM-A33 architecture	Spartan 3	$A + B_1 n + B_2 n^2$ $B_1 \in [6.43, 18.57]$, $B_2 \in [0.06, 0.09]$
	Virtex 2	$A + B_1 n + B_2 n^2$ $B_1 \in [6.79, 18.63]$, $B_2 \in [0.05, 0.09]$

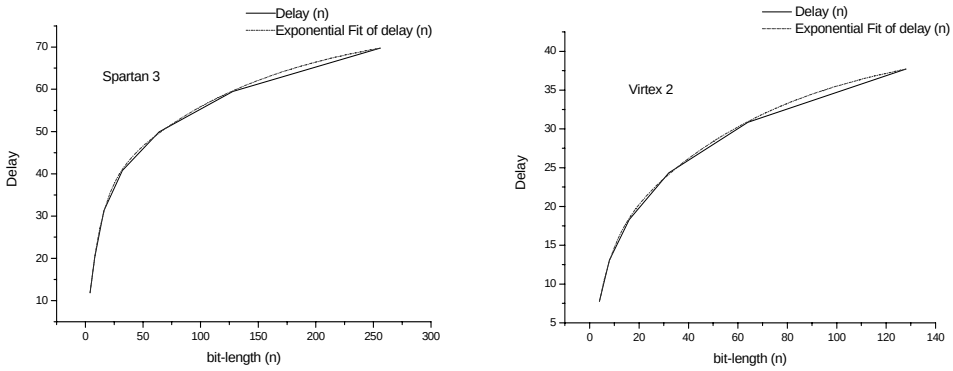
We conclude that the Area (n) occupation of RKM multiplier bases on (A11) and (A33) designs are presented in Section 3 has respectively a third and second polynomial order. The model's parameters of the Area (n) are illustrated in Table 6.

From Figure 12, we conclude that Delay (n) performance of RKM multiplier bases on (A11) and (A33) designs has respectively a second and first exponential order. The exponential fitting equation is as follows:

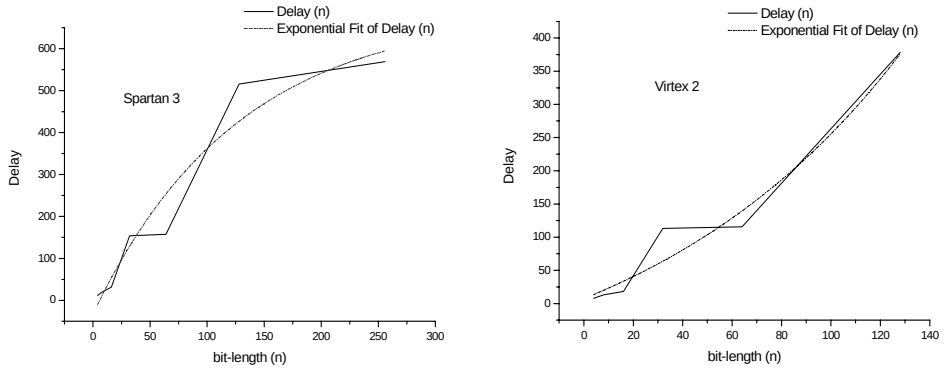
$$y = y_0 + A_1 e^{-x/t_1} + A_2 e^{-x/t_2} \quad (8)$$

With: y_0 the offset; A_1 and A_2 the amplitude; t_1 and t_2 decay constant

The model's parameters are illustrated in Table 7.



(a) A_{11} design



(a) A_{33} design

Fig. 12 Mathematical model (Delay (n)) of RKM multiplier based on: (a) A11 design; (b) A33 design

TABLE VII
DELAY (N) MODEL PARAMETERS OF RKM A11 AND A33 DESIGNS

Parameters		Mode Equation
RKM- A11 design	Spartan 3	$y0 + A_1 \exp(- n / t1) + A_2 \exp(- n / t2)$ $A_1 \in [-43.4, -41.4]$, $t1 \in [122.6, 150.6]$ $A_2 \in [-35, -32.8]$, $t2 \in [9.8, 11.4]$
	Virtex 2	$y0 + A_1 \exp(- n / t1) + A_2 \exp(- n / t2)$ $A_1 \in [-29.5, -27.7]$, $t1 \in [56, 77.6]$ $A_2 \in [18, 19]$, $t2 \in [0.53, 0.57]$
RKM- A33 design	Spartan 3	$y0 + A_1 \exp(- n / t1)$ $A_1 \in [-884, -564]$, $t1 \in [57.9, 196.5]$
	Virtex 2	$y0 + A_1 \exp(- n / t1)$ $A_1 \in [-64.4, 399.4]$, $t1 \in [-198.7, -21.3]$

VII. CONCLUSION

In this paper, we elaborated and presented RKM multiplier designs for large numbers ($n \geq 64$ bit). We explained that there are various ways to implement RKM that depends on the targeted platform and the goal aimed applications. For each method, we detailed the hardware design and the performance metrics (delay, area).

Furthermore, mathematical performance models, as know delay (n) and area (n) of all proposed RKM designs are elaborated and evaluated. A clear idea is available for FPGA's designer about area occupation and delay of RKM's multiplier for different bit range ($n \geq 64$ bit).This feature is advantageous to have suitable trade-offs between Area and Speed for implementing cryptographic schemes in embedded systems. The goal of our RKM design is crucial for achieving high performance and energy efficient cryptographic protocols. However, the effectiveness of based design depends on the adopted design approaches.

Future works will focus on improving and applying our RKM designs to prevent advanced attacks (Side Channel Attacks such as: SPA, DPA).

REFERENCES

[1] D.V.Bailey, C.Paar, "Efficient Arithmetic in Finite Field Extensions with Application in Elliptic Curve Cryptography," Journal of Cryptology, vol. 14, 2001, pp.153–176.

[2] N.S.Chang, C.H.Kim, Y.H .Park, J.Lim,"A non-redundant and efficient design for Karatsuba-Ofman algorithm," In ISC, Springer-Verlag Berlin Heidelberg, 2005,pp. 288-299.

[3] L.S.Cheng, A.Miri, T.H.Yeap, "Improved FPGA implementations of parallel Karatsuba multiplication over GF (2n)," In *Proc.* 23rd Biennial Symposium on Communications conf, 2006.
[4] L.Dong-Ho, and O.Jong-Soo, "Multi-Segment GF (2^m) Multiplication and its Application to Elliptic Curve Cryptography)," in Proc.GLSVLSI'07 conf. Stresa-Lago Maggiore, Italy, pp. 546-551.
[5] Z.Dyka, P. Langendoerfer,, "Area efficient hardware implementation of elliptic curve cryptography by iteratively applying karatsuba's method," in Proc. of the Design, Automation and Test in Europe conf, pp.70-75.
[6] W.El hadj youssef, M.Zghid, M.Machhout, B.Bouallegue, R.Tourki, "Design and Performance testing of Arithmetic Operators Library for Cryptographic Applications," International Journal of Computer Sciences and Engineering Systems (IJCSES), Vol.1, No.3, 2008,pp. 201-212.
[7] M.Ernst, M.Jung, F.Madlener, S. Huss, R.Blumel, "A Reconfigurable System on Chip Implementation for Elliptic Curve Cryptography over GF(2^n) ,"In CHES, LNCS 2523, 2002, pp. 381-399.
[8] F. U.S. Department of Commerce (2000) 'Digital Signature Standard (DSS)', NIST /FIPS PUB 186-2, January.
[9] F.Rodriguez-Henriquez, N.A.Saqib, A.Diaz-Pérez, "A fast parallel implementation of elliptic curve point multiplication over GF (2^m)", Microprocessors and Microsystems, Vol.28, pp.329-339.
[10] Z.Guitouni, W.El hadj youssef, M.Machhout, R.Tourki, "Implementation of Elliptic Curve Point Multiplication in Projective Systems over GF(2^m)", In Proc.Fourth International Multi-Conference on Systems, Signals & Devices (SSD'07), March.
[11] J.Großschadl, R.M.Avanzi, E.Sava, S. Tillich,"Energy-Efficient Software Implementation of Long Integer Modular Arithmetic," In CHES 2005, LNCS 3659, pp. 75–90.
[12] P.Kocher, L.Ruby, G.McGraw, A.Raghunathan, R.Srivaths, "Security as a New Dimension in Embedded System Design," In Proc. DAC 2004, June, San Diego, California, USA.
[13] L. A. B.Kowada, R.Portugal, C.M.H.Figueiredo,"Reversible Karatsuba's Algorithm," Journal of Universal Computer Science, Vol. 12, no. 5, pp.499-511.
[14] N.Nedjah, L.M.Mourelle, "Fast Less Recursive Hardware for Large Number Multiplication Using Karatsuba-Ofman's Algorithm," In Proc. ISCIS 2003, LNCS 2869, pp. 43-50.
[15] N.Nedjah, L.M.Mourelle,"A Review of Modular Multiplication Methods and Respective Hardware Implementations," Informatica, Vol. 30, pp.111-129.
[16] P.L.Montgomery, "Five, Six, and seven-Term Karatsuba-Like Formulae," IEEE Transactions on Computers, Vol. 54, No. 3, pp. 362-69.
[17] W.Qingxian, "The application of elliptic curves cryptography in embedded systems," In Proc. Second International Embedded Software and Systems Conf., pp.16-18.
[18] S.Peter, P.Langendorferv, "An Efficient Polynomial Multiplier in GF (2^m) and its Application to ECC Designs," In Proc. DATE0 Conf EDAA.
[19] J.Von zur Gathen, J.Shokrollahi, "Efficient FPGA-based Karatsuba multipliers for polynomials over F2," In Proc.SAC 2005 conf, LNCS 3897, pp. 359-369.
[20] J.Von zur Gathen, J.Shokrollahi, "Fast arithmetic for polynomials over F2 in hardware," In Proc. IEEE Information Theory Workshop, Punta del Este, Uruguay.
[21] LC.Washington.,"Elliptic Curves: Number Theory and Cryptography,"In Chapman & Hall New York CRC Press.
[22] Weimerskirch, A. and Paar, C. (2003) 'Generalizations of the Karatsuba Algorithm for Efficient Implementations', Ruhr-Universitat-Bochum, Germany, Tech. Rep.
[23] T.Wollinger, G.Guajardo, C.Paar ,"Cryptography in Embedded Systems: An Overview," In Proc. of the Embedded World 2003 Exhibition and Conference, Design & Elektronik, Nuernberg, Germany, February, pp. 735-744.

Mohsen. Machhout was born in Jerba, on January 31 1966. He received MS and PhD degrees in electrical engineering from University of Tunis II, Tunisia, in 1994 and 2000 respectively. Dr Machhout is currently Assistant Professor at University of Monastir, Tunisia. His research interests include implementation of standard cryptography algorithm, key stream generator and electronic signature on FPGA.

Medien. Zeghid received his M.S. degree in Electronic Materials and Dispositifs from the Science Faculty of Monastir, Tunisia, in 2005. Currently, he is a PhD student. His research interests include Security Networks, implementation of standard cryptography algorithm, Multimedia Application, Network on Chip: NoC. He is working in collaboration with LESTER Laboratory, Lorient, France.

Wajih. El Hadj Youssef was born in Monastir, on May 06 1979. He received MS in Informatique industrielle et automatique from National Instituted of Applied Sciences and Technology of Tunis, Tunisia, in 2005. Currently, he is a PhD student. His research interests include implementation of cryptography algorithm, key stream generator and electronic signature on FPGA, security of embedded system,

Belgacem. Bouallegue received his MSc in Physic Microelectronic and his DEA in Electronic Materials and Dispositifs from the Science Faculty of Monastir, Tunisia, in 1998 and 2000, respectively. Currently, he is a PhD student. His research interests include High Speed Networks, Multimedia Application, Network on Chip: NoC, flow and congestion control, interoperability and performance evaluation. He is working in collaboration with LESTER Laboratory, Lorient Cedex France.

Adel. Baganne born in 1968 is presently an Associate Professor at the UBS University and member of the LESTER Lab. He received his Ph.D. degree in Signal Processing and Telecommunications at the University of Rennes, France,in 1997 and the Engineer degree in Electronics from the National Superior Engineering School in Angers (ESEO), France, in 1993. His research interests include communication synthesis, codesign, co-simulation, computer architecture, VLSI design and CAD tools.

Rached. Tourki was born in Tunis, on May 13 1948. He received the B.S. degree in Physics (Electronics option) from Tunis University, in 1970; the M.S. and the Doctorat de 3eme cycle in Electronics from Institut d'Electronique d'Orsay, Paris south University in 1971 and 1973 respectively. From 1973 to 1974 he served as microelectronics engineer in Thomson CSF. He received the Doctorat d'etat in Physics from Nice University in 1979. Since this date he has been professor in Microelectronics and Microprocessors with the physics department, Faculty of Sciences of Monastir. His current research interests include: Digital signal processing and hardware software codesign for rapid prototyping in telecommunications.