AN EXPORTABLE HIGH SPEED ECC PROCESSOR USING RSD PERFORMED IN FPGA

[1]YERRA RAM CHARAN, [2]MANAS RANJAN BISWAL, [3]DR. NIHAR RANJAN PANDA

[1]M. Tech Student, ECE Department, Sankethika Vidya Parishad Engineering College, Visakhapatnam, A.P, India
[2]Assistant Professor, ECE Department, Sankethika Vidya Parishad Engineering College, Visakhapatnam, A.P, India
[3]Associate Professor, Head of the Department, ECE, Sankethika Vidya Parishad Engineering College, Visakhapatnam, A.P, India

***Abstract*: In this proposed paper, an exportable Application-Specific Instruction-set Processor (ASIC) elliptic curve cryptography processor based on redundant signed digit representation. The outcomes of Vertix-6 and Vertix-5 FPGA implementation are performed in this paper. The processor accomplish arthematic operations for NIST recommended curve P256.The design has an methodical modular adder to decrease the carry propagation problem, a high throughput modular divider which outcomes in maximum operating frequency and the processor employs pipelining techniques for Karatsuba–Ofman method to achieve high throughput multiplication.**

Index Terms: Application-Specific Instruction-set Processor (ASIP), Field Programming Gate Array (FPGA)

I. INTRODUCTION

A flexible Hardware processor that supported all the five NIST recommended prime field was reported in [2]. The design attempted to strike a balance between efficiency and flexibility. The processor supported all the five NIST recommended prime fields. The processor used non adjacent form (NAF) scalar multiplication to compute Q=k.P with Affine – Jacobian point operations and JSF scalar multiplication to compute Q= k.P with Chundnovsky – Jacobian point operations. The representation of k in NAF form made the computation of point multiplication possible with reduced number of point operations compared to Binary Scalar multiplication. The disadvantage of the processor was large area and the processor did not support non- NIST primes. The modular inverter used Binary Inversion Algorithm which occupied approximately the same number of slices as the modular multiplier. Data path of modular inverter was 521 bit wide so that additions and subtractions were performed in one pass. The design avoided hardware time multiplexing for computing inversion which sacrificed area saving for sake of efficiency. Regular multiplications were performed using eight 32 bit multipliers. Since fast reduction algorithms are available for specific NIST primes, the proposed design did not use Montgomery multiplication schemes. Both of the inverter and multiplier circuits consumed 90% of the total area.

II. PROPOSED SYSTEM

2.1 Redundant Signed Digits

The RSD representation, first developed by Avizienis, is a carry free arithmetic where integers are represented by the difference of other two integers. An integer X is represented by the difference of its x+ and x− components, where x+ and x- is the positive and negative component. The RSD representation has the edge of performing addition and subtraction without the required of the two's complement representation. Other than this, an overhead is raised due to the redundancy in the integer representation; since an integer in RSD representation requires double word length compared with typical two's complement representation. In radix-2 balanced RSD represented integers, digits of such integers are either 1, 0, or−1.

2.2 Karatsuba–Ofman Multiplication

The complexity of the regular multiplication using the schoolbook method is $O(n^2)$. Karatsuba and Ofman proposed a methodology to perform a multiplication with complexity ($n^{1.58}$) by dividing the operands of the multiplication into smaller and equal segments. Having two operands of length into be multiplied, the Karatsuba–Ofman methodology suggests to split the two operands into high-(H)and low-(L) segments as follows:

$$a_H = (a_{n-1},\ldots..a_{\lceil n/2\rceil}),\ a_L=(a_{\lceil n/2\rceil -1},\ldots,a_0)$$
$$b_H=(b_{n-1},\ldots..,b_{\lceil n/2\rceil}),b_L=(b_{\lceil n/2\rceil-1},\ldots..,b_0)$$

Consider β as the base for the operands, where β is 2 in case of integers and β is x in case of polynomials. Then, the multiplication of both operands are performed as follows: considering $a=a_L+a_H\beta^{\lceil n/2\rceil}$ and $b=b_L+b_H\beta^{\lceil n/2\rceil}$

$$C=AB= (a_L+a_H\beta^{\lceil n/2\rceil})(b_L+b_H\beta^{\lceil n/2\rceil})$$
$$=a_Lb_L+ (a_Lb_H+a_Hb_L)\beta^{\lceil n/2\rceil}+a_Hb_H\ \beta^n \qquad (1)$$

Hence, four half-sized multiplications are needed, where Karatsuba methodology reformulate (1) to

$$C=AB= (a_L+a_H\beta^{\lceil n/2\rceil}) (b_L+b_H\beta^{\lceil n/2\rceil})$$
$$=a_Lb_L+ ((a_L+a_H)(b_L+b_H)-a_Hb_H-a_Lb_L)\ \beta^{\lceil n/2\rceil})+ a_Hb_H\ \beta^n \qquad (2)$$

Therefore, only three half-sized multiplications are needed. The original Karatsuba algorithm is performed recursively, where the operands are segmented into smaller parts until a reasonable size is reached, and then regular multiplications of the smaller segments are performed recursively.

III OVERALL PROCESSOR ARCHITECTURE:

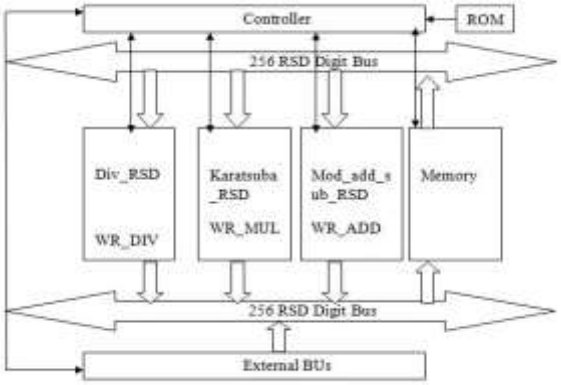

Fig: 1

The proposed P256 ECC processor consists of an AU of 256 RSD digit wide, memory, and two data buses Fig: 1 shows the overall processor architecture. The AU is the core unit of the processor that includes the following blocks: 1) modular addition/subtraction block; 2) modular multiplication block; and 3) modular division block.

3.1 Modular Addition and Subtraction:

Addition is used in the process of multiplication and in the binary GCD modular divider algorithm. In the proposed implementation, radix-2 RSD representation system as carry free representation is used. In RSD with radix-2, digits are represented by 0, 1, and−1, where digit 0 is coded with 00, digit 1 is coded with 10, and digit −1is coded with 01. In Fig. 2, an RSD adder is presented that is built from generalized full adders. The problem with this adder is that it tends to expand the addition result even if there is no overflow, since it restricts the least significant digit (LSD) to be digit−1 only. This unnecessary overflow affects the reduction process later and produces some control complexities in the overall processor architecture.

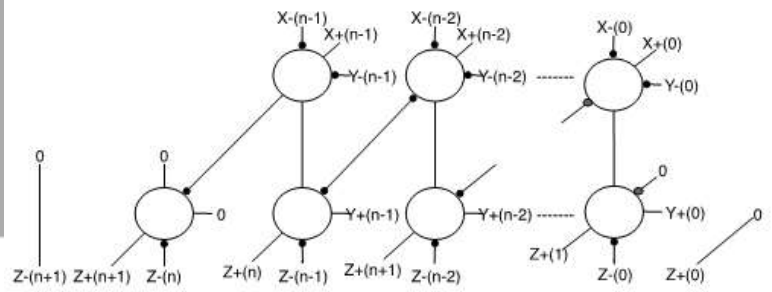

Fig: 2

In order to overcome the problem of overflow introduced in the adder proposed, a new adder is proposed based on the work proposed. The proposed adder consists of two layers, where layer 1 generates the carry and the interim sum, and layer 2 generates the sum, as shown in Fig. 3. Table I shows the addition rules that are performed by layer 1 of the RSD adder, where RSD digits 0,+1, and−1 are represented by Z, P, and N, respectively. It works by assuring that layer 2 does not generate overflow through the use of previous digits in layer 1. The proposed adder is used as the main block in the modular addition component to take advantage of the reduced overflow feature Fig. 4 shows the block diagram of the RSD modular addition block. The advantage of the proposed modular addition scheme is that only the MSD digits of the intermediate results are checked for the reduction process, as shown in Fig. 4. Our modular adder/subtracter consists of one full word RSD adder, two full word multiplexers, and one register with some control signals. One modular addition/subtraction is performed within one, two, or three clock cycles as per the value of the MSD that is retrieved after every addition. Whenever MSD becomes zero, the modular addition/subtraction module stops the operation and the valid out signal is activated. An n+1 RSD digit does not necessarily yield a value larger than the n-digit P256 modulo.

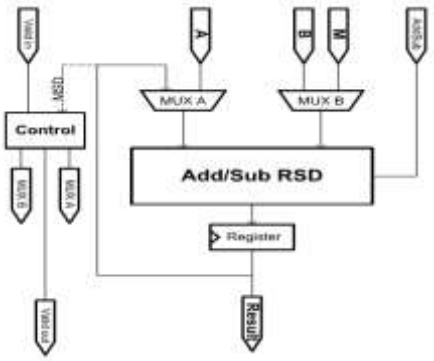

Fig: 3

3.2 Modular Multiplication:

Karatsuba's multiplier recursive nature is considered a major drawback when implemented in hardware. Hardware complexity increases exponentially with the size of the operands to be multiplied. To overcome this drawback, Karatsuba method is applied at two levels. A recursive Karatsuba block that works depth wise, and an iterative Karatsuba that works widthwise.

Recursive Construction of Karatsuba Multiplier:

In general, the reduced complexity of Karatsuba multiplication comes from the fact that four half-word multiplications are replaced by three half-word multiplications with some additions and subtractions. However, the complexity impact increases with the increase of the recursive depth of the multiplier. Hence, it is not sufficient to divide the operands into halves and apply the Karatsuba method at this level only. Operands of size n-RSD digits are divided into two (low and high) equal sized n/2-RSD digits branches. The low branches are multiplied through an n/2 Karatsuba multiplier and the high branches are multiplied through another n/2 Karatsuba multiplier.

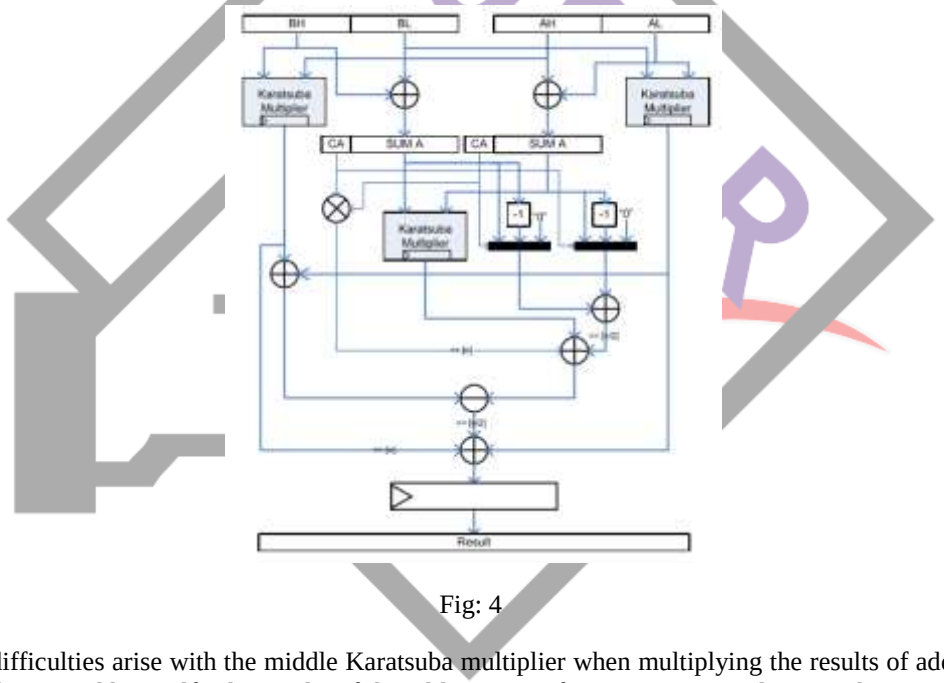

Fig: 4

Implementation difficulties arise with the middle Karatsuba multiplier when multiplying the results of addition of the low and high branches of each operand by itself. The results of the addition are of size n/2+1-RSD digits so that an unbalanced Karatsuba multiplier of size n/2+1 is required. Hence, the carry generated by the middle addition operation needs to be addressed to avoid implementation complexities of the unbalanced Karatsuba multiplier. As form the algorithm represents the recursive construction of the Karatsuba multiplication method at then-digits level. A recursive call of the Karatsuba multiplication module is performed three times for Klow, Khigh, and K1. These three multiplications are performed in parallel through three Karatsuba blocks. Each Karatsuba block performs recursively the same operations for n/2, then for n/4, and so forth. The recursive Karatsuba is constructed by recursive generation down to 4-digit of RSD schoolbook multiplier. The use of schoolbook multiplier at the lower level of the multiplier is due to the fact that Karatsuba method produces delays that cannot be compensated at small operand sizes.

3.3 High-Radix Modular Division:

Binary GCD algorithm is an efficient way of performing modular division since it is based on addition, subtraction, and shifting operations. The complexity of the division operation comes from the fact that the running time of the algorithm is inconsistent and is input dependent.. In the first state, the divider is checked whether it is even or odd. In the second state, the content of the corresponding registers are swapped according to the flag δ. In the last state, division by 4 modulo M is performed. First, division by 2 or by 4 is simply performed by shifting to right 1-digit/2-digits accordingly based on the guarantee that the LSDs are zeros in line 3 and 12 of the algorithm. On the other hand, division by 2 modulo M(division by 4 modulo M) is performed by adding or subtracting the dividend to or from the modulus according to whether the dividend is even or odd and the value of M(mod 4). For

both δ and ρ, a comparison with 0 is necessary. However, an efficient alternative is to initialize a vector of size n with all zeros except the least significant byte (LSB) for δ and the most significant byte (MSB) for ρ. Hence, the counting down of ρ is performed by shifting 1 bit to right and only the LSB is checked for the loop termination. On the other hand, a flag is needed to control the shift direction of δ, where the flag and the value of the LSB are used to determine whether it is less than zero or not. The implementation of the algorithm follows the implementation proposed. The modular divider architecture is shown in Fig. 6. Three RSD adders are used along with three 3×1 multiplexers and one 4×1 multiplexer with some control logic.

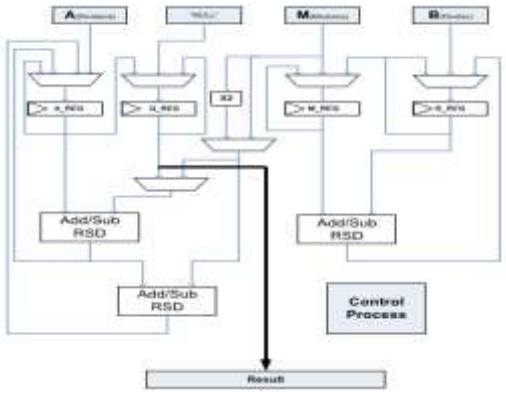

Fig: 5

IV. RESULTS AND DISCUSSION

4.1 Comparison results of Virtex5 and Virtex6

	Virtex5(65nm)	Virtex6(40nm)
Slices	2950	3366
Maximum Frequency(MHz)	38.287	51.868
Delay(ns)	26.118	19.280
Power(w)	1.281	1.12

Table 4.1: Descriptive Statics

4.2 Simulations Outputs:

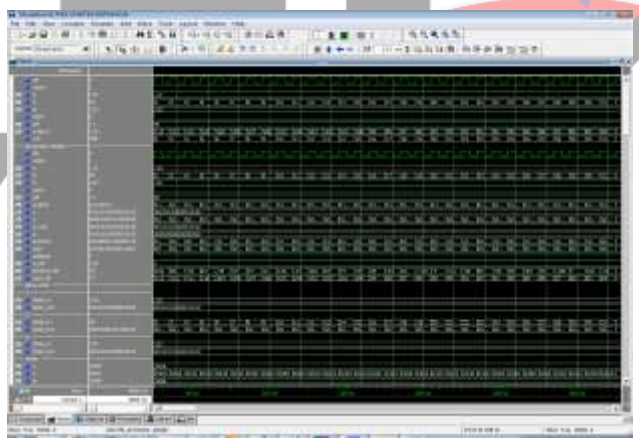

Fig 7: subtraction

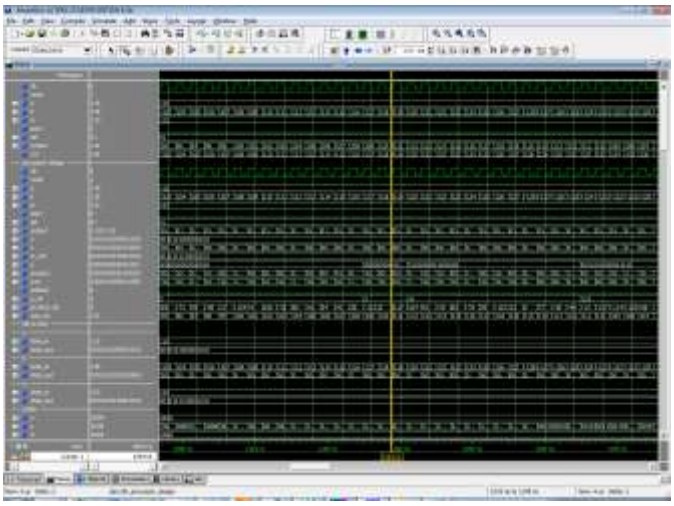

Fig8: addition

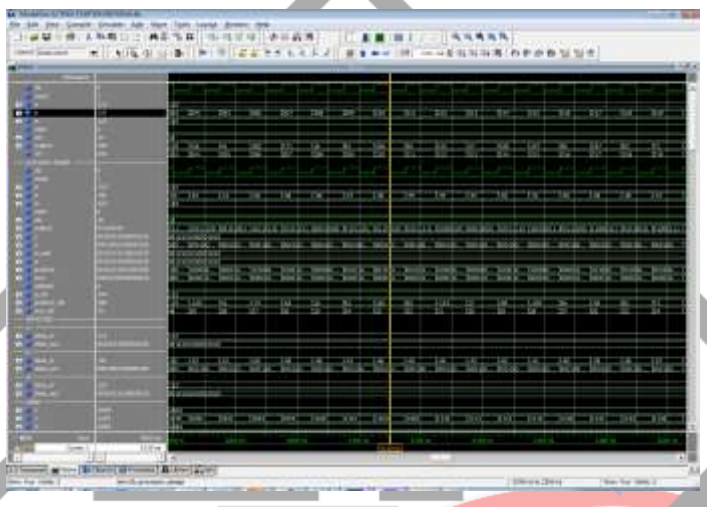

Fig 9: multiplication

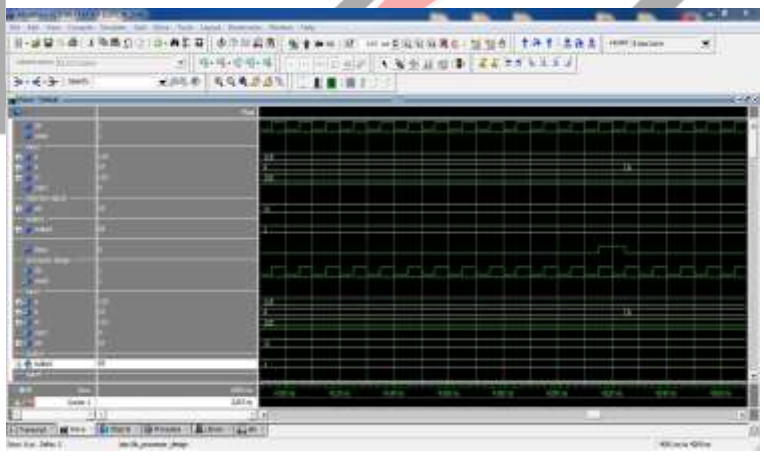

Fig 10: Division

4.3 RTL View

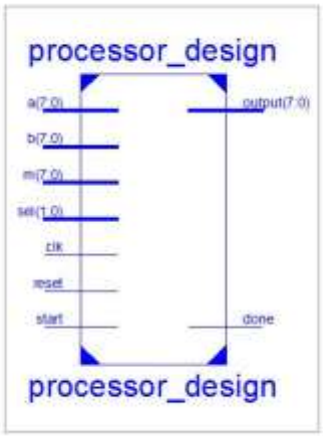

Fig: 11

V Conclusion

In this paper, prime field eight bit ECC processor implementation in FPGA devices like virtex5 and virtex6 has been presented. An RSD as a carry free representation is utilized which resulted in short data paths and increased maximum frequency. We introduced Karatsuba multiplier to achieve high throughput performance by a fully LUT-based FPGA implementation. An efficient binary GCD modular divider with three adders and shifting operations is introduced as well. Furthermore, an efficient modular addition/subtraction is introduced based on checking the LSD of the operands only. The implementation results of the proposed processor showed the shortest data path with a maximum frequency of 51.868 MHz, which is the fastest reported in the literature for ECC processors with fully LUT-based design. A single point multiplication is achieved by the processor within 2.26 ms, which is comparable with ECC processors that are based on embedded multipliers and DSP blocks within the FPGA. The main advantage of our processor is the exportability to other FPGA and ASIC technologies and ability to support different coordinate systems and point multiplication algorithms.

REFERENCES

[1]. G. Chen, G. Bai, and H. Chen, "A high-performance elliptic curve cryptographic processor for general curves over GF(p) based on a systolic arithmetic unit,"IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 54, no. 5, pp. 412–416, May 2007.

[2]. J.-W. Lee, Y.-L. Chen, C.-Y. Tseng, H.-C. Chang, and C.-Y. Lee, "A 521-bit dual-field elliptic curve cryptographic processor with power analysis resistance," inProc. ESSCIRC, Sep. 2010, pp. 206–209.

[3]. H. Marzouqi, M. Al-Qutayri, and K. Salah, "An FPGA implementation of NIST 256 prime field ECC processor," inProc. IEEE 20th Int. Conf. Electron., Circuits, Syst. (ICECS), Dec. 2013, pp. 493–496.

[4]. D. Hankerson, A. Menezes, and S. Vanstone, Guide to Elliptic Curve Cryptography, 1st ed. New York, NY, USA: Springer-Verlag, Jan. 2004.

[5]. F. Blake, G. Seroussi, and N. P. Smart, Advances in Elliptic Curve Cryptography(London Mathematical Society Lecture Note), 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, Apr. 2005.

[6]. Avizienis, "Signed-digit numbe representations for fast parallel arithmetic," IRE Trans. Electron. Comput., vol. EC-10, no. 3, pp. 389–400, Sep. 1961.

[7]. Karatsuba and Y. Ofman, "Multiplication of multidigit numbers on automata,"Soviet Phys. Doklady, vol. 7, p. 595, Jan. 1963.

[8]. Y.-L. Chen, J.-W. Lee, P.-C. Liu, H.-C. Chang, and C.-Y. Lee, "A dualfield elliptic curve cryptographic processor with a radix-4 unified division unit," inProc. IEEE Int. Symp. Circuits Syst. (ISCAS), May 2011, pp. 713–716.