International Journal of Electrical and Electronics Engineers
Vol. No.8 Issue 01, January-June 2016
www.arresearchpublication.com
IJEEE
ISSN 2321 - 2055

OPTIMIZED ARITHMETIC MODULES OF A RSD-BASED ECC PROCESSOR

P. Uma Maheswari[1], V. Vivitha[2], T. Siva Priya[3]

[1,2,3]*M.E VLSI Design,Theni Kammavar Sangam College of Technology, (India)*

ABSTRACT

The design strategy is focused fully on modular arithmetic modules rather than overall elliptic curve cryptography processor architecture. The processor has an efficient modular adder to reduce carry propagation problem, a high throughput modular divider which results in maximum operating frequency and modular multiplier in the processor is optimized based on throughput and modular reduction. The adder is focused for optimization as the addition is needed for accumulation process in multiplication and division.

Keywords- Application-specific instruction-set processor (ASIP), elliptic curve cryptography (ECC), field-programmable gate array (FPGA), Karatsuba–Ofman multiplication, redundant signed digit (RSD).

I. INTRODUCTION

Public Key encryption algorithms are widely used to ensure the data security of network communications. Elliptic curve Cryptography(ECC) is an asymmetric cryptographic system which provides higher security than the Rivest,Shamir and Adleman system(RSA) system.

The basic operation in ECC is scalar point multiplication which multiples a point on the curve by a scalar. A scalar point multiplication is performed by calculation of series point additions and point doublings. Points are added or doubled through series of additions, subtractions, multiplications and divisions of their respective co-ordinates using their geometrical properties. Point coordinates are the elements of finite fields closed under a prime or an irreducible polynomial. Various ECC Processors are proposed in the literature targets binary fields, prime fields or dual field operations.

In prime field ECC processors, carry free arithmetic is essential and results in short datapaths without carry propagation. Redundant schemes such as carry save arithmetic (CSA),redundant signed digits (RSDs) or residue number systems (RNSs) are used in various designs. Efficient addition datapath has to be built since it is a fundamental operation employed in other modular arithmetic operations. Addition is used in the accumulation process during the multiplication operation. Efficient modular addition/subtraction is introduced based on checking the MSD digits of the intermediate results for the reduction process.

Modular multiplication is an essential operation in ECC. Some ECC processors use the divide and conquer approach of Karatsuba multipliers for optimization of multiplication process where others use embedded multipliers and DSP blocks within FPGA fabrics.

The Overall processor architecture is of regular cross bar type and has 256 digit wide data buses. The processor is an application-specific instruction-set processor (ASIP) type to provide program ability and configurability.

Optimization techniques and design techniques are focused towards efficient individual modular arithmetic modules rather than the overall architecture. This architecture allows to replace the individual blocks easily if different algorithms or modular arithmetic techniques are desired. This paper proposes different efficient architectures for the individual modular arithmetic blocks and to improve the performance by modifying it.

In this paper, an RSD as a carry free representation is utilized which avoids lengthy data paths and increased maximum frequency. A modular addition and subtraction is proposed without comparison. A wide range of pipelining and optimization techniques are used to obtain a high throughput iterative Karatsuba multiplier.

II. BACKGROUND

2.1 Elliptic Curve Cryptography

Elliptic curves over a field K are defined by the reduced Weierstrass equation in (1) when the characteristic of the field is two or three. The set of solutions along with a point at infinity O defines the algebraic structure as a group with point addition as the basic operation:

$$E : y^2 = x^3 + ax + b. \quad (1)$$

The smoothness of the curve and distinct roots are guaranteed by $4a^3 + 27b^2 \neq 0.$ Points on the curve are defined by their affine coordinates (x, y). Point coordinates are of type integers for an elliptic curve defined by (1) and are the elements of an underlying finite field with operations performed modulo a prime number. Such elliptic curves are known as prime field elliptic curves.

2.2 Redundant Signed Digits

The RSD representation, first introduced by Avizienisis a carry free arithmetic where integers are represented by the difference of two other integers. An integer X is represented by the difference of its x+ and x− components, where x+ is the positive component and x− is the negative component. The nature of the RSD representation has the advantage of performing addition and subtraction without the need of the two's complement representation. On the other hand, an overhead is introduced 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.

2.3 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$O(n^{1.58})$ by dividing the operands of the multiplication into smaller and equal segments. Having two operands of length n to be multiplied, the Karatsuba–Ofman methodology suggests to split the two operands into high-(H) and low-(L)segments as follows:

$$_{aH}=(a_{n-1,\ldots\ldots\ldots}a_{[n/2]}),\ a_L=(a_{[n/2]}\text{-}1,\ldots\ldots\ldots a_0)$$

$$_{bH}=(b_{n-1,\ldots\ldots\ldots}b_{[n/2]}),\ b_L=(b_{[n/2]}\text{-}1,\ldots\ldots\ldots b_0)$$

Consider β as the base for the operands, where β is 2 incase of integers and β is x in case of polynomials. Then,the multiplication of both operands is performed as follows:

considering $a = aL + a_H\beta^{n/2}$ and $b = bL + b_H\beta^{n/2}$ then

$$C = AB = (a_L + a_H\beta^{[n/2]})(b_L + b_H\beta^{[n/2]}) = a_Lb_L + (a_Lb_H + a_Hb_L)\beta^{[n/2]} + a_Hb_H\,\beta^n \quad (2)$$

Hence, four half-sized multiplications are needed, whereKaratsuba methodology reformulate (2) to

$$C = AB = (a_L + a_H\beta^{[n/2]})(b_L + b_H\beta^{[n/2]}) = a_Lb_L + ((a_L+a_H)(b_L+b_H) - a_Hb_H - a_Lb_L)\beta^{[n/2]} + a_Hb_H \quad (3)$$

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

III. OVERALL PROCESSOR ARCHITECTURE

The proposed P256 ECC processor consists of AU of 256 RSD digits wide, a finite-state machine (FSM), memory, and two data buses. To support the P192 or P224 NIST recommended prime curves the processor can be configured in the pre synthesis phase. Fig.1 shows the overall processor architecture. Two sub control units are attached to the main control unit and has add-on blocks. These two sub control units work as FSMs for point addition and point doubling, respectively.

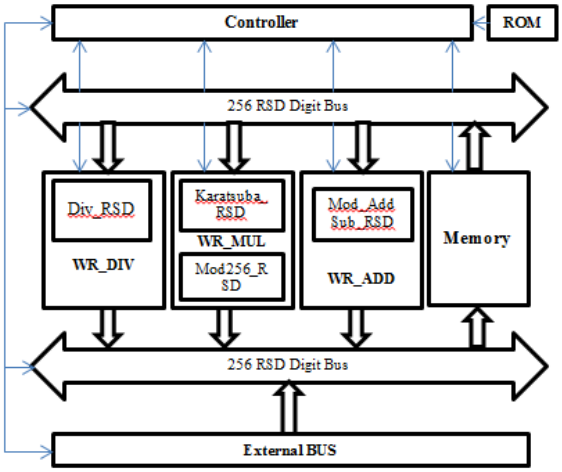


Fig.1 Overall processor architecture

Different coordinate systems are easily supported by adding corresponding sub control blocks that operate according to the formulas of the coordinate system. External data is passed through the external bus enters the processor and sent to the 256 RSD digits input bus. Data is sent in binary format to the processor and a binary to RSD converter stuffs zeros in between the binary bits in order to create the RSD representation. Hence, 256-bits binary represented integers are converted to 512-bits RSD represented integers. Subtracting the negative component from the positive component of the RSD digit converts RSD digits to binary format.

IV. ARITHMETIC UNIT

The AU is the core unit of the processor and includes the following blocks:1) Modular addition/subtractionblock;2) Modular multiplication block; and 3) Modular division block.

4.1 Modular Addition and Subtraction

Addition is used in the accumulation process during the multiplication, as well as, in the binary GCD modular divider algorithm. Radix-2 RSD representation system as carry free representation is used. Digits are

represented by 0, 1, and −1 in RSD with radix-2, where digit 0 is coded with 00, digit 1 is coded with 10, and digit −1 is coded with 01.

Table 1 shows the addition rules that are performed where RSD digits 0, +1, and −1 are represented by Z, P, and N, respectively. Reduced area is taken as an advantage in instantiating adders within the multiplier and the divider.

Table 1

Addition rules for the RSD Adder

a_ib_i	$a_{i-1}b_{i-1}$	Carry	Interim Sum
ZZ	--	Z	Z
ZP/PZ	Neither is N	P	N
ZP/PZ	Atleast one is N	Z	P
ZN/NZ	Neither is N	Z	N
ZN/NZ	Atleast one is N	N	P
PP	--	1	Z
NN	--	N	Z
PN/NP	--	Z	Z

The n-digits modular addition is performed by three levels of RSD addition. Level 1 implements the basic addition of the operands which produces n + 1 digits as a result. If the most significant digit (MSD) of level 1 output has a value of 1/−1, then level 2 adds/subtracts the modulo P256 from the level 1 output correspondingly. The result of level 2 RSD additions has n + 2 digits; however, only the n + 1th digit may have a value of 1/−1. This assertion is backed up by the fact that the operation of level 2 is a reversed operation with the modulo P256, and most importantly, the proposed adder assures that no unnecessary overflow is produced. If the n + 1th digit of level 2 results has a value 1 or −1, then level 3 is used to shrink the output to the n-digit range.

International Journal of Electrical and Electronics Engineers
Vol. No.8 Issue 01, January-June 2016
www.arresearchpublication.com
IJEEE
ISSN 2321 - 2055

Algorithm 1 RSD Modular Addition/Subtraction

Input: $A = a_{n-1} \ldots a_0$ and $B = b_{n-1} \ldots b_0$ and $M = m_{n-1} \ldots m_0$ and AddSub 1-bit

Output : $S = s_{n-1} \ldots s_0 = X + Y \bmod M$

1: **if** AddSub=0 then

2: $T_1 \leftarrow A+B$;

3: **else**

4: $T1 \leftarrow A-B$;

5: **end if**;

6: $T_2 =$ T1 $T1[n] = 0$; T1-M $T1[n] = 1$; T1+M $T1[n] = (-1)$

7: $T_3 =$ T2 $T2[n] = 0$; T2-M $T2[n] = 1$; T2+M $T2[n] = (-1)$

8: **return** $S \leftarrow T_3$

Algorithm1 shows the order of operations performed by the modular addition block. Notice that one modular addition is completed within one, two, or three clock cycles. Fig.3 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.3.

Our modular Adder/Subtractor consists of one full word RSD adder, two full wordmultiplexers, and one register with some control signals. As per the value of the MSD that is retrieved after every addition, one modular addition/ subtraction is performed within one, two, or three clock cycles. The modular addition/subtraction module stops the operation and the valid out signal is activated whenever MSD becomes zero. An n + 1 RSD digit does not necessarily yield a value larger than the n-digit P256 modulo. The subtraction is implemented by negating operand B and this can be performed by swapping the negative and positive components of the RSD representation of the operand.

International Journal of Electrical and Electronics Engineers
Vol. No.8 Issue 01, January-June 2016
www.arresearchpublication.com
IJEEE
ISSN 2321 - 2055

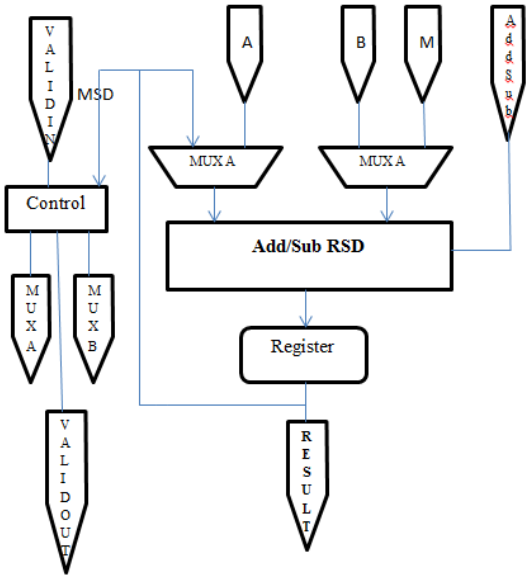


Fig.3 Modular addition subtraction block diagram

4.2 Modular Multiplication

There is a major drawback when Karatsuba's multiplier with recursive nature is implemented in hardware. When the size of the operands to be multiplied is increased hardware complexity increases exponentially.Karatsuba method is applied at two levels to overcome this drawback. A recursive Karatsuba block works depthwise, and an iterative Karatsuba works widthwise. The proposed method consists of two phases: 1) in phase 1, a regular recursive Karatsuba is built through recursive construction down to 1-digit level and 2) the recursive Karatsuba block is used to perform Karatsuba multiplications iteratively.Three recursive Karatsuba blocks are used to perform single widthwise Karatsuba iteration

4.2.1Recursive Construction of Karatsuba Multiplier

Four half-word multiplications are replaced by three half-word multiplications with some additions and subtractions and this reduces the complexity of Karatsuba multiplication. 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. There is an implementation difficulty with the middle Karatsuba multiplier when multiplyingthe 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. The carry generated by the middle addition operation needs to be addressed to avoid implementation complexities of the unbalanced Karatsuba multiplier.

4.2.2Optimization and Pipelining Techniques

The Critical Path Delay (CPD) of the processor is dominated by the Karatsuba multiplier data path. In the 32-digit Karatsuba multiplier data path consists of cascaded adders along with the recursive data path of the 16-

digit Karatsuba. The number of cascaded adders describes the data path of the multiplier. There are seven adders in the multiplier data path as shown in the following equations where A and B are RSD digits of size k:

$$A_{sum} = A_H 2^{k/2} + A_L \quad (4)$$

$B_{sum} = B_H 2^{k/2} + B_L$

$C = A_{sum}^{2k} * B_{sum}^{2k}$

$K_{3A} = \text{Karatsuba}_{16}(A_{sum}, B_{sum})$

$K_1 = \text{Karatsuba}_{16}(A_L, B_L)$

$K_2 = \text{Karatsuba}_{16}(A_H, B_H)$

$$K_{3B} = A'_{sum} + B'_{sum} \quad (5)$$

$$K_{3C} = K_{3A} + K_{3B} 2^{K/2} \quad (6)$$

$$K_3 = K_{3C} + C_{2k} \quad (7)$$

$M_1 = K_1 + K_2$

$M_2 = K_3 - M_1 \quad (8)$

$S' = K_1 + K_2 * 2^{2k} \quad (9)$

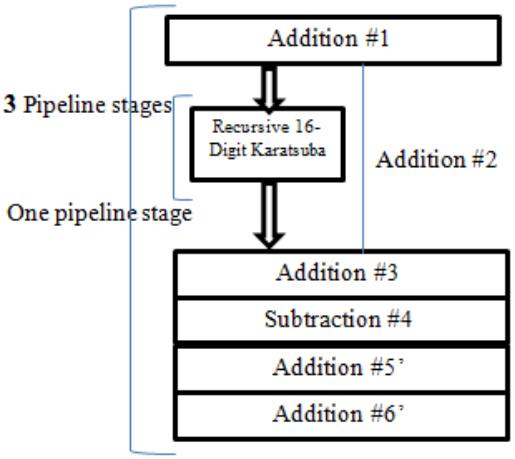
$S = S' + M_2 2^k \quad (10)$

Numbered equations represent the addition operations that contribute to the multiplier's datapath. Two approaches are followed to reduce path delays and increase maximum operating frequency of the system.First approach deals in reducing the number of additions within the CPD. Second, by careful placement of registers within the datapath a highly pipelined system is introduced.

By working around addition numbers in Equations (7),(9) and (10), the number of RSD additions within the datapath is reduced to six additions only. The addition of the carry C in (7) is delayed until the end of the process. Also, the addition in (9) is only a cascade operation when the extra digit at position 2k of operand K1 is removed. Hence, the carry C and the extra digit of K1 from addition (9) are cascaded for a final addition at once. Therefore, additions (7), (9), and (10) are replaced byadditions (11) and (12) as follows, where, operation number (8) become operation number (7) and || represents concatenation.

$$S' = (K_1 - K_1[2k] || K_2 2^{2k}) + M_2 2^k \quad (11)$$

$$S = S' + (C2^{2k} || K1[2k]2^k) \quad (12)$$



(A)Unbalanced data path

International Journal of Electrical and Electronics Engineers
Vol. No.8 Issue 01, January-June 2016
www.arresearchpublication.com
IJEEE
ISSN 2321 - 2055

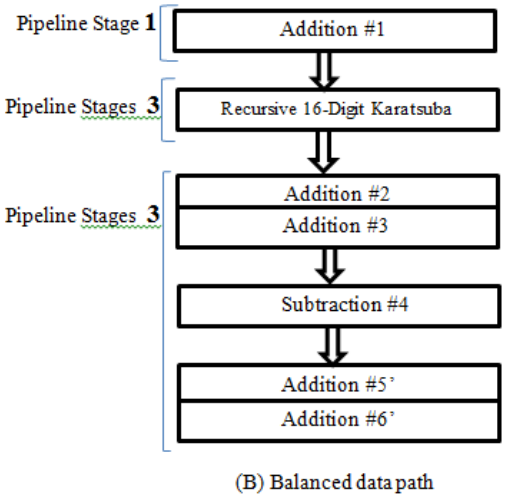


Fig.4 Karatsuba multiplier data path optimization. (A) Unbalanced data path. (B) Balanced data path

Fig.4 shows the datapath of the multiplier. Six adders represent the datapath along with the recursive datapath for the 16-digit Karatsuba multiplier blocks. The 16-digit recursive Karatsuba block is a three stages pipelined datapath, yields an unbalanced datapath, as shown in Fig. 4(A). Hence, an almost balanced datapath with three main stages is introduced, as shown in Fig. 4(B). Two stages consist of three substages datapath as the recursive 16-digit Karatsuba block, and the cascaded adders stage. One initial stage consists of one substage of the middle addition.

4.2.3Extended NIST Reduction

Generalized Mersenne primes are the special type prime numbers that allowfast modular reduction. Regular division is replaced by few additions and subtractions.Such primes are representedas p = f (t), where t is a power of 2. The modulus of the P256 curve is Merssene prime $p = 2^{256}-2^{224}+2^{192}+2^{96}-1$.Due to the redundancy nature of the RSD representation, the multiplication process may produce results that are representedby more than 512 digits and these results are still in the range$-p2 < A < p2$. These one or two extra digits are outside therange of the NIST reduction process. To include these extra digits in the reduction process new formulas are derived.The new reduction process has one extra 256-digit term, D5,along with some modification of the previously existed terms.

This term is added conditionally, whether the extra digit is setor not. Thus, two additions are the total overhead required tohandle the extra digits caused using the RSD representation.The modified reduction formula is $B = T +2S_1 +2S_2 + S_3 + S_4 - D_1 - D_2 - D_3 - D_4 - D_5 \bmod p$, where A_{16} representsthe extra digits produced by RSD Karatsuba multiplier.

$$T = (A_7||A_6||A_5||A_4||A_3||A_2||A_1||A_0)$$

$$S_1 = (A_{15}||A_{14}||A_{13}||A_{12}||A_{11}||0||\ 0\ ||\ 0\ )$$

$$S_2 = (2*A_{16}||A_{15}||A_{14}||A_{13}||A_{12}||0\ ||\ 0||A_{16})$$

$$S_3 =(A_{15}||A_{14}||\ 0||\ 0\ ||-2*A_{16}||A_{10}||A_9||A_8)$$

$$S_4 = (A_8||A_{13}||A_{15}||A_{14}||A_{13}||A_{11}||A_{10}||A_9)$$

$$D_1 =(A_{10}||A_8||\ 0\ ||\ 0||2*A_{16}||A_{13}||A_{12}||A_{11})$$

$$D_2 = (A_{11}||A_9||\ 0\ ||A_{16}||A_{15}||A_{14}||A_{13}||A_{12})$$

$$D_3=(A_{12}||2*16||A_{10}||A_9||A_8||A_{15}||A_{14}||A_{13})$$

$$D_4 = (A_{13}|| 0 ||A_{11}||A_{10}||A_9||A_{16}||A_{15}||A_{14})$$

$$D_5 = (-A_{16}|| 0|| 0|| 0 || 0 || 0 || 0 ||- A_{16}).$$

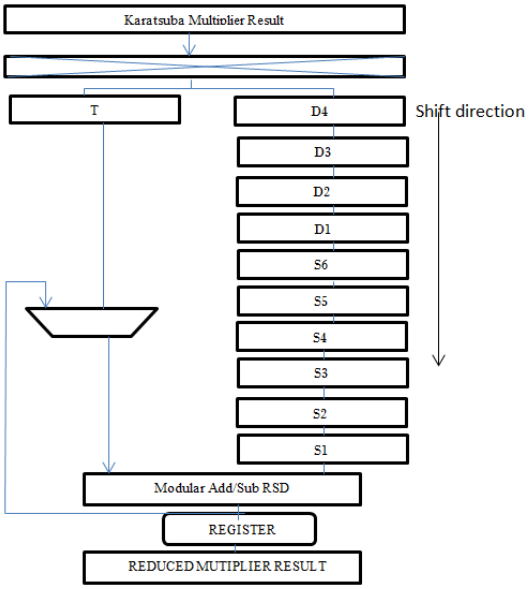


Fig.5 MOD P256 Reduction Block

NIST reduction is reformulated to accommodate the extra digit produced by the RSD Karatsuba multiplier. The resultant reduction scheme has three extra additions. However, through reformulation and combining the original terms with the additional terms, the reduction scheme is optimized. Accordingly, the modular multiplier is built with a Karatsuba multiplier, modular RSD adder, and some registers to hold the 256-digit terms. Fig. 5 shows the block diagram of the Mod P256 RSD multiplier. A controller is used to control the flow of the terms to the modular adder and at every turn, the result of the modular addition is accumulated and fed back to the adder. The cross-bar in Fig.5 shows the wiring of the 32-digit words to their respective locations within the extended NIST reduction registers.

V. IMPLEMENTATION RESULTS

The proposed system was implemented inXilinx Virtex6-XC6VLX75T device. Detailed implementationresults of individual blocks are listed in Table 2.

Such detailed results are useful in understanding the mainblock contributors to the overall hardware resources. Modular multiplier is the largest block withinthe design due to the three recursively built Karatsuba blocks,which operate in parallel. The CPDis shortened with the extensive pipelining techniquesthat are applied to the Karatsuba blocks.The achieved short critical path is due to the improvedpipelining strategies used in Karatsuba multiplier. The use of RSD representationis essential in reducing CPD of the processor. The power consumption is estimated for the proposedprocessor using XPower Analyzer tool in the Xilinx ISE12.1 suit. The power consumption of the proposed systemrunning on Virtex 6 is listed in Table 3.The proposed work is to improve the Add/Sub block performance in speed ,area and power by modifying the existing Add/Sub algorithm.

International Journal of Electrical and Electronics Engineers
Vol. No.8 Issue 01, January-June 2016
www.arresearchpublication.com
IJEEE
ISSN 2321 - 2055

RSD Modular Addition/Subtraction algorithm

Removed Steps

$$T_2 = \begin{cases} T1 & T1[n] = 0 \\ T1\text{-}M & T1[n] = 1 \\ T1+M & T1[n] = (-1) \end{cases}$$

Modified Steps

$$T_3 = \begin{cases} T1 & T1[n] = 0 \\ (T1\text{-}M)\text{-}M & T1[n] = 1 \\ (T2+M)+M & T1[n] = (-1) \end{cases}$$

Table 2

Implementation results in Virtex6 Device

Analyzed metrics	Modular Adder/Subtractor	Modular Multiplier
Slices	80	795
Resources (LUTs)	50	634
Flip-flops	48	342
Maximum Frequency(MHz)	732.869	270.592

Table 3

Comparison results in Virtex6 Device

Add/Sub algorithm	Slices	LUTs	Power utilized	Speed (MHz)
Existing	96	67	54.8%	724.900
Proposed	80	50	54.3%	732.869

VI. CONCLUSION

In this paper, the arithmetic modules of a NIST 256 prime field ECC processor is optimized. RSD as a carry free representation is utilized which resulted in short data paths and increased maximum frequency. Furthermore, an efficient modular addition/subtraction is introduced based on checking the MSD of the operands only. Reduced area is taken as an advantage in instantiating adders within the multiplier. Enhanced pipelining techniques are used within the Karatsuba multiplier to achieve high throughput performance by reducing the number of additions in the datapath of the multiplier. The implementation results of the proposed Adder/Subtractor improves the performance in speed and reduces area and power consumption.

REFERENCES

[1] W. Stallings, Cryptography and Network Security: Principles and Practice, 5th ed. Englewood Cliffs, NJ, USA: Prentice-Hall, Jan. 2010.

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

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

[4] Ç. K. Koç, Ed., Cryptographic Engineering, 1st ed. New York, NY,USA: Springer-Verlag, Dec. 2008.

[5] Y.-L. Chen, J.-W. Lee, P.-C. Liu, H.-C. Chang, and C.-Y. Lee, "A dualfieldelliptic curve cryptographic processor with a radix-4 unified divisionunit," in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), May 2011,pp. 713–716.

[6] G. Chen, G. Bai, and H. Chen, "A new systolic architecture for modulardivision," IEEE Trans. Comput., vol. 56, no. 2, pp. 282–286, Feb. 2007.

[7] Ç. K. Koç, Ed., Cryptographic Engineering, 1st ed. New York, NY,USA: Springer-Verlag, Dec. 2008.

[8] M. E. Kaihara and N. Takagi, "A hardware algorithm for modularmultiplication/division," IEEE Trans. Comput., vol. 54, no. 1, pp. 12–21,Jan. 2005.

[9] S. Yazaki and K. Abe, "VLSI design of Karatsuba integer multipliersand its evaluation," Electron. Commun. Jpn., vol. 92, no. 4, pp. 9–20,2009.

[10] N. Nedjah and L. de Macedo Mourelle, "A reconfigurable recursiveand efficient hardware for Karatsuba–Ofman's multiplication algorithm,"in Proc. IEEE Conf. Control Appl. (CCA), vol. 2. Jun. 2003,pp. 1076–1081.

[11] Y. Wang and R. Li, "A unified architecture for supporting operationsof AES and ECC," in Proc. 4th Int. Symp. Parallel Archit., AlgorithmsProgramm. (PAAP), Dec. 2011, pp. 185–189.

[12] S. Mane, L. Judge, and P. Schaumont, "An integratedprime-field ECDLPhardware accelerator with high-performance modular arithmetic units,"in Proc. Int. Conf. Reconfigurable Comput. FPGAs, Nov./Dec. 2011,pp. 198–203.

[13] M. Esmaeildoust, D. Schinianakis, H. Javashi, T. Stouraitis, and K. Navi,"Efficient RNS implementation of elliptic curve point multiplication overGF(p)," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 21, no. 8,pp. 1545–1549, Aug. 2012.

[14] D. M. Schinianakis, A. P. Fournaris, H. E. Michail, A. P. Kakarountas,and T. Stouraitis, "An RNS implementation of an Fp elliptic curve pointmultiplier," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 56, no. 6,pp. 1202–1213, Jun. 2009.

[15] J.-W. Lee, S.-C. Chung, H.-C. Chang, and C.-Y. Lee, "Efficient poweranalysis-resistant dual-field elliptic curve cryptographic processor usingheterogeneous dual-processing-element architecture," IEEE Trans. VeryLarge Scale Integr. (VLSI) Syst., vol. 22, no. 1, pp. 49–61, Feb. 2013.