A High Speed, Optimized Multiplier Architecture for a DF-ECC Processor

Nagaraja Shylashree
Asst. Prof at RNSIT, Research Scholar
Dept.of ECE,VTU, PESCE, Mandya
Karnataka, India
shylashashi@gmail.com

Venugopalachar Sridhar
Department of E & C Engineering.
VTU, PESCE, Mandya
Karnataka, India
venusridhar@yahoo.com

***Abstract* — This paper presents a High speed, optimized multiplier architecture for a dual-field (DF) processor for elliptic curve cryptography (ECC). This processor can support the required operations in both galois prime field GF(p) and binary field GF(2^m). The performance of the processor is enhanced by the judicious selection of proper type of coordinates in the arithmetic unit. The arithmetic unit is designed to provide dual-field multiplication and addition. The FPGA synthesis results show that the proposed dual-field ECC processor with Karatsuba method can reach a speed up to 124 MHz, consumes a power of 1091mW and occupies 3066 slices.**

Keywords - DF, DF- ECC, ECC, AES, GF, LFSR, M1, M2, FA, FPGA

I. INTRODUCTION

Cryptography is the study of techniques for secure transmission of data through wired or wireless networks [15]. Cryptography is classified in to two types: Symmetric cryptosystems and Asymmetric cryptosystems. In case of symmetric cryptosystems like DES, AES, etc., the sender and the receiver use a common (symmetric) key, whereas asymmetric cryptosystems like RSA, ELGAMAL,etc., use two different keys.

Elliptic curve cryptography (ECC) was proposed by Koblitz and Miller in 1985 [1],[2]. ECC is one of the public key cryptosystems. ECC based algorithms use smaller key sizes compared to other non ECC based algorithms for the same security level.

The dual field ECC processor design proposed here, supports both the prime GF(p) and binary extension field GF(2^m). It combines a bus optimized Montgomery multiplier and on the fly redundant binary converter. Also the proposed processor supports all the existing public and private key encryption standards like CRT, RSA and DSA [4]. The framework for a high-throughput, cost-effective, scalable and flexible architecture ECC processor is proposed. The processor uses a two-phase scheduling for optimization of ECC over GF(p) and GF(2^m) fields. The scalable architecture supports arbitrary elliptic curves and arbitrary finite fields [5]. Montgomery multiplier architecture is operable in both prime GF(p) and binary extension fields GF(2m) for arbitrary prime numbers and irreducible polynomials. A flexible and scalable architecture for large operator sizes has been described in [6], [7]. A processor with a high throughput with reduced power consumption has been designed [8]. High speed modular multiplication [9], [16] has been achieved where parallel implementation has been exploited using serial digit multipliers based on existing mathematical algorithms. In [17] the combinations of various algorithmic techniques and their outcome on speedup, throughput and area aspects have been discussed. Various ECC processor architectures with enhanced features have been proposed in [18]–[22]. A theoretical model to approximate the delay of elliptic curve scalar multiplier architecture (ECSMA) that performs point addition and doubling in a pipelined data path of the ECSMA for GF(2^{163}) is discussed [26]. A new high-speed point multiplier for elliptic curve cryptography on FPGA for different NIST recommended binary fields are proposed in [27] & [28].

The goal of this work is to develop a Dual field High speed processor using hybrid techniques to achieve the best area-speed tradeoff. Here two types of multipliers are implemented and the processor uses the faster one between them depending on the context.

II. BACKGROUND

In the Specifications for public key cryptography mentioned by IEEE 1363 standard [3] and recommended by NIST [11], the standard elliptic curve over GF(2^m) is given by, $y^2+xy = x^3+ax^2+b$, where x, y$\in$ GF(2^m) , and b$\neq$ 0. Similarly the standard elliptic curve over GF(p) is given by $y^2=x^3+ax^2+b$, where x, y $\in$ GF(p) and $4a^3+27b^2 \neq 0$ (mod p). Point Multiplication (PM) is the process of getting kP where P is a point on the given elliptic curve and k is a scalar. KP is obtained as, kP = P+P+,…,+P, i.e addition pf P(k-1) times. PM is the most time consuming operation which involves finite field inversion operations while using affine coordinates. In order to reduce the number of inversions, projective coordinates are used in this work. In this design, we use Lopez's dahab Mixed coordinates, i.e., one point is affine and the other point is projective for point addition and pure projective coordinates for point doubling over GF(2^m). For GF(p), Jacobian's Mixed coordinates for point addition and pure projective coordinates for point doubling are used [12].

A. Montgomery Ladder algorithm

We have adopted Montgomery Ladder algorithm, as it is suitable for our dual -field ECC processor. It is described in Algorithm1. Scalar multiplication factor '*k*' is generated using a pseudo random number generator in order to improve the security. Algorithm1 is compatible for both GF(p) and GF(2^m).

Algorithm 1: Montgomery algorithm

Input: k = (k_{n-1},k_{n-2}-------k_1,k_0), P
Output = [k] P;
$R_0 = 0$; $R_i = P$;
For i = n-1 down to 0
Do
b = k_i ; $R_{i-b} = R_{i-b} + R_b$;
$R_b = 2R_b$;
End for;
Return R_0

III. PROPOSED UNIFIED ARCHITECTURE

The Proposed unified architecture is shown in Fig.1. It consists of two major blocks, dual-field ECC processor block and application block respectively.

The dual-field ECC processor block supports the following functions: generation of random numbers using linear feedback shift register (LFSR).Point addition, point doubling, point scalar multiplication, Montgomery pre and post processing, modular exponentiation and finite-field arithmetic operations. The LFSR is used to generate different scalar multiplier '*k*' values which are stored in RAM locations. Based on the address value, the 'k' value is selected and this '*k*' is used to get *k*P. The dual-field adder and multiplier are used to perform arithmetic operations over the prime field and the binary field. The Montgomery unit controls the modular multiplication, which generates the appropriate results which are then given to the inversion unit to compute affine points. These affine points are fed to the application block.

A. ECC Arithmetic unit

ECC arithmetic unit consists of controller, EC data selector, dual-field multiplier, dual-field adder and Montgomery unit. NIST recommended x and y (affine coordinates) points for 233 bits and 256 bits for both binary and prime field are converted in to X ,Y and Z points (projective coordinates) by assuming Z=1 for both the fields. Along with X, Y, Z , the 'a' and 'b' elliptic curve parameters are the inputs to ECC arithmetic unit. Depending upon signal provided by the controller for EC arithmetic unit, EC data selector selects particular field and decomposes the into a sequence of minor operations using dual-field multiplier and dual-field adder. The EC scalar point multiplication by the Montgomery unit is done by successive point doubling and point addition depending upon the scaling factor '*k*'. To obtain point addition, Lopez Dahab mixed co-ordinates is used over GF(2^m) and Jacobian mixed coordinates is used over GF(p). Similarly, for point doubling, Lopez-Dahab pure projective coordinates is used over GF(2^m) [10] and Jacobian pure projective coordinates is used over GF(p).

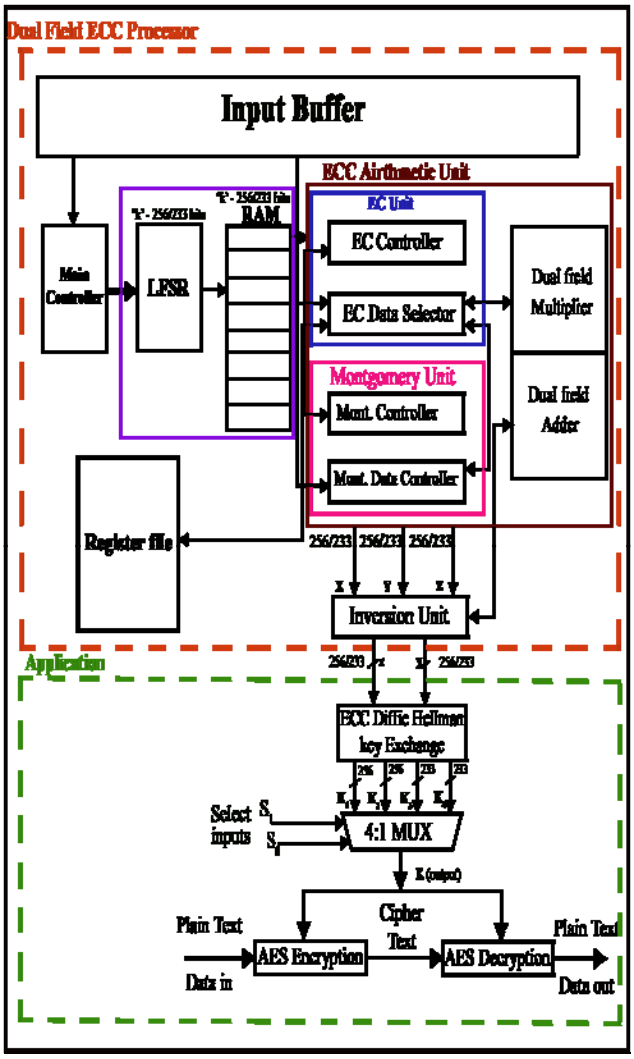

Fig. 1. Proposed Unified Architecture

1) Dual Field Multiplier: is implemented using two methods as follows.

a) Method 1(M1)- Hybrid Karatsuba multiplier :

The hybrid Karatsuba multiplier (combination of simple and general Karatsuba multiplier) divides a larger number into smaller numbers and the result is brought in to range by modulo reduction[12][7]. In this work Hybrid Karatsuba multiplier is used to calculate multiplication for 256/233 bits, the multiplier structure is shown in fig 2.

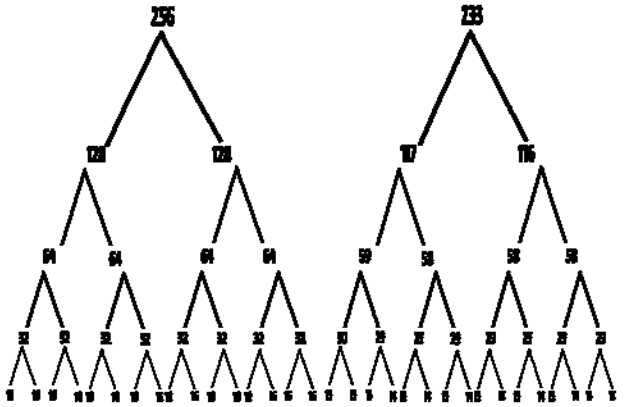

Fig. 2 Hybrid Karatsuba Multiplier for 233 bits and 256 bits.

b) Method 2(M2)-Proposed Shift and Add multiplier for GF(p):

The Black box view of Shift and Add multiplier is shown in Fig.3. Total Product length will be 512 bits. Initially, counter value is '0' and the ready signal is '1'.To initialize the process, Ready and Start signals are kept at logic '1'. After initialization, counter value is set to 256 bits and Ready signal becomes '0'. The input Multiplicand [M] register and Multiplier [Q] register have equal length that is 256 bits. The product will be stored in [A_Q].

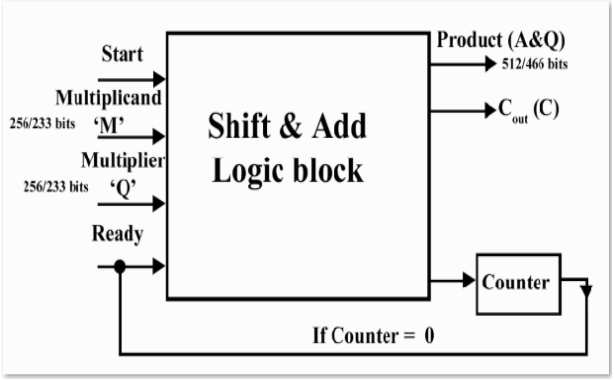

Fig.3.Black box view of Shift and Add Multiplier.

The Proposed functional Shift and Add multiplier is shown in Fig.4.The Multiplier register [Q] is stored in the lower half of the product and the multiplicand [M] is fixed and stored in the memory using a register bank. The adder length is fixed at 256-bits. Initially, product register [A] is loaded with all zero's of size(256 bits) which is equal to that of multiplier(Q).

First, we examine the LSB of the multiplier [Q]. If it is '1', we add multiplicand [M] register with product register [A]. The result is stored in 'C'(Carry out) and product register [A]. Now shift the product register [A_Q] data to the right by one bit using shift right operation. The product will be available in 'C'(Carry out) and [A_Q] register. Now the counter is decremented by '1'. Again check the LSB of the multiplier register [Q], and if it is '0', shift the [A_Q] register data to the right by one bit using shift right operation. Again the counter is decremented by '1' and the cycle continues until the counter value becomes '0' and then the ready signal becomes '1'.

The above technique is used for the binary field multiplication, except the carry propagation is ignored throughout.

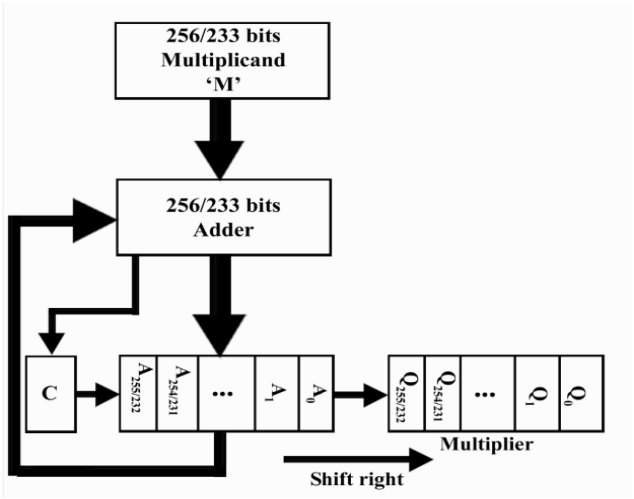

Fig.4.Logic Block of Shift and Add Multiplier for GF(p).

c) Proposed Dual field (DF) Adder:

Dual field adder is constructed using FAs (Full adders) with a control signal is shown in Fig.5. If the control signal is held at '1', the DF-adder acts as prime field adder else(control signal ='0') the DF-adder acts as binary field adder. For the prime field, Mod p of the adder output is to be taken.

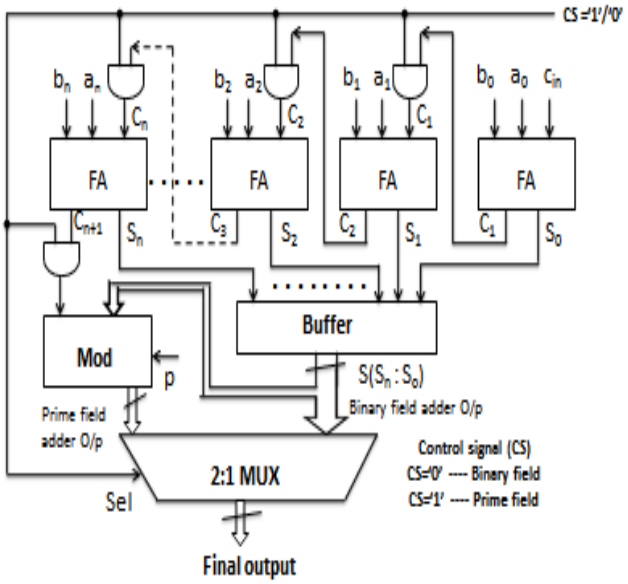

Fig.5.DF- Adder using FAs.

2) Montogomery Unit:

The Montgomery unit contains Montgomery controller and Montgomery Data selector. If the Montgomery controller signal is high, Montgomery Data Selector obtains the output value of the ECC data selector. Based on the curve over GF(p) or GF(2^m), the output value of the ECC data selector are either prime or binary field respectively. The output of Montgomery unit is given to the inversion unit to generate affine points. The two different points are used as the inputs to the Diffie Hellman key exchange, which will generate 2 symmetric keys per field. Keys generated from the prime field are represented as p1 and p2 and the corresponding binary field keys are represented as b3 and b4. These sets of 2 keys are fed to the 4:1 multiplexer. The output K of the multiplexer is one among the four keys based on the select (sel) inputs.

B. Proposed Inversion unit

Extended Euclidean Inverse algorithm (Algorithm 2) is compatible for both the fields. The ECC arithmetic unit provides the output in projective coordinates. The function of Inversion unit is to convert the output in the projective coordinates (X, Y, Z) to affine coordinates (x,y). For binary field (X/Z, Y/Z^2), Lopez Dahab coordinate system is used and for prime field (X/Z^2, Y/Z^3), Jacobian coordinate system is used.

In order to facilitate easy hardware implementation, the basic extended version of Euclidean algorithm is used. In order to remove integer division, the three GCD cases are considered shown by the first three 'if' clauses in the Algorithm 2. Based on the modulus to normalize the final result, the last two 'if statement's behave as correction steps. The evenness or oddness is verified very easily by examining the LSB in hardware.

Algorithm 2: Extended Euclidean Inverse algorithm

Input: a Є [1, p − 1] and p
Output: r Є [1, p − 1] and d,
where r = a^{-1} (mod p) and n ≤ d ≤ 2n
u := p, v := a, r := 0, s := 1, d := 0
while (v > 0)
if (u is even) then
if (r is even) then
u := u/2, r := r/2, d := d + 1
else
u := u/2, r := pr^{-2}, d := d + 1
else if (v is even) then
if (s is even) then
v := v/2, s := s^{-2}, d := d + 1
else
v := v/2, s := rp^{-2}, d := d + 1
else
x := u − v
if (x > 0) then
u := x, r := r − s
if (r = 0) then
r := r + p
else
v := −x, s := s − r
if (s =0) then
s := s + p
if (r > p) then
r := r − p
if (r = 0) then
r := r + p
return r and d

IV. IMPLEMENTATION AND COMPARISONS

We have proposed ECC parallel architecture, of 233 bit and 256 bit DF- ECC processor on an XC5vlx110t-1ff1136 FPGA verification platform to evaluate area, speed and power throughputs. Total on-chip Power is calculated by using Xilinx X-Power analyzer, which consists of dynamic power and quiescent power.

A. Implementation of ECC processors

We have described the ECC parallel architecture of 233 /256 bit DF-ECC processor on an XC5vlx110t-1ff1136 FPGA verification platform to evaluate area, speed and power throughputs. The synthesis results are shown in Table I. The experimental results indicate that the ECC processor using hybrid Karatsuba multiplier/XOR addition (Method 1 or M1) can run at **126.20MHz** over GF(2^m) and **45.426MHz** over GF(p), the same ECC processor using Shift and Add multiplier (Method 2 or M2) can run at **121.206 MHz** over GF(2^m) and **44.356 MHz** over GF(p).

From these results we can conclude that, dual-field data path has shortest critical path for the binary field and longer critical path for the prime field.

The simulation result is shown in Fig. 6. From the simulation diagram, signals pbarb, clk, wr_en, lrst, addr are the inputs and prime1, prime2, binary1, binary2 are the output signals. Here, prime1 and prime2 indicate prime field values. Similarly binary1 and binary2 indicate the binary field values. Based on the field chosen by the signal "pbarb", processor will generate the affine points. When "pbarb" is '0', it indicates prime field and when "pbarb" is '1', it indicates binary field. The Table I shows that our dual -field ECC processor using multiplier M1, occupies 3,066 slices. It consumes 1091mW of power and it can run with a maximum frequency of **124.347MHZ**. Also using M1, DF-ECC processor uses slice LUT's as 3,816, number of BRAM is 28 and the total memory is 990KB, and the same DF-ECC processor using M2 multiplier, occupies 3,021 slices. It consumes 1087mW of power and it can run with a maximum frequency of **119.308MHZ**. Also using M2, DF-ECC processor uses slice LUT's as 3,815, number of BRAM is 28 and the total memory is 990KB.

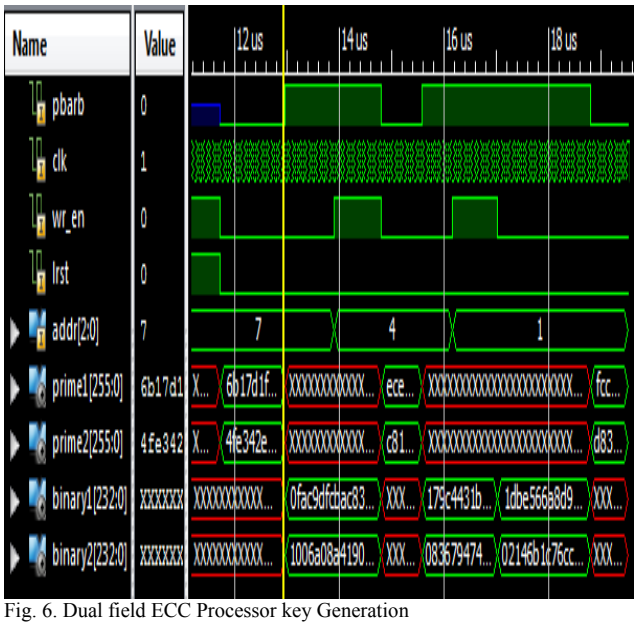

Fig. 6. Dual field ECC Processor key Generation

TABLE I. COMPARISION AMONG THE EXSISTING ECC PROCESSOR DESIGNS USING METHOD 1& 2 MULTIPLIER.

	Platform	*ECC processor field size*	*f_{MAX} in MHZ*	*Area [Slices]+ BRAM's*	*Dual Field*	*Power (W)*
Ours (M1)	**Xilinx Virtex -5**	**233/256**	**124.34**	**3,066 + 28**	**Yes**	**1.091W**
Ours (M2)	**Xilinx Virtex -5**	**233/256**	**119.308**	**3,021 + 28**	**Yes**	**1.087W**
[24]	Xilinx Virtex-2 pro	160/160	100	8,594 +6	Yes	--------
[5]	Xilinx Virtex-2 pro	160/256	94.7	39,531 41,595	Yes	---------
Ours (M1)	**Xilinx XC4V fx100**	**233/256**	**166.311**	**5,988**	**Yes**	---------
[25]	Xilinx XC4Vfx100	163/192	150.5	5,227 CLB's	Yes	---------

In Table I two methods have been incorporated. Comparing the method 1 and method 2, Method 1 multiplier design is more efficient in terms of Speed, where as method 2 multiplier is marginally better in terms of area and power.

As an application, this DF-ECC processor generates two affine points which are used to generate two symmetric keys using Diffie Hellman key exchange algorithm. These keys are used for AES encryption and decryption.

Rijndael proposed AES algorithm in 2001 [14]. Depending upon the key size, it increases the number of rounds as 10, 12 and 14. In this work, the key size is taken as 256 bits and number of rounds used is 14. The encryption/ decryption process starts with plain text/cipher text added with the key size.

The AES Encryption and Decryption is simulated using Xilinx 13.4 ISE simulator. From the simulation result pbarb ,clk, rst, wr_en, nrst, start, reset,reset1,encrypt, encrypt1, addr, sel and data_in are the input signals and p1, p2,k, prime1, prime2, b3, b4, binary1,binary2, data_out,w1(cipher text), data_ready and data_ready1 are the output signals.

Signal pbarb denotes the type of field, when pbarb=0 it indicates prime field and when pbarb=1 it indicates binary field. LFSR is used to generate scalar multiplication factor 'k' and it is stored in the RAM. 'wr_en' signal is used to write or read the data in to the RAM by selecting 1 or 0. 'nrst' is the reset signal to the EC arithmetic unit. 'start' signal is to start the multiplication (shift & add process). For encryption/decryption, start signal is kept '0'.

Based on the address 'addr' value, k value is multiplied with point 'P' on the curve. Signals prime1, prime2, binary1, binary2 are the outputs of DF- ECC processor is shown in Fig.7.

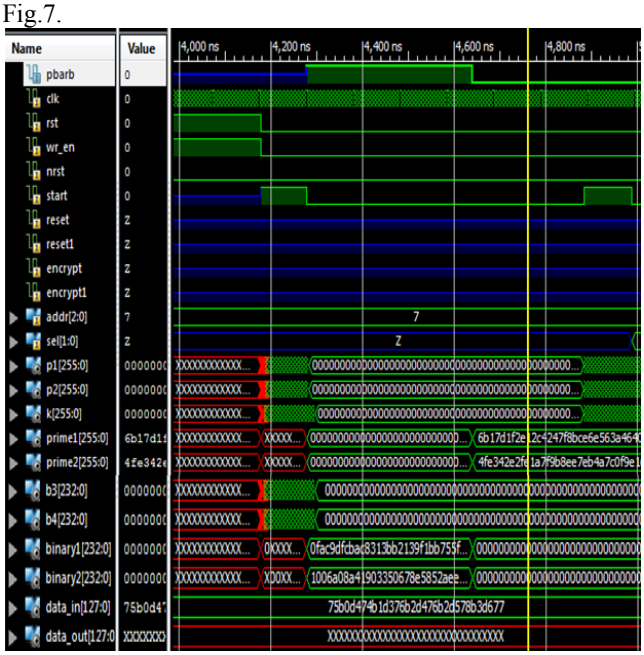

Fig. 7. Dual Field ECC key generation with application.

Signals p1,p2 and b3,b4 generate symmetric keys depending upon the field chosen, is the output of ECC Diffie Hellman key Exchange and signal 'k' is the output of multiplexer. Signal 'k' serves as 'key' for AES Encryption/Decryption is shown in Fig. 8.

The 'reset' and 'encrypt' signals are used for AES encryption. Initially for encryption, select 'reset' as '0' and encrypt' as '1'. Then by making reset as '1'we get the cipher text output in 'w1'. Initially, for decryption 'reset1' and 'encrypt1' signal are kept as '0' .Then by making 'reset1' to '1', we get the plaintext in data_out. On the arrival of cipher text, the 'data_ready' signal is '1' and for Plaintext, data_ready and data_ready1 signals go high and otherwise bot will be low.

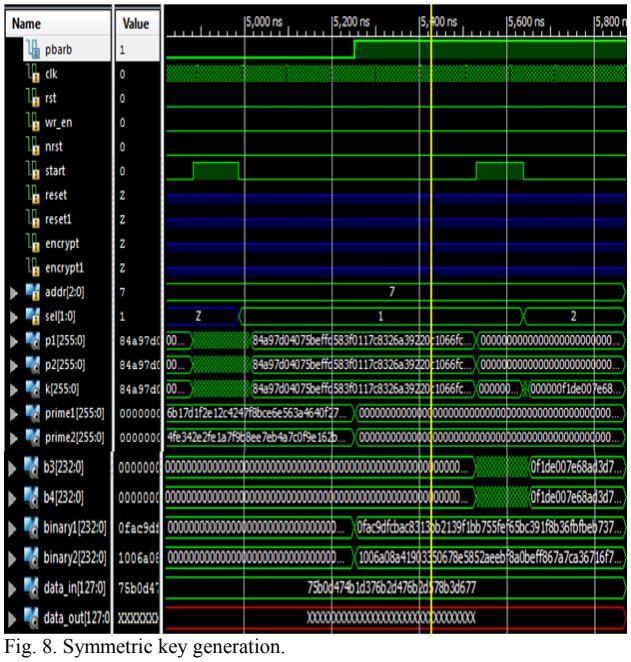

Fig. 8. Symmetric key generation.

Simulation result of AES encryption and decryption are shown Fig. 9 respectively, for binary field. The same procedure can be applied to the prime field.

Table II shows the number of slices occupied, the power consumed and the maximum operating frequency used by the DF-ECC processor, using multiplier M1 for AES application.

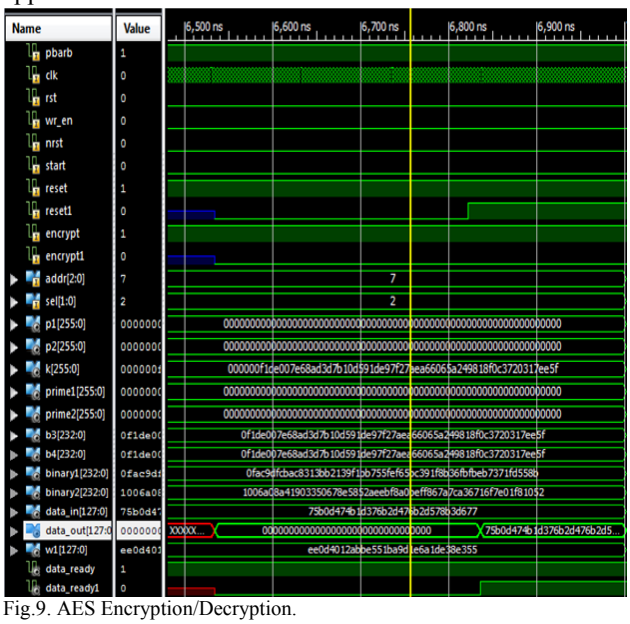

Fig.9. AES Encryption/Decryption.

B. Comparisons

Table I compares existing ECC designs in two different fields separately and Table III compares our DF- ECC processor when scaled up to 233/ 256 bits ECC. In Table III the two methods have been incorporated. Comparing the method1 and method2, our DF-ECC processor using method1 design is more efficient in terms of speed and method2 multiplier design is marginally better in terms of area and power.

TABLE II. DF-ECC PROCESSOR, USING M2 FOR AES APPLICATION

Ours	*Platform*	*Field Size*	f_{MAX} *in MHZ*	*Area [Slices]*	*Power (W)*
(Karatsuba multiplier with AES)	**Xilinx Virtex-5**	**233/256**	**55.389**	**9021**	**1.257**

TABLE III. COMPARISION AMONG THE EXSISTING ECC DESIGNS

	Platform	*Field Size*	f_{MAX} *in MHZ*	*Area [Slices]*	*Power (W)*
Ours (M1)	**Xilinx Virtex-5**	**233-bit GF(2^m)**	**126.20**	**2406**	**1.075W**
		256-bit GF(P)	**45.426**	**2582**	**1.089W**
Ours (M2)	**Xilinx Virtex-5**	**233-bit GF(2^m)**	**121.206**	**2272**	**1.069W**
		256-bit GF(P)	**44.356**	**2436**	**1.081W**
[9]	Xilinx Virtex-2 pro	256-bit GF(P)	43.51	2662 CLB slices	-----
[22]	Xilinx Virtex-2 pro	256-bit GF(P)	37.037	8272 CLB slices	----
[28]	Xilinx Virtex-4	233-bit GF(2^m)	142.53	2648	
[26]	Xilinx Virtex-5	163-bit GF(2^m)	147	3513	
Ours (M1)	**Xilinx XC4vLX 80**	**233-bit GF(2^m)**	**184.504**	**1837**	----
[23]	Xilinx XC4vLX 80	163-bit GF(2^m)	143	24,363	----

The inversion algorithm used in this work occupies lesser area and memory usage, gives high speed and lower dynamic power consumption. The result of algorithm2 is shown in Table IV. Result of Algorithm2 is compared with the existing Algorithm [19], considering bit length as 233 bits. The comparison results are shown in Table IV.

TABLE IV. COMPARISION AMONG THE EXSISTING INVERSION DESIGN

	Platform	f_{MAX} *in MHZ*	*Area [Slices]*	*Area [LUT's]*	*Total memory Usage in KB*	*Power (Dynamic) in mW*
Ours	**Xilinx Virtex -5**	**321.862**	**527**	**1332**	**371924**	**20.22**
[19]	Xilinx Virtex-5	319.38	889	2663	402644	33.52

V. CONCLUSION

We have proposed a new architecture for DF-ECC processor and its design methodology. Our DF-ECC processor supports EC operations involving Point coordinate conversions, point addition, point doubling and point multiplication, with NIST recommended Elliptic curves [3] for both binary and prime fields. By using Virtex 5 Xilinx FPGA device, ECC arithmetic unit design has been implemented with method1 / method2 multiplier. Finally projective points are converted back to affine points using proposed extended Euclidean inversion algorithm. In method1 multiplier design, with a prime field of 256 bits point scalar multiplication with coordinate conversion operates at a maximum frequency of 45.426MHZ, 1089mW power and occupies 2582 slices. On the other hand in binary field, 233 bit point scalar multiplication with coordinate conversion operates at a maximum frequency of 126.20MHZ, 1075mW power and occupies 2406 slices. Method2 multiplier design, with a prime field of 256 bits point scalar multiplication with coordinate conversion operates at a maximum frequency 44.356MHZ, 1081mW power and occupies 2436 slices. On the other hand in binary field, 233 bit point scalar multiplication with coordinate conversion operates at a maximum frequency of 121.206MHZ, 1069mW power and occupies 2272 slices. The experimental results prove that our DF-ECC processor architecture using M1 or M2 provides comparable performance in terms area, time and power.

In future, ECC arithmetic unit performance can be enhanced by using faster multipliers and adders. Also, using the efficient algorithms effectively for modular operations, the current pursuit can be extended to address varied design parameters like area, speed and power.

ACKNOWLEDGMENT

The author would like to thank RNS Institute of Technology for the constant support and Encouragement.

REFERENCES

[1] N. Koblitz, "Elliptic curve cryptosystems," Math. Comput., vol. 48,pp. 203–209, 1987.

[2] V.S. Miller, "Use of elliptic curve in cryptography," in Proc. Crypto,1986, pp. 417–426.

[3] IEEE 1363 standard specifications for public-key cryptography,1363,Jan. 2000.

[4] Akashi Satoh and Kohji Takano, "A Scalable Dual-Field Elliptic Curve Cryptographic Processor," IEEE Transaction on Computers, vol. 52, No. 4,Apr 2003.

[5] Jyu- Yuan Lai and Chih-Tsun Huang, "High-Throughput Cost-Effective Dual-Field Processors and the Design Framework for Elliptic Curve Cryptography," IEEE Trans on VLSI systems, vol. 16, No 11,Nov 2008.

[6] Wanzhong Sun, Zibin Dai and Nianmin Ren2., "A Unified, Scalable Dual-Field Montgomery Multiplier architecture for ECCs," IEEE,2008.

[7] Jyu-Yuan Lai and Chih-Tsun Huang, "A Highly Efficient Cipher Processor for Dual-Field Elliptic Curve Cryptography," IEEE Trans. on Circuits and Systems—II,vol. 56, No. 5, May 2009.

[8] B.MuthuKumar,S.Jeevananthan,"High Speed Hardware Implementation of an Elliptic Curve Cryptography(ECC)Co-Processor," IEEE International conference, 2010.

[9] Jen-Wei Lee, Ju-Hung Hsiao, Hsie-Chia Chang, and Chen-Yi Lee, "An Efficient DPA Countermeasure With Randomized Montgomery Operations for DF-ECC Processor", IEEE Trans. on Circuits and Systems—II,vol. 59, No. 5, May 2012.

[10] J. Lopez and R. Dahab, "Improved algorithms for elliptic curve arithmetic in GF(2^m)" in Proc. Sel. Areas Cryptography: 5th Annu. Int.Workshop (SAC), vol. 1556, pp. 201–212 ,Aug. 1998.

[11] NIST, Recommended Elliptic Curves for Federal Government Use, May 1999 (http://csrc.nist.gov/encryption).

[12] Darrel Hankerson, Alfred Menezes and Scott Vanstone, "Guide to Elliptic Curve Cryptography" Springer, 2004.

[13] NIST, Announcing the advanced encryption standard (AES).Federal Information Processing Standards Publication 197,Tech. Rep., 2001.

[14] William Stallings, "Cryptography and Network Security", Third edition, <www.williamstallings.com/Crypto03e.html.

[15] C.Prema and C.S.Mainkanda babu, "Enhanced high speed modular multiplier using Karatsuba algorithm," presented at the IEEE Int. Conf. Computer Communication and Informatics (ICCCI -2013), Jan.2013.

[16] Joachim von zur Gathen and Jamshid Shokrollahi, "Efficient FPGA based Karatsuba multipliers for polynomials over F_2," In Selected Areas in Cryptography(SAC 2005), Lecture Notes in Computer Science, 359– 369. Springer-Verlag, Kingston, ON, Canada. ISBN3-540-33108-5.

[17] W.Sun and L. Chen, "Design of scalable hardware architecture for dual- field montgomery modular inverse computation," in Proc. Pacific-Asia Conf. Circuits, Commun. Syst., Chengdu, China, May 2009, pp. 409–411.

[18] B. Ansari and M. A. Hasan, "High-performance architecture of elliptic curve scalar multiplication," IEEE Trans. on Computers, vol 57, no. 11, Nov. 2008, pp. 1143–1153.

[19] Gerald Lai, "Analysis of modular Inverse GF(p) Implementation," A survey, Oregon state university, Corvallis, Oregon 97331.

[20] S.M.H Rodríguez and F. Rodríguez-Henríquez, "An FPGA airthmetic logic unit for computing scalar multiplication using the half and- add method," presented at the IEEE Int. Conf reconfig. Comput FPGA's (ReConFig), Puebia, Sep. 2005.

[21] K Sakiyama, L. Batina, B. Preneel, and I. Verbauwhede, "Multi-corecurve-based cryptoprocessor with reconfiguarable modular. arithmetic logic units over GF(2^m)" IEEE Trans. On Computers, vol. 56, no. 9, pp. 1269–1282, Sep. 2007.

[22] Jen-Wei Lee, Szu-Chi Chung, Hsie-Chia Chang and Chen-Yi Lee, "Efficient Power-Analysis-Resistant Dual-Field Elliptic Curve Cryptographic Processor Using Heterogeneous Dual-Processing Element Architecture," IEEE Trans. On VLSI systems,2013.

[23] Chang Hoon Kim , Soonhak Kwon , Chun Pyo Hong , "FPGA implementation of high performance elliptic curve cryptographic processor over GF(2^{163}) Journal of Sys. Architecture,Elsvier,2008.

[24] K. Sakiyama, E. D. Mulder, B. Preneel, and I. Verbauwhede, "A Parallel processing hardware architecture for elliptic curve cryptosystems," inProc. IEEE ICASSP, Toulouse, France, May 2006, vol. 3, pp. 904–907.

[25] Yi Wang, Douglas L. Maskell, Jussipekka Leiwo, "A unified architecture for a public key cryptographic coprocessor," Journal of systems Architecture, Elsvier,2008.

[26] Sujoy Sinha Roy, Chester Rebeiro, and Debdeep Mukhopadhyay, "Theoretical Modeling of Elliptic Curve Scalar Multiplier on LUT - Based FPGAs for Area and Speed," IEEE Trans. On VLSI systems,Vol.21,No.5, May 2013.

[27] Gustavo D. Sutter, Jean-Pierre Deschamps, and José Luis Imaña, "Efficient Elliptic Curve Point Multiplication Using Digit-serial Binary Field Operations," IEEE Trans. On Industrial Elns, Vol. 60, No. 1, Jan 2013.

[28] K.C.Cinnati Loi and Seok-Bum Ko, "High Performance Scalable Elliptic Curve Cryptosystem Processor in GF(2^m),IEEE,2013.