HIGH SPEED POINT ARITHMETIC ARCHITECTURE FOR ECC ON FPGA

Rahila Bilal [1]
1.Sel.Gr. Lect/Research Scholar ,Dept. of ECE ,
THANTHAI PERIYAR GOVT.INSTITUTE OF TECHNOLOGY, VELLORE-2.

Dr.M.Rajaram [2]
2.Professor & HOD, Dept. of EEE,
GOVT.COLLEGE OF ENGINEERING,, TIRUNELVELI .

Abstract *:* Elliptic curve cryptography plays a crucial role in networking and communication security. ECC have evolved in the recent past as an important alternative to established systems like RSA. This paper describes the implementation of an elliptic curve coprocessor based on the FPGA , which can provide a significant speedup for these cryptosystems. The FPGA configuration file is synthesized from VHDL code applying different hardware synthesis products. The implementation of ECC lies in three levels: scalar multiplication, point addition/doubling and finite field modular arithmetic. In this paper, we present a novel fast architecture for the point addition/doubling level in the projective coordinate. The proposed Architecture is based on Binary Field. The Design performs multiplication using Polynomial Basis. Analysis shows that, with reasonable hardware overhead, our architecture can achieve a high speedup for the point addition operation and point Doubling operation.Furthermore, the architecture is parameterized for different data widths to evaluate the optimal resource utilization.

Key words: FPGA, Elliptic curve cryptography, modular arithmetic, projective co-ordinates.

1.INTRODUCTION

Digital communication networks, like the Internet, are incorporated today in many applications that require secure connections. Channels in these networks are often public, i.e. eavesdropping or altering of the transmitted data can not be prevented. Cryptosystems ensure the detection of altered data and prevent the extraction of information from eavesdropped data by unauthorized parties. In order to make security requiring applications economically feasible, it is necessary to implement efficient cryptosystems.

Modern cryptosystems fall in one of two categories: symmetric and asymmetric or public key cryptosystems. In symmetric cryptosystems a single key is used for both, the encryption and the decryption process. This implies that the key must be known by both communicating parties and thus must already have been transmitted through some secure channel. All communication must be planned beforehand. Spontaneous secure communication, which is necessary in many applications (for example online shopping) is not possible. The most expensive operation applied in elliptic curve based cryptosystems is the "multiplication" of a large natural number with a point on an elliptic curve. The coprocessor presented in this paper is an elliptic curve multiplier that speeds up the multiplication operation significantly in comparison to a purely software based implementation.

2.ECC-NEXT GENERATION OF PKC

ECC is much stronger per bit than RSA and is less computationally intensive. Elliptic curve cryptography plays a crucial role in networking and communication security. Elliptic Curve Cryptography (ECC) is a Public Key Cryptography. In Public Key Cryptography each user or the device taking part in the communication generally have a pair of keys, a public key and a private key, and a set of operations associated with the keys to do the cryptographic operations. Only the particular user knows the private key whereas the public key is distributed to all users taking part in the communication.

ECC is defined over the elliptic curve

$$y^2 = x^3 + ax + b$$

Where $4a^3 + 27b^2 \neq 0$

Each value of the 'a' and 'b' gives a different elliptic curve. All points (x, y) which satisfies the above equation plus a point at infinity lies on the elliptic curve. The public key is a point in the curve and the private key is a random number. The advantages of ECC are

*It offers greater security for given key size.

*The smaller key size also makes possible much more compact implementations for a given level of security.

2.1 Elliptic Curve DISCRETE LOGARITHM PROBLEM

The security of ECC relies on the difficulty of solving the Elliptic Curve Discrete Logarithm Problem (ECDLP), i.e. finding k, given P and Q = kP. The problem is computationally intractable for large values of k.

2.2 ECC ARITHMETIC

Elliptic curves are a good candidate for the additive group. Elliptic curves have been used for Public Key Cryptography. It is believed that Elliptic Curve Cryptography is more efficient than other Public Key Cryptography such as RSA in terms of the key length. Finite field is also called Galois Field, it means a field that contains finite number of elements. Binary field is more efficient for the hardware implementation. Therefore, we use binary field in this work. An elliptic curve is defined in a standard two dimensional (x,y) Cartesian coordinate system by an equation of the form:

$$y^2 = x^3 + ax + b$$

An elliptic curve is used to define the members of the set over which the group is calculated as well as the operation between them which define how mathematical operations works in the group. A non-super singular curve over Galois fields with characteristic two is defined as:

$$y^2 + xy = x^3 + ax^2 + b \quad \text{---- (1)}$$

this equation together with a point at infinity O forms an elliptic curve where a,b belongs to GF(2^n) and b ≠ 0.The set GF(2^n) forms an additive Abelian group, which is based on the following definitions

with $P = (x_1, y_1)$, $Q = (x_2 , y_2)$
and $P \neq \pm Q$

Identity: $P + O = O + P = P$

Negation: For $P \neq O$, $-P = (x_1, x_1 + y_1)$

PointAddition:

P+Q=R=(x3,y3)

where (x3,y3) belongs to GF(2^n),

$x_3 = \lambda^2 + \lambda + x_1 + x_2 + a$

$y_3 = \lambda(x_1 + x_3) + x_3 + y_1$ with

$$\lambda = \frac{(y_1+y_2)}{(x_1+x_2)}$$

Point Doubling:

2P=R=(x_3, y_3)

where (x_3,y_3) belongs to GF(2^n),

$$x_3 = \lambda^2 + \lambda + a = x_1^2 + \frac{b}{x_1^2}$$

$y_3 = x_3^2 + \lambda x_3 + x$, with

$$\lambda = x_1 + \frac{y_1}{x_1}$$

2.3 Co- ordinates in ECC

Affine coordinate

Affine coordinate representation can help reduce network bandwidth and memory space for transmission and storage. On the other hand, performing ECSM (Elliptic Curve Scalar Multiplication) in affine coordinate involves a large number of finite field inversion operations. The standard formulae for adding two points on an elliptic curve with the affine coordinates require 1 inverse operation which is costly in the fields of order 2^{163}. The cost ratio of inversion to multiplication is more than 8. Since inversion is costly in hardware we will consider an alternate point representation. This can be done if the elliptic curves are considered with projective coordinates.

Projective Coordinate

In Projective Coordinate, more efficient ECSM algorithms do not involve any finite field inversion operations. Therefore, almost all of the recent hardware implementations of ECSM are in the projective coordinate. A projective plane of the fixed exponential integers (α,ß) over GF(2^n) is defined by creating an equivalence relation of the triples(x,y,z)~(X,Y,Z) if there exists λ ϵ GF(2^n), and λ≠ 0 such that we have (x,y,z)=(λ^{α}X, λ^{β}Y,λZ).

Every point (x,y) on the affine coordinate can be mapped to the projective plane with ϕ: (x,y)→(x,y,1). From the above definition, every equivalent class of the triples on the projective plane(X,Y,Z),Z≠0 can be mapped back to the affine point by x=X/Z^{α} and y=Y/Z^{β}. Currently, there are three popular projective coordinates applied to the ECC system, which are

a) The Homogenous projective coordinate with α =1 and ß =1
b)The Jacobian coordinate with α =1 and ß =3
c)The Lopez Dahab projective
coordinate with α =1 and ß =2
In the third case, x=X/Z and y=Y/Z^2.

Projective versus Affine

The projective and affine implementations share the same hardware design and hence occupy the same circuit area. The total number of cycles required for an elliptic curve multiplication and the execution time required for an elliptic curve multiplication at the maximum frequency reduces for projective coordinates when compared to affine coordinates. The implementation using projective coordinates is always faster than using affine coordinates.

2.4 Elliptic Arithmetic

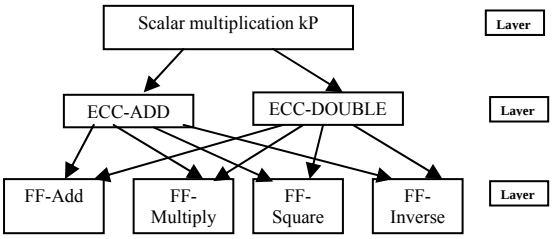


Fig 1. Arithmetic Hierarchy

The elliptic curve operations are performed at three different layers. The top layer computation of ECC is the scalar multiplication, which is based on the point addition and point doubling operations. In the middle layer is the point addition and point doubling which are denoted as ECC-ADD and ECC-DOUBLE respectively. At the bottom layer is the finite field operations, which include finite field multiplication, finite field addition, finite field squaring, and finite field inverse operations.

3. Point multiplication

The core operation in ECC is point multiplication k*P, where k is an integer, P is a point on the curve. A single point multiplication requires multiple computations of point addition (P≠Q) and point doubling(P=Q). The standard method for point multiplication is the double and add algorithm. The algorithm is as follows:

1.k is an integer such that k=(k_l, k_{l-1}…., k_1, k_0) is the binary representation of k with most significant bit k_l=1.

2.Copy original point to temporary variable: temp←P.

3.For index from (l-1) down to 0 do:

a)DOUBLE: temp←temp+temp

b) If k_{index} =1, then also ADD: temp←temp+P

4. Return temp which contains k*P.

This method requires l doublings and w_{k-1} additions where w_k is the binary representation of k. It is possible to provide an exact count for the number of operations necessary to realize point doubling and point addition.

We can see that saving the inversion required for affine coordinate comes at the cost of more multiplications. Since the number of multiplications has increased, the designer is forced to use more temporary registers for the intermediate values as more data dependencies are present.

4.Lopez-Dahab point arithmetic

In the Lopez-Dahab projective coordinate, the point (X, Y,Z) (Z≠0) is corresponding to the point (X/Z, Y/Z^2) in the affine coordinate, and the elliptic curve equation is transformed into the following form:

$$Y^2 + XYZ = X^3Z + aX^2Z^2 + b\,Z^4$$

The point addition formula that does not involve the inversion operation can be derived by converting the point to affine projective as x = X/Z and y = Y/Z^2 at first, then adding the affine points with the general equation for point addition, and finally clearing the denominators. Similarly, the L-D point doubling equation can be derived.

4.1 Lopez-Dahab Algorithm

Consider two distinct point P=(X_1, Y_1, 1) and Q=(X_0, Y_0, Z_0) on the elliptic curve, the result

R=(X_2, Y_2, Z_2) = P+Q

A.Point Addition Algorithm

1.A ←X_1Z_0	12.A ← A.D
2.B ← X_1+A	13.B ← A+B
3.A ← Z_0^2	14.D ←D^2
4.C ← a.A	15.B ← B+D(X_2)
5.A ← Y_1.A	16.F ← X_1.E
6.D ← Y_0+A	17.G ← Y_1.E
7.A ← B.Z_0	18.F ← B+F
8.B ← B^2	19.G ← B+G
9.C ← A+C	20.F ← A.F
10.B ← B.C	21.G ← E.G
11.E ← A^2(Z_2)	22.D ← F+G(Y_2)

B.Point Doubling Algorithm

Consider a point Q=(X_1,Y_1, Z_1) on the elliptic curve, the result R=(X_2,Y_2,Z_2)= 2Q

1. A ← Z_1^2	8. D ← Y_1^2
2. B ← c.A	9. E ← a.A
3. B ← B^2	10. D ← D+E
4. C ← X_1^2	11. D ← D+B
5. A ← A.C(Z_2)	12. D← C.D
6. C ← C^2	13. B ← A.B
7. C ←C+B(X_2)	14. D ← B+D(Y_2)

The algorithms discussed above when implemented in a Sequential way have the advantage that the number of finite field arithmetic modules can be reduced to minimum (like only one adder, one multiplier and one squaring unit are needed for point addition and doubling).

Though the number of arithmetic modules is reduced, such designs introduce long latency (difference in the availability of the first output data in the pipelined system and the sequential system). The latency of the point doubling is $5T_M$+$5T_S$+$4T_A$, and the latency of the point addition is $10T_M$ +$4T_S$ +$8T_A$, where T_M, T_S, T_A denote the latency of the finite field multiplier, squaring unit, and adder respectively), which is not desirable for the applications where high-speed ECC implementation is required.

5.Parallel Architecture

A popular approach to increase the processing speed and to reduce the latency is to apply the parallel processing

technique. By introducing more processing units that can operate in parallel, the results can be obtained much faster. In parallel processing multiple outputs are computed in parallel in a clock period.

Effective sampling period is increased by the level of parallelism. Parallel processing can also be used to reduce the power consumption. Parallel processing increases the sampling rate by replicating hardware so that several inputs can be processed in parallel and several outputs can be produced at the same time.

5.1 Parallel architecture for Point Addition

The latency for Point Addition is calculated as $4T_M+6T_A$, where T_M, T_A denote the latency of the finite field multiplier and adder respectively. The total latency ratio of the serial architecture of Point Addition over the Parallel Point Addition architecture is $(10r_1+4r_2+8)/(4r_1+6)=2.52$ (when r1=15 and r2=2) which is the speedup achieved by using the parallel architecture for Point Addition.

From the architecture shown in Fig 2 there are 10 multipliers, 4 squaring units and 8 adders are used. But 4 multipliers 1 squaring unit and 2 adders are working in parallel. So the modules for multipliers, squaring units and adders are reduced. For an efficient L-D coordinate design we can delay the addition in step 19 by T_A which means Step 19 starts after Step 18 completes. In this way, only one finite field adder is needed. Thus, the total latency is increased by T_A .But $GF(2^n)$ adder is a number of XOR gates in parallel, whose latency is very small, the rescheduling discussed above becomes meaningful. We can reduce the number of multipliers from 4 to 2.This method is to advance the multiplication of Step 1 by T_M-T_S, which makes Step 1 complete before Steps 4 and 5 begin. Also, we need to delay the multiplications of Steps 16 and 17 so that they start after the completion of Step 10.

For such a modified architecture, at most two multipliers are working in parallel at any time instance. Thereby, the total number of multipliers is reduced to two. And the total latency becomes $6T_M+4T_A$, which offers another trade-off between the area and speed. When $r_1=15$ and $r_2=2$, the speedup achieved.

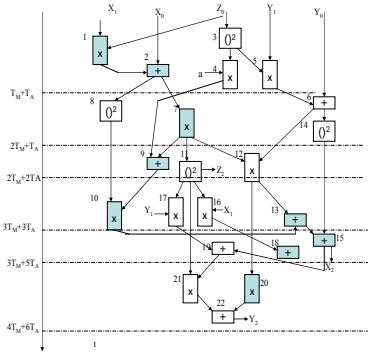


Fig 2. Parallel Architecture for Point Addition

5.2 Parallel architecture for Point Doubling

In Fig.3, the parallel architecture of the Lopez-Dahab point doubling algorithm is shown. We can see the total latency required to compute Y_2 is $3T_M+T_S+3T_A$.

Assume the timing cost ratio of T_M to T_A is r_1 (usually around 15), T_S to T_A is r_2 (around 2), the total latency ratio of the serial architecture of point doubling to the corresponding parallel architecture is $(5r_1+5r_2+4)/(3r_1+r_2+3)=1.78$ (when $r_1=15$ and $r_2=2$), which is the speedup we have achieved by applying the parallel architecture. The different modules in the Point Doubling architecture shown in Fig 3.1 are:

- Multipliers-5
- Squaring units-5
- Adders-4

In this architecture at most 2 multipliers, 2 squaring units and 2 adders are working in parallel at the same time. Hence the number of arithmetic modules can be reduced to 2 multipliers, 2 adders and 2 squaring units.

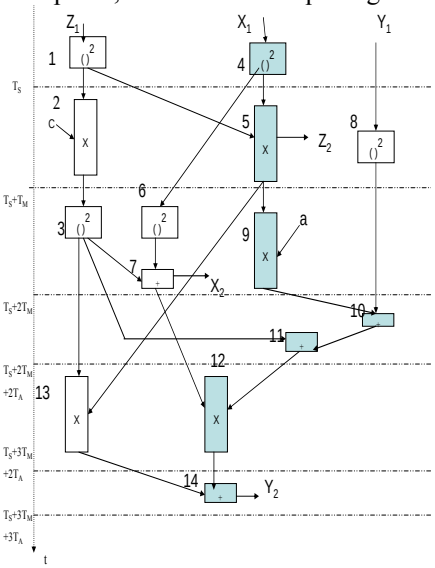


Fig 3. Parallel Architecture for Point Doubling

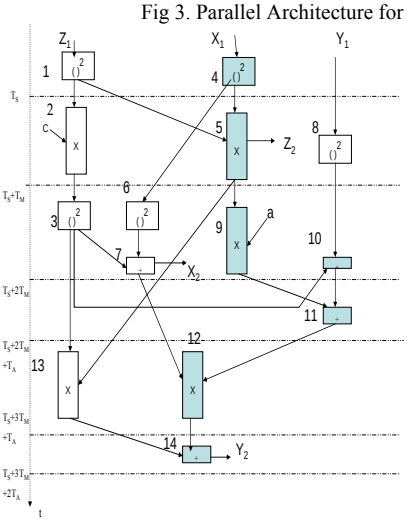


Fig4.ModifiedParallel Architecture Point Doubling

5.3 Modified Point Doubling Architecture

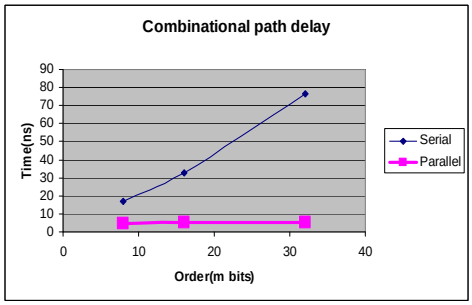
The critical path length in the parallel architecture for L-D point doubling is T_S+3 T_M+3 T_A. This critical path length can be reduced to T_S+3 T_M+2 T_A by the modified .

by the modified architecture of the parallel architecture for point doubling. Also based on the parallel architecture for point doubling totally 5 multipliers, 5 squaring units and 4 adders are used.

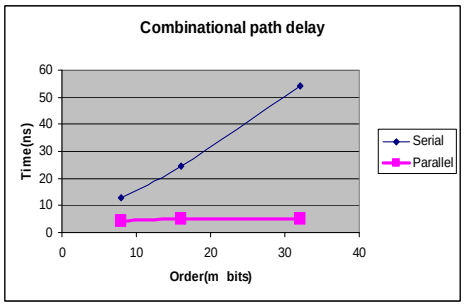
This can be reduced by the modified architecture where at most two multipliers are working in parallel. Similarly at most two squaring units are needed at the same time. If the point doubling implementation has been pipelined and well scheduled, to minimize the number of finite field arithmetic units, thereby reduce the hardware cost to two multipliers, two squaring units and one adder.

6.Results

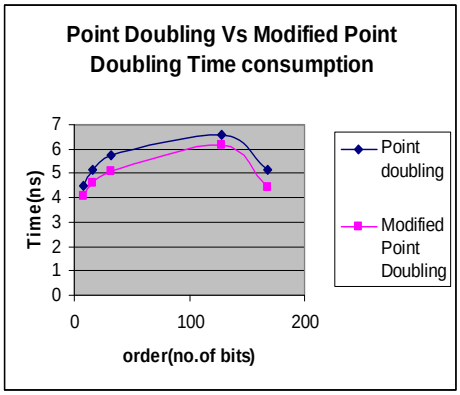
The Fast Point Architecture for the Elliptic Curve Cryptography Point Arithmetic (Point Doubling and Point Addition) presents an efficient approach for ECC arithmetic. The core operation of ECC is based on Scalar Multiplication which involves Point Addition and Point Doubling. The point operations in ECC are carried with the projective coordinates by using the Lopez-Dahab algorithm where the inversion operation is avoided. Codings were developed for different data widths (m= 8,16,32) using serial and parallel architecture .The results are graphed.



Serial Vs Parallel Point Addition



Serial Vs Parallel for point doubling



Point doubling Vs Modified Point doubling time consumption

Table 1. Synthesis report for different bit sizes for Point Addition(Parallel)

m	8	16	32	128	163
slices	270	495	936	3546	4472
LUTs	355	640	1269	5072	5439
Delay (ns)	4.447	5.062	5.100	5.560	5.062

Table 2. Synthesis report for different bit sizes for Point Doubling(Parallel)

	8	16	32	128	163
Slices	90	110	208	1184	1506
LUTs	134	262	300	2491	2364
delay (ns)	4.475	5.145	5.742	6.574	5.128

Table 3. Synthesis report for different bit sizes for Modified Point Doubling(Parallel)

	8	16	32	128	163
Slices	120	220	416	1587	2000
LUTs	171	320	640	2620	2871
delay (ns)	4.385	4.600	5.720	6.351	4.421

Simulation result for m=163 for Point Addition

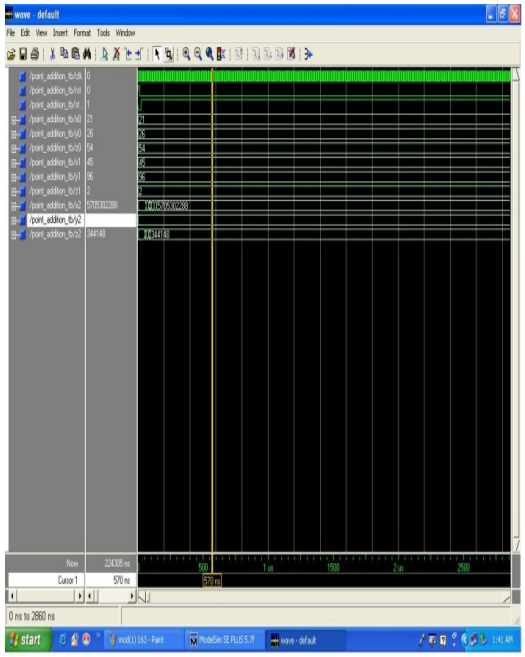


Fig 4.3 Point Addition-Wave(m=163)

Simulation result for m=163 for Point Doubling

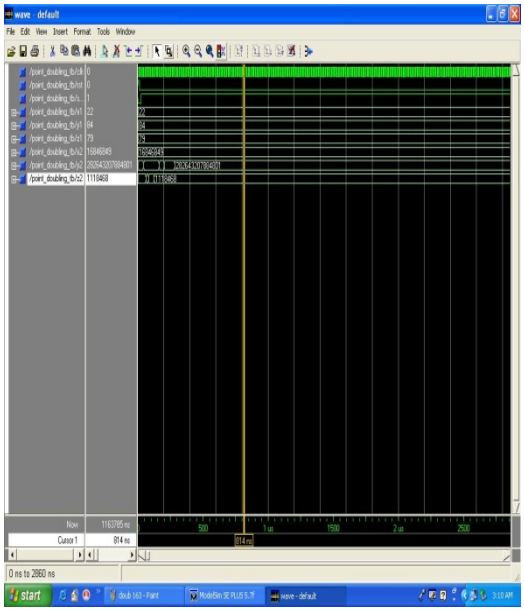


Fig 4.4 Point Doubling-Wave(m=163)

Simulation result for m=163 for modified Point Doubling using VHDL

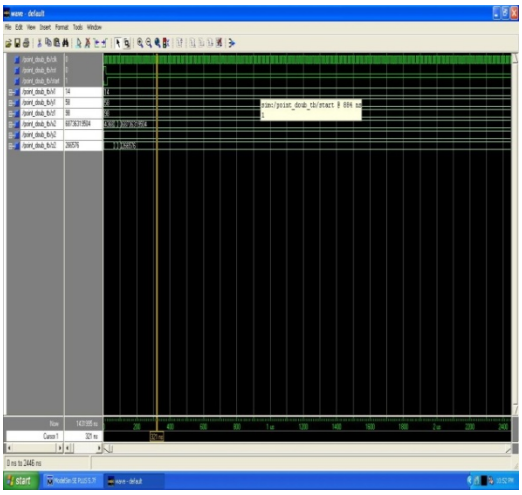


Fig 4.5 Modified Point Doubling-Wave (m=163)

Table 4. Resource utilization and time consumption using various Target devices

DEVICE	SLICES	FLIP FLOPS	LUT'S	TIME DELAY(ns)	FREQUENCY (MHz)
XC3SD1800ACS484	1592	2036	2861	10.06	246.000
XC3S100E VQ100	1592	2017	2861	8.470	256.707
XC2VP2 FG256	1592	2040	2861	6.299	368.793
XC4VLX15 SF363	1594	2035	2865	6.292	391.203
XC5VLX30 FF324	2000	2871	2000	4.421	613.911

7. Conclusion:

The work towards achieving the high speed for he point operation is successfully completed and he results are compared for various field order for different target device Thus a novel architecture for point addition and point doubling with the parallel architecture in projective coordinates is implemented on FPGA.

8.References

[1] Chudnovsky D.V and Chudnovsky G.V,(1986), "Sequences of numbers generated by addition in formal groups and new primality and factorization tests." pp.385-434.

[2] Gupta.V, Gupta.S, Chang.S and Stabila.D,(2002) "Performance analysis of elliptic curve cryptography for SSL", pp. 87-94.
[3] Hankerson. D,Lopez. L, and Menezes. A,(2000) "Software implementation of elliptic curve cryptography over binary fields",pp.1-24.
[4] Jian Huang, (2007)" FPGA implementations of Elliptic Curve Cryptography and Tate Pairing over Binary Field.
[5] Julio Lopez and Ricardo Dahab,(2000) "An overview of elliptic curve cryptography", Technical report, Institute of Computing, State University of Campinas, Brazil.
[6] Lauter.K(2004),"The advantages of elliptic curve cryptography for wireless security," IEEE Wireless Communications, pp. 62-67.
[7] Lopez. J and Dahab. R,(1999) "Improved algorithm for elliptic curve arithmetic in GF(2n)." pp.201-212
[8] Neal Koblitz,(1987) "Elliptic curve cryptosystems," Mathematics of Computation, vol. 48, no. 188, pp. 203-209.
[9] Qingwei Li, Zhongfeng Wang(2008) " Fast Point Operation Architecture for Elliptic Curve Crypography.
[10] Rivest. R, Shamir. A, and Adleman. L,(1978) "A method for obtaining digital signatures and public-key cryptosystems," Communications of the ACM, pp.120-126.
[11] Wang.Y.B, Dong.X.J, and Tian.Z.G, (2007)"FPGA based design of elliptic curve cryptography coprocessor," IEEE 3rd International Conference on Natural Computation, ICNC .
[12] Aug 2000, "IEEE Standard Specifications for Public Key Cryptography", IEEE Standard P1363-2000.

9. Authors Profile

Mrs. Rahila Bilal is working as an Associate Professor , Department of Electronics and Communication Engineering in Thanthai Periyar Government Institute Of Technology, Vellore, Tamilnadu, affiliated to Anna university, Chennai. Has got 23 years of Teaching experience. She has presented many papers in National and International Conferences Has been the co-coordinator in conducting the International Conference and various workshops .

Dr.M.Rajaram, is the Prof. & Head of the department of Electrical and Electronics Engineering , at Government college of Engineering , Tirunelveli, .He has published as many as 120 Technical papers in National and International Journals. He has his credit of having organized many Conferences, Seminars and workshops .He has served as Professor in Government Engineering Colleges in the state of Tamil Nadu for more than 28 years. He has guided and produced many PhD's. He is being a reviewer in many reputed journals and an active member in various technical committees and societies including IEEE and ISTE.