ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

International Journal of Advanced Research in Management, Architecture, Technology and Engineering (IJARMATE) Vol. 3, Special Issue 6, March 2017

FPGA Implementation of High Performance Elliptic Curve Cryptographic Processor

M. Mohamed Alameen
P.G Scholar, Department of Electronics and communications Engineering
Gojan School of Business and Technology
Chennai, Tamilnadu 600052
alameen.mb17@gmail.com

M. Pandiyarajan
Assistant Professor, Department of Electronics and communications Engineering
Gojan School of Business and Technology
Chennai, Tamilnadu 600052
pandiyarajan14@gmail.com

Abstract **- The ultimate purpose of this project has been the implementation in MATLAB of an Elliptic Curve Cryptography (ECC) system, primarily the Elliptic Curve Diffie-Hellman (ECDH) key exchange. We first introduce the fundamentals of Elliptic Curves, over both the real numbers and the integers modulo p where p is prime. Then the theoretical underpinnings of the ECDH system are covered, including a brief look at how this system is broken. Next, we develop the individual elements that will be needed in the implementation of ECDH, such as functions for calculating modular square roots and the addition of points on an EO. We then bring these elements together to create a working ECDH program, and discuss the limitations of the MATLAB environment in which it was created. Finally, we look at the real world application of ECC and its future in the realm of cryptography.**

Index Terms-MATLAB, ECC, EDHC.

I. INTRODUCTION

The goal of this report is to first give a description of the mathematics behind Elliptic Curve Cryptography (ECC), in particular the Elliptic Curve Diffie-Hellman (ECDH) key exchange system, and secondly to de-scribe and develop the algorithms and methods necessary for the implementation of the ECDH system in the MATLAB environment.
Later introduces the fundamental mathematics of the Elliptic Curve, over both the real numbers and the integers modulo p, where p is prime. This includes the addition law denoted by EB, and the construction of the abelian elliptic curve group. Next, in Section 3, we look at the ECDH key exchange system. The basis of this system is the Elliptic Curve Discrete Logarithm Problem (ECDLP), which is discussed in some detail. After outlining the steps necessary to perform an ECDH key exchange between two people, we give a brief overview of the methods available to solve the ECDLP and hence break the ECDH system.

Our focus changes away from the theoretical foundations and towards the practical application of ECC. We describe and develop the tools, methods and algorithms necessary for ECC, starting with the basic operations of modular exponentiation and the calculation of inverses over finite fields, and moving on to the more complicated tasks of finding modular square roots and the addition of points on the EC. Added to that we bring these individual pieces together to construct a functional ECDH system, and then discuss the limitations of this program and the MATLAB environment it was created in. Finally, we discuss how ECC is implemented in the real world, and its future use and standardization.

II. THEORY OF ECC

A. *Elliptical curve cryptography*

Elliptical curve cryptography (ECC) is a public key encryption technique based on elliptic curve theory that can be used to create faster, smaller, and more efficient cryptographic keys. ECC generates keys through the properties of the elliptic curve equation instead of the traditional method of generation as the product of very large prime numbers. The technology can be used in conjunction with most public key encryption methods, such as RSA, and Diffie-Hellman. According to some researchers, ECC can yield a level of security with a 164-bit key that other systems require a 1,024-bit key to achieve. Because ECC helps to establish equivalent security with lower computing power and battery resource usage, it is becoming widely used for mobile applications. ECC was developed by Certicom, a mobile e-business security provider, and was recently licensed by Hifn, a manufacturer of integrated circuitry (IC) and network security products. RSA has been developing its own version of ECC. Many manufacturers, including 3COM, Cylink, Motorola, Pitney Bowes, Siemens and VeriFone have included support for ECC in their products.

B. *Cryptographic schemes*

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

International Journal of Advanced Research in Management, Architecture, Technology and Engineering (IJARMATE) Vol. 3, Special Issue 6, March 2017

Cryptographic schemes several discrete logarithm-based protocols have been adapted to elliptic curves, replacing the group with an elliptic curve and the elliptic curve Diffie–Hellman (ECDH) key agreement scheme is based on the Diffie–Hellman scheme. The Elliptic Curve Integrated Encryption Scheme (ECIES), also known as Elliptic Curve Augmented Encryption Scheme or simply the Elliptic Curve Encryption Scheme.

The Elliptic Curve Digital Signature Algorithm (ECDSA) is based on the Digital Signature Algorithm. The deformation scheme using Harrison's p-adic Manhattan metric

The Edwards-curve Digital Signature Algorithm (EdDSA) is based on Schnorr signature and uses twisted Edwards curves The ECMQV key agreement scheme is based on the MQV key agreement scheme and the ECQV implicit certificate scheme. At the RSA Conference 2005, the National Security Agency (NSA) announced Suite B which exclusively uses ECC for digital signature generation and key exchange. The suite is intended to protect both classified and unclassified national security systems and information. Recently, a large number of cryptographic primitives based on bilinear mappings on various elliptic curve groups, such as the Weil and Tate pairings, have been introduced. Schemes based on these primitives provide efficient identity-based encryption as well as pairing-based signatures, signcryption, key agreement, and proxy re-encryption

III. THEORETICAL BACKGROUND

A. *Fundamentals of Elliptic Curves*

Elliptic Curves are a type of algebraic curve with a general form described by the Diophantine equation.

Andrew Wiles utilized them in his proof of Fermat's Last Theorem, and they are gaining popularity in the realm of cryptography for their security and efficiency over current cryptographic methods. They form a large part of US National Security Agency's (NSA) Suite B of cryptographic algorithms that will, over the next decade, replace those currently in use, such as RSA and the Diffie-Hellman Key Exchange [NSA, a]. An illustration of the superiority of ECC over older methods can be seen by looking at the comparative key size needed to ensure a similar level of security: where RSA would need a key size of 1024 bits, methods based on ECs would only require one of 160 bits [NSA, b].

The elliptic curve can be defined over many fields, ranging from the complex numbers C and the rationales' Q to the real numbers IR and integers modulo p as covered in Sections 2.1 and 2.2 [Silverman, 2005, pg. 100-104]. For any field K we can in general find a group (E(K),EB), that consists of pairs of elements in K that are solutions to Equation 1, plus an abstract point denoted by ∞ , which will be discussed further below. The EB operator that acts upon the elements of the group remains the same algebraically for each field, though the procedure for calculating it may vary slightly.

B. *Elliptic Curves over lR*

An elliptic curve over the field IR, denoted as E (IR), is the set of real solutions to equation 1, with a, b, c, dE IR and a# 0, i.e. the set {(x, y) E IR2 Iy2 = ax 3 + bx2 +ex+ d}, with the addition of an abstract point ∞ [Martin, 2006, pg. 5]. We must also state the condition that to be an elliptic curve Equation 1 should have three distinct roots when y =0, either real or complex. This guarantees that the graph of the curve is non-singular [Silverman, 2005, pg. 94], and hence a tangent line can be found at every point, the importance of which will become apparent later.

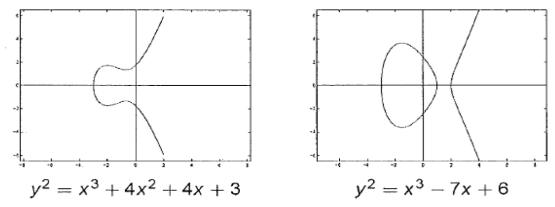


Figure 1. Elliptic Curve Graphs [Vercauteren, 2005]

The graph produced by plotting the elliptic curve on the x and y axes has one of the following forms: From this elliptic curve we can construct an Abelian group by introducing the binary operation EE, called the addition law onE, which over lR!. can be given by a simple geometric construction [Martin, 2006, pg. 8].

Definition 1: Binary operation over lR!.

Let P1, P2 E E(JR), with P1 = (xl,Yl) and P2 = (x2,Y2)·

Case One:

$$P_1 \boxplus P_2 = \begin{cases} P_1 : if\ P_2 = \infty \\ P_2 : if\ P_1 = \infty \end{cases}$$

Case Two:

$$P_1 \boxplus P_2 = \begin{cases} \infty : if\ x_1 = x_2\ and\ y_1 \neq y_2 \\ \infty : if\ x_1 = x_2\ and\ y_1 = y_2 = 0 \end{cases}$$

Case Three:

If Case One and Two are not met, we then have

$$P_1 \boxplus P_2 = P_3 = (x_3, y_3)$$

where

$$x_3 = -x_1 - x_2 - \frac{b}{a} + \frac{m^2}{a} \quad (2)$$

$$y_3 = -y_1 + m(x_1 - x_3) \quad (3)$$

$$m = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & : \textit{if } P_1 \neq P_2 \\ \dfrac{3ax_1{}^2 + 2bx_1 + c}{2y_1} & : \textit{if } P_1 = P_2 \end{cases} \quad (4)$$

To illustrate the geometrical interpretation of the addition law its best to first consider Case Three. Taking the two points H and P2, we connect them with a straight line. This line is guaranteed to intersect the elliptic curve at one other location - we denote this point as P3 1. We then reflect P3 1 along the x axis, and this gives us our answer P 3 .

The gradient of this connecting line is m, and is calculated using different formulas depending on whether the points are distinct or identical. If the points are distinct the gradient is easily calculated using the standard high school method that leads to 'rise over run'. If the points are identical then we must instead use the tangent line at the point, the gradient of which is found by implicit differentiation;

i.e.

$$m = \frac{dy}{dx} = \frac{3ax_1{}^2 + 2bx_1 + c}{2y_1}$$

Cases one and Two are then variations upon this idea. The point oo can be thought of as lying an infinite distance above (or below) the x axis, so if P2 is oo and P 1 is somewhere on the elliptic curve then the straight line connecting the two is considered vertical. P 3 1 is then directly below (or above) P1, and thus the reflection in the x-axis returns us to P1 • Case Two occurs when the two points have the same x value, resulting in P 3 being oo.

C. Elliptic Curves over Zp

Despite the utility offered by elliptic curves over R!., it is only when one uses them over a finite field lF that they can become the basis for practical cryptographic schemes. This is because we work with finite comput-ers that are unable to accurately store and manipulate the elements of an infinite field such as (R), which has an infinite number of elements, some of which will be infinitely recuring decimals. We hence use finite fields, primarily 7/,P (integers modulo p), where pis a prime.

The Elliptic Curve given in Equation 1 now has the limitation that the coefficients a, b, c, d are integers reduced modulo p, as are the variables x and y. We then define our Elliptic Curve as the set of integer solutions modulo p to equation 1 with the addition of the oo point, and denote it as E (Zp)·

Although the geometrical interpretation of EE that we used further no longer applicable, its algebraic definition can still be used, albeit with a slightly modified computational procedure.

Definition 2. : $\boxplus$ *over* $E(\mathbb{Z}_p)$
Let $P_1, P_2 \in \bar{E}(\mathbb{Z}_p)$, *with* $P_1 = (x_1, y_1)$ *and* $P_2 = (x_2, y_2)$.
Case One:

$$P_1 \boxplus P_2 = \begin{cases} P_1 : \textit{if } P_2 = \infty \\ P_2 : \textit{if } P_1 = \infty \end{cases}$$

Case Two:

$$P_1 \boxplus P_2 = \begin{cases} \infty : \textit{if } x_1 = x_2 \textit{ and } y_1 \neq y_2 \\ \infty : \textit{if } x_1 = x_2 \textit{ and } y_1 = y_2 = 0 \end{cases}$$

Case Three:
If Case One and Two are not met, we then have:

$$P_1 \boxplus P_2 = P_3 = (x_3, y_3)$$

where

$$x_3 = -x_1 - x_2 - \bar{b}\bar{a}^{-1} + m^2\bar{a}^{-1} \quad (5)$$

$$y_3 = -y_1 + m(x_1 - x_3) \quad (6)$$

$$m = \begin{cases} (y_2 - y_1)(x_2 - x_1)^{-1} & : \textit{if } P_1 \neq P_2 \\ (3\bar{a}x_1{}^2 + 2\bar{b}x_1 + \bar{c})\bar{2}^{-1}y_1{}^{-1} & : \textit{if } P_1 = P_2 \end{cases} \quad (7)$$

As with the elliptic curves over lR, an abelian group can be formed by combining E (Zp) with the EE operator of Definition 2. Thus, as an abelian group, (.E (Zp), EE) have the following properties, stated without proof [Silverman, 2005, pg. 98]:

- Identity: $P \boxplus \infty = \infty \boxplus P = P$
- Inverse: $P \boxplus -P = \infty$
- Associativity: $(P \boxplus Q) \boxplus R = P \boxplus (Q \boxplus R)$
- Commutativity: $P \boxplus Q = Q \boxplus P$

for all $P, Q, R \in \bar{E}(\mathbb{Z}_p)$.

While the group $(\bar{E}(\mathbb{Z}_p), \boxplus)$ is not in general cyclic, a cyclic subgroup can be generated with the elements $\{\infty, \pm P, \pm 2P, \pm 3P, \ldots\}$, where P is any point on the EC.

IV. EXISTING SYSTEM

Efficient cryptographic architectures are used extensively in sensitive smart infrastructures. Among these architectures are those based on stream ciphers for protection against eavesdropping, especially when these smart and sensitive applications provide life-saving or vital mechanisms. Nevertheless, natural defects call for protection through design for fault detection and reliability. In this paper, we present implications of fault detection cryptographic architectures

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

(Pomaranch in the hardware profile of European Network of Excellence for Cryptology) for smart infrastructures. In addition, we present low-power architectures for its nine-to-seven uneven substitution box [tower field architectures in GF(33)] which has been proposed by Mehran Mzaffari-kermani and Reza Azarderakhsh in the paper Reliable and Error Detection Architecture of Pomaranch for False Alarm Sensitive Cryptographic Application on 2015, Through error simulations, we assess resiliency against false-alarms which might not be tolerated in sensitive intelligent infrastructures as one of our contributions. We further benchmark the feasibility of the proposed approaches through application-specific integrated circuit realizations. Based on the reliability objectives, the proposed architectures are a step-forward toward reaching the desired objective metrics suitable for intelligent, emerging, and sensitive applications

V. RESULTS

A. MATLAB implementation of Elliptic Curve Diffie-Hellman

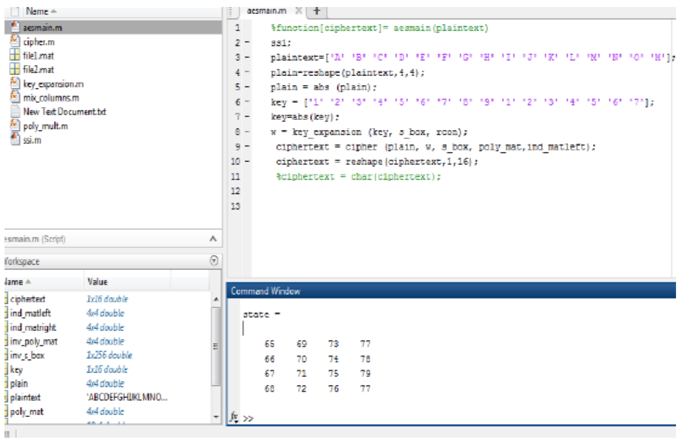
The theoretical underpinnings of ECDH were introduced, and tools to implement this system were developed. We now put these two together to produce a working ECDH key exchange system. Note that the program has been written from the perspective of Alice, with the variable names corresponding to her part in the system. The full MATLAB program is listed below.

The first step in the process is gathering the required data. This consists of the coefficients *a, b, c* and *d* that determine the Elliptic Curve, the prime *p* that specifies the finite field, and a point *P* on this curve. All this information is public, and is to be agreed upon by Alice and Bob. At this stage the program checks that the inputted *p* is indeed prime and, using isecptmod that the given point *P* lies on the Ellip-tic Curve. Alice and Bob then have to each choose a secret number, and this too is inputted into the program. The first of the main calculations can now take place. By using elcmultmod.m the program calculates *A= aP,* and outputs it to the console. Alice can then send this number to Bob.

Once Alice receives *B* from Bob, she inputs it into the console. The program first checks for any transmission errors by verifying that the point lies on the elliptic curve, and then calculates the key *K* = *aB*. The key is then displayed on the console, and the program is complete.

This implementation of ECDH has has been successfully tested with primes ranging up top= 19999423, or roughly 25 bits. While this is nothing near the level 39 of secured in the real world circumstances, it is limited by certain aspects of MATLAB which will be discussed further To aid in the use of this program, a function has been created to locate a random point on the Elliptic Curve to be used as the puplic point *P*. This is performed by first generating a random *x* E *Zp,* then calculating the corresponding *y* value using sqrtmod. m. If no such *y* value exists, another *x* is generated, and the process repeated until one can be found.

B. Simulation results



```
%function[ciphertext]= aesmain(plaintext)
ssi;
plaintext=['A' 'B' 'C' 'D' 'E' 'F' 'G' 'H' 'I' 'J' 'K' 'L' 'M' 'N' 'O' 'M'];
plain=reshape(plaintext,4,4);
plain = abs (plain);
key = ['1' '2' '3' '4' '5' '6' '7' '8' '9' '1' '2' '3' '4' '5' '6' '7'];
key=abs(key);
w = key_expansion (key, s_box, rcon);
 ciphertext = cipher (plain, w, s_box, poly_mat,ind_matleft);
 ciphertext = reshape(ciphertext,1,16);
 %ciphertext = char(ciphertext);
```

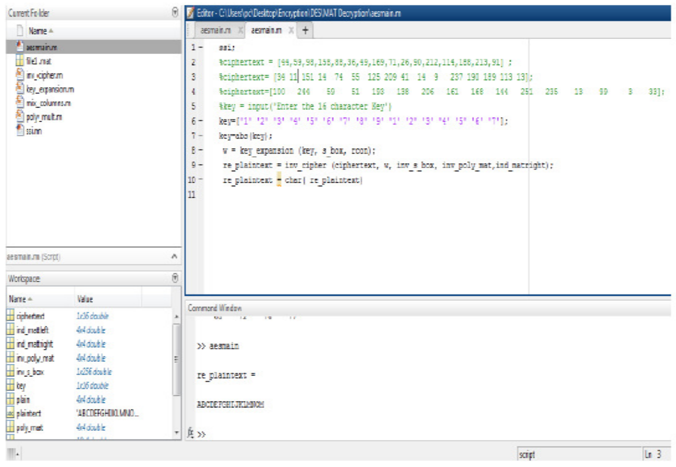
Name	Value
ciphertext	1x16 double
ind_matleft	4x4 double
ind_matright	4x4 double
inv_poly_mat	4x4 double
inv_s_box	1x256 double
key	1x16 double
plain	4x4 double
plaintext	'ABCDEFGHIJKLMNO...
poly_mat	4x4 double

```
state =

    65    69    73    77
    66    70    74    78
    67    71    75    79
    68    72    76    77

>>
```

Fig. 2 Encryption View of Alphabets in to matrix



```
ssi;
%ciphertext = [44,59,98,158,88,36,49,169,71,26,90,212,114,188,213,91] ;
%ciphertext= [34 11 151 14  74  55  125 209 41  14  9   237 190 189 113 13];
%ciphertext=[100   244    59    51   193   138   206   161   168   144   251   235    13    99     3    33];
%key = input('Enter the 16 character Key')
key=['1' '2' '3' '4' '5' '6' '7' '8' '9' '1' '2' '3' '4' '5' '6' '7'];
key=abs(key);
 w = key_expansion (key, s_box, rcon);
 re_plaintext = inv_cipher (ciphertext, w, inv_s_box, inv_poly_mat,ind_matright);
 re_plaintext = char( re_plaintext)
```

Name	Value
ciphertext	1x16 double
ind_matleft	4x4 double
ind_matright	4x4 double
inv_poly_mat	4x4 double
inv_s_box	1x256 double
key	1x16 double
plain	4x4 double
plaintext	'ABCDEFGHIJKLMNO...
poly_mat	4x4 double

```
>> aesmain

re_plaintext =

ABCDEFGHIJKLMNOM

>>
```

Fig. 3 Decryption View of matrix in to Alphabets

Given the limitations of MATLAB discussed above, how is ECC implemented in the real world? The answer lies in the concept of arbitrary precision arithmetic, which allows calculations to be performed on integers of any length. There does not appear to be any packages to implement arbitrary precision 40

Arithmetic in MATLAB, but there are several such packages available for other languages like C/0++ and Java.

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

IJARMATE
Your ulti-MATE Research Paper !!!

International Journal of Advanced Research in Management, Architecture, Technology and Engineering (IJARMATE) Vol. 3, Special Issue 6, March 2017

Hence it is in these languages that real world ECC systems are created.

VI. Conclusions

An Elliptic Curves Cryptography is success fully implemented in simulations by ECDH Key exchange system using MATLAB tool. We can develop an abelian group structure that is then used as the basis for cryptographic schemes such as the ECDH key exchange. The security of these schemes relies on the difficulty of solving the ECDLP.By this method an Encryption and decryption schemes were tested by encrypting the series of alphabets in to matrix form which lies over the elliptic curve then we have decrypted the Matrix by its Key. Over the course of this project I have created a limited but functional implementation of the ECDH key exchange system, using the MATLAB software package. This program allows two parties, Alice and Bob, to generate identical keys with which to then symmetrically encrypt a conversation. If their enemy Eve wishes to eavesdrop on the conversation she must first break the ECDLP. For the relatively small numbers used here this could be done easily, but for real-world programs using very large numbers this would be computationally infeasible.

The limited nature of the implementation is due to the method by which MATLAB stores and works with large integers, and the apparent absence of any package that would allow MATLAB to deal with them differently. However, looking back at the work completed using a SOC Encryption and decryption will be tested over wireless medium. We can see that many of the required algorithms and methods already developed and discussed are independent of the language they are eventually implemented in. Thus if we wanted to create a more effective program capable of handling the large numbers that would ensure security we merely have to 'translate' our current methods into a superior language such as 0/C++, instead of starting again from scratch.

References

[1] Ezra Brown. Square roots from 1; 24, 51, 10 to Dan Shanks. *The College Mathematics Journal,* 30(2):82-95, Mar 1999.

[2] Johannes A. Buchmann. *Introduction to Cryptography. Springer, 2nd edition*, 2004.

[3] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms. MIT Press*, 2nd edition, 2001.

[4] Ben Martin. Elliptic curve cryptography. *Math 409 Notes, University of Canterbury*, 2006.

[5] Federal information processing standard publication 186: Digital signature standard. NIST(*National Institute of Standards and Technology)*,2000.

[6] F. Vercauteren. Elliptic curve discrete logarithm problem. *Lecture Slides, Katholieke Universiteit Leuven*, 2005.

[7] Sebastian Wedeniwski. Primality Tests of Commutator Curves. *PhD thesis*, Eberhard Karls University of Tiibingen, 2001.

[8] Performance Analysis of Elliptic Curve Cryptography on Reconfigurable Hardware, *Proceedings of the World Congress on Engineering* 2008 Vol I, WCE 2008, July 2 - 4, 2008, London, U.K.

[9] Mehran Mzaffari-kermani, Reliable and Error Detection Architecture of Pomaranch for False Alarm Sensitive Cryptographic Application, *IEEE Transactions On Very Large Scale Integration (Vlsi) Systems,* 2015.