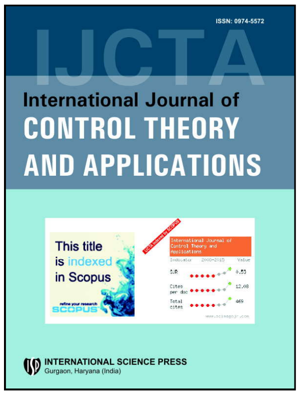


**International Journal of Control Theory and Applications**

ISSN : 0974-5572

© International Science Press

Volume 10 • Number 6 • 2017

# KOM Multiplier for ECC implementation in FPGA

**Aditi Sharma[1] and Rajesh Bhadada[2]**

*[1] PhD Research Scholar Department of Computer Science & Engineering MBM Engineering College, Jai Narain Vyas University,Jodhpur, India, Email: aditi11121986@gmail.com*
*[2] Professor Department of Electronics and Communication Engineering MBM Engineering College, Jai Narain Vyas University,Jodhpur, India, Email: rajesh_bhadada@rediffmail.com*

***Abstract:*** Cryptographic algorithms are having high security for a large amount of data, if their performance can be attained by hardware acceleration as compared to software implantations. We have presented a Karatsuba- Ofman multiplier being developed for generating a secure elliptic curve crypto processor. Modular multiplication of large integers is a key structure for cryptographic algorithms. Karatsuba- Ofman multiplier is based on the divide and conquers methodology. The proposed strategy and performance outcomes in FPGAs for a scalable hardware design. This design is considered as a computing modular multiplication in Galois prime fields GF(p), based on the Karatsuba-Ofman multiplier algorithm.We have also represented the comparison of Karatsuba- Ofman multiplier with the existing multiplier of ECC. ECC multiplication is attained by using a dedicated Galois Field arithmetic simulated on Xilinx FPGA.

***Keywords:*** ECC, Karatsuba-Ofman, FPGA

## 1. INTRODUCTION

In most of the cryptosystems modular exponentiation is used as the powerful operation. It is also considered as the nonlinear scrambling operation. In this operation modular multiplication is done at many times. Modular Multiplication is operated via earliest multiplying then decreasing or else interleaving multiplication with decreased steps. Previous methods are considered as there are very fast multiplication is need, the final method is used where limitation of storage is being considered. In this paper we ponder on specifying Karatsuba – Ofman Multiplier that is proposed by Karatsuba and Ofman. The Karatsuba-Ofman Multiplier or the Karatsuba multiplier is a high-speed multiplier algorithm, exposed by Anatolii Alexeevitch Karatsuba in the year 1960 with availability in 1962. In general multiplication of two m bits digit numbers used to decrease to maximum of single valued digit multiplications. 2m valued digit numbers multiplication operations are reduced with the two m digits additions, multiplications, subtractions, left shifts and m+1 digit multiplications and two 2m digits additions. FPGAs provide constructive performance tool for encryption and decryption algorithms for the implementation where the timely inaccuracy is main aspect for findings and for system on chips where system measures can easily be altered to suit developing security necessities.

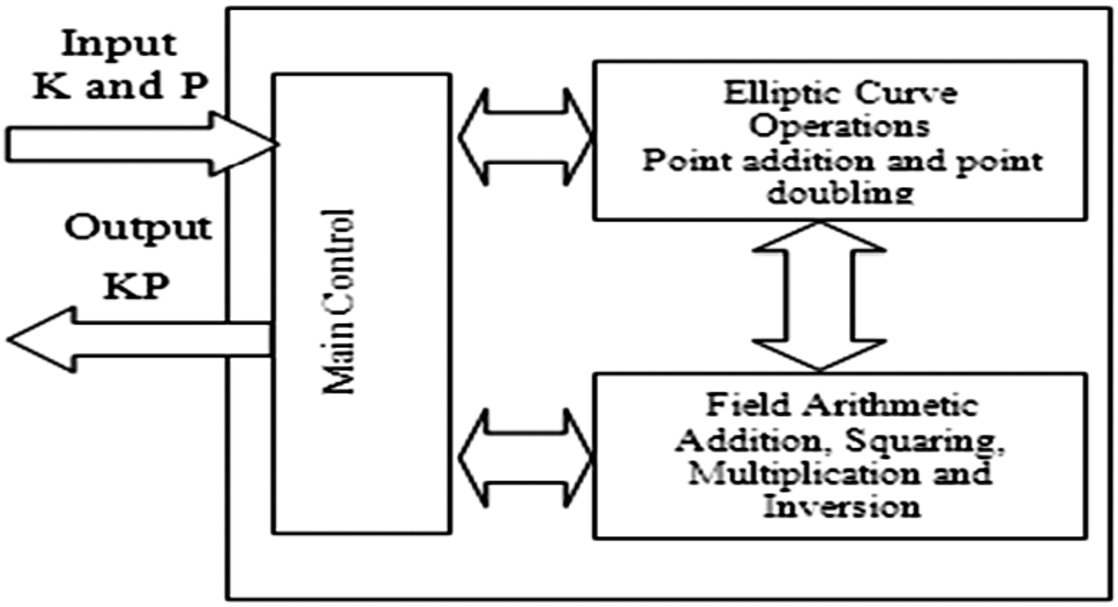

Input
K and P
Output
KP
Main Control
Elliptic Curve Operations
Point addition and point doubling
Field Arithmetic
Addition, Squaring, Multiplication and Inversion

**Figure 1: ALU for ECC Processor**

The rest of the paper is organized as follows. Proposed algorithms are explained in section II with implemented hardware architecture. Simulation Tool with language is explained in Section III. Simulation and Experimental results with comparative studies are presented in section IV. Concluding remarks are given in section IV.

## 2. KARATSUBA – OFMAN ALGORITHM

Let two long integers are A and B. Binary representation of these numbers are as follows:

$$A = \sum_{i=0}^{m-1} a_i \times 2^i,\ B = \sum_{i=0}^{m-1} b_i \times 2^i \tag{1}$$

Here the multiplier product AB is computed effectively. AH and AL, BH and BL equal digits sized are parted by the decomposition of operands correspondingly. These are having the representations as $m$ upper ordered bits and lesser ordered bits of $A$ and $B$ .

Let $y = 2m$. Zero is right padded while y is odd. Multiplier product Z = AB used to be calculated as in equation (4) and equation (5):

$$A = 2^m \left( \sum_{i=0}^{m-1} A_{i+m} 2^i \right) + \sum_{i=0}^{m-1} A_i\ 2^i = A_H 2^m + A_L \tag{2}$$

$$B = 2^m \left( \sum_{i=0}^{m-1} B_{i+m2^i} \right) 2^i + \sum_{i=0}^{m-1} B_i\ 2^i = B_H 2^m + B_L \tag{3}$$

$$Z = AB = \left( A_H 2^n + A_L \right) \tag{4}$$

$$Z = 2^{2m} (A_H B_H) + 2^{2m} (A_H B_L + A_L B_H) + A_L B_L \tag{5}$$

The above equations results the following necessities:-

For calculation of product Z four m-bits multiplications is needed as the requisite of standard multiplication algorithm. Formulation of *T(y)* in equation (6) is done based upon the assumption that T(y) one-bit operations is carried out for multiplication of *y*-bits. For computation of operation of additions and operation of shifts, the count of one bit operations δ*y* is used.

In following equation complexity of multiplication algorithm is set with consideration of*T(1) = 1* However, The computation of Z can be improved by noticing Equation (8):

$$T(y) = 4T(m) + \delta y \tag{6}$$

$$T(y) = \theta\left(y^{\log} 2^4\right) = \theta\left(y^2\right) \tag{7}$$

$$A_H B_L + A_L B_H = (A_H + A_L)(B_H + B_L) - A_H B_H - A_L B_L \tag{8}$$

Following algorithm explains the method of Karatsuba-Ofman's multiplication. Here the number of bits in *A* are |*A*| function ,higher upper half portion number of bits of *A* is returned by *HIGH(A)*, lower half portion number of bits of A is returned by LOW(A), *A*2$^m$ is returned by *RShift*(*A, m*), When *A* and *B* are sized one then *AB* is returned by Onebitmultiplier*(A,B)*. For the extraction of highest half bits and lowest half bits A is right padded with zero, conditioning |A| is odd.

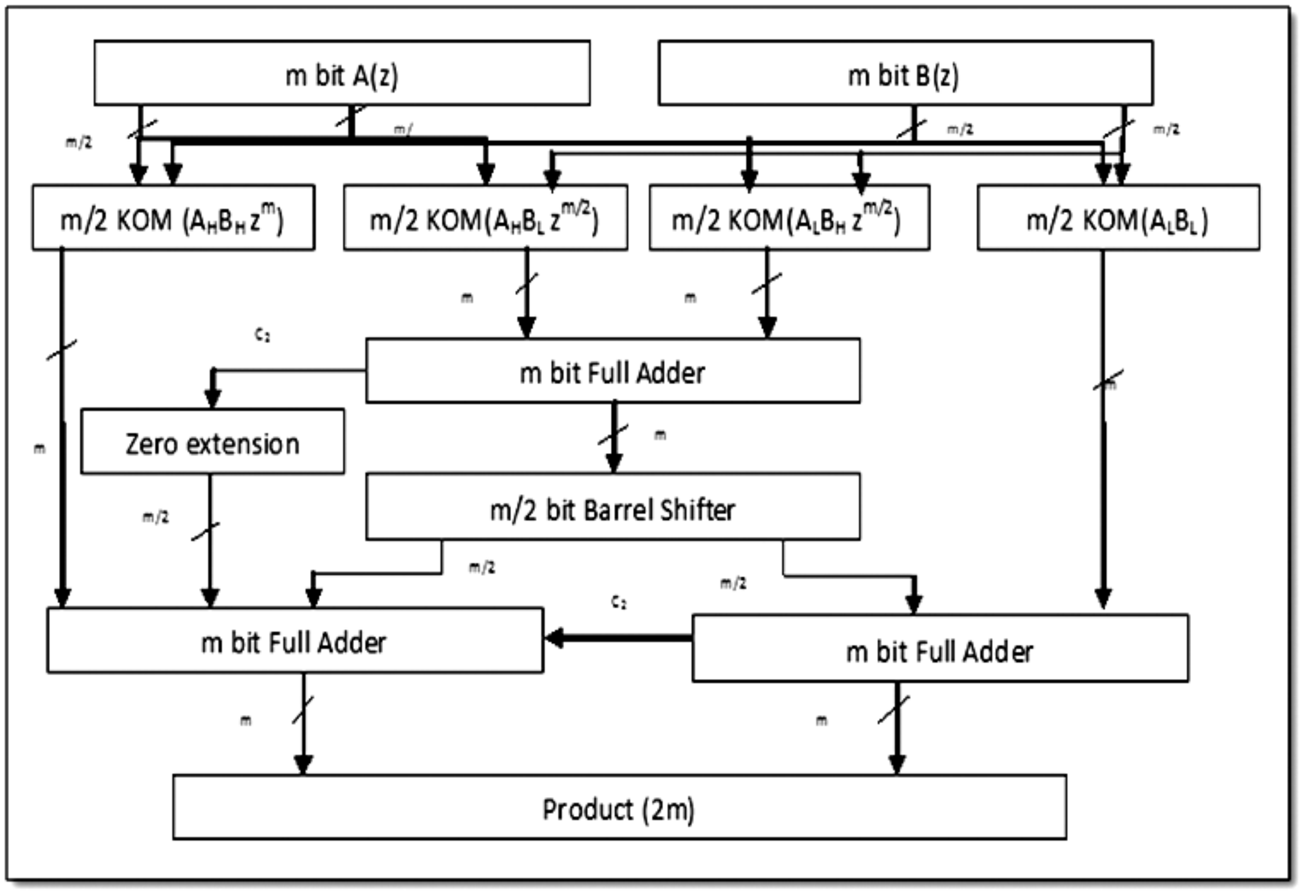


**Figure 2: Karastuba Ofman Multiplier Model**

```
1.1KOM Multiplier algorithm –
Algorithm KOM(A, B)
if |A| = 1
then KOM    := OneBitMultiplier(A, B)
else
    Z1      := KOM(High(A), High(B));
    Z2      := KOM(Low(A), Low(B));
    Z3      := KOM(High(A)+Low(A), High(B)+Low(B));
    KOM     :=RShift(Z1,|A|)+RShift(Z3-Z1-Z2,|A|/2)+Z2;
end.
```

Three m-bits multiplication is used for the computation of Z in KOM(A,B) algorithm. Here addition, subtraction, multiplication are done by ,one bit operations. For getting KOM(A,B) algorithm complexity asymptotically faster we have to consider the value *T*(1) = 1.This comparison is from standard multiplication algorithm.

$$T(y) = 2T(m) + T(m+1) + \delta' y \approx 3T(n) + \delta' y, \tag{9}$$

$$T(y) \approx \theta\left(y \log 2^{3}\right) = \theta\left(y^{1.58}\right) \tag{10}$$

## 3. SIMULATION TOOL & LANGUAGE USED

We have implemented the hardware design using VHDL language and simulated using Xilinx ISE System edition. For electronic device computerization, VHDL and Verilog (Hardware Description language) is used. VHDL is a all-purpose and parallel programming language for the description of FPGA's and IC's, digital and mixed signal systems. VHDL key feature allows synthesisation of required design into the genuine hardware after the subsequently process of behaviour modelling and simulation.

Comprehensive software HDL designs for synthesisation analysis is contrivance produced by Xilinx ISE, Integrated Software Environment. This software tool allows RTL schematic designs to simulate, synthesize and build programs into the target device to perform various timing, RTL diagrams examination, testing analysis. Structure Originator intended for DSP™ is the organisation and industry leading advanced implementation.

FPGAs are used for implantation of excessive-performance Digital Signal Processing systems. The target device is Spartan6 XC6SLX16-CSG324and product version of Xilinx ISE used is 14.3.

**Table 1**
**Karastuba Ofman Multiplier Model**

| | |
|---|---|
| Speed Grade | -5 |
| Delay in Route | 9.438ns |
| Delay in Logic | 11.864ns |
| Delay (Levels of logic =17) | 21.3022ns |

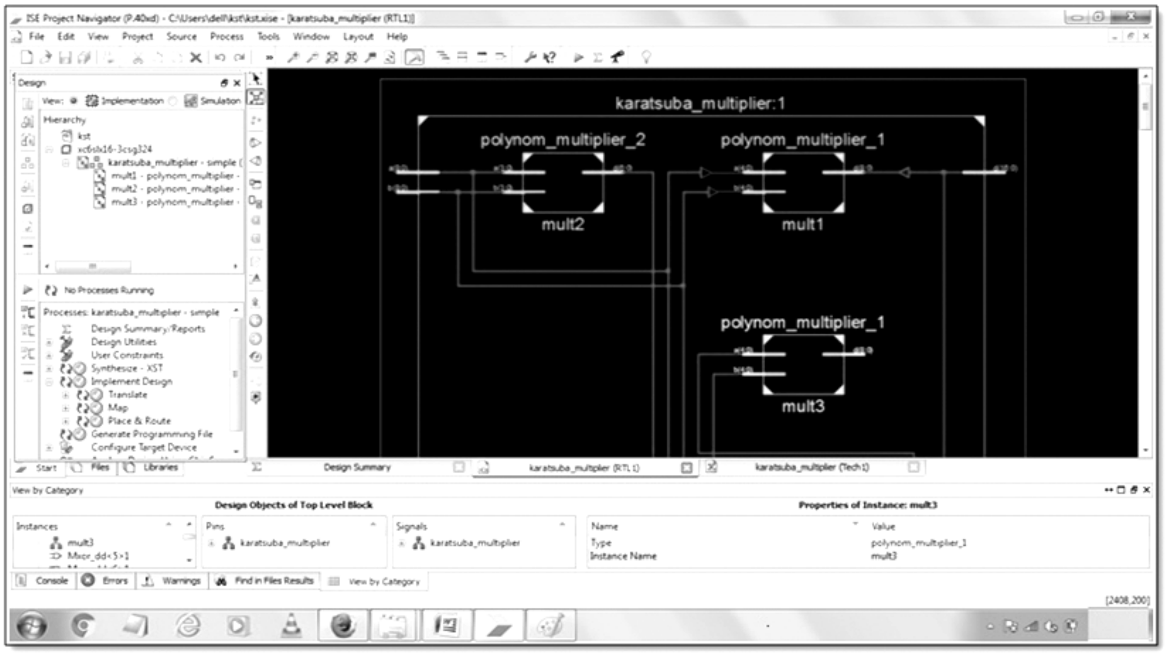
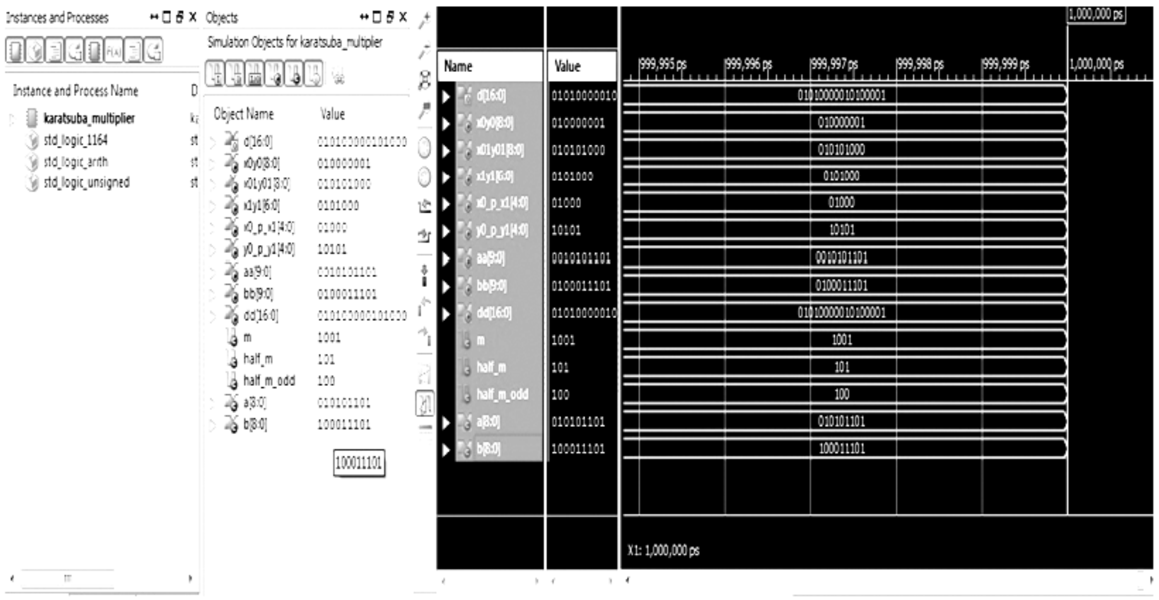


**Figure 3: RTL Schematic Proposed Karatsuba- Ofman Multiplier Model**

## 4. SIMULATION RESULTS

The simulation results are shown for target device is Spartan6 XC6SLX16-CSG324 in terms of timing report and occupied slices for Galois Field 191.

We have used the synthesis tool precision synthesis for the calculation of GF 191 bits which occupies 6265 slices area without implementing clock and timing delay of 0.29 ms. We have compared the results of our proposed multiplier to the existence multiplier which shows the better performance as in Table 2.



| Name | Value |
|---|---|
| d[16:0] | 01010000010100001 |
| x0y0[8:0] | 010000001 |
| x01y01[8:0] | 010101000 |
| x1y1[6:0] | 0101000 |
| x0_p_x1[4:0] | 01000 |
| y0_p_y1[4:0] | 10101 |
| aa[9:0] | 0010101101 |
| bb[9:0] | 0100011101 |
| dd[16:0] | 01010000010100001 |
| m | 1001 |
| half_m | 101 |
| half_m_odd | 100 |
| a[8:0] | 010101101 |
| b[8:0] | 100011101 |

**Figure 4: Simulation Results Proposed Karatsuba- Ofman Multiplier Model**

**Table 2**
**Comparison Result with existing KOM Multiplier Model**

| Ref. | FPGA Device | Field | Occupied slices | Clock (MHz) | Timing delay |
|---|---|---|---|---|---|
| El hadj[7] | Virtex 2600E | 163 | 9581 | Not avail | 2.68ms |
| | | 163 | 1800 | | 5.2ms |
| | | 163 | 7579 | | 3.976ms |
| | | 163 | 1300 | | 4.1ms |
| Smart[8] | XCV 4000XL | 191 | Not avail | Not avail | 17.71ms |
| | | 191 | | | 11.82ms |
| Sakiyama[9] | Virtex II pro | 163 | Not avail | 100 MHz | 0.84ms |
| | | 191 | | 100 MHz | 2.11ms |
| Gura[10] | Virtex II pro | 163 | Not avail | 66.4MHz | 0.143ms |
| | | 193 | | 66.4MHz | 0.187ms |
| | | 233 | | 66.4MHz | 0.225ms |
| Bednara[11] | Virtex XCV1000BG | 191 | Not avail | 50MHz | 3.72ms |
| | | 191 | | 50MHz | 4.07ms |
| | | 191 | | 36MHz | 0.5ms |
| Proposed Work | Spartan6XC6SLX16-CSG324 | 191 | 6265 | Not avail | 0.29ms |
| | | | | 60 MHz | 0.23ms |

## 5. CONCLUSION

In this work, FPGA Implementation of KOM multiplier has been carried out using structural design of VHDL language in Xilinx ISE. We have used Xilinx ISE System edition to perform the simulations for the 8 bit binary numbers. Compared to other multipliers which are being used in the industry, KOM multiplier has low delay levels and high performance which makes it as the best choice by the designers. In this project we have simulated results for the 8 bit binary numbers which consists of four 4*4 KOM multipliers in the hardware level along with summers and barrel shifters for the radix operations. This KOM multiplier is applicable for 191bits as well as 8 bits multiplier which uses very few additions and shifting operations compared to other conventional multipliers which greatly enhance the speed.

## REFERENCES

[1] P. Montgomery, "Modular multiplication without trial division," Mathematics of computation, vol. 44, no. 170, pp. 519–521, 1985.

[2] C. Kaya Koc, T. Acar, and J. Kaliski, B.S., "Analyzing and comparing Montgomery multiplication algorithms," IEEE Micro, vol. 16, no. 3, pp.

[3] E. Oksuzoglu and E. Savas, "Parametric, secure and compact implementation of RSA on FPGA," in International Conference on Reconfigurable Computing and FPGAs, 2008, pp. 391 –396.

[4] A. Karatsuba and Y. Ofman, "Multiplication of many-digital numbers by automatic computers," in Doklady Akad. Nauk SSSR, vol. 145, no. 293-294, 1962, p. 85.

[5] Bhadada, R., & Sharma, A. (2014, December). Montgomery implantation of ECC over RSA on FPGA for public key cryptography application. In *Emerging Technology Trends in Electronics, Communication and Networking (ET2ECN), 2014 2nd International Conference on* (pp. 1-5). IEEE

[6] A. Sharma and R. Bhadada, Security of Database: An approach using Elliptic Curve cryptography in the proceedings of National Conference on New Advances of databases, data mining and Knowledge discovery,(2013).

[7] Youssef Wajih, E.h., Zied, G., Mohsen, M., Rached, T.: Design and Implementation of Elliptic Curve Point Multiplication Processor over GF (2m) IJCSES International Journal of Computer Sciences and Engineering Systems 2(2) (April 2008).

[8] Smart, N.P.: The hessian form of an elliptic curve. In: Koç, Ç.K., Naccache, D., Paar, C. (eds.) CHES 2001. LNCS, vol. 2162, pp. 118–125. Springer, Heidelberg (2001).

[9] Sakiyama, K., De Mulder, E., Preneel, B., Verbauwhede, I.: A Parallel Processing Hardware Architecture for Elliptic Curve Cryptosystems. In:Acoustics,Speech and Signal Processing, ICASSP (May 2006).

[10] Gura, N., Shantz, S., Eberle, H., et al.: An End-to-End Systems Approach to Elliptic Curve Cryptography. In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 349–365. Springer, Heidelberg (2003).

[11] Bednara, M., Daldrup, M., Shokrollahi, J., Teich, J., von zur Gathen, J.: Reconfigurable Implementation of Elliptic Curve Crypto Algorithms. In: 9th Reconfigurable Architectures Workshop (RAW 2002), Fort Laud- erdale, Florida, U.S.A, pp. 157–164 (April 2002).

[12] Yang, Y., Wu, C., Li, Z., & Yang, J. Efficient FPGA implementation of modular multiplication based on Montgomery algorithm. *Microprocessors and Microsystems*, *47*, 209-215(2016).