Realization of Efficient Curve-Based Crypto Processor on FPGA

S.Gayathri, A.Maria Christina Blessy, P.Princy Magdaline
Assistant Professor (ECE), St.Joseph's Institute of technology, Chennai, India.

Abstract – Elliptical curve cryptography plays major role in data security which is more important in today's networking and communication field. The proposal of the project is to realize an efficient GF (2^m) curve-based crypto processor on FPGA. Hybrid Karatsuba multiplier and modifiedItoh Tsujii multiplicative inverse algorithm (ITA) are used for finite field arithmetic operation which increases its efficiency. Compact and faster design can be realized in this proposed method.

Index Terms – Karatsuba multiplier, Itoh Tsujii algorithm (ITA), Field Programmable gate array (FPGA).

1. INTRODUCTION

Elliptic Curve cryptography is one type of public-key cryptography which is based on the algebraic structure of elliptic curves over finite fields. The advantages of ECC over other technique are high level of security and the usage of small keys. In the field of Mobile, Wireless and Network servers, to sustain the high throughput the implementations of high speed crypto-systems are needed. ECC has been extensively used for hardware implementation of FPGA. In 1985 Koblitz and V. Miller [1,2] independently proposed ECC using the group of points on an elliptic curve defined over a finite field in discrete log cryptosystems. The better performance can be obtained in the binary field extension [5] since addition is done by XOR operation where no carry is involved.

The most important operation in ECC is scalar multiplication. It is just calculating *k*P in equation 1.

$$\mathbf{Q} = \boldsymbol{k}\mathbf{P} \qquad (1)$$

Where **Q**- public key which is also a point on the elliptic curve

k- Private key(scalar)

P- Base point on a curve

Elliptic Curve Cryptosystem has three hierarchical layer as shown in Figure 1. The performance of the top layers of the hierarchy depends on the performance of the underlying layers. It is therefore important to have efficient implementations of finite field operations such as squaring, addition, multiplication and inversion. Among these, finite field multiplication and inversion most critically affect the performance of the Elliptic Curve Cryptosystem.

Elliptical curve arithmetic is done by using affine coordinate system. The cost of inversion in affine coordinates is much more expensive so it is done by using projective coordinate representation [6].

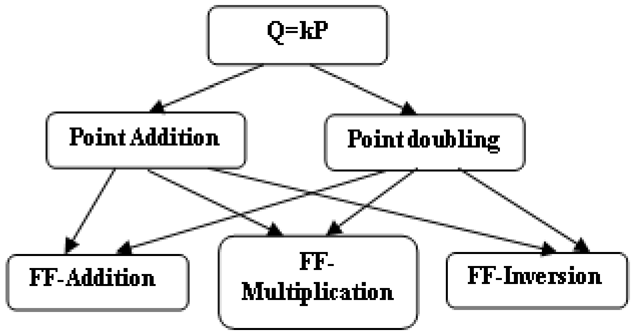


Figure 1: ECCHierarchy

2. MATHEMATICAL REVIEW

Hybrid Karatsubamultiplier [7] is proposed for a high performance elliptic curve crypto processor which requires less amount of space on FPGA.

The Karatsuba algorithm is done by splitting the *n* bit multiplicands into two 2-term polynomials: $A(x)=A_h x^{m/2}+A_1$ and $B(x)=B_h x^{m/2}+B_1$.The multiplication is then done using three $m/2$ bitmultiplications as shown in equation 2. The three $m/2$ bit multiplications are then implemented recursively.

$$C'(x)=(A_h x^{m/2}+A_1)(B_h x^{m/2}+B_1)$$
$$=A_hB_hx^m+(A_h B_1+A_1B_h)x^{m/2}+A_1B_1$$
$$=A_hB_hx^m+((A_h+A_1)(B_h+B_1)+A_hB_h+A_1B_1)x^{m/2}+A_1B_1 \qquad (2)$$

Finite field inversion is commonly done by Euclidean algorithm and Itoh-tsujiialgorithm [8]. Even though the ITA is faster it requires large area which is mainly due to the multiplication unit. All cryptographic applications require toperform finite field multiplications, hence their hardware implementations require a multiplier to be present. This multiplier can be reused by the ITA for inverse computations. In this case the multiplier need not be considered in the area required by the ITA[4].So ITA algorithm is preferred over Euclidean algorithm.

The *Multiplicative Inverse* of an element a∈GF(2^m) is the element a^{-1}∈GF(2^m) such that a a^{-1}≡ a^{-1} a≡1mod(m). From Fermat's Little Theorem, the multiplicative inverse can be written as

$$a^{-1} = a^{2^m-2} = (a^{2^{m-1}-1})^2$$

The implementation of a^{-1} requires *(m-2)* multiplications and *(m-1)* squaring. Addition chains are efficiently usedwhich reduces the number of multiplication in ITA algorithm. An *Addition Chain* for m∈ *N* is a sequence of integers of the form *$U=(u_0,u_1,u_2...u_r)$* satisfying the properties *$u_0=0,u_r=1,u_i=u_j+u_k$ for $k\leq j<i$.* An addition chain for *232* is given by equation 3.

U = (1 2 3 6 7 14 28 58 116 232) (3)

Let $\alpha_k(a) = a^{2^k-1}$ ∈GF(2^m) and $\alpha_{k+j}(a) = (\alpha_k)^{2^j}\alpha_j$. If a ∈GF($2^{233}$) then $a^{-1}= (\alpha_{232}(a))^2$.

3. TOP LAYER

Scalar multiplication (Q=kP)

The basic operations that are performed to compute kP are Point Addition and Point Doubling. The Elliptical curve scalar multiplication (Q=kP) is done by adding k times P, where P is the point on the curve k is an element on the finite field GF(2^m). Algorithm 1 is used to compute scalar multiplication.

Algorithm 1: Elliptical Curve Scalar Multiplier

Input: An integer k≠0 of length l bits and base point P

Output: Q=kP

```
begin
  Q=O
for i=l-2downto 0do
        Q=Double(Q)
        ifk_i=1 then
          Q=Add(Q,P)
        end
end
end
```

4. MIDDLE LAYER

Using affine coordinates point addition operation of two points P=(x_1,y_1) and Q=(x_2, y_2) is the point R= (x_3, y_3) =P+Q where

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

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

$$\lambda =(y_1+y_2)/(x_1+x_2)$$

Point doubling operation is R=(x_3,y_3)=P+P where

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

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

$$\lambda = x_1+y_1/x_1$$

Point addition is performed by using two field multiplications, one squaring and one inversion. Point doubling requires one extra squaring operation. It is well known that projective coordinates avoids the inversion operation in each ECC point addition but it introduces more field multiplication (seven for Point addition and twelve for Point doubling). So, in a hardware implementation, the inversion module is done by using projective coordinates.

5. LOWER LAYER

Finite Field Addition

Finite field addition is performed by using XOR operation so it does not involve any carry.

Finite Field Multiplication

Finite Field Multiplication is done by using Hybrid Karatsuba multiplier which combines Simple Karatsuba multiplier and General karatsuba multiplier.

The advantage of General Karatsuba multiplier is percentage of underutilized LUTs is low. But for large multiplicands, numbers of gates required are high which exceeds the benefit obtained by the full utilized LUTs result in bigger area. So Simple karatsuba multiplier is used for large multiplicands and General Karatsuba is used for smaller multiplicands.

Figure 2 shows the Hybrid Karatsuba multiplier, the initial four recursions are done by using Simple Karatsuba multiplier results in low gate count, while the final recursion is done by using 14 bit and 15bit General Karasuba multiplier which results in low underutilized LUTs.

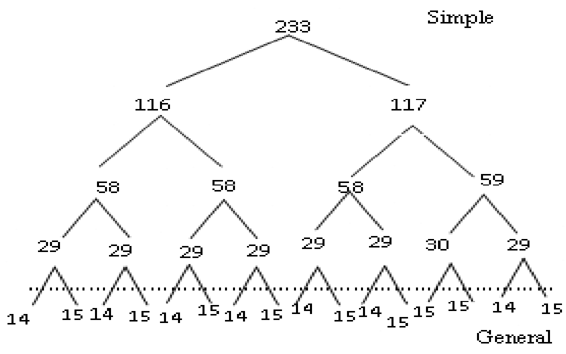


Figure 2-233 bit Hybrid Karasuba Multiplier

	$\alpha_{u_i}(a)$	$\alpha_{u_{j}+u_k}(a)$	**Exponentiation**
1	$\alpha_{u_1}(a)$		a^3
2	$\alpha_{u_2}(a)$	$\alpha_{u_{1+1}}(a)$	$\alpha_1 = a^{4^2-1}$
3	$\alpha_{u_3}(a)$	$\alpha_{u_{2+1}}(a)$	$(\alpha_2)^{4^1}\alpha_1 = a^{4^3-1}$
4	$\alpha_{u_6}(a)$	$\alpha_{u_{3+3}}(a)$	$(\alpha_3)^{4^3}\alpha_3 = a^{4^6-1}$
5	$\alpha_{u_7}(a)$	$\alpha_{u_{6+1}}(a)$	$(\alpha_6)^{4^1}\alpha_1 = a^{4^7-1}$
6	$\alpha_{u_{14}}(a)$	$\alpha_{u_{7+7}}(a)$	$(\alpha_7)^{4^7}\alpha_7 = a^{4^{14}-1}$
7	$\alpha_{u_{28}}(a)$	$\alpha_{u_{14+14}}(a)$	$(\alpha_{14})^{4^{14}}\alpha_{14} = a^{4^{28}-1}$
8	$\alpha_{u_{29}}(a)$	$\alpha_{u_{28+1}}(a)$	$(\alpha_{28})^{4^1}\alpha_1 = a^{4^{29}-1}$
9	$\alpha_{u_{58}}(a)$	$\alpha_{u_{29+29}}(a)$	$(\alpha_{58})^{4^{29}}\alpha_{29} = a^{4^{58}-1}$
10	$\alpha_{u_{116}}(a)$	$\alpha_{u_{58+58}}(a)$	$(\alpha_{116})^{4^{58}}\alpha_{58} = a^{4^{116}-1}$

Table 1: Inverse of a GF(2^m) using Quad-ITA

Finite Field Inversion

Quad ITA is proposed which uses quad circuits (ie rise to the power 4) instead of squarer circuits to obtain the a^{-1}. The procedure for obtaining the inverse of an element using the quad-ITA is shown in Algorithm 2. In the algorithm $\alpha_k(a) = \alpha_k = \alpha^{4^k-1}$.

Algorithm 2: Quad-ITA

Input: The element a∈GF(2^m) and Brauer chain

$$U=\{1,2\ldots.\tfrac{m-1}{2}\}$$

Output: The multiplicative inverse a^{-1}

```
begin

l'=length(U)
    $a^{2=}a*a$
                  $\alpha_{u_i=}a^3 = a^2.a$
for each $u_i \in U(2 \le i \le l')$do
        $p = u_{i-1}$
        $q = u_i - u_{i-1}$
        $\alpha_{u_i=}\alpha_q * \alpha_p^{4^q}$
end
                  $a^{-1} = \alpha_{u_{l'}} * \alpha_{u_{l'}}$

end
```

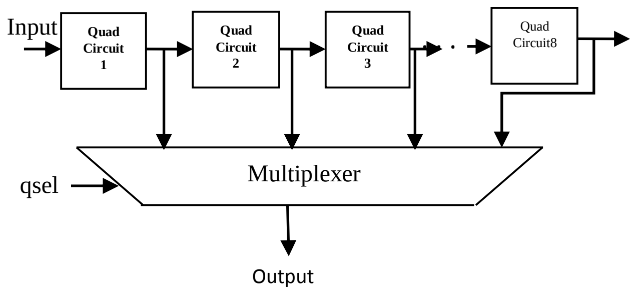
This algorithm computes $a^{-1} = [\alpha_{m-1/2}(a)]^2$.The major advantage of this method is (m-1)/2 addition chains is used instead of (m-1) so the clock cycle is saved and also it uses the FPGA resources better. The Table 1 shows the inverse of a GF(2^m) using QUAD-ITA.

To precompute a³ if the combinational multiplier is used for squarer, two clock cycles are neededso the total clock cycle for the quad block, while $l' + 2$ is the clock cycles used by the multiplier

$$\#Clock\ Cycle_{combinational} = (l' + 2) + \sum_{i=2}^{l'} \left\lceil \frac{u_i - u_{i-1}}{u_s} \right\rceil$$

Quad block

The Quad block consists of cascadedu_s Quad circuits with the input as q_{in} as shown in Figure 3.If the number of power4 required is less than 8 then the multiplier is used to tap out the interim output. If it needs greater than u_s then the output of Quad block is again fed back as the input. q_{sel} will decide which of the eight powers is taken as the output.



The quad block should be designed in such a way that it should be tradeoff between area and clock cycle. If l_p be the LUTs and t_p be the combinational delay for the single quad circuits, then the LUTs and the combinational delay for u_s quad circuits are $u_s l_p$, $u_s\ t_p$.In order to obtain the stable frequency of operation the combinational delay should be less than the maximum combinational delay of the entire design. The maximum combinational delay will occur in Karatsubamultiplier so

$u_s t_p$<Delay of multipliers

The quad circuit should be selected in which the delay should be closer to the multiplier delay. If the quad circuits are reduced the clockcycle is less but the area is more and vice versa. It's found that the best result is obtained with 8 cascaded quad circuits.

The addition chain for the quad block is (m-1)/2 and shortest addition chain is chosen, so that less memory is needed. The memory required for the addition chain is to store the results from intermediate steps.

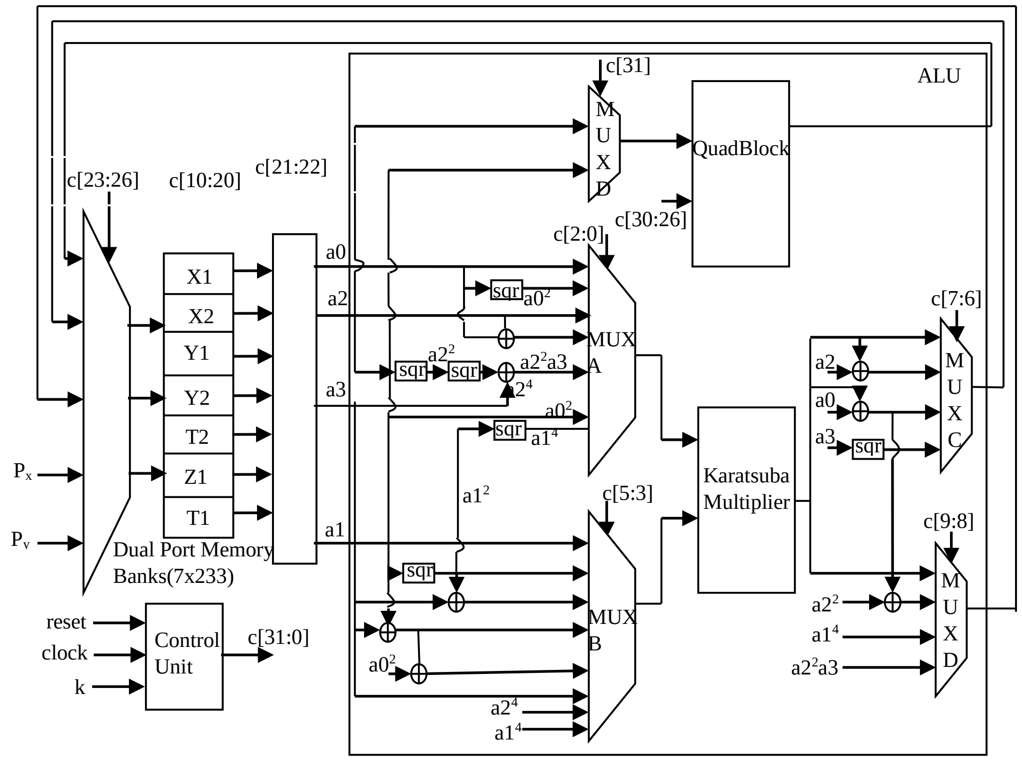


Figure 4Elliptic Curve Crypto Processor

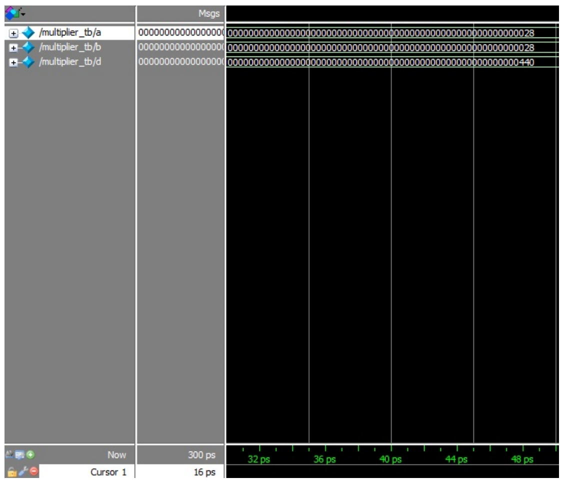
6. ELLIPTICAL CURVE CRYPTO PROCESSOR

The architecture of the processor shown in figure 5 has three main blocks. ALU, Registers andControl unit. The input to the processor is the base point P_x,P_y and the key k and the output is kP. The processor implements the scalar multiplication described in Algorithm. The register bank contain dual port registers whose output is given to the Arithmetic Unit through 5 buses A1, A2, A3, A4, Qin and the input is from the output of the AU through the buses C0, C1, Qout. Elliptical curve operations are controlled by the control signals(c[0]…c[32]) generated every clock cycle.

The Arithmetic coordinate implement Point Addition and point Doubling in Projective coordinate. Quad block uses 8 Quad circuits which is suitable for the Processor to reduce the clock cycle and the area.

7. EXPERIMENTAL RESULTS

Figure 5 and 6 shows the simulation result of 233 bit hybrid Karatsuba multiplier and ALU. The ALU is synthesized using Xilinx ISE synthesis tool (Version 11.1) and the LUTs, Delays(Table 1) forGF(2^{233}) binary field with irreducible trinomial specified in NIST Digital Signal standard are compared for various devices.



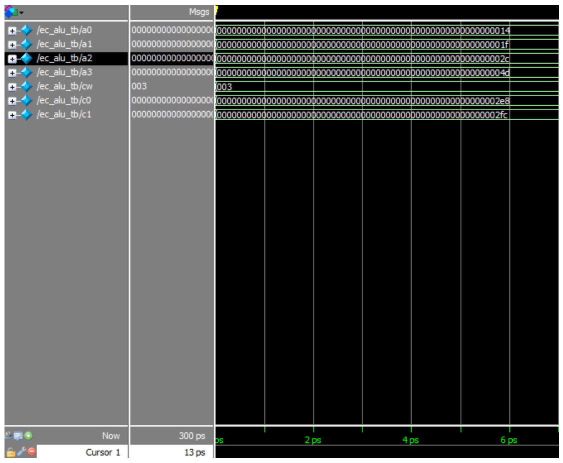
```
                    Msgs
/multiplier_tb/a    0000000000000000  000000000000000000000000000000000000000000000000000000000028
/multiplier_tb/b    0000000000000000  000000000000000000000000000000000000000000000000000000000028
/multiplier_tb/d    0000000000000000  000000000000000000000000000000000000000000000000000000000440

Now                 300 ps            32 ps   36 ps   40 ps   44 ps   48 ps
Cursor 1            16 ps
```

Figure 5 Simulation result of 233 bit Hybrid Karatsuba multiplier

Devices	LUTs	Delay
Virtex4	25264	28.465ns
Virtex5	17119	15.614ns
Spartan3	25264	30ns
Automotive Spartan 2E	25264	52.29ns

Table 2 Comparison of ALU for various Devices



```
/ec_alu_tb/a0  00000000000000000  000000000000000000000000000000000000000000000000000000000014
/ec_alu_tb/a1  00000000000000000  00000000000000000000000000000000000000000000000000000000001f
/ec_alu_tb/a2  00000000000000000  00000000000000000000000000000000000000000000000000000000002c
/ec_alu_tb/a3  00000000000000000  00000000000000000000000000000000000000000000000000000000004d
/ec_alu_tb/cw  003                003
/ec_alu_tb/c0  00000000000000000  0000000000000000000000000000000000000000000000000000000002e8
/ec_alu_tb/c1  00000000000000000  0000000000000000000000000000000000000000000000000000000002fc
```

Figure 6 Simulation result of ALU in the ECC Processor

8. CONCLUSION

Elliptical curve cryptography Processor can be efficiently implemented on FPGA by using Hybrid Karatsuba multiplier and modified ItohTsujii inverse algorithm. The Hybrid Karatsuba multiplier can be used in Elliptic Curves to minimize the LUTs required and increase the operating frequency. Quad block with 8 cascaded Quad circuits for Itoh Tsujii algorithm for inversion will give better performance in area and clock cycles.

REFERENCES

[1] Neal Koblitz. Elliptic Curve Cryptosystems.*Mathematics of Computation*,48:203–209, 1987.

[2] Victor Miller. Uses of Elliptic Curves in Cryptography.*Advances in Cryptology, Crypto'85*, 218:417–426, 1986.

[3] Donald E. Knuth. *The Art of Computer Programming Volumes 1-3 Boxed Set*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA,1998.

[4] Chester Rebeiro, SujoySinha Roy, D. Sankara Reddy, and DebdeepMukhopadhyay*, Revisiting TheItoh-Tsujii Inversion Algorithm For FPGA Platforms*,IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, VOL. 19, NO. 8, AUGUST 2011

[5] U.S. Department of Commerce,National Institute of Standards and Technology. *Digital Signature Standard (DSS)*, 2000.

[6] Julio López and Ricardo Dahab. Fast Multiplication on Elliptic Curves over GF(2^m) Without Precomputation. In *CHES '99: Proceedings of the FirstInternational Workshop on Cryptographic Hardware and EmbeddedSystems*, pages 316–327, London, UK, 1999. Springer-Verlag.

[7] Chester Rebeiro and DebdeepMukhopadhyay, *Hybrid Masked Karatsuba Multiplier For Gf(2^{233})*

[8] Toshiya Itoh and Shigeo Tsujii. A Fast Algorithm for Computing Multiplicative Inverses in GF (2m) using Normal Bases. *Inf. Comput.*, 78(3):171–177, 1988.

[9] J. Guajardo and C. Paar, "Itoh-Tsujii inversionin standard basis andits application in cryptography and codes," *Des.Codes Cryptography*,vol. 25, no. 2, pp. 207–216, 2002.

[10]Francisco Rodríguez-Henríquez and Çetin Kaya Koç. On Fully Parallel Karatsuba Multipliers for GF(2m). In *Proc. of the International Conferenceon Computer Science and Technology (CST).*