ACADEMIA

Accelerating the world's research.

Design of RSA Processor and Field Arithmetic of ECC with Vedic Multipliers for Nodes in Wireless Sensor Networks

IOSR Journals

Related papers

Download a PDF Pack of the best related papers 🗹

	- 10
rango Banago	
discoverus	
	atio

An Optimized Multiplier Architecture for a Dual Field Processor For ECC Journal of Computer Science IJCSIS

Performance Evaluation of Projective Binary Edwards Elliptic Curve Computations with Parallel Archit... Qasem Abu Al-Haija, Mohammad Alkhatib

Hardware architectures for public key cryptography Bart Preneel

Design of RSA Processor and Field Arithmetic of ECC with Vedic Multipliers for Nodes in Wireless Sensor Networks

Leelavathi G, Shaila K, Venugopal K R

Electronics and Communication Engineering VTU-Research Centre, Vivekananda Institute of Technology, Bengaluru, India Principal University Visvesvaraya College of Engineering, IEEE Fellow, Bengaluru, India Corresponding Author: Leelavathi G

Abstract: In Wireless Sensor Nodes due to the resource constraints the fast multipliers are preferred for data processing. In this paper, the RSA processor using Vedic multiplication technique is proposed which is capable of achieving considerable speed and with minimum area utilization. The multiplication of two prime numbers is implemented using Nikhilam and UrdvaTriyagbagam multipliers. The results shows that there is good improvement in delay and device utilization usingUrdvaTriyagbagam method. UrdvaTriyagbagamis utilized in Point addition and Point doubling, which are finite field arithmetic of ECC in both prime and binary field. Multipliers are implemented on RSA and ECC over NIST/SECG GF(p) and GF(2m) curves and estimates the algorithms with respect to performance in speed and memory usage.

Keywords: Ellptic Curve Cryptography, FPGA,NikhilamMultiplier, RSA algorithm, Urdva-tiryagbhyam Multiplier, Wireless Sensor Networks.

Date of Submission: 05-01-2018

Date of acceptance: 03-02-2018

I. Introduction

The increase in application of Wireless Sensor Networks (WSNs) in environments like home, military and commercialrequiresspeeding up of the data processing in the network[1].Cryptography is a practice for building the message secure with encryption and decryption processes.ECC and RSA are the public-key algorithms that have been investigated by the research ommunal for several years. The RSA was developed by Rivest, Shamir and Adleman in 1976.Koblitz and Miller available work on ECC in 1985. Modular exponentiationis the major operation underlying in RSA and its security isin question due to its difficulty of factoring large integers. ECC operates on groups of points over elliptic curves and originates its security from the hardness of the elliptic curve discrete logarithm problem (ECDLP). The Public Key Cryptography (PKC) involves cryptographic processes which are computationally intensive and sensor nodes are strictly resource constrained. In encryption systems multipliers play important role. High speed systems with low power consumption and time delay mainly depends on multiplier execution time. Compared to conventional multiplication, Vedic technique of multiplication involves very less number of operations resulting in fasterand high performance multiplier[2-4]. Compared to RSA, ECCbetter suited for WSNs, since ECC off ers smaller key sizes, performs faster computation, as well as memory, energy and bandwidth are saved.

The construction of Vedic multipliers is based on Vedic Sutras. The Vedic mathematics has been distributed into sixteen different Sutras and their methods are comparable to the working of human mind which is capable of condensing the complex calculations into simpler ones, consuming less power and attains lower chip area[4]. In this work, the multiplication is computed using NikhilamandUrdva-Tiryagbhayam multiplication methodologies for RSAalgorithm that improves the performance in terms of area and speed. Nikhilamincludes minimum number of steps, space, time saving and cerebral calculation. *Urdhva – Tiryagbhayam* formula is appropriate to all cases of multiplication and division of a large number by another large number. This is based on "Vertically and Crosswise" technique andcreates almost all the numeric computations faster and easier. The benefit of multiplier developed on this sutra is that with the rise in the number of bits, area and delay increase at a smaller rate. So this Sutra is used for design of field arithmetic's factor.

Motivation: Security is generally expensive and thespeed is more obviousin WSNs dueto limited resources of the sensor nodes. In orderto increase the speed while appropriately utilizing the available resources, it is important to use multipliers which have better performance with respect to security algorithms. Vedic multipliers allow the encryption and decryption of messages to speed up RSA algorithm and finite field arithmetic of ECC.

Organization: The security techniques, public key cryptography and RSA cryptography, different multiplication algorithms are described in Section II. In Section III, Problem definition, Processor Model, Implementation of Vedic multiplier techniques, RSA Cryptography together with Algorithm and Performance Evaluation are discussed. Section IV comprises Conclusions.

II. Related Work

Shaila et al., [2-3]designs a key predistribution scheme for WSNs. Tiwari et al., [5]proposed Vedic multiplier constructed on algorithm of ancient Indian Vedic Mathematics, for low power and high speed applications and is implemented on ALTERA Cyclone –II FPGA.Kunchugi et al.,[6]discussed the efficiency of UrdhvaTriyagbhyam Vedic method for multiplication that takes11 logic cells for nibble multiplier and propagation time of 4.585ns. Based on the formulas of ancient Indian Vedic Mathematics, a novel complex number multiplier ASIC design that are highly suitable for high speed complex arithmetic circuits is discussed in [7][8].

Hudder et al.,[9]presented a novel architecture to accomplish high speed multiplication using compressors with ancient Vedic mathematical techniques. Kumaravel et al.,[10] examines modular multiplication using Vedic mathematics that increases the speed of performance. The multipliers are implemented with Karastuba-ofman and Booth algorithm considering delay as a parameter. In [11-13], authors implemented Vedic multiplier and compared their work with Booth and array multipliers.

Pritam et al., [14] proposed algorithms based on 1's complement subtraction which will remarkably improve the computational efficiency of scalar multiplication. Soramet al., [15-17]experimentallyevaluated the performance of RSA and ECC. Shaila et al., [18] discussed about constraints of WSNs and proposed a scheme called Modified Blooms Scheme (MBS). Thapiyal et al., [19] utilizes Vedic Mathematics to implement Elliptic Curve Encryption that speeds up the task of multiplication process. Houssain et al., [14] elaborated on the study of hardware implementations of ECC in Wireless Sensor Networks. Both Binary and Primary field are used for the implementation of point addition and point doubling for ECC. Our proposed work provides moderate level of security for resource constrained devices.

III. Problem Definition And Model

A. Problem Definition

In a given WSNs, encryption of data is performed before it is transmitted. This is achieved by using multiplication techniques. Multipliers are the main source in high speed arithmetic logic units. A particular multiplication technique in encryption algorithm is not able to deliver all the desired performance properties for nodes in WSNs, so Public-key cryptography is adopted. Public-key cryptography iscomputationally expensive for WSNs if not accelerated by cryptographic hardware.

B. Objective

The main objective of this work is to design and develop efficient multipliers that increase the performance of RSA and ECC processor with Vedic multiplication methods.

C. Algorithm for RSA Processor

The complete process involves three phases.

```
Phase 1. Key generation
          Selection of prime numbers p, q
a.
         Computation of n = p * q .....(1)
Computation of phi = (p - 1) * (q - 1).....(2)
b.
c.
          Choose public key e, satisfying the condition
d.
1 < e < phi, gcd(e, phi) = 1......(3)
                                       private
          Finding
                          the
                                                       key
                                                                    d,
e.
     d * e
f.
          Private key = (d, n)
          Public key = (e, n)
Phase 2. Encryption
C = M^e mod n \dots \dots
                                .....(5)
Where C is cipher text and M is plaintext
Phase 3. Decryption
M = C^d \mod n.....(6)
```

D. ECC Arithmetic

ECC is computationally intensive since it includes arithmetic operations in finite fields. An elliptic curve cryptosystem functions in a group of points on an elliptic curve defined over a finite field. To improve performance most practical ECC implementations are binary extension fields $GF(2^m)$ and prime fields GF(p). The prime field types permit efficient software implementation, particularly on processors furnished with a fast integer multiplier. For hardware implementation, binary extension fields $GF(2^m)$ are commonly the superior choice.

Due to the computational expenditure of inversion associated to multiplication, projective coordinate methods have been projected which circumvents the inversion operation. Point arithmetic based on the Lopez - Dahab coordinate(LD) is effective for hardware implementation. In this work, Scalar multiplication operations are to be carried out with point adding in mixed coordinates *i.e.*, one point is in affine and another point is in L-D projective coordinate and point doubling in L-D projective coordinates in the binary field. Both point addition and doubling operations are foundation of scalar multiplication. The representation considered in this paper is a normal base that is most suitable for hardware implementation. An elliptic curve over GF (2^M) is defined as the cubic equation E:

$$y^2 + xy = x^3 + ax^2 + b$$
 (7)

Where a, b, x, $y \in GF(2^M)$ and $b \neq 0$. The Weiestrass equation defining an elliptic curve over GF(p):

$$y2 \pmod{p} = (x3+ax+b) \mod p \tag{8}$$

Where x, y are defines elements of GF(p), and a, b are integer modulo p,

IV. RSA Processor

E. RSA Processor- Design Unit

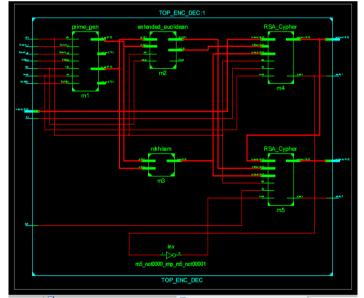


Fig.1 Schematic Diagram of theComplete Processor

Figure 1 shows the complete model in which both private key and public keys are generated. The public key of the sender is distributed over the network that is utilized for decryption process. The encryption process uses the private key of the sender, provides more security for data through digital signature and authentication.

The RSA processor shown in Figure 1 is modeled using Verilog and synthesized on FPGA device Spartan 3 XC3S400-5pq208 for software and hardware implementation. The encryption block (RSA_Cipher m4) computes cipher $C=M^e \mod n$. The message (*M*)in binary form is taken as input along with public key (*e*) and modulus (*n*). The decryption block (RSA_Cipher m5) decodes user message (*M*) from cipher (*C*) by

computing $M = C^d \mod n$, where, dis the private key. Hence, the original message which is in the binary format is recovered safely.

F. Simulation and Hardware Results

Simulation results of Complete RSA processor is illustrated in this section. The primality tester outputs for p,qare shown in Figure 2.

Messaç	jes		
+ 🔷 /nkhilam/res	144	200	144
🛤 🍫 /vikhilem/x	12	10	12
🖬 🍜 /mkhilam/y		20	(12
🗖 🌙 /niktilan/x1	00001100	20010100	00001100
🖬 🍲 /nktslam/y1	00001100	00001010	00001100
🖅 /nitrilam/x2	00000100	00001100	00000100
🖬 🍏 /niktiliam/y2	00000100	20000010	(00000100
🗖 🍲 /hikhilani/x3	00000000	100001010	00000000
🖬 🌍 /nittilam/y:3	00000000	20000000	00000000
The Avenue and	2000000	20000000	

Fig 3. Result of Nikhilam Multiplier

Nikhilam and Urdhva – Tiryagbhayam multiplier results with 8 bit data is shown in Figure 3 and Figure 4.

- A fund out to 120	
🖅 🔶 /mul_8x8/b 150 25	
E	
<u>mu_Sx8/m1</u> 00110000 01010001 01010001	
r → /mul_8x8/m2 00101010 00001001	
<u>-</u> / /mul_8x8/m4 00111111 00000001	
E-\$/mul_8x8/sum3 01000110 00000010	
/mul_8x8/c1 St0	
/mul_8x8/c2 St0	
/mul_8x8/c3 St0	
✓ /glbl/GSR We0	

Fig 4. Results of Urdhva – Tiryagbhayam

📭 🥠 /euclid/m_in	20	20		i i i i i i i i i i i i i i i i i i i
🖬 🍫 /eudid/n_in	15	15		
🖬 🛶 /eudid/gcd	5	0	_	(5
🤣 /euclid/clk	St0	ويعير الكلالة ويهير		
🥠 /euclid/rst	StO			ا السما ک
🖬 🍫 /euclid/m	000000000000000000000000000000000000000	000000000000000	000000000000000000000000000000000000000	
🖬 🍫 /euclid/m_next	000000000000000000000000000000000000000	0000000000010100		
🖬 🍫 /euclid/n	000000000000000101	000000000000000		
🖪 🛷 /eudid/n_next	000000000000000000000000000000000000000	000000000001111	والمحالية المحالية المحالية	
🤣 /euclid/done	1			
🤣 /euclid/done_next	1			
🖅 🔶 /eudid/factor	000000000000000000000000000000000000000	000000000000000		
	000000000000000000000000000000000000000	000000000000000		
🛨 🔷 /euclid/state	01	00		
/euclid/state_next	10	01	والمست زمسان	10

Fig 5. Extended Euclidean algorithm output for GCD

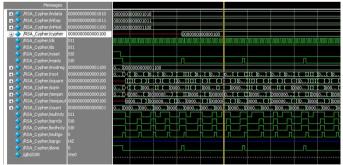


Fig 6. Simulation Results of RSA Processor

The Euclid algorithm gives the proof that the GCD of encryption key and *phi* in the RSA algorithm is one *i.e.*, Euclid algorithm takes the two input values and calculates its gcd, if the gcd is one it gives the output '1' else it gives the gcd of two numbers but it discards the output. The results are provided in Figure 5.

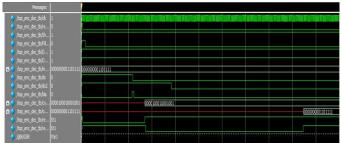


Fig 7.Simulation Result with Input, Output and Recovered Input

Figure 6 and Figure 7displays the complete RSA processor results with encryption and decryption of data. The message(M), encryption key(e) and modulas value(n) are observed as Indata, InExp and Inmod respectively. From the timing diagrams the time required for conversion to cipher text and convert back to original message can be observed.

The Hardware implementation of RSA processor on Xilinx Spartan III FPGA development board is shown in Figure 8. When the rst=0 and fillsel=0, the ready signals should be '1'.

The input 8-bit data that is generated by the LFSR internally for the encryption along with the product result of the Vedic multiplier *i.e.*, modulus value and a message (16'd0000000010111101) isprocessed. The output of the encrypted data is obtained as (0000011111111100). For decryption of data a decryption key, the encrypted data and modulus is required. The output is found as (16'd0000000010111101) hence the original data is obtained and verified.



Fig 8. Hardware Implementation of the RSA Processor

Bus/Signal	X	0	0	160	320	480	640	800 	960 .	1120 1	1280 1	440 1	600 1	760 19	20 20 1	180 22: 	40 240	10 256	0 272	0 2880) 3040) 3200) 3360 	3520	3680 : 	3840	4000
∾ cypher	0000	0000:												0000	10001	00010	01										
🏻 outdata	0000	0000			000000001101111)																						
- ready1	1	1																									
- ready2	0	0																									

Fig 9. Ciphered Output ThroughChipscope

Figure 9 shows the ciphered output, through chipscope for the input data 16'd000000001101111, the encrypted output obtained is 16'd0000100010001001. The original data is retraced using the decryption key with output 16'd0000100010001001.

G. Performance Analysis

In [12] author reports delay from different implementations that varies from 23.18 η seconds to 32.01 η seconds. From Table 1 and Figure 10, it is observed that Nikhilam occupies more area than Urdhva, but the delay is 50% less and power consumption remains the same for both the multipliers.Considering less delay, Nikhilam multiplier can be chosen for implementation, whereas with area the Urdhva performs better. The device utilization on Spartan 3 FPGA, compared with [11] is given in Table 2, shows less device utilization in our implementation.

The multiplier can be selected depending on the type of application. For nibble multiplication, 11 logic cells are utilized and propagation delay is 4.585 η seconds in [6]. The Urdva multiplier occupies 9% of area [9] which is 75% more than our implementation.

I able 1. Perfo	rmance Comparis	on of Multipliers
Parameters/Technique	Nikhilam	Urdhva – Tiryagbhyam
Area	315 slices	94 slices
Delay	12.832nanosec	26.391nanosec
Power(Static)	0.060watts	0.060watts

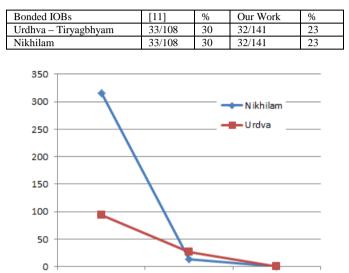


Table 2. Device Utilization Comparison of Multipliers

Observing the results obtained from Table 1-4, it can be concluded that Urdhva gives the best device utilization, with higher frequency and no delay is found. So the Urdhva is selected for the implementation of ECC arithmetic operations, point addition and point doubling.

Fig 10.Comparison of Constraints

delay(n sec)

Power(Static)watts

Area(slices)

Device	Parameters	Nikhilam	Urdhva
Artix 7	Slice Registers	0%	0%
	LUTs	2%	1%
	Frequency	38.806Mhz	194.76Mhz
	Path delay	No	No
Kintex	Slice Registers	0%	0%
	LUTs	0%	0%
	Frequency	50.698Mhz	228.760Mhz
	Path delay	No	No
Virtex	Slice Registers	28%	23%
	LUTs	9%	10%
	Frequency	15.110Mhz	62.888Mhz
	Path delay	No	No
Spartan 3	Slice Registers	0%	0%
	LUTs	0%	0%
	Frequency	43.679 Mhz	202.360 Mhz
	Path delay	No	No

Table 3.Comparison of Multipliers for Device Utilization

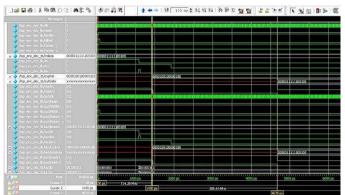


Fig.11 Analysis of Encryption and Decryption Timing

From the Figure 11 the time required for encryption and decryption with 16 bit data is calculated as below. Encryption Timing = (time position of cursor2) - (time position of cursor1)

= (1430 - 30) ps

$$= 1430 \text{ ps}$$

Decryption Timing = (time position of cursor3) - (time position of cursor2)

= (4670 - 1430) ps= 3240 ps

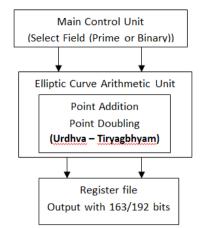
From the simulation results obtained with 16 bit data Encryption decryption timings remains same for Nikhilam and Urdhava. The time required is interms of picoseconds and very less compare to the previous results from literature survey.

Device	Parameters	RSA Encryption and Decryption
Artix 7	Slice Registers	6%
	LUTs	30%
	Frequency	38.806
	Path delay	No
	Static Power	0.107Watts
Kintex	Slice Registers	0%
	LUTs	1%
	Frequency	50.698
	Path delay	No
	Static Power	0.148 Watts
Virtex	Slice Registers	0%
	LUTs	0%
	Frequency	50.698
	Path delay	No
	Static Power	0.671 Watts
Spartan 3	Slice Registers	28%
	LUTs	9%
	Frequency	15.110
	Path delay	No
	Static Power	0.060 Watts

Table 4. Comparison with RSA encryption and decryption

V. ECC Finite Arithmatic Model

The ECC finite arithmetic model architecture mainly consists of the main control unit which contains a control signal which is responsible for selecting the Binary or Primary field. Then it is followed by the elliptic curve arithmetic unit which is used to perform the point addition and point doubling with the aid of Vedic multiplier (Urdhavtriyagbhayam). The architecture is first implemented with 8 bits of input data with dual field. One field is selected at a time then both point addition and point doubling are performed. Further, the elliptic curves defined over the binary field and primary field $GF(2^{163})$ and $GF(2^{192})$ respectively. Urdhavtriyambhayamtechnique is used for point addition and point doubling operations. The block diagram for point addition and point doubling in binary field and prime field (dual field) using Urdhavtriyagbhayamis given in Figure 12. The proposed design has been coded in Verilog HDL, simulated by ModelSim and implemented on Virtex XC5VLX50Tdevice. Figure 13 shows Dual field addition and doubling simulation results for 8bits.





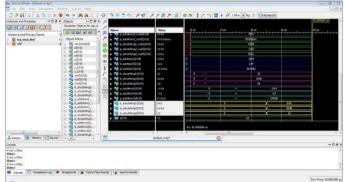


Fig 13. Dual field addition and doubling simulation results(8 bits)

H. Point Addition and Point Doubling Over Binary Field

This point addition requires mainly two points one is in projective coordinate and another point affine coordinate, the outcome will be in the projective directions.

The algorithm shown in Figure 14, is used to perform the point addition over the binary filed using Lopez Dahab mixed coordinates, since it uses two points one is of affine, second point is of projective coordinates.

Inputs:
$$A(x_2, y_2)$$
, $Q(X_4, Y_4, Z_4)$,
Outputs: $R(X_3, Y_3, Z_3)$.
 $A=Y_4+y_2*Z_4^2$;
 $B=X_4+x_2*Z_4$;
 $C=B*Z_4$;
 $Z_3=C*C$;
 $D=x_2*Z_3$;
 $E=A+B*B+aC$;
 $X_3=A*A+C*E$;
 $I=D+X_3$;
 $J=A*C+Z_3$;
 $F=I*J$;
 $K=Z_3*Z_3$;
 $Y_3=F+x_2*K+y_2*K$.

Fig 14. Algorithm for Point addition in Binary Field

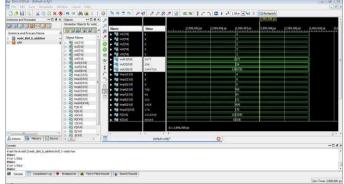


Fig 15. Binary field addition (8 bits)

Project top_dual_filed JTAG Chain + DEV/D McDeviceD (ICGVLX507) System Monitor Console	Mab	th Unit		T) UNET O Mo Function	11.60 (ILA)			Value		100	Radit Bin		Counter	ේ හි ම
Unit' D M(d, AD (E, A) Trigger Setup Waveform Lusting	Add OH	Adive *				Condition I perConditio		J;			Trigger Con	96on Equation NO	17	l;
Signals: DEV: 0 UNIT: 0	Type: Window	-	Windows:			4	Depth: 10	24		-	Position:		0	
Data Port p doubling3	SS Waveform - DEV	10 March 10	ICANI KAUTA	NTO INI A	0-01-01		-							o' 13' 18
p_doubling2	Bars/Sional								0 520 560				40 880 920 5	
p_doubling1	Englithe d	0 0	- and made	hund	b.uch	t	ليسابسنا	miland		ماسيات	سليستليه	dimilian	hadaadaa	- Lundar
p_addition3	>b additon2	0 0	-						-	_				
p_addition1 = 0_dexb@mo3	and the second second second	10000	-											
+ b_doubling2	- b_doubliseg1	1 7 K (26)	-						32					
D_doubling1 = 0_admon2	> b drublling2	1568 1568	-						1559.					
Image: b_addion3	← b_drubling3	34 38							16					
- 0_addion1	> p_addition1	0000 0000	-						1099					
0-y1_1	> p_addition3	0000 0000							0000					
+ 12 + 12	P_widition7	0000-0050-	-						0000000					
- p4	<pre>> p_doubling1</pre>	0000 0000	1						0000					
0- y4 0- x4	P_doubling2	0500 0000							0000).
* spi fied		• 4 • 4 •								· /		3224	12 - EL	- N
Im Trigger Ports	Wassform captured	Jun 14, 2015 1	156:34 AM						X		· · · 0:	0 4 8	A X-0) :	.0

Fig 16. Binary field addition (8 bits) on Chipscope

Figure 15 shows the simulation output for 8 bits in binary field and the Hardware output with Chipscope inFigure 16.

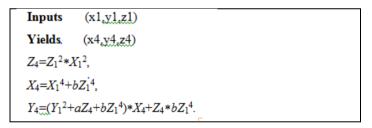


Fig 17. Algorithm for Point doubling in Binary Field

The point doubling algorithm with projective coordinates is shown in Figure 17. The corresponding simulation output and hardware output with chip scope is given in Figure 18 and Figure 19.

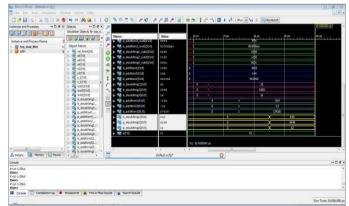


Fig 18. Binary field doubling using mixed coordinates (8 bits)

Project: top_dual_filed JTAG Chain † DEV:0 MpDankce0 (HDSNLX58T) System Monitor Core cle		ch Unit	CSVLX50T) UNIT-0 Myd Function	(A.I) (A.I)	Value	XXX	Rada	Counter disabled	' Ø' 1
URUT D MyLAO (LA) Trigger Setue Waveform Loting		Active ®	1	Trigger Conditi TriggerCond			Trigger Condition Equ MO	adon	
Signals: DEV: 0 UNIT: 0	Type: Window	▼ Wr	ndows	1	Depth: 1024		Position	0	
Data Port P p. doubling3	Waveform - DEV	to MyDevice0 (XC5	WEXSOT) UNITED MYOLAD	(ILA)				đ	' D'
>> p_doubling2 >> p_doubling1	Bus/Signal	x o 🖁	40 80 120 160	200 240 280	320 360 400 440 480	520 560 600 640	680 720 760 800	9 840 888 920 960 1	300
+ p_addition2	* y1_1	02 02				12			3
∽ p_addBon3 ∞ p_addBon3	P x3_3	02 02				12			5
o_doubling3	+ b_addtion1	1577 1571			1	577			3
b_doubling2 b_doubling1	P-b addtion3	256 256				54			5
b_addion2 b_addion3	0-b addition2	1340 1240			124	0720			7
0_000001	P-b doublingi	0000 0000			0	000			3
≈ 21_1 ≈ 11_1	⇒ b deublling2	0000 0000			0	000			3
9- y2	+ b doubling3	0000 0000			0	000			5
や 12 や 34	P-p addition1	0000 0000			0	000			7
0- 94	► p addition3	0000 0000			0	100			3
or s4 or sel filed	4				204				2
Trioper Ports	Waveform captured					X1 0	4 > 0r 0	4 + AIX-011 0	18

Fig 19.Binary field doubling (8 bits) on Chipscope

I. Point Addition and Point Doubling Over Prime Field

For an elliptic curve, in case of point addition over the prime filed one point is the normal point which is represented as (x,y) is projected into projective coordinates (X1,Y1,Z1), where $x=X/Z^2$ and $y=Y/Z^3$ and another point is of affine point represented as (x2,y2). Point addition algorithm and results are given in Figures 20-22.

The point doubling over prime field is carried out by using the pure projective coordinates. It requires only one point which is in projective coordinates and the result point doubling (adding a point witch itself) is also in projective coordinates. In these algorithm a and b are called as the parameters of an elliptic curves. Here point P is the input and the output of point doubling is represented by 2P. The algorithm to perform point doubling over the prime field is represented as follows in Figure 23, 24 and Figure 25.

```
Algorithm

Input: Q = (X_4, Y_4, Z_4), A = (x_2, y_2)

Output: R = (X_3, Y_3, Z_3) = P + Q;

A = X_4;

B = x_2 * Z_1^2;

C = A - B;

D = Y_1;

E = y_2 * Z_1^3;

F = D - E;

G = A + B;

H = D + E;

Z_3 = Z_1 * C;

X_3 = F^2 - G * C^2;

I = G * C^2 - 2 * X_3;

Y_3 = (I * F - H * C^2)/2;
```

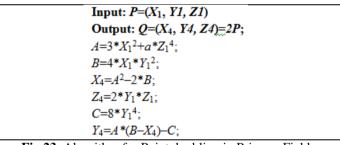
Fig 20. Algorithm for Point addition in Primary Field

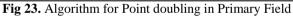


Fig 21. Prime field addition simulation results

roject: kop_duad_filed TAG Chain • DEV 6 M(Device6 (XCSVLX50T) • System:Monitor Console	Trigger Setup - 1	th Unit	(XCSVLXS97) UNI Fund	00		Value		NOX Bin	Counter dissbled	e' Ø'	
ONITIO MyEAD (EA) Trigger Setup Waveform Listing	Add OH	Adhe B	-		ndillion Name Condition0	1		Trigger Co	ndition Equation MD		
isoals: DEV: 0 UNIT: 0	Type: Window	T W	Indons:		t Depth	1024		Position		0	
Data Port	(5) (2) Waveform - DEV	0 HyDevice0 DIC	SVLX50TI UNIT:01	IMLAC (EA)							്ത്
P_doubling3 ⇒ p_doubling3		11		1 160 200 240 3	90 320 360	400 440 480	520 560 600	640 680 720	760 800 84	0 880 920 1	
P p_doubling1	Bun/Sipsal	× 0 5	ll.		dan dan de						
p_addton2 p_addton3	* b_addtion2	0 0	0				0				
- p_addition1	<pre>⇔ b_doubling1</pre>	0 0					0				
► b_doubiling3	> b_doublling2	0 e [1				0				3
b_doubling2 b_doubling1	0- h doubling3	0 0					0				
► b_addbon2	o p addition1	-120 -120	<u>.</u>				1204				- 1
> b_addion3 > b_addion1	P p addition3	-20 -20					-20				
≥1_1	+ p addition2	5212 5212					5212				_
= y1_1 = y2	Contraction of the	0000 0000					0000				
- 12 - 12	0" p_doubling1	1									
5-24	<pre>or p_doublling2</pre>	0000 0000					0000				
4- 14 7- 16	0 p_doubling3	0000 0000	1				0000				
se sei Ned	XI.		-								
Trigger Ports	Waveform captured	Jun 14, 2015 113	57,39 AM				Xi	0 4 + 00	0 4 1	Δ(X-0) :	0

Fig 22.Primary field addition (8 bits) on Chipscope





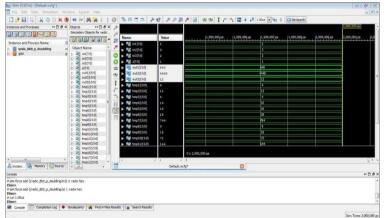


Fig 24. Primary field doubling_simulation

| Mal | ch Unit | Devic

 | HI (XCSVI | |
 | AYLAD | JILA) | | | | Value |
 | | | - | ox | | | T
 | | | | | ំ ព័ |
|-----------------------------|---
--
--
--
--|--|---|--
--	--------------------------------------	---	--	--	---
--	---	---	---	---	--

 | | |
 | 1 | | | | | |
 | | | | 1 | Kgger (| Con-dife
NO | an Equ
 | aton | | | | |
| Type: Wedaw | |

 | Windows | |
 | | | 1 | Dep | th 100 | 24 |
 | | | * | -10 | Position | | _
 | _ | _ | | H) | |
| Waveform - DEV | 7.0 MyDe | vice0

 | | | | | | |
 | | | | | | |
 | | | | | | |
 | | | | | í Ø, |
| Bus/Signal | x | 0

 | 83 1 | 80 | 120 1
 | 68 20 | 0 240 | 280 | 320 36 | 49 | 0 440 | 480
 | 520 | 568 1 | 500 I | 568 1 | 10 72 | 8 76 | 3 810
 | 840 | 890 | 920 | 960 | 1000 |
| P b addition2 | 0 |

 | | | | | | |
 | | | | | | |
 | 0. | | | | | |
 | | | | | |
| ⊳b doubllingt | 0 | 0

 | 1 | |
 | | | | | | | -
 | 8 | | | | | |
 | | | | | |
| 0- h doublling2 | | 0

 | 1 | | _
 | | _ | _ | _ | _ | _ | _
 | 0 | - | _ | - | | - | _
 | _ | _ | - | _ | - |
| >> b doubling) | 1 |

 | 1 | - | -
 | _ | _ | _ | | - | - | -
 | 0 | _ | | - | | - | _
 | _ | - | - | _ | = |
| P p editioni | |

 | 1E | |
 | | | | | | | -
 | 0 | | | | | |
 | | | | | = |
| A CONTRACT OF A DESCRIPTION | |

 | - | _ |
 | | _ | | | _ | | _
 | 0 | _ | _ | _ | _ | _ |
 | _ | _ | _ | _ | - |
| | |

 | | | | | | |
 | | | | | | |
 | A | | | | | |
 | | | | | - |
| 100 | 240 | -

 | - | |
 | | _ | | | | |
 | | - | _ | _ | | | _
 | _ | _ | | _ | - |
| | 1000 |

 | 1 | |
 | | | | | | | _
 | | | | | | |
 | | | | | - |
| | 1000 |

 | 1 | - |
 | | _ | | _ | - | _ |
 | | - | - | _ | _ | - | _
 | _ | _ | - | _ | |
| 1000 | 1.23 |

 | | _ |
 | | _ | | | _ | _ | _
 | | - | _ | _ | _ | _ | _
 | _ | _ | _ | _ | - 14 |
| Waveform captures | |

 | | | | | | | | | | | | | | | | | | | | | | | | | |
 | | | | | | |
 | | x. | | | 1. | - |
 | | | | | a , |
| | 1 1 | Image: Same_OUTLAB Tragge: Same_OUTLAB Image: Same_OUTLAB Same OutLAB Image: Same OutLAB Same OutLAB <td< td=""><td>Image: Loop: Loop:</td><td>Math.bit Math.bit 1 Soft Array 2 Soft Soft Array 3 Soft Soft Without 3 Soft Soft Without 3 Soft Soft Without 3 Soft Soft Soft 4 Array Soft Soft 5 Array Soft Soft 5 Array Soft Soft 5 Array Soft Soft 5 Array Soft Soft <</td><td>Mathematical Content Participation Participation Image: Strategy of the strategy</td><td>Ballo Lind Pandem Image: State in the state in</td><td>Basis lot Pactor 0 </td><td>Image: Control of the second second</td><td>Math.bit Pandam Image: Section of the section of t</td><td>Math. Col. Pandem Math. Col. Pandem Math. Col. Toget: Collaborations Math. Col. Toget: Collaborations Math. Col. Without Social Math. Col. Math. Col. Math. Col. Math. Col.<</td><td>Biological Control (Control (Contro) (Control (Control (Control (Control (Control (Contr</td><td>Model bell Fandam Water Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Nam Image: Condition Name</td><td>Math. Data Fundam Value Image: Status Image: Status Image: Status Image: Status Image: Status Image: Status Image: Status Image: Status Image: Status Image: Status Image: Status</td></td<> <td>Mathematical Columnation Paradiam Value Image: Construction Tragge: Construction Tragge: Construction Image: Constretion Tragge: Constretion</td> <td>Mod 3-bit Fundam Value Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image</td> <td>Math. Dial Pandim Value Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant</td> <td>Mathematical Data Pandam Value Image: Data Data Andree Topper Conduction Home Topper Conduction Home Image: Data Data Image: Data Data Topper Conduction Home Topper Conduction Home Image: Data Ima</td> <td>Max Max Max Reg Image: Im</td> <td>Math. Dat. Fundam Value State Image: Control in them Togger Control in them Togger Control in them Togger Control in them Image: Control in them Togger Control in them Togger Control in them Togger Control in them Image: Control in them Togger Control in them Togger Control in them Togger Control in them Image: Control in them Togger Control in them Togger Control in them Togger Control in them Image: Control in them Togger Control in them Togger Control in them Togger Control in them Togger Control in them Image: Control in them Togger Control in them Togger Control in them Togger Control in them Togger Control in them Image: Control in them Togger Control in them Togger Control in them Togger Control in them Togger Control in them Image: Control in them Togger Control in them Togger Control in them Togger Control in them Togger Control in them Image: Control in them Togger Control in them Togger Control in them Togger Control in them Togger Control in them Image: Control in them Togger Control in them Togger Control</td> <td>Mail Mail Rode Image: Destination of the set of th</td> <td>Middle Fandom Made Rade Image: Condition Store
BiggerCondition Store
BiggerConditet
BiggerCondition Store
BiggerCondition Store
Bigg</td> <td>Math. Dat. Feedam Value Radii Common Name Radii Common Name Radii Common Name Common Name</td> <td>Mach Load Fundam Year Rade Description 0 </td> <td>Math Fandam Yalar Rode Rode Construct 1 International Construction Target Condition Name Target</td> | Image: Loop: | Math.bit Math.bit 1 Soft Array 2 Soft Soft Array 3 Soft Soft Without 3 Soft Soft Without 3 Soft Soft Without 3 Soft Soft Soft 4 Array Soft Soft 5 Array Soft Soft 5 Array Soft Soft 5 Array Soft Soft 5 Array Soft Soft < | Mathematical Content Participation Participation Image: Strategy of the strategy | Ballo Lind Pandem Image: State in the state in | Basis lot Pactor 0 | Image: Control of the second | Math.bit Pandam Image: Section of the section of t | Math. Col. Pandem Math. Col. Pandem Math. Col. Toget: Collaborations Math. Col. Toget: Collaborations Math. Col. Without Social Math. Col. Math. Col. Math. Col. Math. Col.< | Biological Control (Control (Contro) (Control (Control (Control (Control (Control (Contr | Model bell Fandam Water Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Nam Image: Condition Name | Math. Data Fundam Value Image: Status Image: Status Image: Status Image: Status Image: Status Image: Status Image: Status Image: Status Image: Status Image: Status Image: Status | Mathematical Columnation Paradiam Value Image: Construction Tragge: Construction Tragge: Construction Image: Constretion Tragge: Constretion | Mod 3-bit Fundam Value Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image: Condition Name Image | Math. Dial Pandim Value Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant Lange Toget Constant Lange Image: Constant Lange Toget Constant | Mathematical Data Pandam Value Image: Data Data Andree Topper Conduction Home Topper Conduction Home Image: Data Data Image: Data Data Topper Conduction Home Topper Conduction Home Image: Data Ima | Max Max Max Reg Image: Im | Math. Dat. Fundam Value State Image: Control in them Togger Control in them Togger Control in them Togger Control in them Image: Control in them Togger Control in them Togger Control in them Togger Control in them Image: Control in them Togger Control in them Togger Control in them Togger Control in them Image: Control in them Togger Control in them Togger Control in them Togger Control in them Image: Control in them Togger Control in them Togger Control in them Togger Control in them Togger Control in them Image: Control in them Togger Control in them Togger Control in them Togger Control in them Togger Control in them Image: Control in them Togger Control in them Togger Control in them Togger Control in them Togger Control in them Image: Control in them Togger Control in them Togger Control in them Togger Control in them Togger Control in them Image: Control in them Togger Control in them Togger Control in them Togger Control in them Togger Control in them Image: Control in them Togger Control in them Togger Control | Mail Mail Rode Image: Destination of the set of th | Middle Fandom Made Rade Image: Condition Store
BiggerCondition Store
BiggerConditet
BiggerCondition Store
BiggerCondition Store
Bigg | Math. Dat. Feedam Value Radii Common Name Radii Common Name Radii Common Name Common Name | Mach Load Fundam Year Rade Description 0 | Math Fandam Yalar Rode Rode Construct 1 International Construction Target Condition Name Target |

Fig 25. Prime field doubling on Chipscope

A. Performance Analysis

The input message data considered is of 8 bits and the time required for point addition is in terms of pico seconds. The device utilization is also very less i.e 0% for the FPGA used for the implementation. The input size is increased to 163 and 192 bits for both Binary and Primary field. The time required for Point addition and Point doubling is in terms of micro and nano seconds. Simulation waveforms are shown in Figure 26 and Figure 27.

Figure 28 shows the RTL schematic of the implemented dual field processor which performs point addition and point doubling in Binary and Primary fields.

			17.9	44320 us				
Name	Value 15 us	16 us	17 us 18	us	19 us	20 us	21 us	22 us
1 clk	0							
1 rst	1							
la sel_field	0						1	
bin_add1[162:0]	101101010101010		10110	101010101010101	10 10 10 10 10 10 10 10 1	0101010		
 bin_add2[162:0] 	001010101010110		00101	0 10 10 10 10 1 10 10 1	10 10 1 10 10 10 10 10 10	0010101		
bin_add3[162:0]	554555aab551556		5,455	5aab551556aad	4555aab551556aac	154555a		
 bin_add4[162:0] 	1556aad54555aak		1556a	ad54555aab551	56aad54555aab551	1\$56aad		
 bin_add5[162:0] 	001010101010110		00101	010101010110101	10 10 1 10 10 10 10 10 10	0010101		
out_add_bin1[162:0]	7c0556c8198fcd5		7:05	56c8198fcd98f7	7eaafd978a6ab277	f72c1d		
• 🔩 out_add_bin2[162:0]	5d7c19d36cf6c92		5d7c1	19d36cf6c923e0b	27d1d69da59689b7	87614d		
• 🍓 out_add_bin3[162:0]	0b393c06cc2c10b		06393	3c06cc2c10b52f8	d22e5e306722ab14	dcda0b		
• 📢 out_dob_bin1[162:0]	4fb08e9820cf641		tbos	e9820cf64f74a9	57b033655aedda5b	bcd56b		
• 🔣 out_dob_bin2[162:0]	441fa19b899f70b		141fa	a 19b899f70bf87	fa06bbac9e2a68fa74	4af263		
out_dob_bin3[162:0]	1351c94db66cac8		3510	94db66cac58f29	9ed8e6fb26017000	73105a		
• 📲 out1[162:0]	7c0556c8198fcd5	7c0556c8198fcd98f75	7eaafd978a6ab277f7 <mark>2</mark> c1d		*·	4b08e9820cf64f74a	57b033655aedda5b	bcd56b
• 📲 out2[162:0]	5d7c19d36cf6c92	5d7c19d36cf6c923e0b2	7d1d69da59689b787614d	ł	*	441fa19b899f70bf87	fa06bbac9e2a68fa7	4af263
• 📲 out3[162:0]	0b393c06cc2c10b	0b393c06cc2c10b52f8c	22e5e306722ab14dca0b)	X	1351c94db66cac58f2	9ed8e6fb26017000	73105a

Fig 26.Simulatiom waveform of Point Addition and Point Doubling for 163 bits

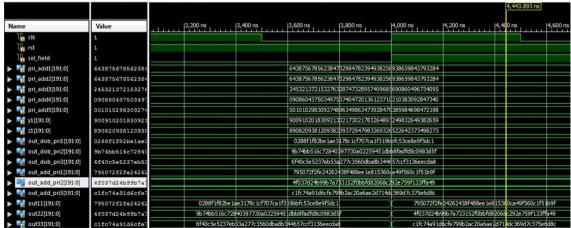


Fig 27.Simulation waveform of Point Addition and Point Doubling for 192 bits

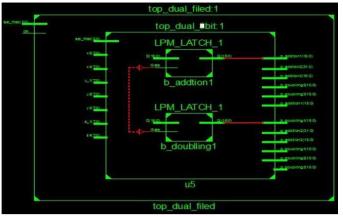


Fig.28 RTL schematic of dual field processor

Generated on: Module: Technology library: Operating condition: Wireload mode: Area mode:	Nov 04 final_t fast	2015 12:32 op_163bits alanced_tree d	
Instance Cell		Dynamic Power(nW)	
inal top 163bits 49	2 35280.158	321274.581	356554.739
u1	0 0.000	10440.347	10440.347
	0 0.000	10445.703	10445.703
^{u1} ^{u2} Fig 29. Powe	0 0.000	10445.703	10445.703
u1 Fig 29. Powe cc:/> repio ort area Generated by: Generated on:	0 0.000 r consumpt	10445.703 ion of dual r(R) RTL Com 015 12:31:3	10445.703 processor piler v14.20-s038
u1 u2 Fig 29. Power c:/> repio ort area Generated by: Generated on: Module:	0 0.000 er consumpt Encounte Nov 04 2 final_to	10445.703 ion of dual	10445.703 processor piler v14.20-s038
u1 u2 Fig 29. Power c:/> repio ort area Generated by: Generated on: Module: Technology library:	0 0.000 r consumpt Encounte Nov 04 2 final_to fast	10445.703 ion of dual r(R) RTL Com 015 12:31:3 p_163bits	10445.703 processor piler v14.20-s038
u1 u2 Fig 29. Power c:/> repio ort area Generated by: Generated on: Module:	0 0.000 r consumpt Encounte Nov 04 2 final_to fast	10445.703 ion of dual r(R) RTL Com 015 12:31:3 p_163bits lanced_tree)	10445.703 processor piler v14.20-s038

final_top_163bits
<none> (D) 3714 Fig 30. Area utilized for the dual processor

492

3714

0

The Verilog HDL code is executed with Cadence RTL Compiler for area, power and delay details. Figure 29, 30 and 31 gives the area utilized, path delay and power consumption. Figure 32 denotes physical layout of the processor implemented.

Generate							14.20-s03
Generate	d on:		▼ 04 20			24 pm	
Module:			nal_top	p_163	oits		
	gy library:				-		
	g condition			lance	d_tree)	
Wireload			closed				
Area mod	e:	tin	ming 1:	ibrar	Υ		
Pin	Туре	Fanout			-	Arrival (ps)	
	Type in port			(ps)	(ps)	(ps)	 R
rst			(fF)	(ps)	(ps)	(ps)	R
rst g12164/A		2	(fF) 4.2	(ps) 0	(ps) +0 +0	(ps) 0	
rst g12164/A g12164/Y	in port	2	(fF) 4.2	(ps) 0	(ps) +0 +0	(ps) 0 853	R
rst g12164/A g12164/Y g11835/A1	in port	2 489	(fF) 4.2	(ps) 0 1179	(ps) +0 +0 +853 +0	(ps) 0 853 853	R

Fig 31.Delay found with dual field processor

Wireload

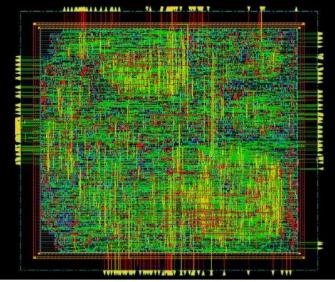


Fig 32. Physical Layout of Dual Filed Processor

VI. Conclusions

RSA processor with *multiplication algorithm*centered on the formula of ancient Indian Vedic mathematics has been projected and implemented with both multipliers. Both the Vedic multiplication methodsUrdhvaTiryambhayam and Nikhilam, have been explored with respect to area, delay and static power consumption. Due to its construction, Urdhva suffers from high carry propagation delay in instance of multiplication, whereas with Nikhilam Sutra the multiplication is reduced to 50% of it. We have developed and implemented the multiplier architecture based on these Sutrasfor RSA processor. Further, it is planned to implement this work for higher bits to speed up the process of encryption and decryption in Wireless Sensor nodes.

Architecture of dual processor has been proposed with a data path capable of doing operations such as point addition and point doubling of the elliptic curves either on prime filed GF (P) or binary extended fields GF (2^m). The design of dual field processor is presented; this proposed method requires less computational time as compared with the previous implementations. Hence, it results in high performance. This is an initialization work to take up the complete Scalar multiplication in ECC for Encryption and Decryption of information.

References

- Antonio de la Piedra, An Braeken, and AbdellahTouhafi. 2012. Sensor Systems Based on FPGAs and Their Applications: A Survey. [1]. Sensors (2012), 12, 12235-12264; DOI:10.3390/s 120912235.
- [2]. Shaila K, S H Manjula, Thriveni J, Venugopal K R and L MPatnaik, "Resilience Against Node Capture Attack using Asymmetric Matrices in Key Predistribution Scheme in Wireless Sensor Networks," in International Journal on Computer Science and Engineering, ISSN:0975-3397, vol. 11, no. 3, pp. 31-41, 2011.
- Lata B T, VidyaRao, Sivasankari H, Tejaswi V, Shaila K, Venugopal K R, L M Patnaik, "SEAD: Source Encrypted Authentic Data [3]. for WirelessSensor Networks," in International Journal of Engineering Research and Development, e-ISSN: 2278-067X, p-ISSN: 2278-800X, vol. 11, no. 3,pp. 01-16, 2015.
- Can Eyupoglu, "Investigation of the Performance Nikhilam Multiplication algorithm", In Proceeding of the world Conference on [4]. Technology, Innovation and Entrepreneurship. Elsevier, Procedia-Social Sciences 195(2015), pp. 1959–1965, 2015.
- Tiwari H.D., Gankhuyag, G., Kim, M., and Cho, B.: "Multiplier Design Based on Ancient Indian Vedic Mathematics," In Proceedings of the IEEE InternationalConference on SoCDesign, ISOCC, Busan, pp. II-65-II-68,2008. [5].
- Kunchigi, V., Kulkarni, L. and Kulkarni. S.: "High Speed and Area Efficient Vedic Multiplier," In Proceedingsof IEEEInternational [6]. Conference on Devices, Circuits and Systems (ICDCS), Coimbatore, pp. 360-364, 2012
- PrabirSaha, Arindam Banerjee, Partha Bhattacharyya, AnupDandapat.:"High Speed ASIC Design of Complex Multiplier Using [7]. Vedic Mathematics", in Proceeding of the IEEE Students' TechnologySymposium, IITKharagpur, pp. 237-242, 2011.
- Saha, P., Banerjee, A., Dandapat, A., and Bhattacharyya , P. : "ASICDesign of a High Speed Low Power Circuit for Factorial [8]. Calculation using AncientVedic Mathematics," in Elsevier Microelectronics Journal, vol. 42, pp. 1343-1352, 2011.
- [9]. Huddar S R., Rupanagudi, S.R., M., Mohan, S.: "Novel High Speed Vedic Mathematics Multiplier using Compressors", in Proceedings of IEEE InternationalMultiConference, pp. 465-469, 2013.
- [10]. Kumaravel, Ramalatha, Marimuthu .: "VLSI Implementation of High Performance RSA Algorithm using Vedic Mathematics", in Proceedings of International conference on Computational Intelligence and Multimedia Applications, pp. 126-129, 2007. SSChopada, Rama Mehta. "Performance Analysis of Vedic Multiplication Technique Using FPGA", in IEEE Bombay Section
- [11]. Symposium (IBSS), India, 2015.
- [12]. Pramod S. Aswale, PriyankaNirgude, Bhakti Patil, RohiniChanderi.: "Design and Implementation of High Speed Multiplier based on Vedic Mathematics", in International Journal of Computer Applications, Vol.155, No.8,2016.
- R Raju., S.Veerakumar .: "Design and Implementation of Low Power and High performance Vedic Multilier", in Proceedings of [13]. IEEE International Conference on Communication and Signal Processing, pp. 0601-0605, 2016.

- [14]. PritamGajkumar Shah, Xu Huang, Dharmendra Sharma.: "Algorithm based on one's complement for fast scalar multiplication in ECC for Wireless Sensor Network", in Proceedings of IEEE International Conference on Advanced Information Networking and Applications Workshops, pp. 571-576, 2010.
- [15]. SoramRanbir Singh, Ajoy Kumar Khan, SoramRakesh Singh.: "Performance Evaluation of RSA and Elliptic Curve Cryptography", in Proceedings of IEEE International Conference on 2nd International Conference on Contemporary Computing and Informatics, pp. 302-306, 2016.
- [16]. SoramRanbir Singh, Ajoy Kumar Khan, TalhellambamSonamani Singh.: "ACritical Review on Elliptic Curve Cryptography", in Proceedings of International Conference on Automatic Control and Dynamic Optimization Techniques, pp. 13-19, 2016.
- [17]. Xianjin Fang, Yanting Wu.: "Investigation into the Elliptic Curve Cryptography", in Proceedings of International Conference on Information Management, pp. 412-415, 2017.
- [18]. Shaila K, Nalini L, Tejaswi V, Thriveni J, Venugopal K R, L M Patnaik, "Secure QoS-Aware Data Fusion to Prevent Node Misbehavior in Wireless Sensor Networks," in International Journal of ComputerScience and Network Security, ISSN: 0975-3397, vol. 11, no. 3, pp. 31- 41, 2011.
- [19]. HimanshuThapliyal and M B Srinivas, "An Efficient Metho d of Elliptic Curve Encryption Using Ancient Indian Vedic Mathematics," in Proceedings of the 48th IEEE Midwest Symposium on Circuit and Systems, ISBN:0-7803-9197-7, vol. 1, pp. 826-828, 2005.
- [20]. HilalHoussain, MohamadBadra and Turki F Al-Somani," Comparative Study of Elliptic Curve Cryptography Hardware Implementations in Wireless Sensor Networks," in International Journal of RFID Securityand Cryptography (IJRFIDSC), vol. 1, no. 1/2, pp. 67-74, March/June2012.
- [21]. Gustavo D Sutter, Jean-Pierre Deschamps and Jos Luis Imaa, "Efficient Elliptic Curve Point Multiplication Using Digit-Serial Binary Field Operations," *in IEEE Transactions on Industrial Electronics*, ISSN:0278-0046, vol. 60, no.1, pp. 217-225, 2013.
- [22]. J W Lee and H C Chang, "Efficient Power-Analysis-Resistant Dual-Field Elliptic Curve Cryptographic Processor Using Heterogeneous Dual-Processing-Element Architecture," in IEEE Transactions on VeryLarge Scale Integration Systems, ISSN:1063-8210, vol. 22, no. 1, pp.49-61, 2014.
- [23]. Gustavo D Sutter, Deschamps, Jean-Pierre and Imana, Jos'e Luis, "Modular Multiplication and Exponentiation Architectures for Fast RSA Cryptosystem Based on Digit Serial Computation.," IEEE Transactions on Industrial Electronics, vol. 58, no.7 pp. 3101-3109, 2011.

Leelavathi G "Design of RSA Processor and Field Arithmetic of ECC with Vedic Multipliers for Nodes in Wireless Sensor Networks." IOSR Journal of VLSI and Signal Processing (IOSR-JVSP), vol. 8, no. 1, 2018, pp. 01-15.

DOI: 10.9790/4200-0801010115
