New Algorithm to Convert any Integer in TBNS

Subhashis Maitra
Kalyani Government Engineering Collage
Kalyani , Nadia, West Bengal, India

Amitabha Sinha
West Bengal University of Technology
Saltlake, Kolkata, West Bengal, India

ABSTRACT

Algebraic complexity of different Algorithms in Signal Processing and Cryptography leads to a major problem and Researchers are trying to develop new Algorithms to solve these problems. To enhance the speed of the existing Algorithms, different number system have been found for point multiplication in elliptic curve cryptography and coefficient multiplication in digital signal processing manly for digital filter design. Among the different number system, DBNS, DBC, HBTJSF, w-NAF are efficient. Recently, to increase the speed again, TBNS, SDTBNS have been developed. There are different method to convert any integer or fraction into TBNS and hence SDTBNS. Here a new algorithm will be discussed which increase the conversion efficiency.

Keywords

DBC, DBNS, Digital Filter, DSP, ECC, HBTJSF, JSF, TBC, TBHJSF, TBNS, w-NAF.

1. INTRODUCTION

The complexities of multiplication and addition in multi-exponentiation operation and points addition in ECC are the major problem in current signal processing and different cryptographic algorithms. This problem also arises in case of the coefficient multiplication in digital filter design. To speed-up these multiplication operation, Shamir proposed an algorithm [1][2] that eliminates the unnecessary separate computation of the two expressions. Shamir stated in [3], that two integers x and y can be expanded in the binary form at the same time and shown an extra savings of doublers and multipliers. He also proposed that if ‘n’ represents the bit length of the largest exponent, then average numbers of multipliers and doublers required are ‘3n/4’ and ‘n’. In ECC, where the operation [x]P + [y]Q is an important part to perform, the elements can be easily inverted using Shamir’s algorithm. Here the scalars x and y can be represented using a 2×n matrix as,

$x = (x_{n-1}\ x_{n-2} \ldots\ldots\ldots\ldots\ x_1\ x_0)$ and

$y = (y_{n-1}\ y_{n-2} \ldots\ldots\ldots\ldots\ y_1\ y_0)$ where x_i and $y_i \in \{-1, 0, 1\}$

For example 405 and 424 in NAF can be represented as

$405 = (1\ 0\ \bar{1}\ 0\ 0\ 1\ 0\ 1\ 0\ 1)$

$424 = (1\ 0\ \bar{1}\ 0\ 1\ 0\ 1\ 0\ 0\ \bar{1})$

Hence 210 and 324 simultaneously can be represented in 2x9 matrix as

$$\binom{405}{423} = \begin{pmatrix} 1\ 0\ \bar{1}\ 0\ 0\ 1\ 0\ 1\ 0\ 1 \\ 1\ 0\ \bar{1}\ 0\ 1\ 0\ 1\ 0\ 0\ \bar{1} \end{pmatrix} \qquad (1)$$

It is to be noted that there are 7 non-zero columns. Shamir claimed that in this method of representation, only 5n/9 additions and n + 1 doublings on average are required to calculate [x]P + [y]Q.

Solinas in [4][5], introduced Joint Sparse Form(JSF) in order further reduce the average number of non-zero columns. In JSF, the integers 405 and 423 can be represented as

$$\binom{405}{423} = \begin{pmatrix} 1\ 1\ 0\ 1\ 0\ \bar{1}\ 0\ \bar{1}\ \bar{1} \\ 1\ 1\ 0\ 1\ 0\ 1\ 0\ 0\ \bar{1} \end{pmatrix} \qquad (2)$$

Here it is to be noted that the non-zero columns have been reduced to 6. Hence requirement of doublers and adders will also be reduced. Solinas claimed in [4], that to calculate [x]P + [y]Q, n number of doublers and n/2 adders are required on average, where n is the number of column in the Solinas’s matrix.

V. Dimitrov et. al. and C. Doche et al introduced in [1][6][8] a new number system, known as Joint Double-Base Number System (JDBNS) to further reduce the number of non-zero columns. In JDBNS, two integers n and m can be represented as

$$\binom{n}{m} = \sum_{i=1}^{l} \binom{c_i}{d_i} 2^{a_i}.3^{b_i}\text{ , where } c_i, d_i \in \{\,1, 0, -1\} \qquad (3)$$

Latter Dimitrov and Cooklev introduced in [6], Hybrid Binary-Ternary Number System (HBTNS) [1][7] to speed-up modular exponentiation. In HBTNS, an integer can be represented as a sum of powers of 2 and powers of 3, i.e., it mixes bits and trits(ternary digits). The use of base 3 naturally reduces the number of digits required to represent a n-bit integer. It has been shown by Dimitrov et.al. in[6] that the digit length is almost 12% smaller than the binary length. More importantly, this number system is also very sparse. The average number of non-zero digits in HBTNS is n/3 for an n-bit number.

For example, 405 and 423 in HBTNS can be represented as

Digits[405] = [0 0 0 0 1 0 1], Base[405] = [3 3 3 3 2 2 2] and

Digits[423] = [0 0 1 1 1 1 0 1], Base[423] = [3 3 2 2 2 2 2 2] (4)

Hence 405 and 423 can be represented as the sum of the product terms of the powers of the bases 2 and 3 using the above digits[] and base[] as

$405 = 3^4.2^0 + 3^4.2^2$ and $423 = 3^2.2^0 + 3^2.2^1 + 3^2.2^2 + 3^2.2^3 + 3^2.2^5$

It is to be noted that the binary length for both 405 and 423 is 9. In [9], Ciet et. al. has introduced an Algorithm known as Joint Binary-Ternary Algorithm [JBTA]. Using this Algorithm, a pair of integer, suppose n and m can be represented in the same way as shown by equ.(3), but here the upper limit of ‘i' will be less than that using JDBNS. This Algorithm, as claimed by Ciet et.al. in [9], further reduces the number of nonzero columns and hence speeds up modular exponentiation. They claimed that the average number of bits eliminated at each step of Algorithm in [9], denoted by ‘K’, is greater than or equal to 2.53519 and density which measures the ratio of the number of terms in the representation of any integer in JBTA and the binary length of the number, is proportional to the reciprocal of ‘K’ and its maximum value is 0.45.

The expression for K is given by

$$K = \sum_{\alpha=0}^{\infty} \sum_{\beta=0}^{\infty} p_{\alpha,\beta}\,(\alpha + \beta \log_2 3) \qquad (5)$$

Where $p_{\alpha,\beta} = \frac{1}{2^{2\alpha+1}3^{2\beta-3}}$

Again in [8], Adikari et. al. described an Algorithm known as Hybrid Binary-Ternary Joint Sparse Form [HBTJSF][10][11] to further reduce the number of nonzero columns and hence to speed-up modular exponentiation and scalar multiplication. They claimed that to calculate [x]P + [y]Q, only 0.43n doublers, 0.36n triplers and 0.32n adders are required. Here the average number of non-zero columns is 0.32n, where n is the bit length.

Example 1: Representation of 405 and 423 in HBTJSF.

Solution: $405 = [\,1\ 0\ 0\ 0\ 0\ \bar{3}\ 0\ 0]$
$423 = [\,1\ 0\ 0\ 0\ 0\ \bar{1}\ 0\ 0]$
$\text{Base}[] = [3\ 3\ 2\ 3\ 2\ 2\ 2\ 2]$ (6)

It is to be noted that in HBTJSF representation the number of non-zero columns is reduced to 3.

From (6), it is clear that $405 = 3^3.2^4 - 3^3.2^0$ and
$423 = 3^3.2^4 - 3^2.2^0$.

Since HBTJSF uses the digit set { -2, -1, 0, 1, 2, 3}, total number of pre-computation required in this case is 14 which are P ± Q, P ± 2Q, P ± 3Q, 2P ± Q, 2P ± 3Q, 3P ± Q and 3P ± 2Q to calculate [x]P + [y]Q. In the next section a new Algorithm known as Triple-Base Hybrid Joint Sparse Form (TBHJSF)[] and its novelty for use in modular exponentiation and scalar multiplication will be discussed.

2. TRIPLE-BASE HYBRID JOINT SPARSE FORM

The algorithm proposed by S. Maitra et. al. in [12], Triple-Base Hybrid Joint Sparse Form[TBHJSF] is basically a modification of HBTJSF[8]. Here a third base, 5 is to be used in Base[] array. S. Maitra et. al. proposed that the base 5 has been chosen here in order to perform decimal shifting[since 5*2 =10 when multiplied with 2.9 gives 29 and hence if 29 can be represented in TBHJSF, 2.9 can also be computed from the representation of 29]. In this Algorithm, a pair of integers can be represented in TBHJSF by first checking whether the two integers are divisible by 5. If they are divisible by 5, the digits for both the integers will be set to 0, otherwise they should be checked whether they are divisible by 3. If they are divisible by 3, the digits for both the integers will be set to 0, otherwise the integers should be again checked whether they are divisible by 2. If they are divisible by 2, the digits for both the integers will be set to 0, otherwise both the integers are made divisible by 30(2*3*5) by adding or subtracting x, where x ∈ { -14, -13, -----, 0, 1, 2, --------- 15} and then the sum are divided by 2. The quotients are then treated as the two integers and the previous steps are then repeated until the quotients reach to zero. This Algorithm has been proposed in [12]

Algorithm I

Input: Two positive integers m_1 and m_2 ;
Output: Arrays Digit1[], Digit2[], Base[];

```
i = 0;
while $m_1 > 0$ or $m_2 > 0$, do
    if $m_1 \equiv 0 (\text{mod } 5)$ and $m_2 \equiv 0 (\text{mod } 5)$ then
        base[i] = 5;
        Digit1[i] = Digit2[i] = 0;
        $m_1 = m_1/5$ , $m_2 = m_2/5$;
    else if $m_1 \equiv 0 (\text{mod } 3)$ and $m_2 \equiv 0 (\text{mod } 3)$ then
        base[i] = 3;
        Digit1[i] = Digit2[i] = 0;
        $m_1 = m_1/3$ , $m_2 = m_2/3$;
    else if $m_1 \equiv 0 (\text{mod } 2)$ and $m_2 \equiv 0 (\text{mod } 2)$ then
        base[i] = 2;
        Digit1[i] = Digit2[i] = 0;
        $m_1 = m_1/2$ , $m_2 = m_2/2$;
    else
        base[i] = 2;
        Digit1[i] = $m_1$ mods 30, Digit2[i] = $m_2$ mods 30 ;
        $m_1 = (m_1$ - Digit1[i])/2, $m_2 = (m_2$ - Digit2[i])/2 ;
    end if ;
    i = i + 1;
end while;
return Digit1[],Digit2[], base[];
```

Example 2: Representation of 1234 and 2302 in TBHJSF.

Sol: $1234 = (\,1\quad 0\quad 0\quad \bar{9}\quad 0\quad 0\quad \overline{13}\quad 0\,)$
$2302 = (\,1\quad 0\quad 0\quad 8\quad 0\quad 0\quad 11\quad 0\,)$
$\text{Base}[] = (2\quad 3\quad 5\quad 2\quad 3\quad 5\quad 2\quad 2\,)$
(7)

Using this Algorithm, the number of non-zero columns can be reduced to a large extent at the cost of the size of the pre-computation look-up-table. Using the above Algorithm, 1234 and 2302 in TBNS can be represented as
$1234 = 2^3.3^2.5^2 - 9.2^2.\ 3^1.5^1 - 13.2$ and $2302 = 2^3.3^2.5^2 + 8.2^2.\ 3^1.5^1 + 11.2$

Hence we require only three doublers, two triplers, two pentuplers and two adders to compute [x]P + [y]Q, where x and y are 1234 and 2302 . From the above example it clear that the number of adders required to represent any integer in TBNS[13][14][15] has been drastically reduced with respect to DBNS, HBTJSF, w-NAF etc. Here in the next section we will propose a new Algorithm to represent any integer in TBNS form and hence to compute scalar multiplication in ECC and co-efficient multiplication in DSP.

3. PROPOSED ALGORITHM

Here we will discuss a new Algorithm to convert any two integers in Joint Triple-Base Number System (JTBNS). It is supposed that x and y are two positive integers. At first they are divided by $2^{v_2(x,y)}.3^{v_3(x,y)}$ in order to obtain p and q, where $p, q \in C$, C being the set of all positive integer 'm' and 'n' so that $v_2(m,n) = v_3(m,n) = 0$. Here $v_2(m,n) = \min\{\, v_2(m), v_2(n)\}$, where $v_2(m)$ and $v_2(n)$ are the 2-adic valuation of m and n respectively. $v_2(m,n)$ gives the largest power of the base 2, that can divides 'm' and 'n' simultaneously[8][9]. Then the function, gain(p, q) is called. The common powers of 2, 3 and 5 are then eliminated in $p \leftarrow D$ and $q \leftarrow E$, where D and E are the coefficients to maximize the factor gain(p, q). Hence the result will be new pair of positive integers in C. The process is the repeated. Since 'p' and 'q' decrease at each step and remain positive and at last a new pair can be obtained so that $p \leq 1$ and $q \leq 1$. Then the process will be terminated. The different steps of the proposed algorithm are being discussed here.

Algorithm II

Input: Two positive integers x and y such that x > 1 and y > 1

Output: Gain, A_i, B_i, C_i, D_i and E_i, a Joint Triple-Base Chain.

1. $i \leftarrow 0$, gain $\leftarrow 0$, $d_i \leftarrow 0$, $e_i \leftarrow 0$.

2. Divide the two integers x and y by the maximum value of a_i, b_i and c_i, where a_i, b_i and c_i are the

```
   power of 2, 3 and 5 in the form as $2^{a_i}.3^{b_i}.5^{c_i}$ either by adding 1 or subtracting 1 from the integers to make them divisible by $2^{a_i}.3^{b_i}.5^{c_i}$. Gain will be $2^{\max(a_i)}.3^{\max(b_i)}.5^{\max(c_i)}$ and i ← 1.
3. $A_i \leftarrow 0$, $B_i \leftarrow 0$ and $C_i \leftarrow 0$, $D_i \leftarrow 0$, $E_i \leftarrow 0$.
4. Now x ← $(x \pm d_i)/(2^{\max(a_i)}.3^{\max(b_i)}.5^{\max(c_i)})$ and y ← $(y \pm e_i)/(2^{\max(a_i)}.3^{\max(b_i)}.5^{\max(c_i)})$, $A_i \leftarrow A_i + \max(a_i)$, $B_i \leftarrow B_i + \max(b_i)$ and $C_i \leftarrow C_i + \max(c_i)$ , $D_i \leftarrow d_i$, $E_i \leftarrow e_i$.
5. Repeat step1 to 4
6. Return Gain, $A_i$, $B_i$, $C_i$, $D_i$ and $E_i$
```

Example 3: Convert 1234 and 2302 in JTBNS
Solution:

i	Gain(G)	integer x	integer y	D_i	E_i	A_i	B_i	C_i
0	0	1234	2302	0	0	0	0	0
1	2^1	1234 ÷ 2 = 617	2302 ÷ 2 = 1151	0	0	0	0	0
2	$2^1.3^1$	(617+ 1) ÷6 = 103	(1151+ 1) ÷ 6 = 192	$\bar{1}$	$\bar{1}$	1	0	0
3	3^1	(103 –1) ÷3 = 34	(192 + 0) ÷3 = 64	1	0	2	1	0
4	5^1	(34+1) ÷5 = 7	(64+1) ÷5 = 13	$\bar{1}$	$\bar{1}$	2	2	0
5	2^2	(7 + 1) ÷4 = 2	(13 - 1) ÷4 = 3	$\bar{1}$	1	2	2	1
6	3^1	(2 + 1) ÷ 3 = 1	(3 + 0) ÷ 3 = 1	$\bar{1}$	0	4	2	1
7	---	1 – 1 = 0	1 – 1 = 0	1	1	4	3	1

Hence

$$\binom{1234}{2302} = \binom{0}{0}.\ 2^0.3^0.5^0 + \binom{\bar{1}}{\bar{1}}.2^1.3^0.5^0 + \binom{1}{0}.\ 2^2.3^1.5^0 + \binom{\bar{1}}{\bar{1}}.2^2.3^2.5^0 + \binom{\bar{1}}{1}.2^2.3^2.5^1 + \binom{\bar{1}}{0}.2^4.3^2.5^1 + \binom{1}{1}.\ 2^4.3^3.5^1 \quad (8)$$

4. COMPLEXITY ANALYSIS

Here we will compute the average density of JTBC obtained by Algorithm 2 and the average values of the maximal power of 2, 3 and 5 in the JTBC expansion. The average density gives the number of non-zero columns in the expansion and hence gives the average number of additions and the average values of the maximal power of 2, 3 and 5 gives the number of doublers, triplers and pentuplers required to compute x[P] + y[Q]. To find out the density of expansion, two positive integers x and y, where x, y ∈ C and three fixed values α, β and γ are taken into consideration. To find the probability of the gain (x, y) = $p_{\alpha,\beta,\gamma}$ to occur, the total number of pairs having the desired gain in a certain square are to be found out and then that number is to be divided by the total number of pairs in $C \cap S_{\alpha,\beta,\gamma}$, where C = { (x, y) $\mid$ $v_2(x,y) = v_3(x,y) = v_5(x,y) = 0$ } and $S_{\alpha,\beta,\gamma}$ = { (x, y)) $\mid$ $v_2(x,y) = v_3(x,y) = v_5(x,y) = 0$ and gain(x, y) = $2^{\alpha}.3^{\beta}.5^{\gamma}$ }.

Now the total number of pairs in the square $R_{\alpha,\beta,\gamma}$ is given by the following Lemma.

Lemma 1: The cardinal number of the square $R_{\alpha,\beta,\gamma}$ = [1, $2^{\alpha}.3^{\beta}.5^{\gamma}$]^2 is $2^{2\alpha}.3^{2\beta}.5^{2\gamma}$

Proof: Let A be a set of two positive integers x and y, i.e. A = { x, y }. Then the square set, A^2 = { (x, x), (x, y), (y, x), (y, y) }. Here the cardinal number of A is 2 and that of A^2 is 4= 2^2. Now the cardinal number of [1, $2^{\alpha}.3^{\beta}.5^{\gamma}$] is $2^{\alpha}.3^{\beta}.5^{\gamma}$. So the cardinal number of $R_{\alpha,\beta,\gamma}$ = [1, $2^{\alpha}.3^{\beta}.5^{\gamma}$]^2 is $(2^{\alpha}.3^{\beta}.5^{\gamma})^2 = 2^{2\alpha}.3^{2\beta}.5^{2\gamma}$.

Example 4: The cardinal number of the square $R_{1,1,0}$ = [1, $2^1.3^1.5^0$]^2 = { (1,1), (1,2), (1,3), (1,4), (1,5), (1,6), (2,1), (2,2), (2,3), (2,4), (2,5), (2,6), (3,1), (3,2), (3,3), (3,4), (3,5), (3,6), (4,1), (4,2), (4,3), (4,4), (4,5), (4,6), (1,1), (5,2), (5,3), (5,4), (5,5), (5,6), (6,1), (6,2), (6,3), (6,4), (6,5), (6,6) } is 36.

Lemma 2: The cardinality of $C \cap S_{\alpha,\beta,\gamma}$, where C = { (x, y) $\mid$ $v_2(x,y) = v_3(x,y) = v_5(x,y) = 0$ } is equal to $2^{2\alpha+1}.3^{2\beta-1}.5^{2\gamma}$

Proof: In the square $S_{1,1,0}$, the number of pairs are (1,1), (1,2), (1,3), (1,4), (1,5), (1,6), (2,1), (2,2), (2,3), (2,4), (2,5), (2,6), (3,1), (3,2), (3,3), (3,4), (3,5), (3,6), (4,1), (4,2), (4,3), (4,4), (4,5), (4,6), (5,1), (5,2), (5,3), (5,4), (5,5), (5,6), (6,1), (6,2), (6,3), (6,4), (6,5), (6,6) among which (2,2), (2,4), (2,6), (3,3), (3,6), (4,2), (4,4), (4,6), (6,2), (6,3), (6,4) and (6,6), for these pairs, $v_2(x,y) \neq v_3(x,y) \neq v_5(x,y) = 0$. Hence

$C \cap S_{\alpha,\beta,\gamma}$ = **{** (1,1), (1,2), (1,3), (1,4), (1,5), (1,6), (2,1), (2,3), (2,5), (3,1), (3,2), (3,4), (3,5), (4,1), (4,3), (4,5), (5,1), (5,2), (5,3), (5,4), (5,5), (5,6), (6,1), (6,5)} and hence the cardinal number of $C \cap S_{\alpha,\beta,\gamma}$ = 24 = $2^{2..1+1}.3^{2..1-1}.5^{2.0}$.
Thus for any values of α, β and γ , the cardinal number of $C \cap S_{\alpha,\beta,\gamma} = 2^{2\alpha+1}.3^{2\beta-1}.5^{2\gamma}$.

Let us now find out the probability $p_{\alpha,\beta,\gamma}$ of the gain(x, y) which is the ratio of the total number of pairs having the desired gain in a certain square, $S_{\alpha,\beta,\gamma}$ and the total number of pairs in $C \cap S_{\alpha,\beta,\gamma}$. Now for a larger square,

$S_{\alpha+\varepsilon+1\,,\beta+\,\rho+1,\gamma+\,\tau+1}$, total number of pairs having the gain $2^{\alpha}.3^{\beta}.5^{\gamma}$ and the total number of pairs in C have the same factor $2^{2\varepsilon}.3^{2\rho}.5^{2\tau}$.

Lemma 3: In practice, for small pairs (x, y) and (x + i. $2^{\alpha+1}.3^{\beta+1}.5^{\gamma+1}$, y + j. $2^{\alpha+1}.3^{\beta+1}.5^{\gamma+1}$), where (i, j) $\in [\,0, 2^{\varepsilon}.3^{\rho}.5^{\tau} - 1\,]^2$ cannot have the same gain $2^{\alpha}.3^{\beta}.5^{\gamma}$

Proof: To prove this lemma let us consider a positive pair (39, 139) belongs to C.
Gain of the pair (39, 139) is $2^2.3^0.5^1$, when the coefficients are (-1, -1) or ($\bar{1}$, $\bar{1}$).

Let us consider another pair (39 + $1.2^3.3^1.5^2$, 139 + 2. $2^3.3^1.5^2$) = (639, 1339) belongs to C.
Gain of the pair (639, 1339) = $2^2.3^0.5^1$, when the coefficients are (-1, -1) or ($\bar{1}$, $\bar{1}$).
But for large pairs, Lemma 3 is not true. In that case, we can consider Lemma 4, being discussed in the following.

Lemma 4: For any three nonnegative integers, $,\beta,\gamma$, gain of a pair ($x + i.2^{\delta}.3^{\theta}.5^{\mu}, y + j.2^{\delta}.3^{\theta}.5^{\mu}$) belongs to C, where δ, θ and μ are nonnegative integers such that $2^{\delta} > 2^{\alpha}.3^{\beta}.5^{\gamma}, 3^{\theta} > 2^{\alpha}.3^{\beta}.5^{\gamma}$ and $5^{\mu} > 2^{\alpha}.3^{\beta}.5^{\gamma}$, and (i, j) belongs to square set , will be equal to the gain of the pair (x, y) belongs to C.

Proof: Let us assume that gain $(x + i.2^{\delta}.3^{\theta}.5^{\mu}, y + j.2^{\delta}.3^{\theta}.5^{\mu})$ = $2^{\alpha_1}.3^{\beta_1}.5^{\gamma_1}$ > gain (x, y). If the coefficients corresponding the gain(x, y) is D and E then

$$v_2\left(x - D + i.2^{\delta}.3^{\theta}.5^{\mu}\right) \geq \alpha_1,$$
$$v_2\left(y - E + j.2^{\delta}.3^{\theta}.5^{\mu}\right) \geq \alpha_1,$$
$$v_3\left(x - D + i.2^{\delta}.3^{\theta}.5^{\mu}\right) \geq \beta_1,$$
$$v_3\left(y - E + j.2^{\delta}.3^{\theta}.5^{\mu}\right) \geq \beta_1 \text{ and}$$
$$v_5\left(x - D + i.2^{\delta}.3^{\theta}.5^{\mu}\right) \geq \gamma_1,$$
$$v_5\left(y - E + j.2^{\delta}.3^{\theta}.5^{\mu}\right) \geq \gamma_1$$

Now using the rule $v_p(m,n) = \min\{v_p(m), v_p(n)\}$, where $v_p(m)$ and $v_p(n)$ are the p-adic valuation of m and n respectively and $v_p(m) \neq v_p(n)$ and $v_p(m,n)$ gives the largest power of the base p, we get

$v_2(x - D) \geq \alpha_1, v_3(x - D) \geq \beta_1$ and $v_5(x - D) \geq \gamma_1$, $v_p(x - D) \neq v_p(i.2^{\delta}.3^{\theta}.5^{\mu})$, p = 2, 3 and 5. Now if $v_2(x - D) = v_2(i.2^{\delta}.3^{\theta}.5^{\mu})$, then it can be prove that $v_2(x - D) \geq \delta$, again if $v_3(x - D) = v_3(i.2^{\delta}.3^{\theta}.5^{\mu})$, then $v_3(x - D) \geq \theta$ and if $v_5(x - D) = v_5(i.2^{\delta}.3^{\theta}.5^{\mu})$, then $v_3(x - D) \geq \mu$. In a similar way we can prove that $v_2(y - E) \geq \alpha_1, v_3(y - E) \geq \beta_1$ and $v_5(y - E) \geq \gamma_1$, $v_p(y - E) \neq v_p(j.2^{\delta}.3^{\theta}.5^{\mu})$, p = 2, 3 and 5 and if $v_2(y - E) = v_2(j.2^{\delta}.3^{\theta}.5^{\mu})$, $v_2(y - E) \geq \delta$, if $v_3(y - E) = v_3(j.2^{\delta}.3^{\theta}.5^{\mu})$, then $v_3(y - E) \geq \theta$ and if $v_5(y - E) = v_5(j.2^{\delta}.3^{\theta}.5^{\mu})$, then $v_5(y - E) \geq \mu$. Hence we can say that $v_2(x - D)$ and $v_2(y - E)$ are larger than minimum(α_1, δ). Similarly $v_3(x - D)$ and $v_3(y - E)$ are larger than minimum(β_1, θ) and $v_5(x - D)$ and $v_5(y - E)$ are larger than minimum(γ_1, μ). Now it has been assumed that $2^{\delta} > 2^{\alpha}.3^{\beta}.5^{\gamma}, 3^{\theta} > 2^{\alpha}.3^{\beta}.5^{\gamma}$ and $5^{\mu} > 2^{\alpha}.3^{\beta}.5^{\gamma}$, hence $2^{\alpha_1}.3^{\beta_1}.5^{\gamma_1} \geq 2^{\delta}.3^{\theta}.5^{\mu}$ and hence gain(x, y) $> 2^{\alpha}.3^{\beta}.5^{\gamma}$ which is contrary to the hypothesis and hence

$$\text{gain}\,(x + i.2^{\delta}.3^{\theta}.5^{\mu}, y + j.2^{\delta}.3^{\theta}.5^{\mu}) = \text{gain}(x, y). \qquad (9)$$

Lemma 5: The probability, $p_{\alpha,\beta,\gamma}$ is bounded above by $\frac{1}{2^{2\alpha+1}3^{2\beta-3}5^{2\gamma-1}}$ for any non-negative integers α, β and γ.

Proof : There are 30 integers in the set [1, $2^{\alpha+1}3^{\beta+1}5^{\gamma+1}$] which are divisible by $2^{\alpha}3^{\beta}5^{\gamma}$. For example, in [1, $2^{1+1}3^{1+1}5^{1+1}$] , the integers are 30, 60, 90, 120, 150, ----------------, 900 and these are divisible by 30 = $2^1 3^1 5^1$. Again, taking into consideration the coefficients that belong to { 1, -1}, each elements can be represented into three integers, suppose for the element x_0, there are three elements like x_0 – 1, x_0 and x_0 + 1 that are divisible by $2^{\alpha}3^{\beta}5^{\gamma}$. Hence, in total there will be 90 integers in the set [1, $2^{\alpha+1}3^{\beta+1}5^{\gamma+1}$] which are divisible by $2^{\alpha}3^{\beta}5^{\gamma}$. Hence in the square $S_{\delta,\theta,\mu}$, the pairs having gain = $2^{\alpha}3^{\beta}5^{\gamma}$, will be of the form ($x_0 + i.2^{\alpha+1}3^{\beta+1}5^{\gamma+1}, y_0 + j.2^{\alpha+1}3^{\beta+1}5^{\gamma+1}$), where x_0 and y_0 are one of the 90 elements. Here i and j belong to [0, $2^{\delta-\,\alpha-1}3^{\theta-\,\beta-1}5^{\mu-\,\gamma-1} - 1]^2$. Hence the maximum number of pairs in the square $S_{\delta,\theta,\mu}$ is $2^{2(\delta-\,\alpha)}.3^{2(\theta-\beta-1)}.5^{2(\mu-\gamma)-\,1}$ with a gain $2^{\alpha}3^{\beta}5^{\gamma}$. Hence the probability $p_{\alpha,\beta,\gamma} = \frac{1}{2^{2\alpha+1}3^{2\beta-3}5^{2\gamma-1}}$.
Now if the gain(x, y) = $2^{\alpha}3^{\beta}5^{\gamma}$, the sizes of x and y will be reduced by $\alpha + \beta\log_2 3 + \gamma\log_2 5$ bits and hence the average number of bits(denoted as ψ) eliminated at each step of Algorithm II is given by

$$\Psi = \sum_{\alpha=0}^{\infty}\sum_{\beta=0}^{\infty}\sum_{\gamma=0}^{\infty} p_{\alpha,\beta,\gamma}(\,\alpha + \beta\log_2 3 + \gamma\log_2 5)$$
$$= \sum_{\alpha=0}^{\infty}\sum_{\beta=0}^{\infty}\sum_{\gamma=0}^{\infty} \frac{\alpha + \beta\log_2 3 + \gamma\log_2 5}{2^{2\alpha+1}3^{2\beta-3}5^{2\gamma-1}} \qquad (10)$$

If we take, $\alpha = 5, \beta = 4, \gamma = 2$, we get $\psi \geq 3.1253$. This values of α, β and γ are assumed because they cover almost 100% of the cases. Density measure the ratio of the number of terms in the representation of any integer in any system and the binary length of the number and this is proportional to the reciprocal of ψ and its maximum value is 0.4 where the maximum value of density in JBTRS is 0.45. Since the density is less than that for JBTRS, the number of terms in the representation of any integer in JTBNS is also less and hence the number of adder required is also less which shows the novelty of the Proposed Algorithm. Again since the sequence in the JTBNS chain is arranged in descending order of the power of the bases of 2, 3 and 5, the first product term give the maximum power of the bases of 2, 3 and 5. If these are represented as a_1 , b_1 and c_1, then they can be written as a_1 = max{ a_i, a_j}, b_1 = max{b_i, b_j} and c_1 = max{c_i, c_j}, where a_i, a_j, b_i, b_j and c_i, c_j are the power of the base 2, 3 and 5 respectively in the joint expansion of the two integers m and n. The value of a_1, b_1 and c_1 are max{ 0.3971 $\log_2 m$, 0.3971 $\log_2 n$}, max{ 0.1985 $\log_2 m$, 0.1985 $\log_2 n$} and max{ 0.1241 $\log_2 m$, 0.1241 $\log_2 m$} respectively which are calculated based on the equation (11). Equ. (11) gives the average gain at each step and its value is equal to 5.0367. But the average gain at each step using JDBNS as mentioned in [8] is 4.3774 and the value of a_1 and b_1 accordingly are max{0.4569$\log_2 m$, 0.4569$\log_2 n$} and max{ 0.3427 $\log_2 m$, 0.3427 $\log_2 n$} respectively, where a_1= max {a_i, a_j} and b_1 = max{ b_i, b_j}, a_i, a_j and b_i , b_i being the exponents of the bases of 2 and 3 respectively in the in the joint expansion of the two integers m and n. Table 1 gives a comparative study of JTBNS with respect to JDBC, JSF and JBTRS.

$$\text{Average Gain at each step} = \sum_{\infty=5}^{\infty}\sum_{\beta=1}^{\infty}\sum_{\gamma=1}^{\infty}\frac{\begin{array}{c}2(\infty-1+\beta\log_2 3+\gamma\log_2 5)+2\max(\infty-1+\gamma\log_2 5,1+\beta\log_2 3)+2\\ \max(\infty-1+\beta\log_2 3,1+\gamma\log_2 5)+2\max(1+\beta\log_2 3+\gamma\log_2 5,\infty-1)+\\ 2\max((1+\gamma\log_2 5,\infty-1+\beta\log_2 3)+2\max(1+\beta\log_2 3,\infty-1+\gamma\log_2 5)\end{array}}{2^{\infty-3}.3^{\beta}.5^{\gamma}} \quad (11)$$

Table 1
Comparative study of JTBNS with respect to JSF (Joint Sparse Form), JDBC(Joint Double Base Chain) and JBTRS(Joint Binary Ternary Representation System) method for two integers 542788 and 462444.

Method	Density	Number of doublers	Number of triplers	Number of pentuplers	Number of adders
JSF	0.5	19	-	-	9
JDBC	0.5	14	3	-	9
JBTA	0.45	11	5	-	8
HBTNS	0.4	10	5	-	4
JTBNS	0.4	7	5	1	6

From the above Table, it is clear that JTBNS is only comparable with HBTNS, but HBTNS requires a pre-computation Look-up-table. Also the number of doublers required for HBTNS is more than that required for JTBNS at the cost of adders. But the complexity to design a doubler is more than to design an adder. Hence JTBNS is advantageous from all respect.

5. CONCLUSIONS

From the above discussions it is clear that using JTBNS Algorithm, it is possible to find out the product of the samples with the coefficients in designing a digital filter. Also in ECC, JTBNS finds it application to perform scalar multiplication and multi-scalar exponentiation. The conversion of a pair of integer can be achieved easily using the proposed algorithm with greater efficiency. The VHDL architecture following this Algorithm performs the said coefficient multiplication in DF and scalar multiplication in case of ECC. The design of the architecture that converts an integer into JTBNS form is also simple using less number of hardware as shown in Table 1.

6. REFERENCES

[1] Christophe Doche, David R. Kohel, and Francesco Sica, “Double-Base Number System for Multi- scalar Multiplications”, Draft, September, 9, 2008.

[2] Doche, C., Habsieger, L., “A Tree-Based Approach for Computing Double-Base Chains”, in: Y. Mu, W. Susilo and J. Seberry(Eds.), ACISP 2008, LNCS 5107, PP. 433-446, 2008, Springer-Verlag Berlin Heidelberg 2008.

[3] Avanzi, R.M., Cohen, H., Doche, C., Frey, G., Nguyen, K., Lange, T., Vercauteren, F.: Handbook of Elliptic and Hyperelliptic Curve Cryptography, in: Discrete Mathematics and its Application, Chapman and Hall/CRC, Boca Raton(2005).

[4] J. A. Solinas, “Low-weight binary representations for pairs of integers”, Center for Applied Cryptographic Research, University of Waterloo, Waterloo, ON, Canada, Research Report CORR 2001-41, 2001.

[5] Avanzi, R.M., Dimitrov, V.S., Doche, C., Sica, F.: Extending Scalar Multiplication using Double Bases, in: Lai, X., Chen, K.(Eds.), ASIACRYPT 2006, LNCS, vol. 4284, pp. 130 – 144, Springer, Heidelberg(2006).

[6] V. Dimitrov and T. V. Cooklev, “Two algorithm for modular exponentiation based on nonstandard arithmetic”, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Science, vol. E78-A, no. 1, pp. 82 -87, Jan. 1995, special issue on cryptography and information security.

[7] A.D. Booth, “A Signed binary multiplication technique”, Quarterly Journal of Mechanics and Applied Mathematics, vol. 4, no. 2, pp. 236 – 240, 1951, reprinted in E. E. Swartzlander, Computer Arithmetic, vol. 1, IEEE Computer Society Press Tutorial, Los Alamitos, CA, 1990.

[8] J. Adikari, V. Dimitrov, and L. Imbert. Hybrid Binary-Ternary Joint Sparse Form and its Application in Elliptic Curve Cryptography. Preprint, Available at: http://eprint.iacr.org/2008.

[9] M. Ciet, M. Joye, K. Lauter, and P. L. Montgomery. Trading Inversions for Multiplications in Elliptic Curve Cryptography. Des. Codes Cryptogr., 39(2):189–206, 2006.

[10] C. Doche and L. Imbert, “Extended double-base number system with applications to elliptic curve cryptography”, in Progress in Cryptography, INDOCRYPT’06,ser. Lecture Notes in Computer Science, vol. 4329, Springer, 006, pp. 335 – 348.

[11] M. Ciet, T. Lange, F. Sica, and J. J. Quisquater. Improved algorithms for efficient arithmetic on elliptic curves using fast endomorphism. In Advances in Cryptology – Eurocrypt 2003, volume 2656 of Lecture Notes in Comput. Sci., pages 388– 400. Springer-Verlag, 2003.

[12] S. Maitra, A. Sinha, “Triple-Base Hybrid Joint Sparse Form and its Applications”, International Journal of Computer Applications (0975 – 8887), vol. 43, No. 3, April, 2012.

[13] Pavel Sinha, Amitabha Sinha, Krishanu Mukherjee and Kenneth Alan Newton, “Triple Base Number Digital and Numerical Processing System”, Patent filed under E. S. P. Microdesign Inc., Pennsylvania, U.S.A., U. S. Pat. App. No. 11/488, 138.

[14] S. Maitra, A. Sinha, “A Single Digit Tripple Base Number System – A New Concept for Implementing

High Performance Multiplier Unit for DSP Applications", Proceedings of the sixth International Conference on Information, Communication and Signal Processing (ICICS2007), December, 10-13,2007.

[15] V. S. Dimitrov, G. A. Jullien and W. C. Miller, "Theory and Application of Double-Base Number System", IEEE Transaction on Computers, vol. 48, No. 10, pp-1098-1106, October, 1999.