

A New Aspect of Dual Basis for Efficient Field Arithmetic

Chang-Hyi Lee¹ and Jong-In Lim²

¹ Samsung Advanced Institute of Technology,
chang@saitgw.sait.samsung.co.kr
² Korea University,
jilim@tiger.korea.ac.kr

Abstract. In this manuscript we consider the special type of dual basis for finite fields, $GF(2^m)$, where the variants of m are presented in the following contents. Here we introduce our field representing method for its efficient arithmetic (of field multiplication and field inversion). It revealed a very effective role for both software and VLSI implementations, but the aspect of hardware design for its structure is out of this manuscript and so, here, we deal only the case of its software implementation (the efficiency of hardware implementation is appeared in another article submitted to IEEE Transactions on Computers). A brief description of this advantageous characteristics is that (1) the field multiplication can be constructed only by $k(\leq \frac{m}{2})$ rotations and the same amount of vector XOR processes, (2) there is needed no additional work load as basis changing (from standard to the dual basis or from the dual basis to standard basis as the conventional dual based arithmetic does), (3) the field squaring is only bit-by-bit permutation and it has a good regularity for its implementation, and (4) the field inversion process is available for both cases of its implementation using Fermat's Theorem and of its implementation using *almost inverse* algorithm [14], especially the case of using the *almost inverse* algorithm has an additional advantage in finding (computing) its complete inverse element.

1 Introduction

Field arithmetic is fundamental to the implementation of Elliptic Curve Cryptosystem (ECC), one of the public key cryptosystems suggested by N. Koblitz [1] and V. Miller [2]. These requirements need the efficient implementations for field arithmetic operations. Among these operations, the field multiplication and the field inversion are very critical to the time/hardware complexity for their software and hardware implementations. In general, there are three methods to represent a field element in $GF(q^m)$, depending upon what types of field bases are used, polynomial basis (standard basis), or normal basis, or dual basis. It is known that the dual basis multipliers are the most hardware efficient multipliers available, both in bit-serial and in bit-parallel designs [3], [4], [5]. But in its software implementation as well as such a hardware implementation with dual

basis, one should pay some additional costs, such as basis conversion and a little of complexity of field squaring. Here, in this manuscript, we insist that there exists a special type of dual basis over which the performance and implementing effectiveness of field arithmetic (field multiplication and field inversion) surpass those over optimal normal basis[7][8][9] or trinomial standard basis (optimized polynomial basis).

Whatever basis is used for those field arithmetic processing, there are simplified and effective forms of bases, so called *trinomial standard basis*, optimal dual basis (using the trinomial dual basis)[5], optimal normal basis[10] (of which multiplication table has the least number of nonzero entries).

As said previously, in our special type of dual basis of $GF(2^m)$ over $GF(2)$, the field arithmetic processes can be accomplished with gaining both benefits of dual basis and the same effects using normal basis (or optimal normal basis), where m is the field extending dimension over $GF(2)$ on which the first type of optimal normal basis exists, i.e.

$$m = 4, 10, 12, 18, 28, 36, 52, 58, 60, 66, 82, 100, 106, 130, 138, 148, \\ 162, 172, 178, 180, 196, 210, 226, 268, 292, 316, 346, 348 \dots$$

Note that our dual basis is not a self-dual normal basis[11]. In the following subsection, we describe some mathematical background for general dual basis and in section 2, there is presented our method to implement field arithmetic processes. In section 3, we describe their implementing algorithms (for the aspect of their VLSI design, we have described it in another paper submitted to IEEE Trans. on Computers)

1.1 Dual Basis

Throughout this paper it is assumed that the reader is familiar with the basic theory of finite fields, and for more details one may refer to [12]. Let $Tr(\cdot)$ be the trace function of $GF(q^m)$ to $GF(q)$ defined by

$$Tr(a) = \sum_{i=0}^{m-1} a^{q^i},$$

which is a special linear functional (linear transformation) of $GF(q^m)$ to $GF(q)$. In general, there are q^m linear functionals of $GF(q^m)$ to $GF(q)$ and by them we can define the general duality between two bases $\{\varphi_i\}$ and $\{\psi_i\}$ for $GF(q^m)$ such as the following.

Definition 1. Let f be a nontrivial linear functional of $GF(q^m)$ to $GF(q)$. Any two bases, $\{\varphi_i\}$ and $\{\psi_i\}$, are called dual with respect to f if they satisfy the following equation.

$$f(\varphi_i \psi_j) = \begin{cases} 1 & , \text{if } i = j \\ 0 & , \text{if } i \neq j \end{cases}$$

Theorem 1. *Every basis for $GF(q^m)$ has its unique dual basis with respect to any given nontrivial linear functional f of $GF(q^m)$ to $GF(q)$.*

Theorem 2. *If $\{\varphi_i\}$ and $\{\psi_i\}$ are dual bases for $GF(q^m)$ to each other, then any element $a \in GF(q^m)$ represented in $\{\varphi_i\}$ can be rewritten in the form of*

$$a = \sum_{i=0}^{m-1} f(a\varphi_i)\psi_i$$

over the basis $\{\psi_i\}$.

Here we describe a bit serial multiplier due to Berlekamp[3] which uses a self-dual basis representation. Let

$$\varphi = \{1, \alpha, \alpha^2, \dots, \alpha^{m-1}\}$$

be a polynomial(standard) basis of $GF(2^m)$ over $GF(2)$, $p(t)$ is the minimal polynomial of α , and let

$$\psi = \{\psi_0, \psi_1, \dots, \psi_{m-1}\}$$

be its self-dual basis with respect to trace function $Tr(\cdot)$, i.e. $\psi_j = \alpha^j$, $j = 0, 1, \dots, m-1$ and $Tr(\alpha^i \psi_j) = 1$ if $i = j$ and $= 0$ if $i \neq j$. And let $p(t) = \sum_{i=0}^{m-1} p_i t^i$ is the minimal polynomial of α over $GF(2)$, i.e. the defining irreducible polynomial for $GF(2^m)$. Then for each $x \in GF(2^m)$, we have

$$x = \sum_{i=0}^{m-1} x_i \alpha^i = \sum_{i=0}^{m-1} Tr(x \alpha^i) \psi_i = \sum_{i=0}^{m-1} [x]_i \psi_i,$$

where $[x]_i$ are the coordinates of x with respect to the dual basis ψ . Then we have

$$[\alpha x]_j = Tr(\alpha x \cdot \alpha^j) = Tr(\alpha^{j+1} x) = [x]_{j+1},$$

for $0 \leq j \leq m-2$, and

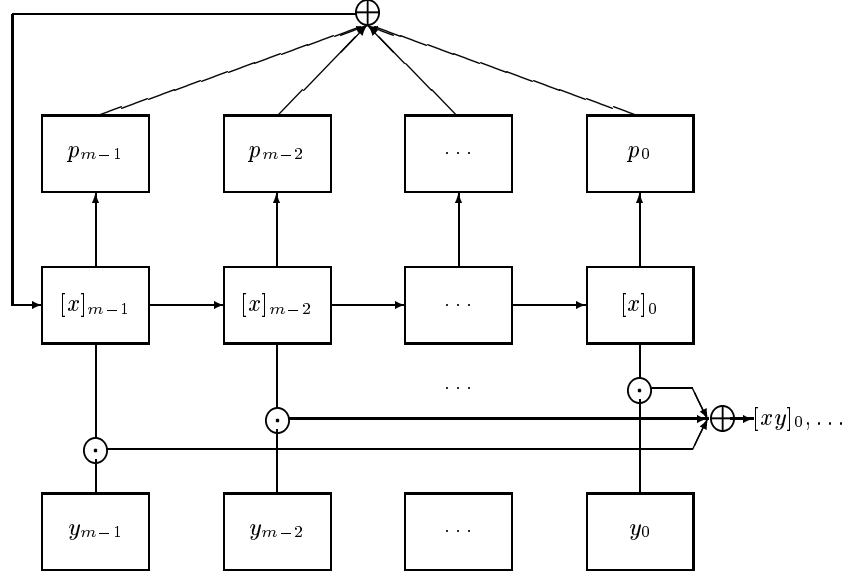
$$\begin{aligned} [\alpha x]_{m-1} &= Tr(\alpha^m x) = Tr(\sum_{i=0}^{m-1} p_i \alpha^i x) \\ &= \sum_{i=0}^{m-1} p_i Tr(\alpha^i x) = \sum_{i=0}^{m-1} p_i [x]_i. \end{aligned}$$

This mechanism is in charge of the very critical role in the field multiplication based on a dual basis. For the concrete explanation, let $y = \sum_{i=0}^{m-1} y_i \alpha^i \in GF(2^m)$, then for each k , $0 \leq k \leq m-1$,

$$[xy]_k = \left[x \sum_{j=0}^{m-1} y_j \alpha^j \right]_k = \sum_{j=0}^{m-1} y_j [\alpha^j x]_k = \sum_{j=0}^{m-1} y_j [\alpha^k x]_j.$$

Thus the product of x and y in dual coordinates can be obtained by the computation diagram in Fig.1, in that figure, \oplus denotes the bit-by-bit XOR operation

Fig. 1. Block diagram for general dual basis multiplication



and \odot denotes the bit-by-bit logic AND operation which are the two field operations in $GF(2)$.

As seen in Fig.1, in the general case(non self-dual), the product of two elements in $GF(2^m)$ using dual basis representation requires the extra work to convert one of the two elements into the representation over its dual basis(or from dual to standard basis) and it can be easily checked to note that for the simplicity of the upper XOR-summation part in Fig.1, one selects a very simple(small number of nonzero terms) irreducible polynomial, i.e., trinomial, etc., for *optimized dual basis*[10].

From the Fig.1, we can easily get the following theorem in generalized form.

Theorem 3. *Let α be a root of the defining irreducible polynomial for the field, $GF(q^m)$. And let $\psi = \{\psi_i\}$ be the dual basis to the standard basis $\varphi = \{\alpha^i\}$ with respect to a linear functional f of $GF(q^m)$ to $GF(2)$.*

If, for $x, y, z \in GF(q^m)$, $xy = z$ then the following relation holds. (in the following, $[\cdot]_j$ means the j^{th} coefficient over the dual basis ψ .)

$$\begin{bmatrix} f(y) & f(y\alpha) & \cdots & f(y\alpha^{m-1}) \\ f(y\alpha) & f(y\alpha^2) & \cdots & f(y\alpha^m) \\ \vdots & \vdots & \ddots & \vdots \\ f(y\alpha^{m-1}) & f(y\alpha^m) & \cdots & f(y\alpha^{2m-2}) \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{m-1} \end{bmatrix} = \begin{bmatrix} f(z) \\ f(z\alpha) \\ \vdots \\ f(z\alpha^{m-1}) \end{bmatrix} \quad (1)$$

$$= ([z]_0, [z]_1, \dots, [z]_{m-1})^t$$

Let's denote the above matrix equation by $M(\mathbf{y}) \cdot \mathbf{x}^t = \mathbf{z}^t$, where the superscript, t , denotes the transposition.

2 Description of the theoretic evolution

2.1 Circular Dual Basis

Throughout the following, it is assumed the characteristic of $GF(q)$ is 2, i.e. q is a power of 2. In this section we propose a new type of dual basis named *circular dual basis* and extract some useful results from it for the field arithmetic.

Definition 2. Let $p(t)$ be a nonzero polynomial over $GF(q)$, where q is a prime power, and $p(0) \neq 0$. Then the least positive integer e for which $p(t)$ divides $t^e - 1$ is called the order (or period) of p and denoted by $\text{ord}(p(t))$.

For the convenience, let's call the polynomial

$$p(t) = 1 + t + t^2 + \dots + t^m \in GF(q)[t]$$

by *compact polynomial* of degree m over $GF(q)$.

Theorem 4. For a positive integer $m \geq 2$, if $m + 1$ is prime number and m is the multiplicative order of q modulo $m + 1$, then there exists a unique irreducible compact polynomial $p(t)$ of which degree m and of which order is $m + 1$. That is, $p(t) = 1 + t + t^2 + \dots + t^m$ is irreducible over the field and it divides $t^{m+1} - 1$.

Proof. By theorem 3.5 in [12], the number of monic irreducible polynomials in $GF(q)[t]$ of degree m and order e is equal to $\phi(e)/m$, where ϕ is a Euler's function. And in our case, $e = m + 1$, and so $\phi(e) = m$. Hence the unique monic polynomial of degree m which divides $t^{m+1} - 1$ is the very *compact polynomial*. \square

We call by *circular polynomial* such compact and *irreducible* polynomial of which order is $m + 1$ as in the above and call by *circular dual basis* the dual basis of circular polynomial. For the concrete example, considering those over the field $GF(2)$, we are led to the following result.

Theorem 5. Over the field $GF(2)$, there are circular polynomials of which degrees are one of the following values m ;

$$\begin{aligned} m &= 4, 10, 12, 18, 28, 36, 52, 58, 60, 66, 82, 100, 106, 130, 138, 148, \\ &162, 172, 178, 180, 196, 210, 226, 268, 292, 316, 346, 348 \dots \\ &= \text{the set of all dimensions for which the first} \\ &\quad \text{type of optimal normal basis over } GF(2) \text{ exist.} \end{aligned}$$

Proof. By the previous theorem, each degrees m of the circular polynomial must be a positive number such that $q = 2$ is a primitive element modulo the prime number $m + 1$. And these are the same with the set of numbers which are the degrees for which the first type of optimal normal basis exists[10][13]. \square

Throughout the following, it is assumed that $p(t) = \sum_{i=0}^m t^i$ is a circular polynomial of degree m over $GF(q)$, i.e. the irreducible polynomial over $GF(q)$ of degree m which divides $(t^{m+1} - 1)$. The m -dimensional extended field $GF(q^m)$ over $GF(q)$ defined by the circular polynomial $p(t)$ has the standard basis

$$\varphi = \{1, \alpha, \alpha^2, \dots, \alpha^{m-1}\},$$

where α is a root of $p(t)$.

Theorem 6. *If we fix the linear functional in theorem 2.3 by trace map $Tr(\cdot)$ of $GF(q^m)$ to $GF(q)$, the dual basis $\psi = \{\psi_0, \psi_1, \dots, \psi_{m-1}\}$ to the standard basis φ are represented in the following form:*

$$\begin{aligned} \psi_i &= \alpha + \alpha^{-i} \\ &= \alpha + \alpha^{m+1-i}. \end{aligned}$$

Those can be also represented in the basis elements α^i 's of φ such as;

$$\psi_0 = 1 + \alpha, \tag{2}$$

$$\psi_1 = s + \alpha, \quad s = \sum_{i=0}^{m-1} \varphi_i, \tag{3}$$

$$\psi_j = \alpha^{m+1-j} + \alpha, \quad j = 2, 3, \dots, m-1. \tag{4}$$

Proof. By the theorem 1.5 in [13], if

$$g(t) = (t - \alpha)(\beta_0 + \beta_1 t + \dots + \beta_{m-1} t^{m-1}), \tag{5}$$

then the dual basis is $\psi = \{\psi_i\}$, where

$$\psi_i = \frac{\beta_i}{g'(\alpha)}, \quad i = 0, 1, \dots, m-1. \tag{6}$$

Through the expansion of equation(5), we can easily get the representations of β_i 's in the polynomial form of α^{-1} ;

$$\beta_i = \sum_{j=1}^{i+1} (\alpha^{-1})^j, \quad i = 0, 1, \dots, m-1.$$

Furthermore $g'(\alpha) = 1 + \alpha^2 + \alpha^4 + \dots + \alpha^{m-2}$, and note that

$$\begin{aligned} (1 + \alpha)g'(\alpha) &= 1 + \alpha + \alpha^2 + \dots + \alpha^{m-1} = \alpha^{-1} \\ \Rightarrow g'(\alpha) &= \frac{1}{\alpha(1 + \alpha)}. \end{aligned} \tag{7}$$

Hence, by equation(6) and equation(7), we get

$$\begin{aligned}
\psi_i &= \frac{\beta_i}{g'(\alpha)} \\
&= (\alpha + \alpha^2) \sum_{j=1}^{i+1} (\alpha^{-1})^j \\
&= (\alpha + \alpha^2)(1 + \alpha + \alpha^2 + \dots + \alpha^{m-i-1}) \\
&= \alpha + \alpha^{m+1-i}.
\end{aligned}$$

□

From the above, we easily get the following basis changing matrix D_0 from the dual basis ψ to the standard basis φ ;

$$(x_i)^t = D_0([x]_i)^t = \begin{bmatrix} 1 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & \cdots & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & \cdots & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & \cdots & 0 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 1 & 0 & 1 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & \cdots & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} [x]_0 \\ [x]_1 \\ [x]_2 \\ [x]_3 \\ \vdots \\ [x]_{m-2} \\ [x]_{m-1} \end{bmatrix} \quad (8)$$

Here we introduce one(bit) additional coefficient $s[x]$ for the utilized representation of $\mathbf{x} = ([x]_i) \in GF(2^m)$ over the circular dual basis such as the following:

$$\begin{aligned}
s[x] &= \sum_{i=0}^{m-1} [x]_i \\
\mathbf{x} &= ([x]_0, [x]_1, \dots, [x]_{m-1}) \text{ over the dual basis} \\
\bar{\mathbf{x}} &:= ([x]_0, [x]_1, \dots, [x]_{m-1}, s[x]) \quad (9)
\end{aligned}$$

We call such the representation $\bar{\mathbf{x}}$ by circular dual based representation. By this notion, the equation(8) can be rewritten as:

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{m-1} \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & \cdots & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & \cdots & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 1 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} [x]_0 \\ [x]_1 \\ [x]_2 \\ [x]_3 \\ \vdots \\ [x]_{m-1} \\ s[x] \end{bmatrix} \quad (10)$$

$$\Rightarrow \text{let's abbreviate this eq. by } \mathbf{x}^t = D\bar{\mathbf{x}}^t \quad (11)$$

Using the above utilities, we got the following theorem.

Theorem 7. Let $\mathbf{x}, \mathbf{y}, \mathbf{z} \in GF(2^m)$, $\mathbf{z} = \mathbf{x}\mathbf{y}$,

$$\bar{\mathbf{x}} = ([x]_0, [x]_1, \dots, [x]_{m-1}, s[x])$$

$$\bar{\mathbf{y}} = ([y]_0, [y]_1, \dots, [y]_{m-1}, s[y])$$

$$\bar{\mathbf{z}} = ([z]_0, [z]_1, \dots, [z]_{m-1}, s[z])$$

be the their circular dual based representations. Then, the equation(1) can be rewritten as the following form:

$$\begin{aligned} \bar{\mathbf{z}} &= \bar{\mathbf{x}} \begin{bmatrix} [y]_0 & [y]_1 & [y]_2 & \cdots & [y]_{m-2} & [y]_{m-1} & s[y] \\ s[y] & [y]_0 & [y]_1 & \cdots & [y]_{m-3} & [y]_{m-2} & [y]_{m-1} \\ [y]_{m-1} & s[y] & [y]_0 & \cdots & [y]_{m-4} & [y]_{m-3} & [y]_{m-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ [y]_2 & [y]_3 & [y]_4 & \cdots & s[y] & [y]_0 & [y]_1 \\ [y]_1 & [y]_2 & [y]_3 & \cdots & [y]_{m-1} & s[y] & [y]_0 \end{bmatrix} \\ &= \sum_{i=0}^m [x]_i \text{Rot}_r(\bar{\mathbf{y}}, i), \quad [x]_m := s[x], \end{aligned} \quad (12)$$

where $\text{Rot}_r(\bar{\mathbf{y}}, i)$ denotes the rotated vector of $\bar{\mathbf{y}}$ to the right by i positions.

Proof. Note that

$$\begin{aligned} \text{Tr}(\mathbf{y}\alpha^i) &= [y]_i \quad i = 0, 1, \dots, m-1, \\ \text{Tr}(\mathbf{y}\alpha^m) &= \text{Tr}(\mathbf{y} \sum_{i=0}^{m-1} \alpha^i) = \sum_{i=0}^{m-1} [y]_i = s[y], \\ &\alpha^{m+1} = 1. \end{aligned}$$

Using these notions, the equation(1) can be rewritten as in the following one bit expanded form.

$$\begin{bmatrix} [y]_0 & [y]_1 & \cdots & [y]_{m-1} & s[y] \\ [y]_1 & [y]_2 & \cdots & s[y] & [y]_0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ [y]_{m-1} & s[y] & \cdots & [y]_{m-3} & [y]_{m-2} \\ s[y] & [y]_0 & \cdots & [y]_{m-2} & [y]_{m-1} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{m-1} \\ 0 \end{bmatrix} = \begin{bmatrix} [z]_0 \\ [z]_1 \\ \vdots \\ [z]_{m-1} \\ \delta \end{bmatrix}, \quad (13)$$

$$\Rightarrow \text{let's abbreviate this eq. by } M(\bar{\mathbf{y}})\mathbf{x}^t = \bar{\mathbf{z}}^t \quad (14)$$

where δ is not determined yet. It will be turn out to be $\sum_{i=0}^{m-1} [z]_i$.

Now, using the equation(10) and (11), we replace the standard based representation of \mathbf{x} appeared in the above equation by our circular dual based representation, then we got:

$$MD\bar{\mathbf{x}}^t = \bar{\mathbf{z}}^t \quad (15)$$

Apply the transposition function to the both sides, then we are led to the new equation:

$$\bar{\mathbf{x}}D^tM^t = \bar{\mathbf{z}} \quad (16)$$

Hence we get to know that the left hand side of equation(16) is the right hand side of the theorem. Furthermore, the fact that the sum of all row vectors in D^tM^t is zero-vector gives us the result $\delta = \sum_{i=0}^{m-1} [z]_i$. This completes the proof. \square

In the following section, with these results we deal the field multiplication, squaring circuit, and field inversion.

3 Description of the technique(algorithm)

In this section we describe the algorithms for efficient field arithmetic operations based on the previous results.

3.1 Field Multiplication

As one sees in the previous section, we got to know that

$$\begin{aligned} \mathbf{x} &= ([x]_0, [x]_1, \dots, [x]_{m-1}, [x]_m), [x]_m = \sum_{i=1}^{m-1} [x]_i \\ \mathbf{y} &= ([y]_0, [y]_1, \dots, [y]_{m-1}, [y]_m), [y]_m = \sum_{i=1}^{m-1} [y]_i \\ \mathbf{z} &= \mathbf{xy} \\ &= \sum_{i=0}^m [x]_i Rot_r(\mathbf{y}, i). \end{aligned}$$

From this we can easily construct the following very simple field multiplication algorithm in pseudo-code(Table 1); But note that $\sum_{i=0}^m Rot_r(\mathbf{y}, i) = zero\ vector$.

Table 1. Field multiplication algorithm(1)

```

INPUT :  $\mathbf{x}, \mathbf{y}$  : circular dual based elements in  $GF(2^m)$ .
OUTPUT :  $\mathbf{z} = \mathbf{xy}$  in circular dual based representation.
 $\mathbf{z} \leftarrow 0$  /* initialize  $\mathbf{z}$  to be zero vector */
for ( $i = 0; i \leq m; i++$ )
{
    if ( $[x]_i \neq 0$ )  $\mathbf{z} \leftarrow \mathbf{z} \oplus Rot_r(\mathbf{y}, i)$ ;
}

```

From this notion, we get more developed algorithm(Table 2);

Hence we can achieve $\mathbf{z} = \mathbf{xy}$ by only $k(\leq \frac{m}{2})$ vector rotations and the same amount of vector XOR operations.

Table 2. Field multiplication algorithm(2)

INPUT : \mathbf{x}, \mathbf{y} : circular dual based elements in $GF(2^m)$.
OUTPUT : $\mathbf{z} = \mathbf{xy}$ in circular dual based representation.
 $\mathbf{z} \leftarrow 0$ /* initialize \mathbf{z} to be zero vector */
if ((# of nonzero entries of \mathbf{x}) $> \frac{m}{2}$) negate \mathbf{x} ;
for ($i = 0; i \leq m; i++$)
{
if ($[x]_i \neq 0$) $\mathbf{z} \leftarrow \mathbf{z} \oplus Rot_r(\mathbf{y}, i)$;
}

3.2 Squaring

From now on, it is assumed that all coefficients of any element in $GF(2^m)$ over $GF(2)$ were represented over the circular dual basis defined in the above.

Theorem 8. Let $\mathbf{x}, \mathbf{y} \in GF(2^m)$, $\mathbf{y} = \mathbf{x}^2$, and

$$\begin{aligned}\mathbf{x} &= ([x]_0, [x]_1, \dots, [x]_{m-1}, [x]_m = s[x]) \\ \mathbf{y} &= ([y]_0, [y]_1, \dots, [y]_{m-1}, [y]_m = s[y]).\end{aligned}$$

Then the following equation holds;

$$\mathbf{y}^t = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & 0 & \dots & 1 & 0 & 0 & \dots & 0 & 0 \end{bmatrix} \mathbf{x}^t \quad (17)$$

That is,

$$[y]_{2i} = [x]_i, \quad i = 0, 1, \dots, \frac{m}{2} \quad (18)$$

$$[y]_{2j-1} = [x]_{j+\frac{m}{2}}, \quad j = 1, 2, \dots, \frac{m}{2}. \quad (19)$$

Proof. Since the characteristic of $GF(2^m)$ is 2, the squaring map is a linear transform. So it is sufficient for us to check the behavior of basis elements (ψ_i 's) via the squaring transformation. First, recall that $\psi_j = \alpha + \alpha^{-j \bmod (m+1)}$ for $j \neq 1$, then

$$\psi_j^2 = \alpha^2 + \alpha^{-2j}$$

$$\begin{aligned}
&= (\alpha^2 + \alpha) + (\alpha^{-2j} + \alpha) \\
&= \psi_{m-1} + \psi_{2j}, \text{ for } j = 0, 1, \dots, \frac{m}{2} - 1 \\
\psi_{m/2}^2 &= \alpha^2 + \alpha^{m+2} \\
&= \alpha^2 + \alpha, \text{ since } \alpha^{m+1} = 1 \\
&= \psi_{m-1} \\
\psi_{m/2+j}^2 &= \alpha^2 + \alpha^{m+1-(2j-1)} \\
&= (\alpha^2 + \alpha) + (\alpha^{m+1-(2j-1)} + \alpha) \\
&= \psi_{m-1} + \psi_{2j-1} \text{ for } j = 1, 2, \dots, \frac{m}{2} - 1
\end{aligned}$$

Hence

$$\begin{aligned}
[y]_{2j} &= [x]_j, \quad j = 0, 1, \dots, \frac{m}{2} - 1, \\
[y]_{2j-1} &= [x]_{m/2+j}, \quad j = 1, 2, \dots, \frac{m}{2} - 1 \\
[y]_{m-1} &= \sum_{i=0}^{m-1} [x]_i = s[x]
\end{aligned}$$

and moreover,

$$\begin{aligned}
s[y] &= \sum_{i=0}^{m-1} [y]_i \\
&= \sum_{i=0}^{m/2-1} [x]_i + \sum_{i=1}^{m/2-1} [x]_{m/2+i} + s[x] \\
&= [x]_{m/2}.
\end{aligned}$$

This proves the theorem. \square

In the above theorem, the squaring is only a bit-by-bit permutation based on the circular dual basis, that is this squaring process needs no logic operations. Let $EXPAND_k(\cdot)$ be the function of $GF(2^k)$ to $GF(2^{2k-1})$ defined as, for given $\mathbf{x} \in GF(2^k)$:

$$\begin{aligned}
EXPAND_k(\mathbf{x})_{2j+1} &= 0, \quad j = 0, 1, 2, \dots, k-2 \\
EXPAND_k(\mathbf{x})_{2j} &= \mathbf{x}_j, \quad j = 0, 1, 2, \dots, k-1
\end{aligned}$$

Then the squaring can be described by the following simple algorithm (Table 3), in pseudo-code;

where the function $EXPAND_k(\cdot)$ can be easily implemented by small memory of table and its performance was presented in section §6.

3.3 Conversion to Standard Basis

We showed that the basis conversion matrix of circular dual basis to its standard basis is the matrix D_0 appeared in the equation(8). From that equation, we get

Table 3. Squaring algorithm

INPUT : $\mathbf{x} = ([x]_0, [x]_1, \dots, [x]_{m-1}, [x]_m = s[x]);$
OUTPUT : \mathbf{x}^2 in circular dual based form;
 $\mathbf{x}^L \leftarrow ([x]_0, [x]_1, \dots, [x]_{\frac{m}{2}});$
 $\mathbf{x}^H \leftarrow ([x]_{\frac{m}{2}+1}, [x]_{\frac{m}{2}+2}, \dots, [x]_m, 0);$
Return $\mathbf{x}^2 \leftarrow EXPAND_{\frac{m}{2}}(\mathbf{x}^L) \oplus (EXPAND_{\frac{m}{2}}(\mathbf{x}^H) \gg 1);$

the following conversion equation in one dimensional expanded form(here, x_i 's denote the coordinates over the standard basis and $[x]_i$'s denote the coordinates over its circular dual basis of \mathbf{x}):

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{m-1} \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} [x]_0 \\ [x]_1 \\ [x]_2 \\ [x]_3 \\ \vdots \\ [x]_{m-1} \\ [x]_m \end{bmatrix} + \begin{bmatrix} [x]_1 \\ [x]_1 \\ [x]_1 \\ [x]_1 \\ \vdots \\ [x]_1 \\ [x]_1 \end{bmatrix}. \quad (20)$$

This equation generates the following simple representation-conversion algorithm(Table 4) of a field element from circular dual basis to its standard basis:

Table 4. Conversion to the standard based representation

INPUT : $\bar{\mathbf{x}} = ([x]_0, [x]_1, \dots, [x]_m)$ over Cir.Dual Basis.
OUTPUT : $\mathbf{x} = (x_0, x_1, \dots, x_{m-1}, 0)$ over Stand.Basis.
1. $\mathbf{x} \leftarrow$ rotate $\bar{\mathbf{x}}$ by one bit to the left;
2. if $([x]_1 \neq 0)$ $\mathbf{x} \leftarrow$ negation of \mathbf{x} ;
3. $\mathbf{x} \leftarrow$ take the reciprocal of \mathbf{x} ;

But, simply, we need only the line 1 and 2 for the inversion algorithm in the following subsection.

3.4 Field Inversion

We point out that, from the previous results, we can construct inversion algorithm by use of the hereto made multiplication and squaring algorithms and Fermat's Theorem like as the optimized one in [10][9][7]. But here we construct it by some more fast inversion algorithm using *almost inverse algorithm*[14] and its advantage in our circular dual basis.

Let $p(t) = 1 + t + \dots + t^m$ be a circular polynomial, i.e. irreducible polynomial and $p(\alpha) = 0$ in $GF(2^m)$, then $p(t)$ is a self-reciprocal, i.e. $t^m p(\frac{1}{t}) = p(t)$ and $\alpha^{m+1} = 1$. Let $\alpha^{-1} = u$ and let

$$\begin{aligned} A &= a_0 + a_1\alpha + a_2\alpha^2 + \dots + a_{m-1}\alpha^{m-1} := A(\alpha) \in GF(2^m) \\ &= \alpha^{m-1}(a_{m-1} + a_{m-2}u + a_{m-3}u^2 + \dots + a_0u^{m-1}) \\ &= \alpha^{m-1}A(u) \in GF(2^m). \end{aligned}$$

Then the coefficient strings of $A(\alpha)$ and $A(u)$ in α and u respectively are reciprocal to each other. We can apply the *almost inverse* algorithm to $A(u)$ with the reduction polynomial $p(u) \in GF(2)[u]$. That is, We can find a polynomial in u , $B(u)$, such that

$$A(u)B(u) = u^k \pmod{p(u)} \quad (21)$$

for some integer k , $0 \leq k \leq 2(m-1)$. And so $B(\alpha) = \alpha^{m-1}B(u)$ is an almost inverse of $A(\alpha)$ with reduction polynomial $p(\alpha)$. Now the exact inverse of $A(\alpha)$ comes to the following (by multiplying $\alpha^{2(m-1)}$ to both sides of equation(21));

$$\begin{aligned} A(\alpha)^{-1} &= \alpha^{m-1}B(u)(u^k \alpha^{2(m-1)})^{-1} \\ &= B(\alpha)\alpha^{(k-2(m-1)) \bmod m+1} \end{aligned} \quad (22)$$

Note that, following the above notations, first, to converting the circular based element \mathbf{x} into the u -polynomial type of $A(u)$ is very simple and easy (see the section §3.3) and second, α^j , $j = 1, 2, \dots, m$, corresponds to the circular dual based element, δ_{m-j+1} , of which all components are zero except only the $(m-j+1)_{th}$ component. The later tells us that the product of a circular dual based element \mathbf{x} and α^j in the circular dual based form can be easily achieved only by one right-rotation of \mathbf{x} by $m-j+1$ positions (see section §3.1).

Therefore we can describe the inversion algorithm (Table 5) for the circular dual based representation by the following pseudo-code (in the code, $ROTL_s(*, i)$ denotes the left-rotation of $*$ by i bits in the total size of s bits, and so does $ROTR_s(*, i)$ except the rotating direction converted):

In the algorithm (Table 5), the lines indexed 11-12 in table-5 carry out the coefficient conversion process of circular dual basis to the reciprocal(u -polynomial) of the standard basis, the lines indexed 21-26 in table-5 carry out the *almost inverse* algorithm, and, finally, the lines 31-36 carry out the coefficient conversion process of the u -polynomial to the circular dual basis. Note that, in the lines 22, 32, the division by u and the multiplication by u are simply accomplished by the left and right shift operations respectively. From this we see that, in the above inverse calculating algorithm, the representation converting parts have very negligible implementation complexity comparing with just the *almost inverse* algorithm.

In this section we did not mention about our algorithm to solve a quadratic equation, since its performance do not have an influence on the performance or efficiency of the whole elliptic curve arithmetic. But even in that case our circular dual based representation gives a very efficient resolution routine with very high (recursive) regularity and without any basis conversion and any table memory.

Table 5. Field inversion algorithm

INPUT : \mathbf{x} in circular dual based form;
OUTPUT : \mathbf{x}^{-1} in circular dual based form;

11. $A(u) \leftarrow ROTL_{m+1}(\mathbf{x}, 2)$;
12. if $(A(u)_m \neq 0)$ negate $A(u)$ in the first $m + 1$ bits;

21. set: $k = 0, B(u) = 1, C(u) = 0, F(u) = A(u), G(u) = p(u)$;
22. While $(F(0) = 0)$ do $F = F/u, C = C * u, k = k + 1$;
23. if $(F = 1)$ goto 31;
24. if $(\deg(F) < \deg(G))$, exchange F, G and B, C ;
25. $F = F \oplus G, B = B \oplus C$;
26. goto 22;

31. $t \leftarrow$ XOR sum of all coefficients in $B(u)$;
32. if $(t \neq 0)$ $B = \neg B + t * u^m$; /* $\neg B$: negation of $B(m$ bits) */
33. $\mathbf{x}^{-1} \leftarrow ROTR_{m+1}(B, 2)$;
34. $k \leftarrow (k - 2 * (m - 1)) \bmod m + 1$;
35. if $(k = 0)$ return \mathbf{x}^{-1} ;
36. return $\mathbf{x}^{-1} = ROTR_{m+1}(\mathbf{x}^{-1}, m - k + 1)$;

4 Attributes and advantages of the techniques

4.1 Advantages over other representation methods

There are many advantages of this representation method for field arithmetic. With hereto presented results, its advantages are described as the following.

1. There needed no basis changing process for field multiplication as the conventional dual basis does.
2. The field multiplication can be implemented by only vector rotations and vector XOR operations, and so its code is much simpler and its performance is much better than those over optimized normal bases.
3. The squaring of field element is just a simple bit-by-bit permutation and has a very high regularity.
4. The field inversion using *almost inverse*[14] algorithm can be implemented to calculate the exact inverse element with almost the same cost of just the *almost inverse* algorithm process, since the work to compute the production of the factor x^k is reduced into only one rotating operation.

4.2 Limitations of the method

As said in the previous sections, §1 and §2, the circular dual basis exists only for the finite field in which the first type of optimal normal basis exists. This, however, is not a severe limitation for the cryptographic system, since there are sufficiently many suitable fields for cryptographic goals.

Notes and Comments.

- In the practical implementation of the circular dual based field arithmetic for ECC(Elliptic Curve Cryptosystem) over $GF(2^m)$, the binary representation of a field element takes the reversed order of the previously written form for one's easy comprehension, i.e.

$$a = (a_0, a_1, \dots, a_{m-1}, a_m = s[a]) \rightarrow (a_m = s[a], a_{m-1}, \dots, a_1, a_0).$$

And the additional dummy bit, $s[a]$, should be appeared just during the inner ECC arithmetic process.

- Following the above representation form, one would become aware of the following two facts.

- a. multiplicative identity in $GF(2^m)$ is $(\overbrace{1, 1, \dots, 1}^{m+1}, 0)$
- b. For $a = (a_m, a_{m-1}, \dots, a_1, a_0) \in GF(2^m)$, $Tr(a) = a_0$

5 Intellectual property issues

There are several patents related to the use of general and conventional dual basis for encoder and decoder. In the below we present some of them. And we are going to apply our circular dual based method for a patent as an optimized implementation scheme for efficient field arithmetic based on such dual basis representation.

- US 5,467,297 : *Finite field inversion*
- US 4,994,995 : *Bit-serial division method and apparatus*
- US 4,567,568 : *Apparatus for dividing the elements of Galois field*
- US 4,410,989 : *Bit serial encoder*

6 Performance

In the following table 6, the process timings of our methods for various field operations in $GF(2^{178})$ are presented, wherein they are compared with other efficient software implementations[17][18] for field arithmetic in $GF((2^{16})^{11})$ by use of composite Galois Fields and by use of the multiplication table for $GF(2^{16})$ and trinomial standard basis. The present circular dual based implementation of our method, as one sees in the above table, shows that it is very efficient for its software implementation(as well as hardware implementation) and by noting that the platform, DEC alpha 3000, has a very efficient RISC architecture for parallel processing of multi-instructions and that the simple processing structure of our circular dual based field arithmetic has a good parallelism, we see that it would be more faster than the others on the same platforms, DEC alpha 3000, and more carefully optimized code would give more developed performance than our referencing C-code's.

Table 6. Timings comparison for various field operations

| | Our Method in $GF(2^{178})$ | Method in [18] | | Method in [17] in $GF(2^{176})$ |
|--------------------------|--|--|--|---|
| | | in $GF(2^{177})$ | in $GF(2^{176})$ | |
| Field Repres. Type | Circular Dual Basis over $GF(2)$ | Trinomial Stand.Basis over $GF(2)$ | Trinomial Stand.Basis over $GF(2^{16})$ | Trinomial Stand.Basis over $GF(2^{16})$ |
| Total Table Memory | \leq 1Kbytes | \leq 1Kbytes | \geq 256Kbytes | \geq 256Kbytes |
| Plat- forms | Pentium 133MHz V.C++comp. | Pentium 133MHz Watcom 10.6 ANSI-C comp. | Pentium 133MHz Watcom 10.6 ANSI-C comp. | DEC alpha 3000, 175MHz 64-bit word |
| Square | 1.76 μ s | 2.7 μ s | 5.9 μ s | 4.23 μ s |
| Mult. | 50 μ s | 71.8 μ s | 62.7 μ s | 38.6 μ s |
| Invers. | 160 μ s | 225 μ s | 160 μ s | 158.7 μ s |

7 Conclusion

In this contribution paper for IEEE P1363a, we present a newly proposed aspect of dual basis, named *circular dual basis*, and considered its efficiency for field arithmetic operations. The proposed circular dual based implementation of various field operations seems to be more efficient both than those software implementations by trinomial standard bases (and other bases, but consider the notion that trinomial standard based representation is the most efficient method for the software implementation of field operations) and than those hardware implementations by optimal normal bases (this part was dealt in our submitted paper to IEEE Trans. on Computers). Finally we point out that our method is also available and efficient on such composite Galois fields, $GF((2^n)^m)$, that $gcd(m, n) = 1$ and m is one of the numbers in theorem 5. This would be comprehended by intuition.

References

1. N. Koblitz, *Elliptic curve cryptosystems*, Math. Comp., 48 (1987), 203-209.
2. V. Miller, *Uses of elliptic curves in cryptography*, Advances in Cryptology: Proceedings of Crypto'85, Lecture Notes in Computer Science 218 (1986), Springer-Verlag, 417-426.
3. E.R. Berlekamp, *Bit-serial Reed-Solomon encoder*, IEEE Trans. Information Theory, vol. IT-28, pp.869-874, Nov.1982.
4. C.C. Wang, T.K. Truong, H.M. Shao, L.J. Deutsch, J.K. Omura, and I.S. Reed, *VLSI architecture for computing multiplications and inversions in $GF(2^m)$* , IEEE Trans. Comput., vol. C-34, pp.1230-1234, Aug. 1985.
5. Sebastian T.J. Fenn, Mohammed Benaissa, and David Taylor, *$GF(2^m)$ Multiplication and Division Over the Dual Basis*, IEEE Trans. Comput., vol.45, pp.319-327, March 1996.

6. J.L. Massey and J.K. Omura, *Computational Method and Apparatus for Finite Field Arithmetic*, U.S. Patent Application, submitted 1981.
7. G.B. Agnew, R.C. Mullin and S.A. Vanstone, *An implementation of elliptic curve cryptosystems over $F_{2^{155}}$* , IEEE Journal on Selected Areas in Communications, Vol.11, no.5(June 1993), pp.804-813.
8. D.W. Ash, I.F. Blake, and S. Vanstone, *Low complexity normal bases*, Discrete Applied Math. 25(1989), pp.191-210.
9. T. Itoh, S. Tsujii, *A fast algorithm for computing multiplicative inverses in $GF(2^m)$ using normal bases*, Information and Computation, 78:171-177, 1988.
10. R.C. Mullin, I.M. Onyszchuk, S.A. Vanstone, and R.M. Wilson, *Optimal Normal Bases in $GF(p^n)$* , Discrete Applied Maths., pp.142-169, 1988/1989.
11. A. Lempel and M.J. Weinberger, *Self complementary normal bases in finite fields*, SIAM J. Disc. Math., 1 (1988), 193-198.
12. R. Lidl and H. Niederreiter, *An Introduction to Finite Fields and Their Applications*, Cambridge Univ. Press, 1986.
13. Alfred J. Menezes, *Applications of Finite Fields*, Kluwer Academic Publishers, 1993.
14. R. Schroepel, H. Orman, S. O'Malley and O. Spatscheck, *Fast key exchange with elliptic curve systems*, Advances in Cryptology, Proc. Crypto'95, LNCS963, D. Coppersmith, ED., Springer-Verlag, 1995, pp.43-56.
15. P.K.S. Wah and M.Z. Wang, *Realization and application of the Massey-Omura lock*, in Proc. Int. Zurich Sem., Mar.1984.
16. S.T.J. Fenn and M. Benaissa, and D. Taylor, *Finite Field Inversion Over the Dual Basis*, IEEE Trans. on VLSI Systems, vol.4, No.1, March 1996.
17. J. Guajardo and C. Paar, *Efficient Algorithms for Elliptic Curve Cryptosystems*, Advances in Cryptology, CRYPTO'97, pp.342-356, 1997.
18. Erik De Win, Antoon Bosselaers, and Servaas Vandenberghe, *A Fast Software Implementation for Arithmetic Operations in $GF(2^n)$* , Advances in Cryptology, ASIACRYPT'96, pp.65-76, 1996.