Division and Bit-Serial Multiplication over GF(q^m)[†]

M. A. Hasan and V. K. Bhargava

Abstract

Division and bit-serial multiplication in finite fields are considered. Using coordinates of the *supporting* elements it is shown that when field elements are represented by polynomials, division over GF(q^m) can be performed by solving a system of m linear equations over GF(q). For a canonical basis representation, a relationship between the division and the discrete time Wiener-Hopf equation of degree m over GF(q) is derived. This relationship leads to a bit-serial multiplication scheme which can be easily realized for all irreducible polynomials.

1 Introduction

Operations in finite fields GF(2^m) are quite distinct from binary arithmetic. The elements of GF(2^m) can be represented by m binary digits. For such representation addition and subtraction operations are simple, but division and multiplication operations are not. In recent years, the realization of multiplication operation in finite fields has received wide attention and several approaches have been presented. The use of the dual basis and normal basis for representing the elements of finite fields has lead to interesting realizations of the multiplication operation.

Berlekamp [1] has developed a bit-serial multiplication algorithm over GF(2^m) for the encoding of Reed-Solomon codes. A block diagram for computing $c = ab$ over GF(2^m) using Berlekamp's bit-serial multiplication scheme is shown in Fig. 1. When both the multiplicand and the multiplier are represented by the primal basis (which is expected for most practical cases), the above multiplication scheme requires two basis transformations in addition to Berlekamp's bit-serial multiplication circuit. The latter requires only $2m$ shift registers, m AND gates and $m + W_H(g) - 3$ XOR gates where $W_H(g)$ denotes Hamming weight of the irreducible polynomial $g(z)$ chosen for that particular finite field GF(2^m). However, the circuits for the basis transformations are not always simple. A lucid explanation of Berlekamp's bit-serial multiplication scheme can be found in McEliece [2].

Berlekamp's algorithm is very efficient in the sense that it requires minimum circuitry. It has the additional advantage that multiplication by a fixed constant can be hard-wared. However, the algorithm to multiply two elements of GF(2^m) requires to represent one factor by a canonical basis and the other factor by the corresponding dual basis and the product is obtained in the dual basis. The involvement of two different bases is not advantageous specially when the multiplier is to be

[†] Appeared in *IEE Proc. Computers and Digital Techniques*, pp. 230-236, May 1992. This work was supported, in part, by the Association of Universities and Colleges of Canada under a Canadian Commonwealth Scholarship and, in part, by the Natural Sciences and Engineering Research Council of Canada. The work was done when the authors were with the Department of Electrical and Computer Engineering, University of Victoria, Victoria, B.C., Canada V8W 3P6.

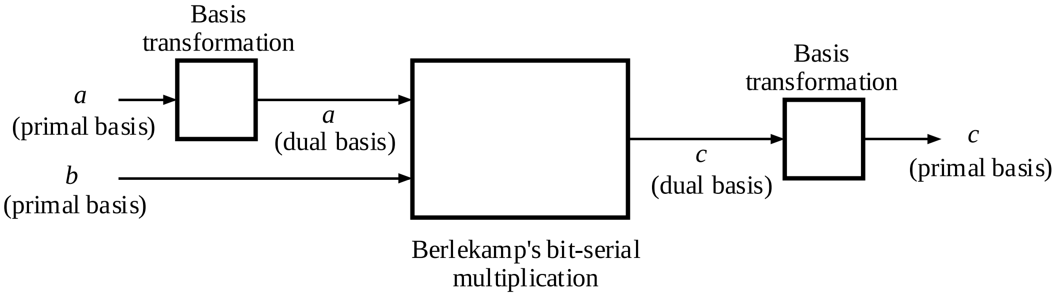

Figure 1: Involvement of dual basis in Berlekamp's bit-serial multiplication scheme.

used as a part of a larger device; because it would be necessary in general to enhance circuitry to change bases [2].

There are some cases for which the basis transformation is very easily accomplished. In fact it can be done just by a permutation of the coordinates. Such easily accomplished basis change depends on the irreducible polynomial chosen. Morii, Kasahara and Whiting have shown in [3] that it is not necessary to use the dual basis for the realization of an efficient bit-serial multiplication when the irreducible polynomial is a trinomial. They have also shown that when the irreducible polynomial is of the form of $g(z) = z^m + z^{k+2} + z^{k+1} + z^k + 1$, $0 < k < m - 2$, only a simple transformation of the bases is necessary to have an efficient bit-serial multiplication circuit.

Recently Wang and Blake [6] have proved that the element transformation into the dual basis can be performed by a simple permutation of the coefficients if and only if the irreducible polynomial is a trinomial. They have developed a bit-serial multiplication scheme which can be realized for all irreducible polynomials over $\mathrm{GF}(2^m)$. However, the involvement of some form of basis transformation circuits both at the input and at the output is still there. As a result, it is desirable to develop an algorithm where the number of gates and registers required for the basis transformation circuit can be reduced for the realization of the bit-serial multiplication in finite fields.

In this paper, a new scheme is developed extending the works of [1], [3] and [6] to perform bit-serial multiplication over $\mathrm{GF}(q^m)$. The scheme uses the coefficients of the so-called *supporting* elements, viz., α^0, α^1, $\cdots$, α^{2m-2} where $\{\alpha^0, \alpha^1, \cdots, \alpha^{m-1}\}$ is the basis for the finite field. It is also shown that when the field elements are represented as polynomials using any suitable basis, the division over $\mathrm{GF}(q^m)$ can be performed by solving a system of m linear equations of general form over $\mathrm{GF}(q)$; and for the canonical basis representation, the division can be performed by solving a discrete time Wiener-Hopf equations (DTWHE) over $\mathrm{GF}(q)$ of $2m - 1$ constants.

The organization of this paper is as follows. Section 2 provides the preliminaries and the definition of the *supporting* elements. The algorithm to perform division over finite fields appears in Section 3. The relationship between the DTWHE and division in finite fields is derived in Section 4. Using this relationship a realization of the bit-serial multiplication scheme is developed in Section 5. Finally conclusions are drawn in Section 6.

2 Preliminaries

GF(q^m) is an extension field of GF(q) where q is a prime and m is a positive integer. The extension field has q^m elements. Let

$$g(z) = \sum_{i=0}^{m} g_i z^i | g_i \in \mathrm{GF}(q)$$

be an irreducible monic polynomial of degree m; $g(z)$ has a root α in GF(q^m). Then every element of GF(q^m) can be represented as a polynomial of powers of α over GF(q) i.e., GF$(q^m) = \{a_0\alpha^{k_0} + a_1\alpha^{k_1} + a_2\alpha^{k_2} + \cdots + a_{m-1}\alpha^{k_{m-1}} | a_i \in \mathrm{GF}(q)$ for $0 \leq i \leq m-1\}$ where $\{\alpha^{k_0},\ \alpha^{k_1},\ \cdots,\ \alpha^{k_{m-1}}\}$ is the basis of GF(q^m). We denote the row vector $\mathbf{a}$ as

$$\mathbf{a} = [a_0,\ a_1,\ \cdots, a_{m-1}] .$$

The set H is defined in this paper as

$$H \quad = \quad \{\alpha^{k_i+k_j}\} \quad i,j = 0,\ 1,\ \cdots,\ m-1. \tag{1}$$

The elements of the set H are hereafter referred to as the *supporting* elements. The coordinates of these *supporting* elements are used in our analyses. To distinguish these coordinates we denote them by adding superscripts as follows.

$$\alpha^n = \sum_{i=0}^{m-1} p_i^{[n]} \alpha^{k_i}. \tag{2}$$

Thus $p_i^{[n]}$ is the i-th coordinate of the *supporting* element α^n. We denote $\mathbf{p}_i^{[k_j]}$ as a column vector whose components are the ith coordinates of the supporting elements $\alpha^{k_0+k_j}$, $\alpha^{k_1+k_j}$, $\cdots$, $\alpha^{k_{m-1}+k_j}$; i.e.,

$$\mathbf{p}_i^{[k_j]} = \left[p_i^{[k_0+k_j]},\ p_i^{[k_1+k_j]},\ \cdots, p_i^{[k_{m-1}+k_j]}\right]^T . \tag{3}$$

3 Division Algorithm

The conventional way to perform division $c(\alpha)/a(\alpha)$ in a finite field is to first compute the multiplicative inverse of $a(\alpha)$ and then multiply the inverse with $c(\alpha)$. The following theorem states that the division in the finite field can be computed in an alternate way.

Theorem 1: Let $g(z)$ be an irreducible polynomial over GF(q) and $a(\alpha)$, $b(\alpha)$ and $c(\alpha)$ be any three elements in GF(q^m). Let the elements be represented by a suitable basis of the form $\{\alpha^{k_0},\ \alpha^{k_1},\ \cdots,\ \alpha^{k_{m-1}}\}$. Then the division $b(\alpha) = c(\alpha)/a(\alpha)$ (mod $g(\alpha)$), $a(\alpha) \neq 0$, in the finite field GF(q^m) can be performed by solving the following equation over GF(q)

$$\begin{bmatrix} \mathbf{a}\cdot\mathbf{p}_{m-1}^{[k_{m-1}]} & \mathbf{a}\cdot\mathbf{p}_{m-1}^{[k_{m-2}]} & \cdots & \mathbf{a}\cdot\mathbf{p}_{m-1}^{[k_0]} \\ \mathbf{a}\cdot\mathbf{p}_{m-2}^{[k_{m-1}]} & \mathbf{a}\cdot\mathbf{p}_{m-2}^{[k_{m-2}]} & \cdots & \mathbf{a}\cdot\mathbf{p}_{m-2}^{[k_0]} \\ \cdots & \cdots & \cdots & \cdots \\ \mathbf{a}\cdot\mathbf{p}_{0}^{[k_{m-1}]} & \mathbf{a}\cdot\mathbf{p}_{0}^{[k_{m-2}]} & \cdots & \mathbf{a}\cdot\mathbf{p}_{0}^{[k_0]} \end{bmatrix} \begin{bmatrix} b_{m-1} \\ b_{m-2} \\ \cdots \\ b_0 \end{bmatrix} = \begin{bmatrix} c_{m-1} \\ c_{m-2} \\ \cdots \\ c_0 \end{bmatrix} \tag{4}$$

where "$\mathbf{x}\cdot\mathbf{y}$" denotes the inner product of $\mathbf{x}$ and $\mathbf{y}$.

Proof: The polynomial representations of $a(\alpha)$, $b(\alpha)$ and $c(\alpha)$ are

$$a(\alpha) = \sum_{l=0}^{m-1} a_l \alpha^{k_l} | a_l \in \mathrm{GF}(q),$$

$$b(\alpha) = \sum_{j=0}^{m-1} b_j \alpha^{k_j} | b_j \in \mathrm{GF}(q)$$

and

$$c(\alpha) = \sum_{i=0}^{m-1} c_i \alpha^{k_i} | c_i \in \mathrm{GF}(q).$$

Then

$$\begin{aligned} c(\alpha) &= a(\alpha)b(\alpha) \pmod{g(\alpha)} \\ &= \sum_{l=0}^{m-1} a_l \alpha^{k_l} \sum_{j=0}^{m-1} b_j \alpha^{k_j} \pmod{g(\alpha)} \\ &= \sum_{j=0}^{m-1} b_j \sum_{l=0}^{m-1} a_l \alpha^{k_l+k_j} \pmod{g(\alpha)}. \end{aligned}$$

Using (2) we can write

$$\begin{aligned} c(\alpha) &= \sum_{j=0}^{m-1} b_j \sum_{l=0}^{m-1} a_l \sum_{i=0}^{m-1} p_i^{[k_l+k_j]} \alpha^{k_i} \\ \sum_{i=0}^{m-1} c_i \alpha^{k_i} &= \sum_{i=0}^{m-1} \sum_{j=0}^{m-1} b_j \sum_{l=0}^{m-1} a_l p_i^{[k_l+k_j]} \alpha^{k_i}. \end{aligned}$$

Equating the coefficients of α on both sides of the above equation we obtain

$$c_i = \sum_{j=0}^{m-1}\left(\sum_{l=0}^{m-1} a_l p_i^{[k_l+k_j]}\right) b_j \qquad i = m-1,\ m-2,\ \cdots,\ 0 \tag{5}$$

which represents the system of m linear equations in $b_0,\ b_1,\ \cdots,\ b_{m-1}$ of (4). Q.E.D.

From (4) we see that when the coordinates of $a(\alpha)$, $c(\alpha)$ and the *supporting* elements are known, $b(\alpha) = c(\alpha)/a(\alpha)$ (mod $g(\alpha)$) can be computed by solving the system of m linear equations in m unknowns over GF(q). For the convenience of representation, we denote (4) as $\mathbf{Ub} = \mathbf{c}$ where $\mathbf{U} \equiv [u_{i,j}]_{i,j=0}^{m-1} = \left[\mathbf{a}\cdot\mathbf{p}_{m-1-i}^{[k_{m-1-j}]}\right]_{i,j=0}^{m-1}$, $\mathbf{b} = [b_{m-1-i}]_{i=0}^{m-1}$ and $\mathbf{c} = [c_{m-1-i}]_{i=0}^{m-1}$. We summarize the steps involved in the division algorithm as follows.

Algorithm 1:

Step 1) Determine the coordinates of the *supporting* elements.

Step 2) Construct Eq. (4).

Step 3) Solve Eq. (4) for $\mathbf{b}$ to get the required result.

The essence of computing division over GF(q^m) using the above algorithm is the inversion of $\mathbf{U}$ over GF(q). The computational complexity involved with the inversion of an $m \times m$ matrix of general form like $\mathbf{U}$ is $O(m^3)$. In the next section, we derive another division algorithm where $\mathbf{U}$ is transformed to a Toeplitz matrix and the later can be inverted by efficient algorithms, for example, [5] and [7]. We now provide an example to compute division using *Algorithm 1.*

Example 1: Let the irreducible polynomial chosen for the field GF(2^3) be $g(z) = 1 + z^2 + z^3$. It is required to divide α^4 by α^2 over GF(2^3). The solution would be trivial if both the divisor and the dividend are given as powers of α in which case the division can be performed by simply subtracting the power of the divisor from that of the dividend. Unfortunately field elements are usually represented as polynomials of the powers of α using suitable bases. Here we consider the canonical and normal bases representations. For the canonical basis representation

$$\alpha^4 \equiv \sum_{i=0}^{2} c_i\alpha^i = 1 + \alpha + \alpha^2 \tag{6}$$

$$\alpha^2 \equiv \sum_{i=0}^{2} a_i\alpha^i = \alpha^2 \tag{7}$$

and for the normal basis representation

$$\alpha^4 \equiv \sum_{i=0}^{2} c_i \alpha^i \quad = \quad \alpha^4 \tag{8}$$

$$\alpha^2 \equiv \sum_{i=0}^{2} a_i \alpha^i \quad = \quad \alpha^2 \tag{9}$$

We now follow *Algorithm 1* step by step to compute the division.

Case I- Canonical basis representation.

Step 1) Here

$$H = \{1,\ \alpha,\ \alpha^2,\ \alpha^3,\ \alpha^4\}$$

and the coordinates of the *supporting* elements are obtained from the following.

$$\begin{aligned}
\alpha^0 &= p_0^{[0]} + p_1^{[0]}\alpha + p_2^{[0]}\alpha^2 \\
\alpha^1 &= p_0^{[1]} + p_1^{[1]}\alpha + p_2^{[1]}\alpha^2 \\
\alpha^2 &= p_0^{[2]} + p_1^{[2]}\alpha + p_2^{[2]}\alpha^2 \\
\alpha^3 &= 1 + \alpha^2 = p_0^{[3]} + p_1^{[3]}\alpha + p_2^{[3]}\alpha^2 \\
\alpha^4 &= 1 + \alpha + \alpha^2 = p_0^{[4]} + p_1^{[4]}\alpha + p_2^{[4]}\alpha^2
\end{aligned}$$

Thus,

$$p_1^{[0]} = p_2^{[0]} = p_0^{[1]} = p_2^{[1]} = p_0^{[2]} = p_1^{[2]} = p_1^{[3]} = 0$$

and

$$p_0^{[0]} = p_1^{[1]} = p_2^{[2]} = p_0^{[3]} = p_2^{[3]} = p_0^{[4]} = p_1^{[4]} = p_2^{[4]} = 1.$$

Step 2) For the canonical basis representation $k_i = i$. So with $m = 3$, $\mathbf{U}$ is given by as follows.

$$\mathbf{U} = \begin{bmatrix} \mathbf{a}\cdot\mathbf{p}_2^{[2]} & \mathbf{a}\cdot\mathbf{p}_2^{[1]} & \mathbf{a}\cdot\mathbf{p}_2^{[0]} \\ \mathbf{a}\cdot\mathbf{p}_1^{[2]} & \mathbf{a}\cdot\mathbf{p}_1^{[1]} & \mathbf{a}\cdot\mathbf{p}_1^{[0]} \\ \mathbf{a}\cdot\mathbf{p}_0^{[2]} & \mathbf{a}\cdot\mathbf{p}_0^{[1]} & \mathbf{a}\cdot\mathbf{p}_0^{[0]} \end{bmatrix}$$

Substituting the values of the coordinates of the *supporting* elements from **Step 1** we obtain

$$\mathbf{U} = \begin{bmatrix} a_0 + a_1 + a_2 & a_1 + a_2 & a_2 \\ a_2 & a_0 & a_1 \\ a_1 + a_2 & a_2 & a_0 \end{bmatrix}.$$

Now using the coordinates of the elements $c(\alpha)$ and $a(\alpha)$ from (6) and (7), we can write in the form of Eq. (4) as follows.

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} b_2 \\ b_1 \\ b_0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Step 3) Solving the system of three linear equations in three unknowns we obtain $b_0 = 0,\ b_1 = 0$ and $b_2 = 1$; so $b(\alpha) = \alpha^2$.

Case II- Normal basis representation.

Step 1) In this case

$$H = \{\alpha,\ \alpha^2,\ \alpha^3,\ \alpha^4,\ \alpha^5,\ \alpha^6\}$$

and

$$\begin{aligned}
\alpha^1 &= p_0^{[1]}\alpha + p_1^{[1]}\alpha^2 + p_2^{[1]}\alpha^4 \\
\alpha^2 &= p_0^{[2]}\alpha + p_1^{[2]}\alpha^2 + p_2^{[2]}\alpha^4 \\
\alpha^3 &= \alpha + \alpha^4 = p_0^{[3]}\alpha + p_1^{[3]}\alpha^2 + p_2^{[3]}\alpha^4 \\
\alpha^4 &= p_0^{[4]}\alpha + p_1^{[4]}\alpha^2 + p_2^{[4]}\alpha^4 \\
\alpha^5 &= \alpha^2 + \alpha^4 = p_0^{[5]}\alpha + p_1^{[5]}\alpha^2 + p_2^{[5]}\alpha^4 \\
\alpha^6 &= \alpha + \alpha^2 = p_0^{[6]}\alpha + p_1^{[6]}\alpha^2 + p_2^{[6]}\alpha^4
\end{aligned}$$

which give

$$p_1^{[1]} = p_2^{[1]} = p_0^{[2]} = p_2^{[2]} = p_1^{[3]} = p_0^{[4]} = p_1^{[4]} = p_0^{[5]} = p_2^{[6]} = 0$$

and

$$p_0^{[1]} = p_1^{[2]} = p_0^{[3]} = p_2^{[3]} = p_2^{[4]} = p_1^{[5]} = p_2^{[5]} = p_0^{[6]} = p_1^{[6]} = 1.$$

Step 2) For the normal basis representation in $\mathrm{GF}(2^3)$, $k_i = 2^i$; so we can write

$$\mathbf{U} = \begin{bmatrix} \mathbf{a}\cdot\mathbf{p}_2^{[2^2]} & \mathbf{a}\cdot\mathbf{p}_2^{[2^1]} & \mathbf{a}\cdot\mathbf{p}_2^{[2^0]} \\ \mathbf{a}\cdot\mathbf{p}_1^{[2^2]} & \mathbf{a}\cdot\mathbf{p}_1^{[2^1]} & \mathbf{a}\cdot\mathbf{p}_1^{[2^0]} \\ \mathbf{a}\cdot\mathbf{p}_0^{[2^2]} & \mathbf{a}\cdot\mathbf{p}_0^{[2^1]} & \mathbf{a}\cdot\mathbf{p}_0^{[2^0]} \end{bmatrix}.$$

For the finite field being considered here, $\alpha^8 = \alpha$. Using this relationship and substituting the values of the coordinates of the *supporting* elements we have

$$\mathbf{U} = \begin{bmatrix} a_0 & a_0 + a_1 & a_1 + a_2 \\ a_0 + a_1 & a_2 & a_0 + a_2 \\ a_1 + a_2 & a_0 + a_2 & a_1 \end{bmatrix}. \tag{10}$$

Now using Eqs. (4) and (10) and substituting the coordinates of the elements $c(\alpha)$ and $a(\alpha)$ from (8) and (9) we have the following system of linear equations

$$\begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} b_2 \\ b_1 \\ b_0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}.$$

Step 3) The solution of these equations gives $b_0 = 0$, $b_1 = 1$ and $b_2 = 0$ for the normal basis representation of $b(\alpha)$; consequently $b(\alpha) = \alpha^2$.

4 DTWHE and Division in Finite Fields

Definition [5]: The discrete time Wiener-Hopf equation (DTWHE) is defined as a system of linear inhomogeneous m equations with m unknowns x_i $(i = 0,\ 1\ ,\ \cdots,\ m-1)$ $\in \mathrm{GF}(q)$, $2m-1$ constants coefficients y_i $(i = 0,\ 1\ ,\ \cdots,\ 2m-2)$ $\in \mathrm{GF}(q)$ that are not all zero, and m constants z_i $(i = 0,\ 1\ ,\ \cdots,\ m-1)$ $\in \mathrm{GF}(q)$ such that

$$\begin{bmatrix} y_{m-1} & y_{m-2} & \cdots & y_1 & y_0 \\ y_m & y_{m-1} & \cdots & y_2 & y_1 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ y_{2m-2} & y_{2m-3} & \cdots & y_m & y_{m-1} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \cdots \\ x_{m-1} \end{bmatrix} = \begin{bmatrix} z_0 \\ z_1 \\ \cdots \\ z_{m-1} \end{bmatrix}. \tag{11}$$

Eq. (11) is referred as the DTWHE of degree m over $\mathrm{GF}(q)$.

In our forthcoming analyses, the elements of $\mathrm{GF}(q^m)$ are represented by the canonical basis $\{1,\ \alpha,\ \alpha^2,\ \cdots,\ \alpha^{m-1}\}$. In this section we show that if the elements of $\mathrm{GF}(q^m)$ are represented by the canonical basis, then a division over $\mathrm{GF}(q^m)$ can be performed by solving a DTWHE of degree m over $\mathrm{GF}(q)$. The motivation behind obtaining a system of linear equations of the form of DTWHE is due to the lower computational complexity involved in solving a DTWHE [5] to perform division and its possible application for the development of a bit-serial multiplication scheme.

Lemma 1: For the canonical basis representation of the elements of $\mathrm{GF}(q^m)$ i.e., $\alpha^k \equiv \sum_{i=0}^{m-1} p_i^{[k]}\alpha^i$,

$$p_j^{[k+1]} = \begin{cases} -p_{m-1}^{[k]}g_j & \bmod q \quad j=0 \\ p_{j-1}^{[k]} - p_{m-1}^{[k]}g_j & \bmod q \quad 1 \le j \le m-1 \end{cases} \tag{12}$$

where $g(z) = \sum_{i=0}^{m-1} g_i z^i + z^m$ is the irreducible monic polynomial over $\mathrm{GF}(q)$.

Proof:

$$\alpha^{k+1} = \sum_{i=0}^{m-1} p_i^{[k]}\alpha^{i+1}.$$

Substituting $j = i+1$,

$$\alpha^{k+1} = \sum_{j=1}^{m} p_{j-1}^{[k]}\alpha^j = \sum_{j=1}^{m-1} p_{j-1}^{[k]}\alpha^j + p_{m-1}^{[k]}\alpha^m.$$

Since $g(\alpha)=0$, $\alpha^m = -\sum_{j=0}^{m-1} g_j\alpha^j$, thus we have

$$\sum_{j=0}^{m-1} p_j^{[k+1]}\alpha^j = \sum_{j=1}^{m-1} p_{j-1}^{[k]}\alpha^j - p_{m-1}^{[k]}\sum_{j=0}^{m-1} g_j\alpha^j.$$

The coefficients of α^j $(0 \le j \le m-1)$ on both sides yield the proof.

Using (3), we can also write (12) in vector notation as follows.

$$\mathbf{p}_j^{[k+1]} = \begin{cases} -\mathbf{p}_{m-1}^{[k]}g_j & \bmod q \quad j=0 \\ \mathbf{p}_{j-1}^{[k]} - \mathbf{p}_{m-1}^{[k]}g_j & \bmod q \quad 1 \le j \le m-1 \end{cases} \tag{13}$$

Before presenting Theorem 2 we see that for the canonical basis representation of the elements of GF(q^m), Eq. (4) can be written as

$$\begin{bmatrix} \mathbf{a}\cdot\mathbf{p}_{m-1}^{[m-1]} & \mathbf{a}\cdot\mathbf{p}_{m-1}^{[m-2]} & \cdots & \mathbf{a}\cdot\mathbf{p}_{m-1}^{[0]} \\ \mathbf{a}\cdot\mathbf{p}_{m-2}^{[m-1]} & \mathbf{a}\cdot\mathbf{p}_{m-2}^{[m-2]} & \cdots & \mathbf{a}\cdot\mathbf{p}_{m-2}^{[0]} \\ \cdots & \cdots & \cdots & \cdots \\ \mathbf{a}\cdot\mathbf{p}_{0}^{[m-1]} & \mathbf{a}\cdot\mathbf{p}_{0}^{[m-2]} & \cdots & \mathbf{a}\cdot\mathbf{p}_{0}^{[0]} \end{bmatrix} \begin{bmatrix} b_{m-1} \\ b_{m-2} \\ \cdots \\ b_0 \end{bmatrix} = \begin{bmatrix} c_{m-1} \\ c_{m-2} \\ \cdots \\ c_0 \end{bmatrix} \tag{14}$$

over GF(q) with $\mathbf{U} = [u_{i,j}]_{i,j=0}^{m-1} = \left[\mathbf{a}\cdot\mathbf{p}_{m-1-i}^{[m-1-j]}\right]_{i,j=0}^{m-1}$. Let $\mathbf{r}_i$ denote the ith row of the matrix $\mathbf{U}$. We now present the following theorem.

Theorem 2: Let $\mathbf{r}'_i$ denote the ith row of the new matrix, say $\mathbf{U}'$, obtained after the elementary row operations of

$$\mathbf{r}'_i \;=\; \mathbf{r}_i - \sum_{k=1}^{i} \mathbf{r}'_{i-k} g_{m-k} \pmod{q} \quad (i = 1,\ 2,\ \cdots,\ m-1). \tag{15}$$

The above row operations transform (14) to the DTWHE of

$$\begin{bmatrix} s_{m-1} & s_{m-2} & \cdots & s_1 & s_0 \\ s_m & s_{m-1} & \cdots & s_2 & s_1 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ s_{2m-2} & s_{2m-3} & \cdots & s_m & s_{m-1} \end{bmatrix} \begin{bmatrix} b_{m-1} \\ b_{m-2} \\ \cdots \\ b_0 \end{bmatrix} = \begin{bmatrix} w_0 \\ w_1 \\ \cdots \\ w_{m-1} \end{bmatrix} \tag{16}$$

over GF(q) where

$$s_k \;=\; \mathbf{a}\cdot\mathbf{p}_{m-1}^{[k]} \pmod{q} \quad (k = 0,\ 1,\ \cdots,\ 2m-2) \tag{17}$$

and

$$w_i \;=\; \begin{cases} c_{m-1} & \text{if } i = 0 \\ c_{m-1-i} - \sum_{l=1}^{i} w_{i-l} g_{m-l} \pmod{q} & \text{if } i = 1,\ 2,\ \cdots,\ m-1 \end{cases} . \tag{18}$$

A proof of the theorem appears in Appendix A. Below is an example to demonstrate the elementary row operations of (15) giving a DTWHE.

Example 2: Let the irreducible polynomial chosen for the field $\mathrm{GF}(3^2)$ be $g(z) = 2 + z + z^2$. Following the first two steps of *Algorithm 1*, as we did in *Example 1*, we obtain

$$\begin{bmatrix} a_0 + 2a_1 & a_1 \\ a_1 & a_0 \end{bmatrix} \begin{bmatrix} b_1 \\ b_0 \end{bmatrix} = \begin{bmatrix} c_1 \\ c_0 \end{bmatrix}.$$

Then applying the elementary row operation (15) we have

$$\begin{bmatrix} a_0 + 2a_1 & a_1 \\ -a_0 - a_1 & a_0 + 2a_1 \end{bmatrix} \begin{bmatrix} b_1 \\ b_0 \end{bmatrix} = \begin{bmatrix} c_1 \\ c_0 - c_1 \end{bmatrix}$$

which is a DTWHE over $\mathrm{GF}(3)$ of degree 2.

Theorem 2 has actually established a relationship between the DTWHE and division in finite fields. If the elements of $\mathrm{GF}(q^m)$ are represented by the canonical basis, then a division over $\mathrm{GF}(q^m)$ can be performed by simply solving the DTWHE (16) over $\mathrm{GF}(q)$. We summarize the division algorithm as follows.

Algorithm 2:

Step 1) Determine the coordinates of the *supporting* elements.

Step 2) Construct the DTWHE (16).

Step 3) Solve the DTWHE.

Like *Algorithm 1* in Section 3, the essence of computing division over $\mathrm{GF}(q^m)$ by using *Algorithm 2* is the inversion of an $m \times m$ matrix. However, in *Algorithm 2*, the matrix is a Toeplitz matrix and the computational complexity for its inversion is $O(m \log^2 m)$ [5].

Algorithm 2 is similar to the approach of [3] in the sense that when the field elements are represented by the canonical basis, both of them compute division by solving DTWHEs of degree m. The advantage of *Algorithm 2* is that it requires, for the construction of the DTWHE, the determination of the coordinates of only $2m - 1$ supporting elements, whereas the approach of [3] requires the determination of $\mathrm{Tr}(\beta\alpha^i)$ $(i = 0,\ 1,\ \cdots,\ 3m - 3)$, where $\beta \in \mathrm{GF}(q^m)$. Moreover, the relationship between the DTWHE and division in finite field as given in Eq. (16) leads to an attractive bit-serial multiplication scheme. This is discussed in the following section.

5 Bit-Serial Multiplication

If the coordinates of $a(\alpha)$ and $b(\alpha)$ are known, Eq. (16) can be used to obtain a bit-serial multiplication circuit. The conceptual diagram for a bit-serial multiplier using Eq. (16) is shown in Fig. 2. The inputs $a_{m-1},\ a_{m-2},\ \cdots,\ a_0$ are used to generate $s_0,\ s_1,\ \cdots,\ s_{2m-2}$ which are sequentially shifted in to the registers $R_0,\ R_1,\ \cdots,\ R_{m-1}$. At each clock pulse the contents of these registers are multiplied by $b_0,\ b_1,\ \cdots,\ b_{m-1}$ respectively. This is equivalent to multiply one row vector of the square matrix of (16) by the column vector $[b_0,\ b_1,\ \cdots,\ b_{m-1}]^T$ yielding the elements of the vector $[w_0,\ w_1,\ \cdots,\ w_{m-1}]^T$. The latter is then transformed to get the required results $c_{m-1},\ c_{m-2},\ \cdots,\ c_0$.

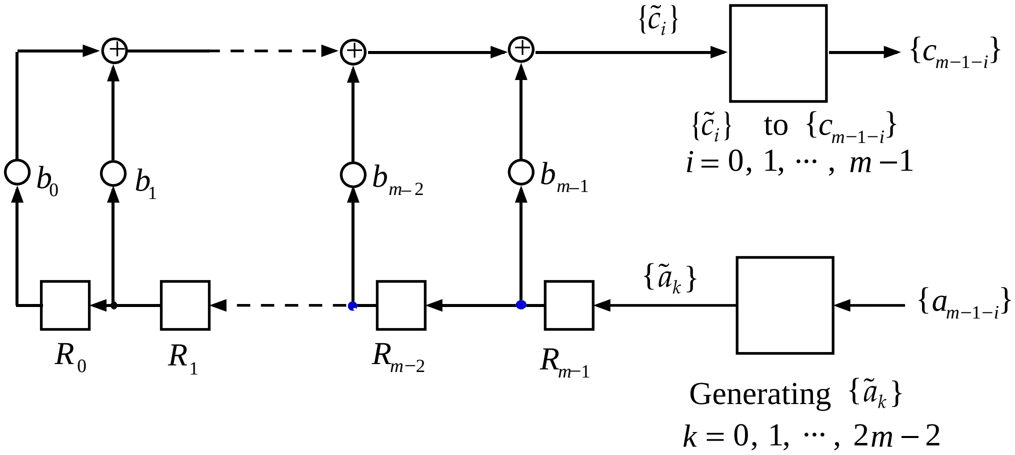

Figure 2: Conceptual diagram for bit-serial multiplication.

We now discuss shift register configurations to generate the constants $s_0,\ s_1,\ \cdots,\ s_{2m-2}$ and to transform $\mathbf{w}$ to $\mathbf{c}$.

5.1 LFSR Configuration for $\{s_0,\ s_1,\ \cdots,\ s_{2m-2}\}$

Since $g(z)$ is an irreducible monic polynomial over GF(q) and $\alpha \in$ GF(q^m) satisfies $g(\alpha) = 0$, we have

$$\alpha^m = -\sum_{i=0}^{m-1} g_i \alpha^i.$$

Again α^m is an element of the set H and we can write from Eq. (2)

$$\alpha^m = \sum_{i=0}^{m-1} p_i^{[m]} \alpha^i.$$

Thus

$$p_i^{[m]} = -g_i \quad i = 0,\ 1,\ 2,\ \cdots,\ m-1.$$

Fig. 3 is a well known configuration for α-multiplication over GF(q^m). In Fig. 3 all the registers and the connecting lines are assumed to function under q-valued logic. If the coordinates $x_0,\ x_1,\ \cdots,\ x_{m-1}$ of the element $x(\alpha) \equiv \sum_{i=0}^{m-1} x_i \alpha^i$ are stored in the registers $B_0,\ B_1,\ \cdots,\ B_{m-1}$ respectively, then after one clock pulse the registers contain the coordinates of the product $\alpha x(\alpha)$. Thus if the registers initially contain the coordinates of the *supporting* element α^{m-1}, then the contents of the last register B_{m-1} with successive clock pulses (up to $m-1$ pulses) are expressed as follows:

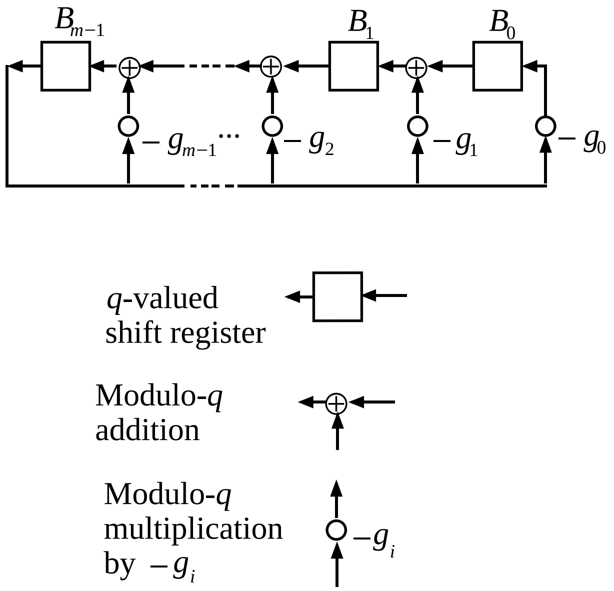

Figure 3: LFSR configuration for multiplication by α.

[hbt]

$$p_{m-1}^{[m-1+j]} \equiv d_j \;=\; \begin{cases} 1 & \text{for } j=0 \\ -\sum\limits_{i=0}^{j-1} g_{m-j+i}\, p_{m-1}^{[m-1+i]} \pmod{q} & \text{for } j = 1,\ 2,\ \cdots,\ m-1 \end{cases} . \tag{19}$$

Corollary 1:

$$s_l \;=\; \sum_{i=0}^{l} a_{m-1-l+i} d_i \quad (l = 0,\ 1,\ \cdots,\ m-1). \tag{20}$$

Eq. (19) in conjunction with Eqs. (12) and (20) is used to derive the following recursive relationship, presented as a corollary, which results in a LFSR (linear feed back shift register) configuration for the generation of s_k.

Corollary 2: The constant coefficients s_k $(k = 0,\ 1,\ \cdots,\ 2m-2)$ of the DTWHE (16) can be derived as

$$s_k = \begin{cases} a_{m-1} & (k=0) \\ a_{m-1-k} - \sum\limits_{l=0}^{k-1} s_l g_{m-k+l} \pmod{q} & (1 \le k \le m-1) \\ -\sum\limits_{l=0}^{m-1} s_{k-1-l} g_{m-1-l} \pmod{q} & (m \le k \le 2m-2) \end{cases} . \tag{21}$$

Proofs of the above two corollaries are given in Appendix B.

Combining the three cases of s_k viz., $k = 0$, $1 \leq k \leq m-1$ and $m \leq k \leq 2m-2$ we see that s_k $(k = 0,\ 1,\ \cdots,\ m-1)$ can be generated sequentially in the register R_{m-1} of Fig. 4 provided that the LFSR initially contain zero and the input to the configuration is the sequence $\{a_{m-1},\ a_{m-2},\ \cdots,\ a_0,\ 0,\ 0,\ \cdots,\ 0\}$ of $2m-1$ elements.

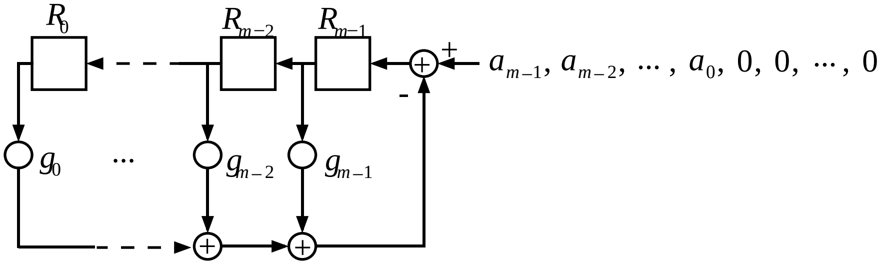

Figure 4: Generation of $\tilde{a}_k$.

5.2 Transformation of w to c

From Eq. (18), i.e.,

$$w_i = \begin{cases} c_{m-1} & \text{if } i = 0 \\ c_{m-1-i} - \sum\limits_{l=1}^{i} w_{i-l}g_{m-l} \pmod q & \text{if } i = 1,\ 2,\ \cdots,\ m-1 \end{cases}$$

we have

$$c_{m-1-i} = \begin{cases} w_0 & \text{if } i = 0 \\ \sum\limits_{l=0}^{i} w_{i-l}g_{m-l} \pmod q & \text{if } i = 1,\ 2,\ \cdots,\ m-1 \end{cases}. \tag{22}$$

A feed forward shift register configuration to transform w's to c's is shown in Fig. 5. It is assumed that the configuration is of q-valued logic and the registers are initially empty. The input to the configuration is the sequence $\{w_0,\ w_1,\ \cdots,\ w_{m-1}\}$ and the corresponding output is $\{c_{m-1},\ c_{m-2},\ \cdots,\ c_0\}$. After m clock pulses the registers $R'_m,\ R'_{m-1},\ \cdots,\ R'_1$ contain $w_{m-1},\ w_{m-2},\ \cdots,\ w_0$ respectively.

Fig. 6 shows the complete configuration of q-valued logic for the bit-serial multiplication scheme. All registers are initially empty. The input to the circuit is the sequence $\{a_{m-1},\ a_{m-2},\ \cdots,\ a_0,\ 0,\ 0,\ \cdots,\ 0\}$ of $2m$ elements. As $a_{m-1},\ a_{m-2},\ \cdots,\ a_0$ enter into the LFSR, $s_0,\ s_1,\ \cdots,\ s_{m-1}$ are loaded into $R_0,\ R_1,\ \cdots,\ R_{m-1}$ respectively; and at this point the switch S closes. At the next m clock cycles $c_{m-1},\ c_{m-2},\ \cdots,\ c_0$ are obtained at the output sequentially as shown in Fig. 6.

If the irreducible monic polynomial is $g(z) = \sum\limits_{i=0}^{m-1} g_i z^i + z^m | g_i \in \mathrm{GF}(q)$ with $g_k \neq 0$ where k $(1 < k < m)$ is the least nonzero positive integer, then only $m-k+1$, instead of m, stages of registers

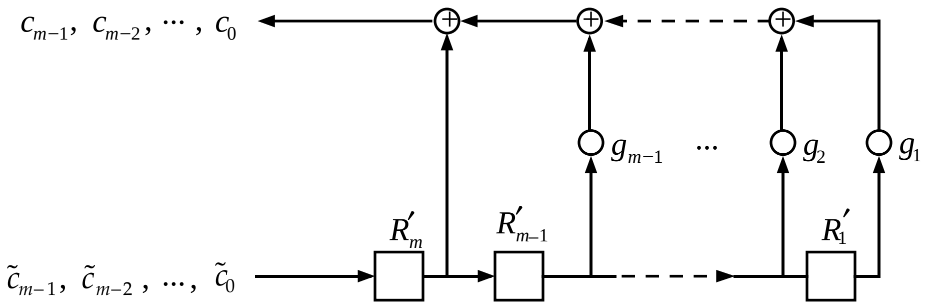
$c_{m-1}, c_{m-2}, \cdots, c_0$
g_{m-1} ··· g_2 g_1
R'_m R'_{m-1} R'_1
$\tilde{c}_{m-1}, \tilde{c}_{m-2}, \dots, \tilde{c}_0$

Figure 5: Transformation of $\tilde{\mathbf{c}}$ to $\mathbf{c}$

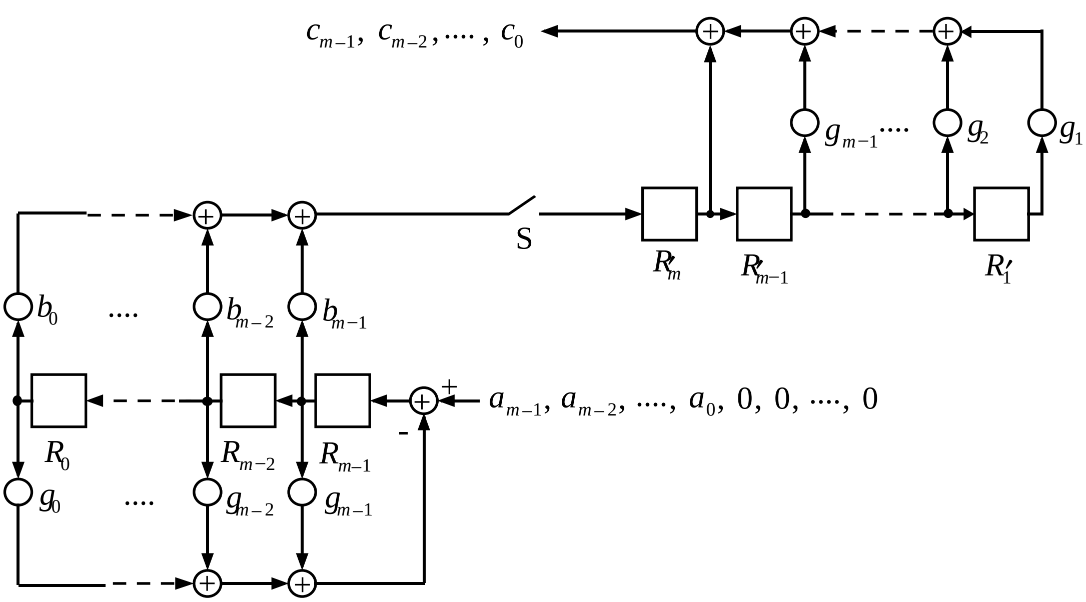
$c_{m-1}, c_{m-2}, \dots., c_0$
g_{m-1}···· g_2 g_1
S
R'_m R'_{m-1} R'_1
b_0 .... b_{m-2} b_{m-1}
+
$a_{m-1}, a_{m-2}, \dots., a_0, 0, 0, \dots., 0$
-
R_0 R_{m-2} R_{m-1}
g_0 .... g_{m-2} g_{m-1}

Figure 6: The bit-serial multiplication circuit.

are required in the feed forward shift register of Fig. 6. Thus by choosing a suitable $g(z)$ with k as large as possible, if an option exists, the number of registers for the bit-serial multiplication circuit can be reduced. For example, both $1+z+z^2+z^3+z^{31}$ and $1+z^{28}+z^{29}+z^{30}+z^{31}$ are irreducible polynomials over GF(2) of weight five [4]. However, the former requires 31 stages of registers in the feed forward shift register while the latter requires only 4.

5.3 Comparison

If the irreducible polynomial is a trinomial or a pentanomial of the form of $g(z) = 1 + z^k + z^{k+1} + z^{k+2} + z^m$, $(0 < k < m-2)$, then the bit-serial multiplication scheme of [3] for GF(2^m) requires very simple basis transformation. However, irreducible trinomials and pentanomials do not exist for all m. A computer search for finding suitable basis transformation as suggested in [3] is a case dependent approach. From that point of view the bit-serial multiplication scheme proposed here has the advantage of being applicable for all irreducible polynomials.

The bit-serial multiplication scheme presented in this paper and that developed in [6] take the multiplicand as well as the multiplier represented by the canonical basis and generate the product which is also represented by the canonical basis. Functionally these two schemes are equivalent;

however, a reduction in the number of gates and shift registers is obtained by using the scheme presented here. This is shown in Table 1 for GF(2^m). To determine the number of gates it is assumed that if $g_i = 1$ ($i = 0,\ 1,\ \cdots,\ m-1$) then the feed back (or feed forward) connection with weight g_i in Fig. 6 exists and a XOR gate is used.

Components	Circuit of [6]	Circuit presented here
No. of shift registers	$5m$	$3m$
No. of 2-input AND gates	$2m$	m
No. of 2-input XOR gates	$2(m-1) + 3[W_H(g) - 2]$	$(m-1) + 2[W_H(g) - 2] + 1$

Table 1: Comparison of no. of gates and registers of the bit-serial multiplication circuits of [6] and of this paper.

6 Conclusion

When the elements of the finite field GF(q^m) are represented by powers of α, computing division involving two elements can be performed by simply subtracting the power of the divisor from that of the dividend. For practical reasons, however, the elements are usually represented as a polynomial using a suitable basis. An algorithm for computing division in finite fields has been derived in this paper . The algorithm is general in the sense that it can be applied for any basis chosen for the field; it requires the solution of a system of m linear equations of the general form over GF(q) to perform a division in GF(q^m). It has been shown that if the field elements are represented by the canonical basis of the form of $\{1,\ \alpha,\ \cdots,\ \alpha^{m-1}\}$, the division can be performed with a lesser order of computational complexity by solving a discrete time Wiener-Hopf equation of degree m. The relationship between the finite field division and the discrete time Wiener-Hopf equation has lead to the development of a bit-serial multiplication scheme. The attractive feature of the multiplication scheme is that it can be easily realized for all irreducible polynomials and in many cases it would require fewer number of gates and shift registers compared to other available bit-serial multiplication schemes.

Appendix A

Proof of Theorem 2: For the sake of simple notation we denote (16) as $\mathbf{U}'\mathbf{b} = \mathbf{w}$ with $\mathbf{U}' \equiv \left[u'_{i,j}\right]_{i,j=0}^{m-1} = [s_{m-1+i-j}]_{i,j=0}^{m-1}$ and $\mathbf{w} \equiv [w_i]_{i=0}^{m-1}$. The transformation from $\mathbf{c}$ of (14) to $\mathbf{w}$ directly follows from (15). So it is required to show that the diagonal elements of $\mathbf{U}'$ are equal and $u'_{i,j} = s_{m-1+i-j} = \mathbf{a} \cdot \mathbf{p}_{m-1}^{[m-1+i-j]}$ $\quad i, j = 0,\ 1,\ \cdots, m-1$. This in shown by induction.

Using (13), it is easy to verify that after the operations (15) on row 1, the latter becomes

$$u'_{1,j} = s_{m-j} = \mathbf{a} \cdot \mathbf{p}_{m-1}^{[m-j]} \quad j = 0,\ 1,\ \cdots,\ m-1.$$

Similarly, after completion of the operations (15) up to the $(i-1)$th row, for $1 \leq i \leq m-1$ and $0 \leq j \leq m-1$ we can write

$$\begin{aligned}
u'_{i,j} &= u_{i,j} - \sum_{k=1}^{i} u'_{i-k,j} g_{m-k} && (23) \\
&= \mathbf{a} \cdot \mathbf{p}_{m-1-i}^{[m-1-j]} - \sum_{k=1}^{i} s_{m-1+i-j-k}\, g_{m-k} \\
&= \mathbf{a} \cdot \left(\mathbf{p}_{m-1-i}^{[m-1-j]} - \sum_{k=1}^{i} \mathbf{p}_{m-1}^{[m-1+i-j-k]} g_{m-k} \right)
\end{aligned}$$

$$(24)$$

Using (13) repeatedly we obtain

$$\begin{aligned}
u'_{i,j} &= \mathbf{a} \cdot \left(\mathbf{p}_{m-i}^{[m-j]} - \sum_{k=1}^{i-1} \mathbf{p}_{m-1}^{[m-1+i-j-k]} g_{m-k} \right) \\
&= \mathbf{a} \cdot \left(\mathbf{p}_{m-i+1}^{[m-j+1]} - \sum_{k=1}^{i-2} \mathbf{p}_{m-1}^{[m-1+i-j-k]} g_{m-k} \right) \\
&\vdots \\
&= \mathbf{a} \cdot \mathbf{p}_{m-1}^{[m-1+(i-1)-(j-1)]} = s_{m-1+(i-1)-(j-1)} = u'_{i-1,j-1}. && (25)
\end{aligned}$$

Appendix B

Proof of Corollary 1: From Eq. (17),

$$s_k = \sum_{l=0}^{m-1} a_l p_{m-1}^{[l+k]}.$$

Substituting $l = m - 1 - k + i$, we obtain

$$s_k = \sum_{i=k-(m-1)}^{k} a_{m-1-k+i} p_{m-1}^{[m-1+i]}.$$

In the canonical basis, for $0 \leq j \leq m-1$

$$p_{m-1}^{[j]} = \begin{cases} 1 & \text{if } j = m-1 \\ 0 & \text{otherwise.} \end{cases} \tag{26}$$

So

$$\begin{aligned} s_k &= \sum_{i=0}^{k} a_{m-1-k+i} p_{m-1}^{[m-1+i]} \\ &= \sum_{i=0}^{k} a_{m-1-k+i} d_i \qquad \text{Q.E.D.} \end{aligned}$$

Proof of Corollary 2: With $k=0$, if we substitute (26) in (17) then we obtain $s_0 = a_{m-1}$. We then consider s_k $(1 \leq k \leq m-1)$. With the help of Eq. (20) we can write the R.H.S. of (21) as

$$\text{R.H.S} \;=\; a_{m-1-k} - \sum_{l=0}^{k-1} \left(\sum_{i=0}^{l} a_{m-1-l+i} d_i \right) g_{m-k+l}. \tag{27}$$

Let

$$\begin{aligned} X &= \sum_{l=0}^{k-1} \left(\sum_{i=0}^{l} a_{m-1-l+i} d_i \right) g_{m-k+l} \\ &= \sum_{l=0}^{k-1} \left(a_{m-1-l} d_0 + a_{m-1-l+1} d_1 + a_{m-1-l+2} d_2 + \cdots + a_{m-1} d_l \right) g_{m-k+l} \end{aligned}$$

Using the fact that $a_j = 0$ when $j \geq m$ or $j < 0$, we have

$$\begin{aligned} X &= a_{m-1} d_0 g_{m-k} \\ &\quad + \left(a_{m-2} d_0 + a_{m-1} d_1 \right) g_{m-k+1} \\ &\quad + \left(a_{m-3} d_0 + a_{m-2} d_1 + a_{m-1} d_2 \right) g_{m-k+2} \\ &\quad \vdots \\ &\quad + \left(a_{m-k} d_0 + a_{m-k+1} d_1 + \cdots + a_{m-1} d_{k-1} \right) g_{m-1} \\ &= a_{m-1} \left(g_{m-k} d_0 + g_{m-k+1} d_1 + \cdots + g_{m-1} d_{k-1} \right) \\ &\quad + a_{m-2} \left(g_{m-k+1} d_0 + g_{m-k+2} d_1 + \cdots + g_{m-1} d_{k-2} \right) \\ &\quad \vdots \\ &\quad + a_{m-k} \left(g_{m-1} d_0 \right) \end{aligned}$$

Substituting (19)

$$\begin{aligned} X &= -a_{m-1}d_k - a_{m-2}d_{k-1} - a_{m-3}d_{k-2} - \cdots - a_{m-k}d_1 \\ &= -\sum_{l=1}^{k} a_{m-1-k+l}d_l \end{aligned}$$

Thus (27) becomes

$$\begin{aligned} \text{R.H.S.} &= a_{m-1-k} - X \\ &= \sum_{l=0}^{k} a_{m-1-k+l}d_l = s_k = \text{L.H.S.} \end{aligned}$$

Using (13) repeated in (17), it is straight-forward to show that

$$s_k = -\sum_{l=0}^{m-1} s_{k-1-l}g_{m-1-l} \qquad m \leq k \leq 2m-2.$$

Acknowledgment

The authors thank Dr. M. Z. Wang for many valuable discussions during the preparation of this paper and to the referees for their comments to improve the presentation.

References

[1] E. R. Berlekamp, "Bit-serial Reed-Solomon encoder," *IEEE Trans. Inform. Theory* vol. IT-28, pp. 869-874, Nov. 1982.

[2] R. J. McEliece, *Finite Fields for Computer Scientists and Engineers,* Kluwer Academic, Massachusetts, 1987.

[3] M. Morii, M. Kasahara and D. L. Whiting, "Efficient bit-serial multiplication and the discrete-time Wiener-Hopf equations over finite fields," *IEEE Trans. Inform. Theory,* vol. IT-35, No. 6, pp. 1177-1183, Nov. 1989.

[4] W. W. Peterson and E. J. Weldon, *Error-Correcting Codes,* MIT, Cambridge, Massachusetts, 1972.

[5] Y. Sugiyama, "An algorithm for solving discrete-time Wiener-Hopf equations based on Euclid's algorithm," *IEEE Trans. Inform. Theory,* vol. IT-32, pp. 394-409, May 1986.

[6] M. Z. Wang and I. F. Blake, "Bit serial multiplication in finite fields," *SIAM J. Disc. Math.,* vol. 3, No. 1, pp. 140-148, Feb. 1990.

[7] W. F. Trench, "An algorithm for the inversion of finite Toeplitz matrices," *J. Soc. Indus. Appl. Math.,* vol. 12, pp. 512-522, Sep. 1964.