International Journal of Foundations of Computer Science

GF(2^n) redundant representation using matrix embedding for irreducible trinomials

Yongjia Wang
School of Software, Tsinghua University, Beijing, China. wangyj1121@126.com

Xi Xiong
School of Software, Tsinghua University, Beijing, China. xixiong91@gmail.com

Haining Fan
School of Software, Tsinghua University, Beijing, China. fhn@tsinghua.edu.cn

Received (Day Month Year)
Accepted (Day Month Year)
Communicated by (xxxxxxxxxx)

By embedding a Toeplitz matrix-vector product (MVP) of dimension n into a circulant MVP of dimension $N = 2n + \delta - 1$, where δ can be any nonnegative integer, we present a GF(2^n) multiplication algorithm. This algorithm leads to a new redundant representation, and it has two merits: 1. The flexible choices of δ make it possible to select a proper N such that the multiplication operation in ring GF(2)$[x]/(x^N + 1)$ can be performed using some asymptotically faster algorithms, e.g. the Fast Fourier Transformation (FFT)-based multiplication algorithm; 2. The redundant degrees, which are defined as N/n, are smaller than those of most previous GF(2^n) redundant representations, and in fact they are approximately equal to 2 for all applicable cases.

Keywords: Finite fields; redundant representation; matrix-vector product.

1. Introduction

When GF(2^n) is viewed as an n-dimensional vector space, field elements can be represented as n-bit vectors in a basis of GF(2^n) over GF(2). Types of bases are various, for example, polynomial bases, normal bases, dual bases and shifted polynomial bases (SPB) and so on. Besides these representations, redundant representations become attractive because of their cheap squaring and modular operations [1] [19].

Most previous redundant representations can be classified as two groups: one is originated from polynomial bases and the other from normal bases. In 1995, Gao et al. presented a multiplication algorithm in normal bases generated by Gauss periods [10] [11]. After converting the representation from normal bases to some polynomials, they embedded a field into a larger cyclotomic ring, and then performed ring multiplication using some asymptotically faster multiplication algorithm, e.g., Fast

Fourier Transform (FFT)-based multiplication algorithm. Especially, they embedded $\mathrm{GF}(2^n)$ into cyclotomic ring $\mathrm{GF}(2)[x]/(x^{2n+1}+1)$ when a type II optimal normal basis exists. In 2007, this approach was improved by Gathen et al.. They used a fast transformation between type II optimal normal bases and suitable polynomial representations, whose complexity is $\mathcal{O}(n\log_2 n)$ bit operations for the general case [22]. In 2010, Bernstein and Lange improved the results of [22] in several ways [2]. They reduced the size of the suitable polynomial from $n+1$ to n, and they also reduced the transformation cost. These works mainly focus on type II optimal normal bases. Another redundant representations originated from general normal bases is [5], where the ordered set $\{1,\gamma,\gamma^2,\gamma^{2^2},\ldots,\gamma^{2^{n-1}}\}$ was used to design $\mathrm{GF}(2^n)$ quadratic parallel multipliers. Especially, they discussed the case that rank of $\{\gamma,\gamma^2,\gamma^{2^2},\ldots,\gamma^{2^{n-1}}\}$ is $n-1$.

Compared to normal bases-based redundant representations, most previous works on redundant representations follow the polynomial approach, namely, they embed $\mathrm{GF}(2^n)$ into a finite quotient ring $\mathrm{GF}(2)[x]/(x^N+1)$, and therefore map a $\mathrm{GF}(2^n)$ multiplication operation into a $\mathrm{GF}(2)[x]/(x^N+1)$ multiplication. The later can be performed using some asymptotically faster multiplication algorithm.

Redundant representations first appeared in finite field $\mathrm{GF}(2^n) := GF(2)[x]/(f(x))$ generated by all-one-polynomial $f(x)=\sum_{i=0}^{n}x^i$. In 1984, Itoh and Tsujii applied the simplicity of multiplication in quotient ring $\mathrm{GF}(2)[x]/(x^N+1)$ (where $N=n+1$) to the $\mathrm{GF}(2^n)$ multiplication [16]. In this case, the n-bit vector of a $\mathrm{GF}(2^n)$ element is mapped to the $(n+1)$-bit vector of a $\mathrm{GF}(2)[x]/(x^N+1)$ element. Therefore, the redundant degree, which is defined as N/n, is $(n+1)/n\approx 1$ for these special $\mathrm{GF}(2^n)$s. Besides multiplication, Silverman also analyzed other operations in these fields [21]. Combining Karatsuba's algorithm and redundant representation, Chang, Hong and Cho presented a low complexity bit-parallel multiplier in 2005 [3]. In 2008, Namin, Wu and Ahmadi designed a novel serial-in parallel-out multiplier in these fields [19].

In 1998, Drolet generalized this idea and introduced $\mathrm{GF}(2^n)$ redundant representations systematically [4]. His results were corrected and improved later by Geiselmann, Muller-Quade and Steinwandt [13]. Similarly, Wu, Hasan, Blake and Gao presented simple and highly regular architectures for finite field multipliers using a redundant representation, and their architectures can provide area-time trade-offs [25] [24]. In 2001, Geiselmann and Lukhaub showed that $\mathrm{GF}(2^n)$ arithmetic, especially exponentiation, in redundant representation is perfectly suited for low power computing [12]. In 2003, Katti and Brennan generalized the idea of quotient ring $\mathrm{GF}(2)[x]/(x^N+1)$ to quotient rings $\mathrm{GF}(2)[x]/(x^N+x^k+1)$ and $\mathrm{GF}(2)[x]/(x^N+x^{k_1}+x^{k_2}+1)$ [17], and in the same year, Geiselmann and Steinwandt generalized redundant representations to finite fields of arbitrary characteristic [14].

The major disadvantage of the above redundant representations is that redundant degrees are often large, for example, the average redundant degree for $151\le n\le 250$ is about 4.58 [24]. Recently, Akleylek and Ozbudak presented a modified redundant representation [1]. Their results improved some of the previous complexity values significantly, or more precisely, redundant degrees are decreased to about 1 or 2 for some $\mathrm{GF}(2^n)$s. But for some other values of n's, no improvement on redundant degrees is reported in their paper, for example, cases that n's are prime.

Besides the disadvantage of large redundant degree, all these polynomial-based methods suffer another disadvantage: for a fixed $\mathrm{GF}(2^n)$, there is only one choice of a smaller N. Because of this limitation, it might be hard to select a proper fast algorithm to perform multiplication in $\mathrm{GF}(2)[x]/(x^N+1)$, for example, FFT does not help when N is a prime [21].

In this article, a different embedding method is used to overcome the above two disadvantages. Instead of following the polynomial approach, we apply the matrix approach to perform the embedding step. We map a $\mathrm{GF}(2^n)$ multiplication operation into a multiplication in the quotient ring $\mathrm{GF}(2)[x]/(x^N+1)$, where $N = 2n+\delta-1$ and δ can be any non-negative integer. The flexible choices of δ make it possible to select a proper N such that the multiplication operation in ring $\mathrm{GF}(2)[x]/(x^N+1)$ can be performed using some asymptotically fast algorithms. Furthermore, our redundant degrees ($N/n \approx 2$) are smaller than those of most previous $\mathrm{GF}(2^n)$ redundant representations for all applicable values of n's. As a comparison, reference [1] provided only 54 composite values of n's such that $15 \leq n \leq 1956$ and their redundant degrees are approximately equal to 1 or 2. But for over 50% (composite and prime) values of n's in this range, or even a larger range $1 \leq n \leq 10,001$, redundant degrees of our method are approximately equal to 2 [20]. Even though, we must note that among these 54 values of n's in [1], there are 34 values of n's such that their redundant degrees are slightly greater than 1.

This paper is organized as follows: The equivalence between circulant Matrix-Vector Product (MVP) and $\mathrm{GF}(2)[x]/(x^N+1)$ multiplication is introduced in Section 2. In Section 3, the new 4-step $\mathrm{GF}(2^n)$ SPB multiplication algorithm is described. Explicit formulae of the new SPB redundant representation are given in Section 4, and an example is presented in Section 5. Considerations for other bases are included in section 6. Finally, a few concluding remarks are made in Section 7.

2. Equivalence between circulant MVP and $\mathrm{GF}(2)[x]/(x^N+1)$ multiplication

Given two $\mathrm{GF}(2)[x]/(x^N+1)$ elements $p = \sum_{i=0}^{N-1} p_i x^i$ and $q = \sum_{i=0}^{N-1} q_i x^i$, let $P = (p_0, p_1, \ldots, p_{N-1})^T$ be the coordinate column vector of p, and Q is defined similarly. The product $r = pq = \sum_{i=0}^{N-1} r_i x^i$ in ring $\mathrm{GF}(2)[x]/(x^N+1)$ can be computed in three steps.

We first compute the conventional polynomial product of p and q:

$$r = pq = \sum_{t=0}^{2N-2} r_t x^t = l + l_+,$$

where $l = \sum_{t=0}^{N-1} r_t x^t$, $l_+ = \sum_{t=N}^{2N-2} r_t x^t$ and

$$r_t = \sum_{\substack{i+j=t \\ 0 \le i,j < N}} p_i q_j = \begin{cases} \sum_{i=0}^{t} p_i q_{t-i}, & 0 \le t \le N-1; \\ \sum_{i=t+1-N}^{N-1} p_i q_{t-i}, & N \le t \le 2N-2. \end{cases}$$

Then we reduce l_+ using equation $x^i = x^{i-N}$, where $N \le i \le 2N-2$, and obtain

$$l_+ \bmod (x^N + 1) = \sum_{t=N}^{2N-2} r_t x^t \bmod (x^N + 1) = \sum_{t=0}^{N-2} r_{t+N} x^t.$$

Finally, we get the product r of p and q in GF(2)$[x]/(x^N + 1)$:

$$\begin{aligned} r &= \sum_{i=0}^{N-1} r_i x^i = (l + l_+) \bmod (x^N + 1) \\ &= \sum_{t=0}^{N-1} r_t x^t + \sum_{t=0}^{N-2} r_{t+N} x^t \\ &= \sum_{t=0}^{N-2} \left(\sum_{i=0}^{t} p_i q_{t-i} + \sum_{i=t+1}^{N-1} p_i q_{t+N-i} \right) x^t + \left(\sum_{i=0}^{N-1} p_i q_{N-1-i} \right) x^{N-1} \\ &= (1, x, x^2, \ldots, x^{N-1}) \begin{pmatrix} q_0 & q_{N-1} & q_{N-2} & \cdots & q_1 \\ q_1 & q_0 & q_{N-1} & \cdots & q_2 \\ q_2 & q_1 & q_0 & \cdots & q_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ q_{N-1} & q_{N-2} & q_{N-3} & \cdots & q_0 \end{pmatrix} \begin{pmatrix} p_0 \\ p_1 \\ p_2 \\ \vdots \\ p_{N-1} \end{pmatrix} \\ &= (1, x, x^2, \ldots, x^{N-1}) \overline{T} P. \end{aligned}$$

Clearly, the $N \times N$ matrix $\overline{T}$ in the above equation is a circulant matrix and the result of the circulant MVP $\overline{T}P$ is just the coordinate column vector $R = (r_0, r_1, \ldots, r_{N-1})^T$ of r. Especially, the first row of $\overline{T}$ is

$$\overline{T}_{(1)} = (q_0, q_{N-1}, q_{N-2}, \ldots, q_1). \tag{1}$$

In the next section, we will use this well-known fact to derive new redundant representations.

3. New GF(2^n) SPB multiplication algorithm

In this part we introduce the main idea of our multiplication algorithm using the shifted polynomial basis (SPB) of GF(2^n) over GF(2). We first introduce the definition of the SPB.

If $f(x) = x^n + x^k + 1$ $(n > 2)$ is an irreducible trinomial over GF(2), then all elements of GF(2^n) can be represented using a polynomial basis $W = \{x^i | 0 \leq i \leq n-1\}$. Let v be an integer, the ordered set $x^{-v}W = \{x^{i-v} | \ 0 \leq i \leq n-1\}$ is called the SPB of GF(2^n) over GF(2) with respect to W. It was shown that the best values of v are k or $k-1$ when the SPB is used to design parallel multipliers [6]. In this article, we select $v = k$. However, we note that the proposed embedding method can be similarly used for the case $v = k-1$.

Given two GF(2^n) elements $a = x^{-v}\sum_{i=0}^{n-1} a_i x^i$ and $b = x^{-v}\sum_{i=0}^{n-1} b_i x^i$ represented in the above SPB, the proposed algorithm can be divided into four steps. The first two steps also appear in designing Toeplitz MVP-based subquadratic GF(2^n) multipliers, and detailed descriptions can be found in [7]. The following part presents these results briefly.

Step 1: Representing the product of a and b as a Mastrovito MVP.

The SPB Mastrovito multiplier was introduced in [6]. Let $A = (a_0, a_1, \ldots, a_{n-1})^T$ be the coordinate column vector of the field element $a = x^{-v}\sum_{i=0}^{n-1} a_i x^i$, B and C are defined similarly. The coordinate column vector C of $c = ab$ can be represented as $C = ZA$ in the following equation:

$$\begin{aligned} c = x^{-v}\sum_{i=0}^{n-1} c_i x^i = ab &= \left(\sum_{i=0}^{n-1} a_i x^{i-v}\right) b \\ &= (x^{-v}b, x^{-v+1}b, \ldots, x^{-1}b, b, \ldots, x^{n-v-1}b)A \\ &= (x^{-v}, x^{-v+1}, \ldots, x^{n-v-1})ZA. \end{aligned}$$

The $n \times n$ matrix $Z = (z_{i,j})_{0 \leq i,j \leq n-1}$, which depends on only B and $f(x)$, is called the Mastrovito matrix, and $C = ZA$ is the Mastrovito MVP formula to compute the product of a and b in GF(2^n).

Step 2: Transforming the Mastrovito MVP $C = ZA$ into a Toeplitz MVP.

Using the transformation matrix U of [7] , the above Mastrovito MVP $C = ZA$ can be transformed into Toeplitz MVP $D = TA$, where T is a Toeplitz matrix, or more precisely,

$$C = ZA = U^{-1}UZA = U^{-1}TA = U^{-1}D, \tag{2}$$

where $U = \begin{pmatrix} 0 & I_{(n-v)\times(n-v)} \\ I_{v\times v} & 0 \end{pmatrix}$, $I_{v\times v}$ is the $v \times v$ identity matrix and $T = UZ$ is an $n \times n$ Toeplitz matrix.

Step 3: Embedding the Toeplitz MVP $D = TA$ into a circulant MVP.

We give a small example to illustrate the idea of this embedding. The following Toeplitz MVP of dimension 3

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} t_0 & t_{-1} & t_{-2} \\ t_1 & t_0 & t_{-1} \\ t_2 & t_1 & t_0 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix}$$

can be embedded into either the following circulant MVP of dimension 6

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ r_3 \\ r_4 \\ r_5 \end{pmatrix} = \begin{pmatrix} t_0 & t_{-1} & t_{-2} & 0 & t_2 & t_1 \\ t_1 & t_0 & t_{-1} & t_{-2} & 0 & t_2 \\ t_2 & t_1 & t_0 & t_{-1} & t_{-2} & 0 \\ 0 & t_2 & t_1 & t_0 & t_{-1} & t_{-2} \\ t_{-2} & 0 & t_2 & t_1 & t_0 & t_{-1} \\ t_{-1} & t_{-2} & 0 & t_2 & t_1 & t_0 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

or the following circulant MVP of dimension 5

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ s_3 \\ s_4 \end{pmatrix} = \begin{pmatrix} t_0 & t_{-1} & t_{-2} & t_2 & t_1 \\ t_1 & t_0 & t_{-1} & t_{-2} & t_2 \\ t_2 & t_1 & t_0 & t_{-1} & t_{-2} \\ t_{-2} & t_2 & t_1 & t_0 & t_{-1} \\ t_{-1} & t_{-2} & t_2 & t_1 & t_0 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ 0 \\ 0 \end{pmatrix}.$$

Generally, given an $n \times n$ Toeplitz matrix

$$T = \begin{pmatrix} t_0 & t_{-1} & t_{-2} & \cdots & t_{-(n-1)} \\ t_1 & t_0 & t_{-1} & \cdots & t_{-(n-2)} \\ t_2 & t_1 & t_0 & \cdots & t_{-(n-3)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ t_{n-1} & t_{n-2} & t_{n-3} & \cdots & t_0 \end{pmatrix},$$

T can be embedded into a $(2n-1+\delta) \times (2n-1+\delta)$ circulant matrix $\overline{T}$ (see, for example, [18]), where δ is an arbitrary nonnegative integer. As a circulant matrix, $\overline{T}$ can be uniquely determined by its first row $\overline{T}_{(1)}$:

$$\overline{T}_{(1)} = (t_0, t_{-1}, t_{-2}, \ldots, t_{-(n-2)}, t_{-(n-1)}, \underbrace{0, \ldots, 0}_{\delta}, t_{n-1}, t_{n-2}, \ldots, t_2, t_1).$$

The rest rows of $\overline{T}$ are the cyclic right shift by one bit of the previous one. To simplify the explanation, we let $\delta = 0$ in this article, i.e.,

$$\overline{T}_{(1)} = (t_0, t_{-1}, t_{-2}, \ldots, t_{-(n-2)}, t_{-(n-1)}, t_{n-1}, t_{n-2}, \ldots, t_2, t_1). \tag{3}$$

In order to embed the Toeplitz MVP $D = TA$ into a circulant MVP of dimension $N = 2n - 1$, which is denoted by R, the n-bit column vector A should also be extended to an N-bit column vector P by adding $(N - n) = (n - 1)$ extra 0's to A:

$$P = (p_0, p_1, \ldots, p_{2n-1})^T = (a_0, a_1, \ldots, a_{n-1}, \underbrace{0, \ldots, 0}_{n-1})^T. \tag{4}$$

Due to the property of the above embedding and the definition of P in equation (4), it is clear that the first n bits of the resulting circulant MVP

$R = (r_0, r_1, \ldots, r_{2n-2})^T = \overline{T}P$ are just the n-bit Toeplitz MVP $D = TA = (c_v, c_{v+1}, \ldots, c_{n-1}, c_0, c_1, \ldots, c_{v-1})$. Therefore, we have

$$R = (r_0, r_1, \ldots, r_{2n-2})^T = (c_v, c_{v+1}, \ldots, c_{n-1}, c_0, c_1, \ldots, c_{v-1}, \underbrace{r_n, r_{n+1}, \ldots, r_{2n-2}}_{n-1})^T. \tag{5}$$

After this step, we have embedded a Toeplitz MVP of dimension n, which corresponds to a GF(2^n) multiplication operation, into a circulant MVP of dimension $N = 2n - 1$. Because of the equivalence between the circulant MVP of dimension N and the multiplication operation in quotient ring GF(2)$[x]/(x^N + 1)$, we can also rewrite the circulant MVP $R = \overline{T}P$ as a multiplication in the quotient ring GF(2)$[x]/(x^N + 1)$. After obtaining the N-bit product vector R in equation (5) using some asymptotically faster multiplication algorithm, we reach the final step.

Step 4: Inversive coordinate transformation from D to C.

We have shown that the first n bits of the circulant MVP R in equation (5) are just the n-bit Toeplitz MVP $D = TA = (c_v, c_{v+1}, \ldots, c_{n-1}, c_0, c_1, \ldots, c_{v-1})^T$. Therefore, the coordinate column vector C of $c = ab$ in equation (2) can be obtained by first extracting the first n bits of R, i.e., the n-bit vector D, and then applying the following inversive coordinate transformation to D :

$$C = U^{-1}D = U^{-1}(c_v, c_{v+1}, \ldots, c_{n-1}, c_0, c_1, \ldots, c_{v-1})^T = (c_0, c_1, \ldots, c_{n-2}, c_{n-1})^T.$$

Compared to previous polynomial-based embedding methods, the proposed method is much more flexible since parameter δ in $N = 2n - 1 + \delta$ can be any nonnegative integer. Furthermore, the redundant degree N/n is approximately equal to 2 for all values of n if δ is small. On the other hand, the modified redundant representation of [1] provide small redundant degrees for only some values of n's. For the other values of n's, no improvement on redundant degrees is reported in their paper, for example, cases that n's are prime. As for the redundant representations in [4], [13], [24] and [17], their major disadvantage is that the redundant degrees are large for some values of n.

In table 1, we compare the redundant degrees N/n of different redundant representations for some values of $n \in [248, 299]$. These values are not successive because there are not proper or efficient redundant representations for the unlisted values of n. The values of δ in $N = 2n + \delta - 1$ are also listed in the table.

In this section, we have introduced the proposed idea at matrix level. In order to apply this idea to practical implementations, we need explicit formulae of elements in matrix $\overline{T}$ and vector P. So, we present a detailed description of step 2 and 3 in the next section.

4. Explicit formulae and complexities for irreducible trinomials

The key point of the redundant representation is to perform a GF(2^n) multiplication operation using a GF(2)$[x]/(x^N + 1)$ multiplication module. Therefore, we must map the two GF(2^n) elements a and b into two GF(2)$[x]/(x^N + 1)$ elements p and q first

Table 1. The redundant degrees N/n for some values of $n \in [248, 299]$

n	δ	[24] 2002	[13] 2002	[17] 2003	[1] 2012
248	994	1489/n=6.00	-	-	-
249	672	1169/n=4.69	-	-	-
250	126	625/n=2.50	-	-	-
251	2	-	503/n=2.00	-	-
281	2	-	563/n=2.00	-	-
293	2	-	587/n=2.00	-	-
285	-240	-	-	329/n=1.15	-
291	-274	-	-	307/n=1.05	-
260	-254	-	-	-	5*53/n=1.02
262	3	-	-	-	2*263/n=2.01
270	3	-	-	-	2*271/n=2.01
290	-284	-	-	-	5*59/n=1.02
Proposed: The redundant degrees are approximately 2 for all values of n.					

(or map the two n-bit coordinate column vector A and B to two N-bit coordinate column vector P and Q respectively). The mapping from a to p is simple: adding $(N-n) = (n-1)$ extra 0's to the n-bit vector A, and it is given in equation (4). We now derive the explicit formula that maps b to q (or B to Q).

In step 1, we have introduced the Mastrovito MVP equation $C = ZA$, where the $n \times n$ matrix $Z = (z_{i,j})_{0 \le i,j \le n-1}$ depends on only B and f. Since explicit expressions of $z_{i,j}$ are different according to the form of the trinomial $x^n + x^v + 1$, we only discuss the case "$n+1 \le 2v$ and $v \le n-2$" in this work. In this case, the following explicit expressions of $z_{v+t,i}$ can be found in [6]:

$$z_{v+t,i} = \begin{cases} b_{2v-n+t-i}, & 0 \le i \le 2v-n+t, \\ b_{2v+t-i}, & 2v-n+t+1 \le i \le v+t, \\ b_{v+n+t-i} + b_{2v+t-i}, & v+t+1 \le i \le n-1, \end{cases}$$

where $0 \le t \le n-v-2$.

For the other irreducible trinomial cases, i.e., "$n+1 > 2v > 0$ or $v = n-1$", we note that explicit expressions of $z_{v+t,i}$ can be derived using the method in [6], and the proposed architecture is also applicable.

After step 2 (transforming the Mastrovito MVP $C = ZA$ into the Toeplitz MVP $D = TA$), row v of matrix Z, i.e., $Z_{(v)}$, will become the first row of T, i.e., $T_{(1)}$. By the above equation, we get explicit expressions of this row:

$$\begin{aligned} Z_{(v)} = T_{(1)} = (&\underbrace{b_{2v-n}, b_{2v-n-1}, \ldots, b_0}_{2v-n+1}, \underbrace{b_{n-1}, b_{n-2}, \ldots, b_v}_{n-v}, \\ &\underbrace{b_{n-1} + b_{v-1}, b_{n-2} + b_{v-2}, \ldots, b_{v+1} + b_{2v-n+1}}_{n-v-1}). \end{aligned}$$

In step 3, we want to embed the Toeplitz MVP $D = TA$ into the circulant MVP $R = \overline{T}P$. Therefore, we also need explicit expressions of the first column of T to form the right half of the first row of $\overline{T}$ (see equation (3)). These explicit expressions can be obtained from the first column of Z, which are also listed in [6]:

$$Z^{(1)} = (\underbrace{b_0 + b_v, b_1 + b_{v+1}, \dots, b_{n-v-1} + b_{n-1}}_{n-v}, \underbrace{b_0 + b_{n-v}, b_1 + b_{n-v+1}, \dots, b_{2v-n-1} + b_{v-1}}_{2v-n}, \underbrace{b_{2v-n}, b_{2v-n+1}, \dots, b_{v-1}}_{n-v})^T.$$

After multiplying U to Z in step 2, we obtain the first column of T:

$$T^{(1)} = (\underbrace{b_{2v-n}, b_{2v-n+1}, \dots, b_{v-1}}_{n-v}, \underbrace{b_0 + b_v, b_1 + b_{v+1}, \dots, b_{n-v-1} + b_{n-1}}_{n-v}, \underbrace{b_0 + b_{n-v}, b_1 + b_{n-v+1}, \dots, b_{2v-n-1} + b_{v-1}}_{2v-n})^T.$$

Now we can form the first row of the $N \times N$ circulant matrix $\overline{T}$ from the first row and column of T:

$$\begin{aligned}\overline{T}_{(1)} = (&\underbrace{b_{2v-n}, b_{2v-n-1}, \dots, b_0}_{2v-n+1}, \underbrace{b_{n-1}, b_{n-2}, \dots, b_v}_{n-v}, \\ &\underbrace{b_{n-1} + b_{v-1}, b_{n-2} + b_{v-2}, \dots, b_{v+1} + b_{2v-n+1}}_{n-v-1}, \\ &\underbrace{b_{2v-n-1} + b_{v-1}, b_{2v-n-2} + b_{v-2}, \dots, b_0 + b_{n-v}}_{2v-n}, \\ &\underbrace{b_{n-v-1} + b_{n-1}, b_{n-v-2} + b_{n-2}, \dots, b_0 + b_v}_{n-v}, \underbrace{b_{v-1}, \dots, b_{2v-n+1}}_{n-v-1}).\end{aligned} \tag{6}$$

Equation (1), namely,

$$\overline{T}_{(1)} = (q_0, q_{N-1}, q_{N-2}, \dots, q_1)$$

reveals the relationship between GF(2)$[x]/(x^N + 1)$ element $q = \sum_{i=0}^{N} q_i x^i$ and the first row $\overline{T}_{(1)}$ of circulant matrix $\overline{T}$. Therefore, by comparing equation (1) with (6), we obtain the following mapping relationship between $Q = (q_0, q_1, \dots, q_{N-1})^T$ and $B = (b_0, b_1, \dots, b_{n-1})^T$:

$$q_t = \begin{cases} b_{t+2v-n}, & 0 \le t \le n-v-1, \\ b_{t+v-n} + b_{t+2v-n}, & n-v \le t \le 2n-2v-1, \\ b_{t+v-n} + b_{t+2v-2n}, & 2n-2v \le t \le n-1, \\ b_{t+v-n+1} + b_{t+2v-2n+1}, & n \le t \le 2n-v-2, \\ b_{t+2v-2n+1}, & 2n-v-1 \le t \le 3n-2v-2, \\ b_{t+2v-3n+1}, & 3n-2v-1 \le t \le 2n-2. \end{cases} \tag{7}$$

This transformation can be performed in parallel at a cost of $2n - v - 2 - (n - v - 1) = n - 1$ XOR gates at 1 XOR gate delay.

Now we analyse the complexity of each step. In **Step 1**, the number of XOR gates required to generate Mastrovito matrices for all irreducible trinomials are given in [6] as follows:

$$\begin{cases} n-1, \, n \neq 2v, \\ n/2, \quad n = 2v. \end{cases}$$

Because the transformation matrix U in **Step 2** and its inverse U^{-1} in **Step 4** involve only permutations of elements, no gate is required in these two steps. **Step 3**, i.e., embedding the Toeplitz MVP $D = TA$ into the circulant MVP $R = \overline{T}P$, does not require any gate either. Therefore, the total complexity of the proposed multiplication algorithm is $n-1$ XOR gates and 1 XOR gate delay plus $\mathcal{M}(n, N)$, which denotes the complexity to multiply the n-term polynomial p and the N-term polynomial q. It is clear that the value of $\mathcal{M}(n, N)$ is the dominate part of the total complexity. The multiplication of p and q can be performed using different algorithms, e.g., the schoolbook method, the Karatsuba algorithm and its generalization – the Toom-Cook algorithm and some asymptotically faster multiplication algorithm like FFT-based multiplication algorithm.

In [1], [13], [24] and [17], the schoolbook method is used to estimate the whole complexities of multipliers in these papers. Based on this method, the complexities of a GF(2)$[x]/(x^N + 1)$ multiplication operation are at most N^2 AND gates and $N(N-1)$ XOR gates [1]. Table 2 compares the AND and XOR complexities of a GF(2)$[x]/(x^N + 1)$ multiplication operation for values of n listed in table 1. For some values of n, the redundant degrees presented in [1] and [17] are slightly greater than 1, therefore, their complexities are comparable to quadratic parallel multipliers. Because the redundant degrees of the proposed method are about 2 for all values of n, it is not a better choice for small values of n. But for large values of n, the FFT or fast cyclic convolution algorithms can be used to compute the product $p \cdot q$, and therefore we can take the advantage of these asymptotically fast multiplication algorithms. In order to apply these algorithms, the value of N must satisfy certain strict conditions. Previous redundant representations provide only a *single* minimal embedding for one GF(2^n). Therefore, it may be hard to select a proper fast algorithm. However, the value of N in our method is $N = 2n - 1 - \delta$, where δ can be any nonnegative integer. The flexible choices of δ make it possible to select a proper N such that a proper asymptotically faster algorithm can be used.

Finally, we note that the squaring operation, which is an important step in the exponentiation computation, is almost free in most other redundant representations. But it is not free in the proposed redundant representation. Nevertheless it is still cheap for irreducible trinomials. In fact, the first n bits of the resulting circulant MVP R in (5), i.e., $c_v, c_{v+1}, \ldots, c_{n-1}, c_0, c_1, \ldots, c_{v-1}$, are the product of two SPB elements a and b. If a squaring operation is required after a multiplication operation, an SPB squarer can be used to perform this operation. The complexities of such an SPB squarer are quite low: $\lceil (n-1)/2 \rceil$ XOR gates and $1T_X$ gate delay [23] [9].

Table 2. #AND and #XOR of a GF(2)[x]/($x^N + 1$) multiplication operation for some values of n

		Proposed Method		Other Methods		Improvement rates	
n	N/n	#AND	#XOR	#AND	#XOR	AND	XOR
248	6.00	245025	244530	2217121	2215632 [24]	88.9%	89.0%
249	4.69	247009	246512	1366561	1365392 [24]	81.9%	81.9%
250	2.50	249001	248752	390625	390000 [24]	36.3%	36.2%
251	2.00	251001	250500	253009	252506 [13]	0.01%	0.01%
281	2.00	314721	314160	316969	316406 [13]	0.01%	0.01%
293	2.00	342225	341640	344569	343982 [13]	0.01%	0.01%
285	1.15	323761	323192	108241	107912 [17]	-199%	-199%
291	1.05	337561	336980	94249	93942 [17]	-258%	-259%
260	1.02	269361	268842	70225	81196 [1]	-284%	-231%
262	2.01	273529	273006	276676	345319 [1]	1.1%	20.9%
270	2.01	290521	289982	293764	366663 [1]	1.1%	20.9%
290	1.02	335241	334662	87025	100654 [1]	-285%	-232%

5. An example

We now present an example to illustrate the proposed multiplication algorithm. Let $\{x^{i-3} | 0 \le i \le 4\}$ be the SPB of GF(2^5) generated by $f(x) = x^5 + x^3 + 1$.

Step 1: Representing the product of a and b as a Mastrovito MVP.

Given two GF(2^5) elements $a = x^{-3}\sum_{i=0}^{4} a_i x^i$ and $b = x^{-3}\sum_{i=0}^{4} b_i x^i$, the coordinate column vector $C = (c_0, c_1, c_2, c_3, c_4)^T$ of $c = ab$ can be represented by the following Mastrovito MVP:

$$C = ZA = \begin{pmatrix} b_0+b_3 & b_2 & b_1 & b_0 & b_4 \\ b_1+b_4 & b_0+b_3 & b_2 & b_1 & b_0 \\ b_0+b_2 & b_1+b_4 & b_0+b_3 & b_2 & b_1 \\ b_1 & b_0 & b_4 & b_3 & b_4+b_2 \\ b_2 & b_1 & b_0 & b_4 & b_3 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \end{pmatrix}.$$

It is easy to see that

$$\begin{cases} c_0 = (b_0+b_3)a_0 + b_2a_1 + b_1a_2 + b_0a_3 + b_4a_4, \\ c_1 = (b_1+b_4)a_0 + (b_0+b_3)a_1 + b_2a_2 + b_1a_3 + b_0a_4, \\ c_2 = (b_0+b_2)a_0 + (b_1+b_4)a_1 + (b_0+b_3)a_2 + b_2a_3 + b_1a_4, \\ c_3 = b_1a_0 + b_0a_1 + b_4a_2 + b_3a_3 + (b_4+b_2)a_4, \\ c_4 = b_2a_0 + b_1a_1 + b_0a_2 + b_4a_3 + b_3a_4. \end{cases} \tag{8}$$

Step 2: Transforming the Mastrovito MVP $C = ZA$ into a Toeplitz MVP.

After multiplying

$$U = \begin{pmatrix} 0 & I_{2\times 2} \\ I_{3\times 3} & 0 \end{pmatrix}$$

to Z, Mastrovito matrix Z is transformed to the following Toeplitz matrix

$$T = UZ = \begin{pmatrix} b_1 & b_0 & b_4 & b_3 & b_4+b_2 \\ b_2 & b_1 & b_0 & b_4 & b_3 \\ b_0+b_3 & b_2 & b_1 & b_0 & b_4 \\ b_1+b_4 & b_0+b_3 & b_2 & b_1 & b_0 \\ b_0+b_2 & b_1+b_4 & b_0+b_3 & b_2 & b_1 \end{pmatrix}.$$

Step 3: Embedding the Toeplitz MVP $D = TA$ into a circulant MVP.

Then Toeplitz matrix T is embedded into the 9×9 circulant matrix $\overline{T}$ whose first row is

$$\overline{T}_{(1)} = (b_1, b_0, b_4, b_3, b_2 + b_4, b_0 + b_2, b_1 + b_4, b_0 + b_3, b_2),$$

and we obtain the circulant MVP $R = (c_3, c_4, c_0, c_1, c_2, r_5, r_6, r_7, r_8) = \overline{T}P$, where P is defined as

$$P = (a_0, a_1, a_2, a_3, a_4, 0, 0, 0, 0)^T \tag{9}$$

The circulant MVP $R = \overline{T}P$ is equivalent to the product of p and q in quotient ring $\mathrm{GF}(2)[x]/(x^9+1)$. The coordinate column vector P of $p = (1, x, x^2, \ldots, x^8)P$ is given by equation (9), and the coordinate column vector Q of $q = (1, x, x^2, \ldots, x^8)Q$ can be determined by equation (7) as follows:

$$Q = (b_1, b_2, b_0 + b_3, b_1 + b_4, b_0 + b_2, b_2 + b_4, b_3, b_4, b_0)^T. \tag{10}$$

After multiplying p and q in $\mathrm{GF}(2)[x]/(x^9 + 1)$, we get

$$\begin{aligned} r &= pq \bmod (x^9 + 1) \\ &= b_1a_0 + b_0a_1 + b_4a_2 + b_3a_3 + (b_4 + b_2)a_4 \\ &\quad +[b_2a_0 + b_1a_1 + b_0a_2 + b_4a_3 + b_3a_4]x \\ &\quad +[(b_0 + b_3)a_0 + b_2a_1 + b_1a_2 + b_0a_3 + b_4a_4]x^2 \\ &\quad +[(b_1 + b_4)a_0 + (b_0 + b_3)a_1 + b_2a_2 + b_1a_3 + b_0a_4]x^3 \\ &\quad +[(b_0 + b_2)a_0 + (b_1 + b_4)a_1 + (b_0 + b_3)a_2 + b_2a_3 + b_1a_4]x^4 \\ &\quad +r_5x^5 + r_6x^6 + r_7x^7 + r_8x^8. \end{aligned}$$

Step 4: Inversive coordinate transformation.

Finally, we apply the inverse coordinate transformation on the first five bits of R, i.e., coefficients of $1, x, x^2, x^3$ and x^4 in the above equation, and get the coordinate column vector C of $c = ab$ in $\mathrm{GF}(2^n)$. It is easy to check that coordinates of C obtained using this new method are equal to those given in (8).

6. Considerations for other bases of GF(2^n) over GF(2)

Besides SPB, the proposed matrix embedding method is also applicable to other bases of $\mathrm{GF}(2^n)$ over GF(2). To this end, multiplication operations in these bases must be transformed into Toeplitz MVPs first. For polynomial bases of $\mathrm{GF}(2^n)$

generated by irreducible trinomials $f(x) = x^n + x^k + 1$ $(2k < n)$, two methods were presented to transform a polynomial basis multiplication into a Toeplitz MVP in [15].

The first method is similar with the transformation of Step 2 in Section III. Given two GF(2^n) elements $a = \sum_{i=0}^{n-1} a_i x^i$ and $b = \sum_{i=0}^{n-1} b_i x^i$ represented in polynomial basis. Let $c = \sum_{i=0}^{n-1} c_i x^i = ab \bmod f(x)$. Define $A = (a_0, a_1, \ldots, a_{n-1})^T$ be the coordinate column vector of a, B and C are defined similarly.

In order to compute the coordinate column vector C of c, we may first multiply polynomials a and b:

$$s = ab = \sum_{t=0}^{2n-2} s_t x^t,$$

where

$$s_t = \begin{cases} \sum_{i=0}^{t} b_{t-i} a_i, & 0 \le t \le n-1, \\ \sum_{i=t+1-n}^{n-1} b_{t-i} a_i, & n \le t \le 2n-2. \end{cases} \tag{11}$$

Then we perform the reduction operation:

$$\begin{aligned} c = s \bmod f(x) &= \sum_{i=0}^{n-1} c_i x^i \\ &= \sum_{t=0}^{n-1} s_t x^t + \sum_{t=n}^{2n-k-1} s_t (x^{t-n+k} + x^{t-n}) \\ &\quad + \sum_{t=2n-k}^{2n-2} s_t (x^{t-2n+2k} + x^{t-2n+k} + x^{t-n}) \\ &= \sum_{t=0}^{n-1} s_t x^t + \sum_{t=k}^{n-1} s_{t+n-k} x^t + \sum_{t=0}^{n-k-1} s_{t+n} x^t \\ &\quad + \sum_{t=k}^{2k-2} s_{t+2n-2k} x^t + \sum_{t=0}^{k-2} s_{t+2n-k} x^t + \sum_{t=n-k}^{n-2} s_{t+n} x^t. \end{aligned} \tag{12}$$

These two steps can be combined into a MVP $C = ZA$. Multiplying the transformation matrix

$$U_k = \begin{pmatrix} 0 & I_{(n-k)\times(n-k)} \\ I_{k\times k} & 0 \end{pmatrix}$$

to Z, we obtain Toeplitz matrix $U_k Z$.

Reference [15] presented a brief description of the second transformation. We now presented a detailed description and proof of this transformation.

Definition 1. *Let Z_i^j denote the $(j - i + 1) \times n$ submatrix of Z formed by selecting rows $i, i + 1, \ldots, j - 1, j$, where $i \le j$.*

Definition 2. *Let $J_{i,j}$ represent the $n \times n$ elementary transformation matrix that adds row i to row j, and E be $\prod_{i=0}^{k-2} J_{i,i+k}$.*

Proposition 3. *$U_{2k-1}EZ$ is a Toeplitz matrix whose each element is a sum of at most two terms.*

Proof. We denote EZ as $\tilde{Z}$. By definition 2, transformation E adds Z_0^{k-2} to Z_k^{2k-2}, and $\tilde{Z}_k^{2k-2}$ is a Toeplitz matrix because Z_0^{k-1} and Z_k^{n-1} are two Toeplitz matrices. Now $\tilde{Z}$ consists of three Toeplitz submatrices:

$$\tilde{Z} = \begin{pmatrix} \tilde{Z}_0^{k-1} \\ \tilde{Z}_k^{2k-2} \\ \tilde{Z}_{2k-1}^{n-1} \end{pmatrix} = \begin{pmatrix} Z_0^{k-1} \\ Z_0^{k-2} + Z_k^{2k-2} \\ Z_{2k-1}^{n-1} \end{pmatrix}.$$

We first prove that the $(2k-1) \times n$ submatrix

$$\begin{pmatrix} \tilde{Z}_0^{k-1} \\ \tilde{Z}_k^{2k-2} \end{pmatrix}$$

is a Toeplitz matrix. To this end, we only need to find out the relationship between row $k-1$ and row k, which correspond to c_{k-1} and $c_0 + c_k$ respectively and can be obtained using (11) and (12):

$$\begin{aligned} c_{k-1} &= s_{k-1} + s_{k+n-1} \\ &= \sum_{i=0}^{k-1} b_{k-i-1}a_i + \sum_{i=k}^{n-1} b_{k+n-1-i}a_i, \end{aligned}$$

$$\begin{aligned} c_0 + c_k &= s_0 + s_k + 2s_n + 2s_{2n-k} + s_{k+n} \\ &= s_0 + s_k + s_{k+n} \\ &= a_0b_0 + \sum_{i=0}^{k} b_{k-i}a_i + \sum_{i=k+1}^{n-1} b_{k+n-i}a_i. \end{aligned}$$

A careful observation reveals that the first $n-1$ elements of row $k-1$ are equal to the last $n-1$ elements of row k. Therefore $\tilde{Z}$ contains only two Toeplitz submatrices: $\tilde{Z}_0^{2k-2}$ and $\tilde{Z}_{2k-1}^{n-1}$. Because row 0 and row $n-1$ of $\tilde{Z}$ are the same as those of Z, $U_{2k-1}EZ$ is a Toeplitz matrix.

Since each element of $\tilde{Z} = EZ$ is a sum of no more than two terms and premultiplication of U_{2k-1} to $\tilde{Z}$ only moves the upper $2k-1$ rows down below the lower $n-2k+1$ rows of $\tilde{Z}$, each element of Toeplitz matrix $U_{2k-1}EZ$ is a sum of at most two terms. □

For GF(2^n)s that Type II optimal normal bases exist, the optimal normal basis multiplication can be transformed into the summation of a Toeplitz MVP and a circulant MVP of dimensions n, see for example [8]. Therefore, the proposed matrix embedding method is applicable. Furthermore, reference [7] indicated that GF(2^n) multiplications in dual, weakly dual, and triangular bases can also be rewritten as Toeplitz MVPs. Therefore, the proposed method works for these bases too.

7. Conclusions

We have presented a new redundant representation to perform GF(2^n) multiplication. Compared to previous methods, it has flexible choice of the value of N and its redundant degree is approximately 2. In this work, we focus on the SPB and only discuss the case that GF(2^n) is generated by $f(x) = x^n + x^v + 1$ where "$n+1 \leq 2v$ and $v \leq n-2$". Explicit formulae that mapping a to p and b to q are derived for this case.

One important step in this method is that the GF(2^n) product formula must be rewritten as a Toeplitz MVP. For other cases of irreducible trinomials and the following two types of pentanomials: $x^n + x^{k+1} + x^k + x^{k-1} + 1$ and $x^{4s} + x^{3s} + x^{2s} + x^s + 1$, their SPB product formulae can also be transformed to Toeplitz MVPs. Detailed description of these transformation matrixes can be found in [7] . Therefore, the proposed method is also applicable to these irreducible polynomials.

References

[1] S. Akleylek and F. Ozbudak, Modified redundant representation for designing arithmetic circuits with small complexity, *IEEE Transactions on Computers* **61** (2012) 427–432.

[2] D. J. Bernstein and T. Lange, Type-II optimal polynomial bases, *Proceedings of WAIFI 2010*, *LNCS-6087* (2010), pp. 41–61.

[3] K. Chang, D. Hong and H. Cho, Low complexity bit-parallel multiplier for $GF(2^m)$ defined by all-one polynomials using redundant representation, *IEEE Transactions on Computers* **54** (2005) 1628–1630.

[4] G. Drolet, A new representation of elements of finite fields $GF(2^m)$ yielding small complexity arithmetic circuits, *IEEE Transactions on Computers* **47** (1998) 938–946.

[5] H. Fan and Y. Dai, New $GF(2^n)$ parallel multiplier using redundant representation (2004).

[6] H. Fan and Y. Dai, Fast bit-parallel $GF(2^n)$ multiplier for all trinomials, *IEEE Transactions on Computers* **54** (2005) 485–490.

[7] H. Fan and M. A. Hasan, A new approach to subquadratic space complexity parallel multiplier for extended binary fields, *IEEE Transactions on Computers* **56** (2007) 224–233.

[8] H. Fan and M. A. Hasan, Subquadratic computational complexity schemes for extended binary field multiplication using optimal normal bases, *IEEE Transactions on Computers* **56** (2007) 1435–1437.

[9] H. Fan and M. A. Hasan, Relationship between $GF(2^m)$ Montgomery and shifted polynomial basis multiplication algorithms, *IEEE Transactions on Computers* **55**(9) (2006) 1202–1206.

[10] S. Gao, J. von zur Gathen and D. Panario, Gauss periods and fast exponentiation in finite fields, *Proceedings of LATIN'95: Theoretical Informatics*, *LNCS-911* (1995), pp. 311–322.

[11] S. Gao, J. von zur Gathen, D. Panario and V. Shoup, Algorithms for exponentiation in finite fields, *J. Symbolic Computation* **29** (2000) 879–889.

[12] W. Geiselmann and H. Lukhaub, Redundant representation of finite fields, *Proceedings of Practice and Theory in Public Key Cryptography*, *LNCS 1992* (2001), pp. 339–352.

[13] W. Geiselmann, J. Muller-Quade and R. Steinwandt, On "a new representation of

elements of finite fields $GF(2^m)$ yielding small complexity arithmetic circuits", *IEEE Transactions on Computers* **51** (2002) 1460–1461.

[14] W. Geiselmann and R. Steinwandt, A redundant representation of $GF(q^n)$ for designing arithmetic circuits, *IEEE Transactions on Computers* **52** (2003) 848–853.

[15] M. A. Hasan, On matrix-vector product based sub-quadratic arithmetic complexity schemes for field multiplication, *Proceedings of SPIE 6697*, (2007), p. 669702.

[16] T. Itoh and S. Tsujii, Structure of parallel multipliers for a class of field $GF(2^m)$, *IEEE Transactions on Computers* **33** (1984) 357–360.

[17] R. Katti and J. Brennan, Low complexity multiplication in a finite field using ring representation, *IEEE Transactions on Computers* **52** (2003) 418–427.

[18] R. Kumar, A fast algorithm for solving a Toeplitz system of equations, *IEEE Transactions on Acoustics, Speech, and Signal Processing* **33** (1985) 254–267.

[19] A. H. Namin, H. Wu and M. Ahmadi, A new finite-field multiplier using redundant representation, *IEEE Transactions on Computers* **57** (2008) 716–720.

[20] G. Seroussi, Table of low-weight binary irreducible polynomials (1998).

[21] J. H. Silverman, Fast multiplication in finite fields $GF(2^n)$, *Proceedings of CHES 99: Cryptographic Hardware and Embedded Systems*, *LNCS-1717* (1999), pp. 122–134.

[22] J. von zur Gathen, A. Shokrollahi and J. Shokrollahi, Efficient multiplication using type 2 optimal normal bases, *Proceedings of WAIFI 2007*, *LNCS-4547* (2007), pp. 56–68.

[23] H. Wu, Montgomery multiplier and squarer for a class of finite fields, *IEEE Transactions on Computers* **51**(5) (2002) 521–529.

[24] H. Wu, M. A. Hasan, I. F. Blake and S. Gao, Finite field multiplier using redundant representation, *IEEE Transactions on Computers* **51** (2002) 1306–1316.

[25] H. Wu, M. Hasan and I. Blake, Highly regular architectures for finite field computation using redundant basis, *Proceedings of CHES 99: Cryptographic Hardware and Embedded Systems*, *LNCS-1717* (1999), pp. 269–279.