Brief Contributions

An Efficient Optimal Normal Basis Type II Multiplier

B. Sunar, *Member*, *IEEE*, and
Ç.K. Koç, *Senior Member*, *IEEE*

Abstract—This paper presents a new parallel multiplier for the Galois field $GF(2^m)$ whose elements are represented using the optimal normal basis of type II. The proposed multiplier requires $1.5(m^2 - m)$ XOR gates, as compared to $2(m^2 - m)$ XOR gates required by the Massey-Omura multiplier. The time complexities of the proposed and the Massey-Omura multipliers are similar.

Index Terms—Galois field, optimal normal basis, Massey-Omura multiplier, space complexity.

1 Introduction

ARITHMETIC operations in the Galois field $GF(2^m)$ (i.e., addition, subtraction, multiplication, and inversion) have several applications in coding theory, computer algebra, and cryptography [7], [5]. In these applications, time- and area-efficient algorithms and hardware structures are desired for addition, multiplication, squaring, and exponentiation operations. The performance of these operations is closely related to the representation of the field elements. An important advance in this area is the Massey-Omura algorithm [8], which is based on the normal basis representation of the field elements. One advantage of the normal basis is that the squaring of an element is computed by a cyclic shift of the binary representation. Efficient algorithms for the multiplication operation in the canonical basis have also been proposed [6], [3], [4]. The space and time complexities of these canonical basis multipliers are much less than those of the Massey-Omura multiplier.

In recent years, efficient normal basis multipliers for special classes of finite fields have been proposed [2], [4]. These multipliers work only for the optimal normal basis of type I. The Massey-Omura algorithm works for both the optimal normal basis of type I and type II. However, its parallel space complexity is about twice that of these special multipliers. The parallel Massey-Omura algorithm requires $2(m^2 - m)$ XOR gates, while both of the special multipliers in [2], [4] require $m^2 - 1$ XOR gates. This paper presents a new multiplication algorithm for the field $GF(2^m)$ whose elements are represented using the optimal normal basis of type II. The parallel multiplier proposed in this paper requires 25 percent fewer XOR gates than the Massey-Omura multiplier.

We also compare the proposed algorithm to a recently introduced multiplication method [1] for the optimal normal basis of type II, which is based on the palindromic representation of polynomials of length $2m$. The details and an analysis of this multiplication algorithm are not given in [1]; however, we expect that its XOR complexity will be at least $(2m)^2$.

- *B. Sunar is with the Department of Electrical and Computer Engineering, Worcester Polytechnic Institute, Worcester, MA 01609.*
- *Ç.K. Koç is with the Department of Electrical and Computer Engineering, Oregon State University, Corvallis, OR 97331. E-mail: koc@ece.orst.edu.*

Manuscript received 11 Feb. 1999; revised 4 May 2000; accepted 30 Sept. 2000.
For information on obtaining reprints of this article, please send e-mail to: tc@computer.org, and reference IEEECS Log Number 109190.

2 Optimal Normal Bases

The field $GF(2^m)$ is often viewed as an m-dimensional vector space defined over $GF(2)$. A set of m linearly independent vectors (elements of $GF(2^m)$) are chosen to serve as the basis of representation. The following are the most commonly used bases:

- A straightforward choice for a basis is the ordered set $\{1, \beta, \beta^2, \ldots, \beta^{m-1}\}$ where $\beta \in GF(2^m)$. This is called the canonical basis.
- If the set of elements $M = \{\beta, \beta^2, \beta^4, \ldots, \beta^{2^{m-1}}\}$ forms a basis for some $\beta \in GF(2^m)$, then the basis M is called normal basis and the element β is called normal element.

The introduction of the Massey-Omura multiplier [8] was followed by the definition of a special type of normal basis called optimal normal basis. This type of basis minimizes the complexity of the Massey-Omura multiplier. There exists two types of optimal normal basis, as classified in [7]. These bases are historically named as the optimal normal basis of type I and the optimal normal basis of type II.

An optimal normal basis of type II for the field $GF(2^m)$ is constructed using the normal element $\beta = \gamma + \gamma^{-1}$, where γ is a primitive $(2m+1)$th root of unity, i.e., $\gamma^{2m+1} = 1$ and $\gamma^i \neq 1$ for any $1 \leq i < 2m + 1$. It turns out that an optimal normal basis of type II can be constructed if $p = 2m + 1$ is prime and if either of the following two conditions also holds:

- 2 is a primitive root modulo p.
- $p = 7 \pmod 8$ and the multiplicative order of 2 modulo p is m.

The second condition also means that (-1) is a quadratic nonresidue modulo p and 2 generates the quadratic residues modulo p. As enumerated in [7, Table 5.1], there are 117 and 319 m values in the range $m \in [2, 2001]$, for which an optimal normal basis of type I and type II exists, respectively. In other words, the optimal normal basis of type II is three times more likely to occur in this range and, thus, efficient algorithms for this representation would be highly useful. In the following sections, we propose an efficient parallel algorithm for multiplying operands represented in the optimal normal basis of type II.

3 Optimal Normal Basis of Type II

We assume that $p = 2m + 1$ is a prime and either of the aforementioned two conditions holds, i.e., we have an optimal normal basis of type II in $GF(2^m)$ based on the normal element $\beta = \gamma + \gamma^{-1}$, where γ is the primitive pth root of unity. The basis is given as:

$$M = \{\beta, \beta^2, \beta^4, \ldots, \beta^{2^{m-1}}\}. \tag{1}$$

We now show that there exists another basis N which is obtained by a simple permutation of the basis elements in M and construct a new parallel multiplication algorithm in the new basis N. We examine both cases below:

- If 2 is primitive modulo p, then the set of powers of 2 modulo p

$$P_1 = \{2, 2^2, 2^3, \ldots, 2^{2m-1}, 2^{2m}\} \pmod p \tag{2}$$

is equivalent[1] to

1. Note that P_1 and Q_1 are sets, i.e., the elements are unordered.

β_1	β_2	$\cdots$	β_{m-2}	β_{m-1}	β_m
$a_1b_2 + a_2b_1$	$a_1b_3 + a_3b_1$	$\cdots$	$a_1b_{m-1} + a_{m-1}b_1$	$a_1b_m + a_mb_1$	
$a_2b_3 + a_3b_2$	$a_2b_4 + a_4b_2$	$\cdots$	$a_2b_m + a_mb_2$		
$\vdots$	$\vdots$				
$a_{m-2}b_{m-1} + a_{m-1}b_{m-2}$	$a_{m-2}b_m + a_mb_{m-2}$				
$a_{m-1}b_m + a_mb_{m-1}$					

Fig. 1. The construction of C_1.

$$Q_1 = \{1, 2, 3, 4, \ldots 2m\}. \tag{3}$$

Therefore, a basis element of the form $\gamma^{2^i} + \gamma^{-2^i}$ can be written as $\gamma^j + \gamma^{-j}$ for $j \in [1, 2m]$. Furthermore, it is always possible to rewrite $\gamma^j + \gamma^{-j}$ as $\gamma^{(2m+1)-j} + \gamma^{-(2m+1)+j}$; if $j \geq m+1$, then this has the benefit of bringing the power of γ to the range $[1, m]$.

- If the multiplicative order of 2 modulo p is equal to m, then the set of powers of 2 modulo p

$$P_2 = \{2, 2^2, 2^3, \ldots, 2^{m-1}, 2^m\} \pmod p \tag{4}$$

consists of m distinct integers in the range $[1, 2m]$. If $2^i \, (\text{mod} p)$ is in the range $[1, m]$, we leave it as it is. If $2^i \pmod p$ is in the range $[m+1, 2m]$, we write in its place the number $(2m+1) - (2^i \bmod p)$ to bring it to the range $[1, m]$. Since these numbers are all distinct, the set P_2 is equivalent to

$$Q_2 = \{1, 2, 3, 4, \ldots m\}. \tag{5}$$

As a result, following the presented facts, a basis element of the form $\gamma^{2^i} + \gamma^{-2^i}$ for $i \in [1, m]$ can be written uniquely as $\gamma^j + \gamma^{-j}$ with $j \in [1, m]$.

Consequently, the bases M and N are given as

$$M = \{\gamma + \gamma^{-1}, \gamma^2 + \gamma^{-2}, \gamma^{2^2} + \gamma^{-2^2}, \ldots, \gamma^{2^{(m-1)}} + \gamma^{-2^{(m-1)}}\} \tag{6}$$

$$N = \{\gamma + \gamma^{-1}, \gamma^2 + \gamma^{-2}, \gamma^3 + \gamma^{-3}, \ldots, \gamma^m + \gamma^{-m}\} \tag{7}$$

are equivalent. The basis N is obtained from the basis M using a simple permutation. Let A be expressed in the basis M as

$$A = a'_1\beta + a'_2\beta^2 + a'_3\beta^{2^2} + \cdots + a'_m\beta^{2^{m-1}}, \tag{8}$$

where $\beta = \gamma + \gamma^{-1}$. The representation of A in the basis N is given as

$$A = a_1\beta_1 + a_2\beta_2 + a_3\beta_3 + \cdots + a_m\beta_m, \tag{9}$$

where $\beta_i = \gamma^i + \gamma^{-i}$. We can express the permutation between the coefficients $a_j = a'_i$ as

$$j = \begin{cases} k & \text{if } k \in [1, m], \\ (2m+1) - k & \text{if } k \in [m+1, 2m], \end{cases} \tag{10}$$

where $k = 2^{i-1} (\text{mod } 2m+1)$ for $i = 1, 2, \ldots, m$. This permutation is a crucial part of the algorithm. It is used to convert the operands from the normal basis to a representation similar to the canonical basis. The inverse permutation is used to convert the elements back to the normal basis after the operation is completed.

The basis N is not a normal basis, it is a shifted form of the canonical basis [4]. Note that the exponents of the basis elements of the shifted canonical basis are one more than the ones of the canonical basis. We construct an efficient parallel multiplier in the following section using this new basis.

4 New Multiplication Algorithm

We propose a new algorithm for multiplying the elements of $GF(2^m)$ in the basis M as follows:

1. Convert the elements represented in the basis M to the basis N using the permutation.
2. Multiply the elements in the basis N.
3. Convert the result back to the basis M using the inverse permutation.

The first and third steps are implemented without any gates since the permutation operation requires a simple rewiring. The second step is a multiplication operation in the basis N, which we present below. Let $A, B \in GF(2^m)$ be represented in the basis N as

$$A = \sum_{i=1}^{m} a_i\beta_i = \sum_{i=1}^{m} a_i(\gamma^i + \gamma^{-i}) \tag{11}$$

$$B = \sum_{i=1}^{m} b_i\beta_i = \sum_{i=1}^{m} b_i(\gamma^i + \gamma^{-i}). \tag{12}$$

The product of these two numbers $C = A \cdot B$ is written as

$$C = A \cdot B = \left(\sum_{i=1}^{m} a_i(\gamma^i + \gamma^{-i})\right)\left(\sum_{j=1}^{m} b_j(\gamma^j + \gamma^{-j})\right). \tag{13}$$

This product can be transformed to the following form:

$$\begin{aligned} C &= \sum_{i=1}^{m}\sum_{j=1}^{m} a_ib_j(\gamma^{i-j} + \gamma^{-(i-j)}) + \sum_{i=1}^{m}\sum_{j=1}^{m} a_ib_j(\gamma^{i+j} + \gamma^{-(i+j)}) \\ &= C_1 + C_2. \end{aligned} \tag{14}$$

For future reference, the two double summations are denoted as C_1 and C_2, as shown above. The term C_1 has the property that the exponent $(i-j)$ of γ is already within the proper range, i.e., $-m \leq (i-j) \leq m$ for all $i, j \in [1, m]$. Furthermore, if $i = j$, then $\gamma^{i-j} + \gamma^{-(i-j)} = \gamma^0 + \gamma^0 = 0$. Thus, we can write C_1 as

$$C_1 = \sum_{i=1}^{m}\sum_{j=1}^{m} a_ib_j(\gamma^{i-j} + \gamma^{-(i-j)}) = \sum_{\substack{1 \leq i,j \leq m \\ i \neq j}} a_ib_j(\gamma^{i-j} + \gamma^{-(i-j)}). \tag{15}$$

If $k = |i-j|$, then the product a_ib_j contributes to the basis element $\beta_k = \gamma^k + \gamma^{-k}$. For example, the coefficients of β_1 are the sum of all a_ib_j for which $|i-j| = 1$. Fig. 1 shows the elements contributed by the summation C_1 arranged in terms of the order of the basis elements.

β_1	β_2	β_3	$\cdots$	β_{m-2}	β_{m-1}	β_m
	a_1b_1	a_1b_2	$\cdots$	a_1b_{m-3}	a_1b_{m-2}	a_1b_{m-1}
		a_2b_1	$\cdots$	a_2b_{m-4}	a_2b_{m-3}	a_2b_{m-2}
				$\vdots$	$\vdots$	$\vdots$
				$a_{m-3}b_1$	$a_{m-3}b_2$	$a_{m-3}b_3$
					$a_{m-2}b_1$	$a_{m-2}b_2$
						$a_{m-1}b_1$

Fig. 2. The construction of D_1.

Furthermore, the term C_2 is transformed into the following:

$$\begin{aligned} C_2 &= \sum_{i=1}^{m}\sum_{j=1}^{m} a_ib_j(\gamma^{i+j}+\gamma^{-(i+j)}) \\ &= \sum_{i=1}^{m}\sum_{j=1}^{m-i} a_ib_j(\gamma^{i+j}+\gamma^{-(i+j)}) + \sum_{i=1}^{m}\sum_{j=m-i+1}^{m} a_ib_j(\gamma^{i+j}+\gamma^{-(i+j)}) \\ &= D_1 + D_2. \end{aligned} \tag{16}$$

The double summations are denoted by D_1 and D_2, respectively. The exponents of the basis elements $\gamma^{i+j}+\gamma^{-(i+j)}$ in D_1 are guaranteed to be in the proper range $1 \le (i+j) \le m$ for $i = 1,2,\ldots,m$ and $j = 1,2,\ldots,m-i$. If $k = i+j$, then product a_ib_j contributes to the basis element β_k as i and j take these values. Fig. 2 shows the construction of the summation D_1.

On the other hand, the basis elements of D_2 are all out of range. We use the identity $\gamma^{2m+1} = 1$ to bring them to the proper range:

$$\begin{aligned} D_2 &= \sum_{i=1}^{m}\sum_{j=m-i+1}^{m} a_ib_j(\gamma^{i+j}+\gamma^{-(i+j)}) \\ &= \sum_{i=1}^{m}\sum_{j=m-i+1}^{m} a_ib_j(\gamma^{2m+1-(i+j)}+\gamma^{-(2m+1-(i+j))}). \end{aligned} \tag{17}$$

Therefore, if $k = i+j > m$, we replace β_k by β_{2m+1-k}. For example, the term a_mb_m contributes to the basis element β_1 since $2m+1-(m+m) = 1$. Fig. 3 shows the construction of D_2.

The multiplication algorithm in the basis N' constructs C_1, D_1, and D_2, and sums the appropriate terms in order to obtain the product $C = C_1 + D_1 + D_2$. The details of the multiplication operation and its complexity analysis are given in the following section.

5 Details of Multiplication and Complexity Analysis

If these three arrays C_1, D_1, and D_2 are inspected closely, the following observations can be made:

1. All three arrays are composed of the elements of the form a_ib_j for $i,j \in [1,m]$.
2. The height of the ith column in the array C_1 is $2(m-i)$ for $i = 1,2,\ldots,m$. This is the number of terms of the form a_ib_j to be summed in the ith column.
3. The height of the ith column in the array D_1 is equal to $i-1$.
4. The height of the ith column in the array D_2 is equal to i.
5. Therefore, the height of the ith column in the entire array representing the total sum C is found as $2(m-i)+i-1+i = 2m-1$, which follows from observations 2, 3, and 4.
6. If there is an element a_ib_j is present in a column, then the element a_jb_i is also present in the same column. This is true for all three arrays C_1, D_1, and D_2.
7. An element of the form a_ib_i is present only once in a column of either D_1 or D_2.
8. Because of observations 5, 6, and 7, a column of the entire array representing the total sum C contains a single element of the form a_ib_i and $2m-2$ elements of the form a_ib_j, where a_jb_i is also present.

The proposed multiplication algorithm first computes the terms a_ib_j for $i,j \in [1,m]$ using exactly m^2 two-input AND gates. This requires a single AND gate delay T_A because of the parallelism. Let $t_{ij} = a_ib_j + a_jb_i$ for $i = 1,2,\ldots,m$ and $j = i+1,i+2,\ldots,m$. We compute the terms t_{ij} using

$$(m-1)+(m-2)+\cdots+2+1 = \frac{1}{2}m(m-1) \tag{18}$$

two-input XOR gates and a single XOR gate delay T_X. The ith column of the entire array contains exactly $\frac{1}{2}(2m-2) = m-1$ terms of the form t_{ij} and also a single element of the form a_ib_i. These m numbers are summed using a binary XOR tree, which requires $m-1$ XOR gates and a total delay of $\lceil \log_2 m \rceil\, T_X$. Due to parallelism, all m columns require $m(m-1)$ XOR gates and the same amount of delay. Therefore, the construction of the product C requires

$$\begin{aligned} \#\ \text{AND Gates} &= m^2, \\ \#\ \text{XOR Gates} &= \frac{1}{2}m(m-1) + m(m-1) = \frac{3}{2}m(m-1), \\ \text{Gate Delay} &= T_A + T_X + \lceil \log_2 m \rceil\, T_X = T_A + (1+\lceil \log_2 m \rceil)T_X. \end{aligned}$$

On the other hand, the parallel Massey-Omura algorithm uses m^2 AND gates and $2m(m-1)$ XOR gates and computes the

β_1	β_2	β_3	$\cdots$	β_{m-2}	β_{m-1}	β_m
a_mb_m	$a_{m-1}b_m$	$a_{m-2}b_m$	$\cdots$	a_3b_m	a_2b_m	a_1b_m
	a_mb_{m-1}	$a_{m-1}b_{m-1}$	$\cdots$	a_4b_{m-1}	a_3b_{m-1}	a_2b_{m-1}
		a_mb_{m-2}	$\cdots$	a_5b_{m-2}	a_4b_{m-2}	a_3b_{m-2}
				$\vdots$	$\vdots$	$\vdots$
				$a_{m-1}b_4$	$a_{m-2}b_4$	$a_{m-3}b_4$
				a_mb_3	$a_{m-1}b_3$	$a_{m-2}b_3$
					a_mb_2	$a_{m-1}b_2$
						a_mb_1

Fig. 3. The construction of D_2.

Basis $\rightarrow$	β_1	β_2	β_3	β_4	β_5
$C_1 \rightarrow$	$a_1b_2 + a_2b_1$ $a_2b_3 + a_3b_2$ $a_3b_4 + a_4b_3$ $a_4b_5 + a_5b_4$	$a_1b_3 + a_3b_1$ $a_2b_4 + a_4b_2$ $a_3b_5 + a_5b_3$	$a_1b_4 + a_4b_1$ $a_2b_5 + a_5b_2$	$a_1b_5 + a_5b_1$	
$D_1 \rightarrow$		a_1b_1	a_1b_2 a_2b_1	a_1b_3 a_2b_2 a_3b_1	a_1b_4 a_2b_3 a_3b_2 a_4b_1
$D_2 \rightarrow$	a_5b_5	a_4b_5 a_5b_4	a_3b_5 a_4b_4 a_5b_3	a_2b_5 a_3b_4 a_4b_3 a_5b_2	a_1b_5 a_2b_4 a_3b_3 a_4b_2 a_5b_1
$C \rightarrow$	t_{12} t_{23} t_{34} t_{45} a_5b_5	t_{13} t_{24} t_{35} t_{45} a_1b_1	t_{14} t_{25} t_{12} t_{35} a_4b_4	t_{15} t_{13} t_{25} t_{34} a_2b_2	t_{14} t_{23} t_{15} t_{24} a_3b_3

Fig. 4. The construction of C_1, D_1, D_2, and C in $GF(2^5)$.

product in $T_A + (1 + \lceil \log_2 (m-1) \rceil) T_X$ gate delays. The proposed algorithm requires 25 percent fewer XOR gates than the Massey-Omura algorithm.

6 An Example

In this section, we illustrate the construction of the basis N and the new multiplication algorithm for the field $GF(2^5)$. Since $2m+1 = 2 \cdot 5 + 1 = 11$ and 2 is primitive in modulo 11, there exists an optimal basis of type II for the field $GF(2^5)$, which is of the form $M = \{\beta, \beta^2, \beta^4, \beta^8, \beta^{16}\}$, where $\beta = \gamma + \gamma^{-1}$. Using the identity $\gamma^{11} = 1$, we convert the basis M to the basis N. The first three exponents 1, 2, and 4 are in the proper range $[1, 5]$. We have $16 = 5 \pmod{11}$, which brings the exponent 16 to the proper range. In order to bring 8 to the range $[1, m] = [1, 5]$, we use the identity $\gamma^8 = \gamma^{8-11} = \gamma^{-3}$. Thus, we can write

$$
\begin{array}{lllllll}
\beta & = & \gamma + \gamma^{-1} & = & \gamma + \gamma^{-1} & = & \beta_1, \\
\beta^2 & = & \gamma^2 + \gamma^{-2} & = & \gamma^2 + \gamma^{-2} & = & \beta_2, \\
\beta^4 & = & \gamma^4 + \gamma^{-4} & = & \gamma^4 + \gamma^{-4} & = & \beta_4, \\
\beta^8 & = & \gamma^8 + \gamma^{-8} & = & \gamma^{-3} + \gamma^{3} & = & \beta_3, \\
\beta^{16} & = & \gamma^{16} + \gamma^{-16} & = & \gamma^5 + \gamma^{-5} & = & \beta_5.
\end{array}
$$

This gives a new basis which is of the form $N = \{\beta_1, \beta_2, \beta_3, \beta_4, \beta_5\}$. The conversion between these two bases is accomplished using a permutation. Assuming, A expressed in M is given as

$$A = (a'_1, a'_2, a'_3, a'_4, a'_5) = a'_1\beta + a'_2\beta^2 + a'_3\beta^4 + a'_4\beta^8 + a'_5\beta^{16},$$

we find the expression for A in N as

$$A = (a_1, a_2, a_3, a_4, a_5) = a_1\beta_1 + a_2\beta_2 + a_3\beta_3 + a_4\beta_4 + a_5\beta_5.$$

This gives the permutation as

$$(a_1, a_2, a_3, a_4, a_5) = (a'_1, a'_2, a'_4, a'_3, a'_5).$$

We now show the construction of the multiplication circuit. Let the elements A and B be given as inputs expressed in the basis M as

$$
\begin{aligned}
A &= (a'_1, a'_2, a'_3, a'_4, a'_5), \\
B &= (b'_1, b'_2, b'_3, b'_4, b'_5).
\end{aligned}
$$

The computation of the product $C = A \cdot B$ expressed in the basis M as $C = (c'_1, c'_2, c'_3, c'_4, c'_5)$ is computed using the following steps:

- First, we use the permutation to obtain the representations of A and B in the basis N as:

  $$
  \begin{array}{lllll}
  A & = & (a_1, a_2, a_3, a_4, a_5) & = & (a'_1, a'_2, a'_4, a'_3, a'_5), \\
  B & = & (b_1, b_2, b_3, b_4, b_5) & = & (b'_1, b'_2, b'_4, b'_3, b'_5).
  \end{array}
  $$

  This step requires a simple rewiring and no gates.
- Then, we generate the product terms a_ib_j for $i = 1, 2, 3, 4, 5$ and $j = 1, 2, 3, 4, 5$ using $m^2 = 5^2 = 25$ AND gates. This computation requires a single AND gate delay T_A.
- Then, we generate the terms $t_{ij} = a_ib_j + a_jb_i$ for $i = 1, 2, 3, 4, 5$ and $j = i+1, i+2, \ldots, 5$. Thus, we compute

  $$
  \begin{array}{llll}
  t_{12}=a_1b_2+a_2b_1 & t_{13}=a_1b_3+a_3b_1 & t_{14}=a_1b_4+a_4b_1 & t_{15}=a_1b_5+a_5b_1 \\
  t_{23}=a_2b_3+a_3b_2 & t_{24}=a_2b_4+a_4b_2 & t_{25}=a_2b_5+a_5b_2 & \\
  t_{34}=a_3b_4+a_4b_3 & t_{35}=a_3b_5+a_5b_3 & & \\
  t_{45}=a_4b_5+a_5b_4 & & &
  \end{array}
  $$

  This computation requires $\frac{1}{2}m(m-1) = 10$ XOR gates and a single XOR gate delay T_X.
- Then, we obtain the elements of the product as follows:

  $$
  \begin{aligned}
  c_1 &= t_{12} + t_{23} + t_{34} + t_{45} + a_5b_5 \\
  c_2 &= t_{13} + t_{24} + t_{35} + t_{45} + a_1b_1 \\
  c_3 &= t_{14} + t_{25} + t_{12} + t_{35} + a_4b_4 \\
  c_4 &= t_{15} + t_{13} + t_{25} + t_{34} + a_2b_2 \\
  c_5 &= t_{14} + t_{23} + t_{15} + t_{24} + a_3b_3.
  \end{aligned}
  $$

  This step requires an additional $m^2 - m = 20$ XOR gates. This computation is accomplished using additional delay of $\lceil \log_2 5 \rceil T_X = 3T_X$.
- The result is expressed in the basis N which is converted to the basis M using the inverse permutation as follows: $(c'_1, c'_2, c'_3, c'_4, c'_5) = (c_1, c_2, c_4, c_3, c_5)$.

In Fig. 4, we illustrate the construction of the arrays C_1, D_1, D_2, and the final array C. The multiplication circuit requires a total of $m^2 = 25$ AND gates and $1.5(m^2 - m) = 30$ XOR gates. The total computation is performed using $T_A + 4T_X$ gate delays.

7 A Similar Design

Another method for multiplication in the normal basis type II was described in a recent technical report [1]. This method uses the permutation described in Section 3 of this paper and, thus, it is also based on the shifted canonical representation. However, to multiply two polynomials represented in the shifted canonical basis, the palindromic representation is used. The palindromic representation of $a(x) = \sum_{i=1}^{m} a_i x^i$ is the polynomial $\sum_{i=1}^{2m} a_i x^i$, where $a_i = a_{p-i}$ for $i = 1, 2, \ldots, m$. It is proven in [1] that the multiplication of two such palindromic polynomials modulo $x^p - 1$ is equivalent to the optimal normal basis type II multiplication However, an explicit algorithm for multiplying two $2m$-length polynomials modulo $x^p - 1$ is not given in [1]. Therefore, we cannot compare their algorithm to the one presented here. The complexity results will depend on the details of the multiplication algorithm. However, we speculate that the XOR complexity of the method in [1] will be at least $(2m)^2 = 4m^2$ since the operands are of length $2m$.

8 Conclusions

We have presented a new parallel multiplier for the field $GF(2^m)$ whose elements are represented using the optimal normal basis of type II. The proposed bit-parallel multiplier requires $1.5(m^2 - m)$ XOR gates while the Massey-Omura multiplier requires $2(m^2 - m)$ XOR gates. The time complexities of these two multipliers are similar: The parallel Massey-Omura multiplier requires $T_A + (1 + \lceil \log_2(m-1) \rceil)T_X$ delays while the delay of the proposed multiplier is $T_A + (1 + \lceil \log_2 m \rceil)T_X$.

A serial version of the proposed multiplier is under consideration. However, we think that it may not be possible to take advantage of the symmetry $a_i b_j + a_j b_i$ in a serial version of the multiplier. Thus, the design of a serial version may require significant modification on the original algorithm.

Acknowledgments

This research is supported by Secured Information Technology, Inc. The work was performed while B. Sunar was with Oregon State University.

References

[1] I.F. Blake, R.M. Roth, and G. Seroussi, "Efficient Arithmetic in $GF(2^n)$ through Palindromic Representation," Hewlett-Packard, HPL-98-134, Aug. 1998.

[2] M.A. Hasan, M.Z. Wang, and V.K. Bhargava, "A Modified Massey-Omura Parallel Multiplier for a Class of Finite Fields," *IEEE Trans. Computers*, vol. 42, no. 11, pp. 1278-1280, Nov. 1993.

[3] T. Itoh and S. Tsujii, "Structure of Parallel Multipliers for a Class of Finite Fields $GF(2^m)$," *Information and Computation*, vol. 83, pp. 21-40, 1989.

[4] Ç.K. Koç and B. Sunar, "Low-Complexity Bit-Parallel Canonical and Normal Basis Multipliers for a Class of Finite Fields," *IEEE Trans. Computers*, vol. 47, no. 3, pp. 353-356, Mar. 1998.

[5] R. Lidl and H. Niederreiter, *Introduction to Finite Fields and Their Applications*. New York: Cambridge Univ. Press, 1994.

[6] E.D. Mastrovito, "VLSI Architectures for Multiplication over Finite Field GF(2^m)," *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, T. Mora, ed., pp. 297-309, Berlin: Springer-Verlag, 1988.

[7] *Applications of Finite Fields*, A.J. Menezes, ed. Boston: Kluwer Academic, 1993.

[8] J. Omura and J. Massey, "Computational Method and Apparatus for Finite Field Arithmetic," US Patent Number 4,587,627, May 1986.