New Complexity Results for Field Multiplication using Optimal Normal Bases and Block Recombination

M. A. Hasan[1], A. H. Namin[2,*] and C. Negre[1,3]

[1]ECE Department and CACR, University of Waterloo, Waterloo, Ontario, Canada.

[2] AMD, Markham, Ontario, Canada.

[3]Team DALI/ELIAUS, Université de Perpignan, Perpignan, France.

Abstract

In this article, we propose new schemes for subquadratic arithmetic complexity multiplication in binary fields using *optimal normal bases*. The schemes are based on a recently proposed method known as *block recombination*, which efficiently computes the sum of two products of Toeplitz matrices and vectors. Specifically, here we take advantage of some structural properties of the matrices and vectors involved in the formulation of field multiplication using optimal normal bases. This yields new space and time complexity results for corresponding bit parallel multipliers.

Keywords. Binary field, optimal normal basis, subquadratic space complexity multiplier, Toeplitz matrix, block recombination.

1 Introduction

Normal bases for finite fields representation were first proposed in 1888 by Kurt Hensel. Over binary fields of an extension degree of n, a normal basis is a set of the n consecutive squarings of a specially chosen field element. The main advantage in a normal basis is its low cost squaring operation through cyclic shift of the coordinates of an element. In 1989, Mullin *et al.* discovered two low complexity classes of normal bases known as Optimal Normal Bases (ONB) of type I and II [9]. These two types of bases have been shown to have great potential for creating high-speed multipliers suitable for cryptographic applications [9].

*This work was done while A.H Namin was with the University of Waterloo

In the literature, there exist a number of bit parallel normal basis multipliers with quadratic $(O(n^2))$ space complexity, e.g., [8], [6], [7]. Recently much attention has been paid to the design of architectures with subquadratic space complexity. In 2001, Leone presented the first architecture offering subquadratic space complexity for type I ONB via canonical basis multiplication using All One Polynomials.

In [3], Fan and Hasan presented a Toeplitz matrix-vector product scheme to design subquadratic complexity multiplier for both type I and II ONB. In [10], von zur Gathen, Shokrollahi and Shokrollahi have proposed to perform the multiplication in ONB-II using a conversion to a polynomial basis and then use subquadratic or any suitable method for polynomial multiplication. Due to their very efficient basis conversion process, their method outperforms the Fan-Hasan multiplier with regard to space complexity. This method has been slightly improved by Bernstein and Lange [1].

Recently, the authors of [5] have proposed a modification of the Toeplitx matrix multiplier of Fan and Hasan. This method which the authors refer to as *block recombination,* uses decomposition of the multiplier of Fan and Hasan in different blocks, and recombines them in a special way to reduce the space complexity.

In this work we further study the subquadratic complexity multiplication using ONB. We first show that the block recombination method can be used to design multipliers with better space complexity in comparison to Fan-Hasan ONB multipliers. We remark that vector and matrix symmetry properties exist in the matrix vector expression of ONB multiplication. The vector symmetry exploits the fact that the multiplication in ONB type II can be formulated as a summation of two Toeplitz matrices multiplied by a vector and its reverse. The matrix symmetry property makes use of the fact that the Toeplitz matrices used in ONB multiplication are made of a number of smaller Toeplitz matrices, some of which are transpose of each other. Our resulting multiplier has a space complexity slightly higher than the multiplier based on the Bernstein-Lange work [1], but it is about twice faster. This makes the proposed multiplier to have a better area-time product complexity.

The remainder of this paper is organized as follows. In Section 2 we briefly review the Fan-Hasan method to perform Toeplitz matrix-vector product and we recall also the block recombination approach of [5]. Section 3 presents the formulation of ONB multiplication using Toeplitz matrix-vector products. Then in Section 4 we apply block recombination method to ONB-I multiplication. In Section 5 we present the block recombination for the two-way split ONB-II multiplier based on vector and matrix symmetry properties. We then extend this method to the three-way split ONB-II multiplier in Section 6. Finally we compare our proposed schemes to recently published methods for field multiplication that use optimal normal bases and we give some concluding remarks (Section 7).

2 Preliminaries

2.1 Toeplitz matrix-vector product approach

A Toeplitz matrix is an $n \times n$ matrix $T = [t_{i,j}; i,j = 0,1,\cdots,n-1]$ such that $t_{i,j} = t_{i-1,j-1}$. Recently it has been shown that the multiplication operation in the binary field can be expressed as a Toeplitz matrix-vector product [2]. A new scheme was also proposed to reduce the space complexity of the Toeplitz matrix-vector multiplication at the cost of increasing its time complexity.

In this scheme if $2|n$, one can use a two-way split formula shown in the left column of Table 1. This models a Toeplitz matrix-vector product of size n by three Toeplitz matrix-vector products of size $n/2$ [2]. In a similar fashion, if $3|n$ one can use the three-way split shown in the right side of the same table which expresses a Toeplitz matrix-vector product of size n as six Toeplitz matrix-vector products of size $n/3$ each [2].

TABLE 1

Space and Time Complexities of Toeplitz matrix-vector product

Two-way split, $n = 2^i$	Three-way split, $n = 3^i$
$T = \begin{bmatrix} T_1 & T_0 \\ T_2 & T_1 \end{bmatrix} \begin{bmatrix} V_0 \\ V_1 \end{bmatrix} = \begin{bmatrix} P_0 + P_2 \\ P_1 + P_2 \end{bmatrix}$ $\begin{cases} P_0 &= (T_0+T_1)V_1 \\ P_1 &= (T_1+T_2)V_0 \\ P_2 &= T_1(V_0+V_1) \end{cases}$	$T = \begin{bmatrix} T_2 & T_1 & T_0 \\ T_3 & T_2 & T_1 \\ T_4 & T_3 & T_2 \end{bmatrix} \begin{bmatrix} V_0 \\ V_1 \\ V_2 \end{bmatrix} = \begin{bmatrix} P_0+P_3+P_4 \\ P_1+P_3+P_5 \\ P_2+P_4+P_5 \end{bmatrix}$ $\begin{cases} P_0 &= (T_0+T_1+T_2)V_2 \\ P_1 &= (T_0+T_1+T_3)V_1 \\ P_2 &= (T_2+T_3+T_4)V_0 \end{cases} \begin{cases} P_3 &= T_1(V_1+V_2) \\ P_4 &= T_2(V_0+V_2) \\ P_5 &= T_3(V_0+V_1) \end{cases}$

The space and time complexities which result by recursively applying the formulas of Table 1 for two-way split ($n = 2^i$) and three-way split ($n = 3^i$) are summarized in Table 2. In this table, D_A and D_X represent the delay of an AND gate and an XOR gate, respectively.

TABLE 2

Space and Time Complexities of Toeplitz matrix-vector product

Split-Size	# AND	# XOR	Delay
Two-way [2]	$n^{\log_2 3}$	$5.5n^{\log_2 3} - 6n + 0.5$	$D_A + 2\log_2(n)D_X$
Three-way [2]	$n^{\log_3 6}$	$4.8n^{\log_3 6} - 5n + 0.2$	$D_A + 3\log_3(n)D_X$

The Fan-Hasan multiplier architecture can be decomposed in different blocks: the component matrix formation (CMF), the component vector formation (CVF), the component multiplication (CM) and the reconstruction (R). The CMF and the CVF blocks recursively compute the component representation of the Toeplitz matrix T and the vector V, respectively. For example, in the two-way split case, CMF$_2$ and CVF$_2$ are defined as

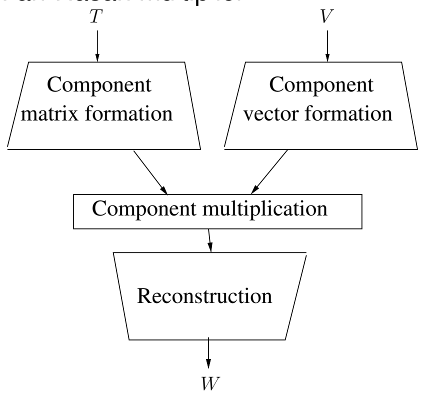
$$\begin{aligned}\mathrm{CMF}_2(T) &= [\mathrm{CMF}_2(T_1+T_0), \mathrm{CMF}_2(T_1), \mathrm{CMF}_2(T_2+T_1)],\\ \mathrm{CVF}_2(V) &= [\mathrm{CVF}_2(V_1), \mathrm{CVF}_2(V_0+V_1), \mathrm{CVF}_2(V_0)].\end{aligned}$$

The component multiplication consists of independent coefficient multiplication. The reconstruction converts the component representation of the product $\hat{W}$ to its vector representation W. For the two-way split case, if we split $\hat{W}$ in three parts $\hat{W} = (\hat{W}_0, \hat{W}_1, \hat{W}_2)$, then $R(\hat{W})$ is recursively calculated as

$$R(\hat{W}) = [R(\hat{W}_0) + R(\hat{W}_1), R(\hat{W}_2) + R(\hat{W}_1)].$$

The resulting block decomposition of the Fan-Hasan multiplier is shown in Fig. 1.

Fig. 1. Block decomposition of the Fan-Hasan multiplier



In [5] the authors have computed the complexity of each block for the two-way split and the three-way split approaches. These complexities are given in Table 3.

2.2 Brief review of block recombination approach

We recall here the basic block recombination method presented in [5]. The basic problem considered in [5] is to perform the following operation which is referred to as two-TMVPs-and-add:

$$T \cdot V + T' \cdot V'.$$

A straightforward architecture to perform this using the Fan-Hasan multiplier is shown in the left part of Fig. 2. In order to reduce the space complexity of this architecture, the authors of [5] have proved a commutative property concerning addition and reconstruction. Specifically, if $\hat{W}$ and $\hat{W}'$ are two vectors in component representation then we have

$$R(\hat{W}) + R(\hat{W}') = R(\hat{W} + \hat{W}').$$

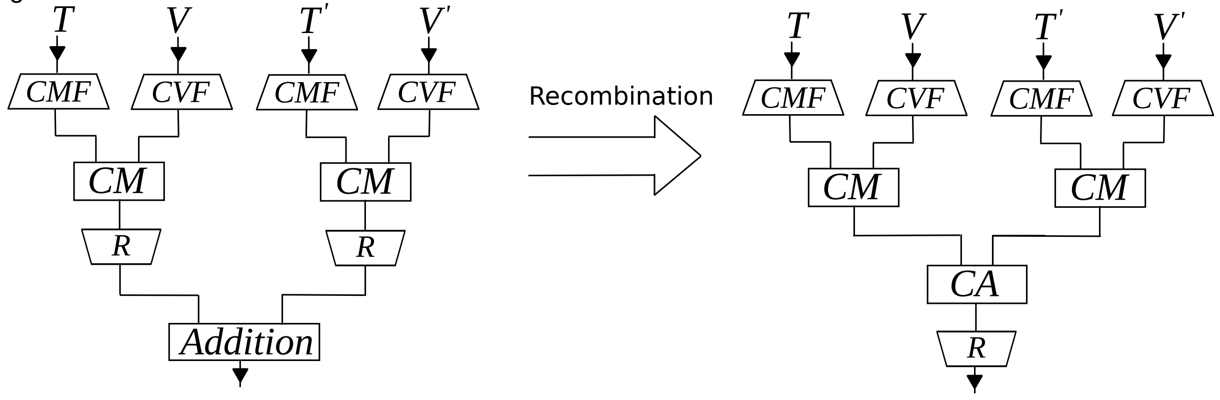
TABLE 3

Space and time complexities of different blocks in the Fan-Hasan multiplier

Computation	Two-way split	Three-way split
Component matrix formation(CMF)	$\mathcal{S}_{2,\oplus}^{CMF}(n) = \frac{5}{2}n^{\log_2(3)} - 3n + \frac{1}{2}$ $\mathcal{D}_{2}^{CMF}(n) = \log_2(n)D_X$	$\mathcal{S}_{3,\oplus}^{CMF}(n) = \frac{9}{5}n^{\log_3(6)} - 2n + \frac{1}{5}$ $\mathcal{D}_{3}^{CMF}(n) = 2\log_3(n)D_X$
Component vector formation (CVF)	$\mathcal{S}_{2,\oplus}^{CVF}(n) = n^{\log_2(3)} - n$ $\mathcal{D}_{2}^{CVF}(n) = \log_2(n)D_X$	$\mathcal{S}_{3,\oplus}^{CVF}(n) = n^{\log_3(6)} - n$ $\mathcal{D}_{3}^{CVF}(n) = \log_3(n)D_X$
Component multiplication (CM)	$\mathcal{S}_{2,\otimes}^{CM}(n) = n^{\log_2(3)}$ $\mathcal{D}_{2}^{CM}(n) = D_A$	$\mathcal{S}_{3,\otimes}^{CM}(n) = n^{\log_3(6)}$ $\mathcal{D}_{3}^{CM}(n) = D_A$
Reconstruction	$\mathcal{S}_{2,\oplus}^{R}(n) = 2n^{\log_2(3)} - 2n$ $\mathcal{D}_{2}^{R}(n) = \log_2(n)D_X$	$\mathcal{S}_{3,\oplus}^{R}(n) = 2n^{\log_3(6)} - 2n$ $\mathcal{D}_{3}^{R}(n) = 2\log_3(n)D_X$

The authors of [5] use this property to reduce the number of reconstruction blocks from two to one as shown in Fig. 2, where block CA performs a component addition.

Fig. 2. Basic two-TMVPs-and-add recombination



Remark 1. In [5] the authors extend the use of this previous basic recombination technique to recombine one single TMVP architecture. They express one TMVP of size n into two instances of two-TMVPs-and-add of size $n/2$ and then recombine each of them using the previously mentioned process. For the sake of simplicity we do not recall this recombination here.

3 Formulation of ONB multiplication using Toeplitz matrix-vector products

Binary field elements can be represented through normal basis $\mathcal{B} = \{\alpha, \alpha^2, \alpha^4, \ldots, \alpha^{2^{n-1}}\}$ for a given normal element $\alpha \in \mathbb{F}_{2^n}$. Among all normal bases, those which provide efficient multiplication have sparse basis representation of the n^2 product $\alpha^{2^i}\alpha^{2^j}$, or in other words, a small *complexity* [9]. Specifically, the optimal normal basis (ONB) are the normal bases which have the sparsest basis product. As shown in [9], there are two types of ONB, type I and type II, and they satisfy respectively one of the following theorems.

Theorem 1 (ONB-I). *Suppose $n+1$ is a prime and 2 is a primitive element in $\mathbb{Z}/(n+1)\mathbb{Z}$. Then the n non-unit $(n+1)$ -th roots of unity form a type I ONB of $\mathbb{F}_{2^n}$ over $\mathbb{F}_2$.*

Theorem 2 (ONB-II). *Let $2n+1$ be a prime and assume that either*

i. 2 is primitive in $\mathbb{Z}/(2n+1)\mathbb{Z}$, or

ii. $2n+1 \equiv 3 \mod 4$ and 2 generates the quadratic residues in $\mathbb{Z}/(2n+1)\mathbb{Z}$.

Then $\alpha = \beta + \beta^{-1}$ generates a type II ONB of $\mathbb{F}_{2^n}$ over $\mathbb{F}_2$, where β is a primitive $(2n+1)$-th root of unity in $\mathbb{F}_{2^{2n}}$.

3.1 TMVP expression for the multiplication in ONB of type I

Let $\mathcal{B} = \{\alpha, \alpha^2, \alpha^4, \ldots, \alpha^{2^{n-1}}\}$ be an ONB-I where α is an irreducible $(n+1)$-th root of unity. For practical purposes, it is generally preferable to use a permuted version of the basis $\mathcal{B}$ (cf. [7], [3]). Indeed, using Theorem 1, we can show that the following basis

$$\mathcal{B}' = \{\alpha, \alpha^2, \alpha^3, \ldots, \ldots, \alpha^n\}$$

is a permuted basis of $\mathcal{B}$.

Now, assume that $A = (a_1, a_2, ..., a_n)$ and $B = (b_1, b_2, ..., b_n)$ are two elements in $\mathbb{F}_{2^n}$ expressed in $\mathcal{B}'$. Then the product of A and B, which we denote as $C = (c_1, c_2, ..., c_n)$, is given by

$$C = \begin{bmatrix} 0 & b_n & b_{n-1} & \cdots & b_4 & b_3 & b_2 \\ b_1 & 0 & b_n & \cdots & b_5 & b_4 & b_3 \\ b_2 & b_1 & 0 & \cdots & b_6 & b_5 & b_4 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ b_{n-3} & b_{n-4} & b_{n-5} & \cdots & 0 & b_n & b_{n-1} \\ b_{n-2} & b_{n-3} & b_{n-4} & \cdots & b_1 & 0 & b_n \\ b_{n-1} & b_{n-2} & b_{n-3} & \cdots & b_2 & b_1 & 0 \end{bmatrix} \cdot \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_{n-2} \\ a_{n-1} \\ a_n \end{bmatrix} + \begin{bmatrix} b_n & b_{n-1} & \cdots & b_2 & b_1 \\ b_n & b_{n-1} & \cdots & b_2 & b_1 \\ b_n & b_{n-1} & \cdots & b_2 & b_1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ b_n & b_{n-1} & \cdots & b_2 & b_1 \\ b_n & b_{n-1} & \cdots & b_2 & b_1 \end{bmatrix} \cdot \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_{n-1} \\ a_n \end{bmatrix} \qquad (1)$$

We remark that the first matrix is Toeplitz, and the second matrix-vector can be computed through only one row-vector product, since all rows of the matrix are the same.

3.2 TMVP expression for multiplication in ONB of type II

Let $\mathcal{B} = \{\alpha, \alpha^2, \alpha^4, \ldots, \alpha^{2^{n-1}}\}$ be an ONB-II where $\alpha = \beta + \beta^{-1}$ satisfies the condition of Theorem 2. It has been proved that the basis

$$\mathcal{B}' = \{\alpha'_1, \ldots, \alpha'_n\}, \text{ where } \alpha'_i = \beta^i + \beta^{-i} \text{ and } \beta \text{ is as defined in Theorem 2,}$$

is a permuted basis of $\mathcal{B}$.

Assume that $A = (a_1, a_2, \ldots, a_n)$ and $B = (b_1, b_2, \ldots, b_n)$ are two elements in $\mathbb{F}_{2^n}$ expressed in the permuted basis $\mathcal{B}'$. Then, it has been shown in [7], [3] that the product of A and B is given by

$$C = \left(\begin{bmatrix} b_2 & b_3 & b_4 & \ldots & b_n & b_n \\ b_3 & b_4 & b_5 & \ldots & b_n & b_{n-1} \\ b_4 & b_5 & b_6 & \ldots & b_{n-1} & b_{n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ b_n & b_n & b_{n-1} & \ldots & b_3 & b_2 \\ b_n & b_{n-1} & b_{n-2} & \ldots & b_2 & b_1 \end{bmatrix} + \begin{bmatrix} 0 & b_1 & b_2 & \ldots & b_{n-2} & b_{n-1} \\ b_1 & 0 & b_1 & \ldots & b_{n-3} & b_{n-2} \\ b_2 & b_1 & 0 & \ldots & b_{n-4} & b_{n-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ b_{n-2} & b_{n-3} & b_{n-4} & \ldots & 0 & b_1 \\ b_{n-1} & b_{n-2} & b_{n-3} & \ldots & b_1 & 0 \end{bmatrix} \right) \cdot \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_{n-1} \\ a_n \end{bmatrix}. \quad (2)$$

As can be seen, the multiplication operation is formulated as a sum of a Hankel matrix-vector and a Toeplitz matrix-vector products. Writing the columns of the Hankel matrix in the reverse order, the multiplication can be expressed as products of two Toeplitz matrix-vector products as follows

$$C = \begin{bmatrix} b_n & b_n & b_{n-1} & \ldots & b_3 & b_2 \\ b_{n-1} & b_n & b_n & \ldots & b_4 & b_3 \\ b_{n-2} & b_{n-1} & b_n & \ldots & b_5 & b_4 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ b_2 & b_3 & b_4 & \ldots & b_n & b_n \\ b_1 & b_2 & b_3 & \ldots & b_{n-1} & b_n \end{bmatrix} \cdot \begin{bmatrix} a_n \\ a_{n-1} \\ a_{n-2} \\ \vdots \\ a_2 \\ a_1 \end{bmatrix} + \begin{bmatrix} 0 & b_1 & b_2 & \ldots & b_{n-2} & b_{n-1} \\ b_1 & 0 & b_1 & \ldots & b_{n-3} & b_{n-2} \\ b_2 & b_1 & 0 & \ldots & b_{n-4} & b_{n-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ b_{n-2} & b_{n-3} & b_{n-4} & \ldots & 0 & b_1 \\ b_{n-1} & b_{n-2} & b_{n-3} & \ldots & b_1 & 0 \end{bmatrix} \cdot \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ xo\vdots \\ a_{n-1} \\ a_n \end{bmatrix}. \quad (3)$$

4 Recombination of ONB-I multiplier

In this section we present a block recombination approach for ONB-I multiplier. Specifically we will use the special structure of the $n \times n$ Toeplitz matrix involved in the ONB-I multiplier.

4.1 Recombination of two-way split approach of ONB-I multiplier

We consider the Toeplitz matrix-vector product in the right side of (1).

$$\begin{bmatrix} 0 & b_n & b_{n-1} & \ldots & b_4 & b_3 & b_2 \\ b_1 & 0 & b_n & \ldots & b_5 & b_4 & b_3 \\ b_2 & b_1 & 0 & \ldots & b_6 & b_5 & b_4 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ b_{n-3} & b_{n-4} & b_{n-5} & \ldots & 0 & b_n & b_{n-1} \\ b_{n-2} & b_{n-3} & b_{n-4} & \ldots & b_1 & 0 & b_n \\ b_{n-1} & b_{n-2} & b_{n-3} & \ldots & b_2 & b_1 & 0 \end{bmatrix} \cdot \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_{n-2} \\ a_{n-1} \\ a_n \end{bmatrix} \quad (4)$$

Assuming that $2|n$ we can split the matrix into four blocks and the vector into two blocks

$$T \cdot V = \begin{bmatrix} T_1 & T_0 \\ T_2 & T_1 \end{bmatrix} \cdot \begin{bmatrix} A_0 \\ A_1 \end{bmatrix} = \begin{bmatrix} T_1 \cdot A_0 + T_0 \cdot A_1 \\ T_2 \cdot A_0 + T_1 \cdot A_1 \end{bmatrix}$$

We would like to apply the method presented in [5] to perform this computation. A direct application of [5] gives an architecture which performs the CMF$_2$ of T_0, T_1 and T_2 and the CVF$_2$ of V_0 and V_1 in parallel. After that it performs component multiplications, component additions and reconstructions. Here we would like to improve this scheme by combining the CMF$_2$ of T_1 and T_2. To do so, we need to rewrite the matrix-vector product $T_0 \cdot A_1$.

$$\begin{aligned} T_0 \cdot A_1 &= \begin{bmatrix} b_{n/2+1} & b_{n/2} & \cdots & b_4 & b_3 & b_2 \\ b_{n/2+2} & b_{n/2+1} & \cdots & b_5 & b_4 & b_3 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ b_{n-1} & b_{n-2} & \cdots & b_{n/2+2} & b_{n/2+1} & b_{n/2} \\ b_n & b_{n-1} & \cdots & b_{n/2+3} & b_{n/2+2} & b_{n/2+1} \end{bmatrix} \cdot \begin{bmatrix} a_{n/2+1} \\ a_{n/2+2} \\ \vdots \\ a_{n-2} \\ a_{n-1} \\ a_n \end{bmatrix} \\ &= a_{n/2+1} \begin{bmatrix} b_{n/2+1} \\ b_{n/2+2} \\ \vdots \\ b_{n-1} \\ b_n \end{bmatrix} + \begin{bmatrix} b_{n/2} & b_{n/2-1} & \cdots & b_4 & b_3 & b_2 & b_1 \\ b_{n/2+1} & b_{n/2} & \cdots & b_5 & b_4 & b_3 & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ b_{n-2} & b_{n-3} & \cdots & b_{n/2+2} & b_{n/2+1} & b_{n/2} & b_{n/2-1} \\ b_{n-1} & b_{n-2} & \cdots & b_{n/2+3} & b_{n/2+2} & b_{n/2+1} & b_{n/2} \end{bmatrix} \cdot \begin{bmatrix} a_{n/2+2} \\ a_{n/2+3} \\ \vdots \\ a_{n-1} \\ a_n \\ 0 \end{bmatrix} \end{aligned} \tag{5}$$

It can be seen that the latter $n/2 \times n/2$ matrix is equal to T_2

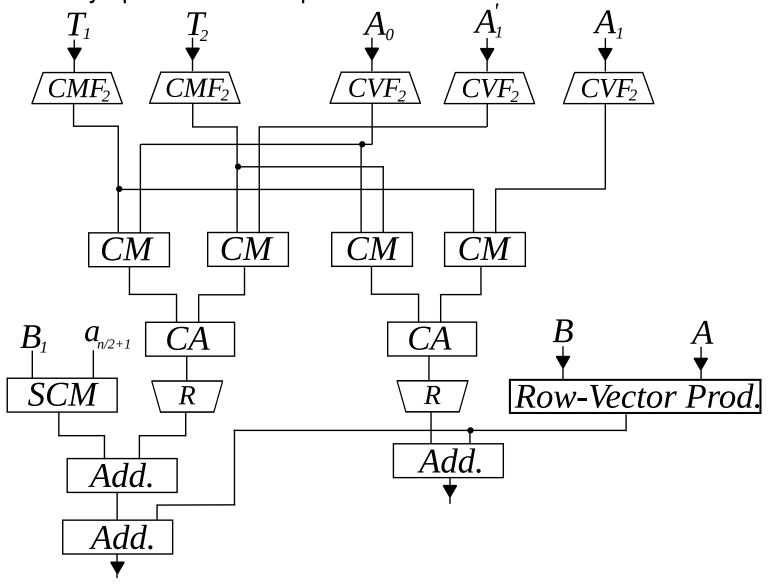
$$T_2 = \begin{bmatrix} b_{n/2} & b_{n/2-1} & b_{n/2-2} & \cdots & b_2 & b_1 \\ b_{n/2+1} & b_{n/2} & b_{n/2-1} & \cdots & b_3 & b_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ b_{n-2} & b_{n-3} & b_{n-4} & \cdots & b_{n/2} & b_{n/2-1} \\ b_{n-1} & b_{n-2} & b_{n-3} & \cdots & b_{n/2+1} & b_{n/2} \end{bmatrix}. \tag{6}$$

This means that we can calculate the matrix vector product $T_0 \cdot A_1$ by performing $T_0 \cdot A_1 = a_{n/2+1}B_1 + T_2 \cdot A'_1$ where

$$A'_1 = \begin{bmatrix} a_{n/2+2} \\ a_{n/2+3} \\ \vdots \\ a_{n-1} \\ a_n \\ 0 \end{bmatrix} \text{ and } B_1 = \begin{bmatrix} b_{n/2+1} \\ b_{n/2+2} \\ \vdots \\ b_{n-1} \\ b_n \end{bmatrix}.$$

We can thus design the architecture for multiplication in ONB-I shown in Fig. 3 which avoids the computation of CMF(T_0). In Fig. 3, the block SCM refers to scalar multiplication, and consists of $n/2$ parallel AND gates and the block Add to an addition through $n/2$ parallel XOR gates. The block *Row-Vector Product* refers to n parallel AND gates and a binary of XOR gates which compute the row-vector in the right side of of (1).

Fig. 3. Recombined two-way split ONB-I multiplier



Complexity evaluation. We assume that n is a power of 2. We count the contribution of each block in terms of the number of AND and XOR gates in the architecture in Fig. 3 and we obtain

$$\begin{aligned}\mathcal{S}_{2,\oplus} &= 2\mathcal{S}_{2,\oplus}^{CMF}(n/2) + 3\mathcal{S}_{2,\oplus}^{CVF}(n/2) + 2\mathcal{S}_{2,\oplus}^{CA}(n/2) + 2\mathcal{S}_{2,\oplus}^{R}(n/2) + \tfrac{5n}{2} - 1,\\ \mathcal{S}_{2,\otimes} &= 4\mathcal{S}_{2,\oplus}^{CM}(n/2) + \tfrac{3n}{2}.\end{aligned}$$

The $3n/2$ AND gates come from the SCM block and the row-vector product, and the $5n/2-1$ XOR gates come from the row-vector products and the additions. We finally obtain the following results using the block complexities given in Table 3.

$$\begin{aligned}\mathcal{S}_{2,\oplus} &= \tfrac{14}{3}n^{\log_2(3)} - 4n + 1,\\ \mathcal{S}_{2,\otimes} &= \tfrac{4}{3}n^{\log_2(3)} + \tfrac{3n}{2}.\end{aligned}$$

Remark 2. If n is not a power of 2 we can still design the subquadratic multiplier shown in Fig. 3 using the previously presented strategy. Indeed, since we have an ONB of type I, then $n+1$ is prime and thus n is even. This implies that the decomposition in four sub-matrices is always possible. If $n/2$ is not a power of 2, we enlarge the $n/2 \times n/2$ Toeplitz-matrices and vectors to a size of the form 2^k3^l. Then we apply Fan-Hasan two-way and three-way split formulas to design the CMF$_2$, CVF$_2$, CM, CA and R blocks we have in Fig. 3.

4.2 Recombination of three-way split approach of ONB-I multiplier

We consider the matrix-vector product given in (1) for multiplication in ONB-I. The following lemma can be used for efficient recombination of three-way split approach for ONB-I.

Lemma 1. *The Toeplitz matrix-vector product of* (1) *can be performed as follows*

$$T \cdot A = \begin{bmatrix} T_2 & T_1 & T_0 \\ T_3 & T_2 & T_1 \\ T_4 & T_3 & T_2 \end{bmatrix} \cdot \begin{bmatrix} A_0 \\ A_1 \\ A_2 \end{bmatrix} = \begin{bmatrix} T_2 \cdot A_0 + a_{n/3+1} B_1 + T_3 \cdot A_1' + T_4 \cdot A_2' \\ T_3 \cdot A_0 + T_2 \cdot A_1 + a_{2n/3+1} B_1 + T_3 \cdot A_2' \\ T_4 \cdot A_0 + T_3 \cdot A_1 + T_2 \cdot A_2 \end{bmatrix} \tag{7}$$

where we have

$$B_2 = \begin{bmatrix} b_{2n/3+1} \\ b_{2n/3+2} \\ \vdots \\ b_n \end{bmatrix}, \quad A_1' = \begin{bmatrix} a_{n/3+2} \\ a_{n/3+3} \\ \vdots \\ a_{2n/3+1} \end{bmatrix}, \quad A_2' = \begin{bmatrix} a_{2n/3+2} \\ a_{n/3+3} \\ \vdots \\ a_n \\ 0 \end{bmatrix}. \tag{8}$$

Proof: The proof is similar to the proof for the two-way split approach. We have to just check that the matrix in (1) can be decomposed in different sub-matrices given in (7). □

Using (7), we remark that we can design a recombined TMVP architecture with only three CMF$_3$ blocks, but we will need five CVF$_3$ blocks. This overall reduces space complexity since a CMF$_3$ is twice bigger than a CVF$_3$ block. The resulting architecture is shown in Fig. 4

Complexity evaluation. We assume that n is a power of 3. The contribution of each block in the number of AND gate and XOR gates of the architecture is as follows

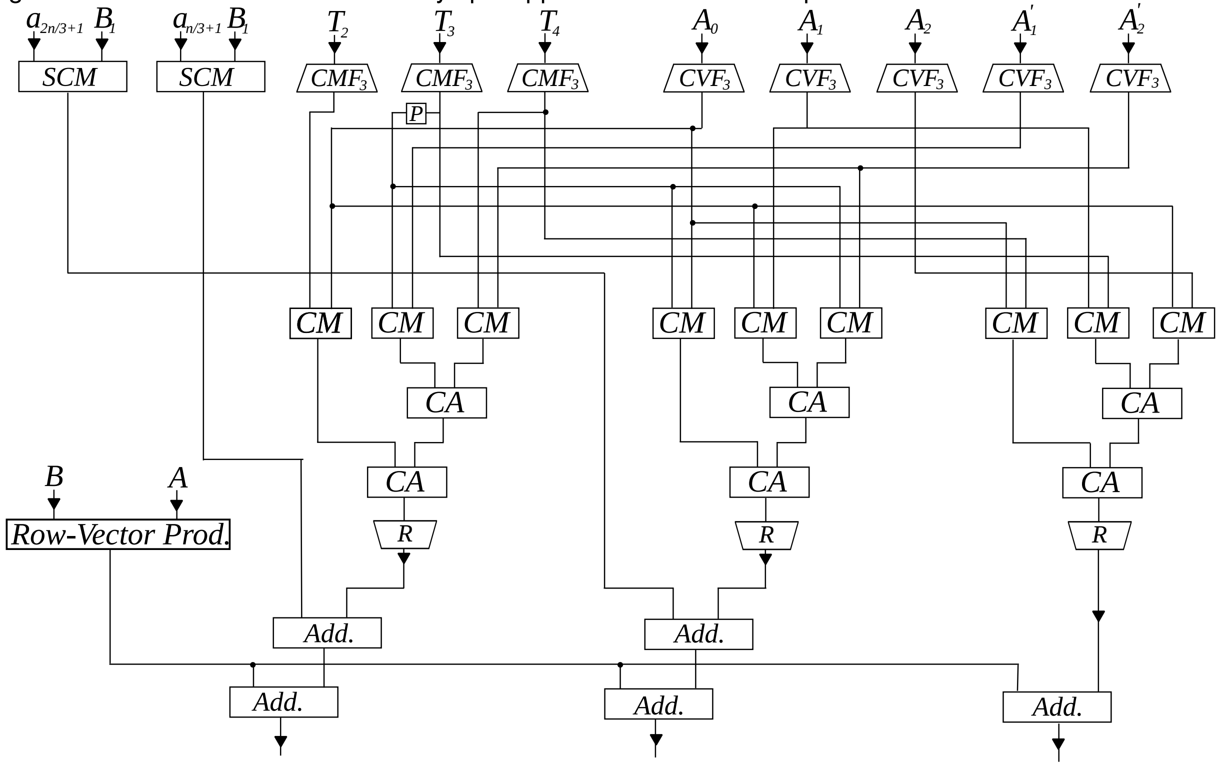
$$\begin{array}{rcl} \mathcal{S}_{3,\oplus} & = & 3\mathcal{S}_{3,\oplus}^{CMF}(n/3) + 5\mathcal{S}_{3,\oplus}^{CVF}(n/3) + 6\mathcal{S}_{3,\oplus}^{CA}(n/3) + 3\mathcal{S}_{3,\oplus}^{R}(n/3) + 8n/3, \\ \mathcal{S}_{3,\otimes} & = & 9\mathcal{S}_{3,\otimes}^{CM}(n/3) + 5n/3. \end{array}$$

Using the block complexities given in Table 3, we have

$$\begin{array}{rcl} \mathcal{S}_{3,\oplus} & = & \frac{56}{15} n^{\log_3(6)} - \frac{43}{9} n + \frac{3}{5}, \\ \mathcal{S}_{3,\otimes} & = & \frac{3}{2} n^{\log_3(6)} + \frac{5n}{3}. \end{array}$$

Remark 3. In situations where n is not a power of 3 we can still design a subquadratic multiplier as shown in Fig. 3 if n can be at least divided by 3. Indeed if n is a factor of 3 we can express the TMVP of ONB-I using (7). Then if $n/3$ is not a power of 3, we can enlarge the size $n/3$ matrices and vectors to a size of the form $2^k 3^l$. Then we can apply Fan-Hasan two-way and three-way formulas to design CMF$_3$, CVF$_3$, CM, CA and R blocks which are in Fig. 4.

Fig. 4. Recombination of the three-way split approach for ONB-I multipliers



5 Recombination of two-way split ONB-II Fan-Hasan multiplier

In this section, we present a block recombination of the Fan-Hasan ONB-II multiplier. We will do this recombination in a slightly different way: we first recombine the two-TMVPs-and-add expression of the ONB-II multiplier, then we set vector symmetry and matrix symmetry in CMF and CVF computation and use these properties for block recombination of the Fan-Hasan ONB-II multiplier.

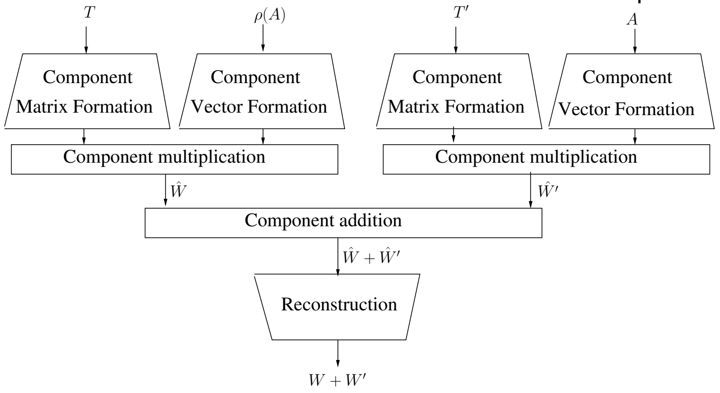
5.1 Recombination of two-TMVPs-and-add architecture

We get back to the addition of two Toeplitz matrix-vector products (two-TMVPs-and-add) given in (3) for the multiplication in ONB-II. In the remainder of this section we note T the first Toeplitz matrix and T' the second matrix in (3). Moreover we will note $\rho(A)$ the vector having the coordinates of A in the reverse order. Then (3) can be rewritten as

$$C = T \cdot \rho(A) + T' \cdot A. \qquad (9)$$

We can directly apply the recombination scheme of [5] presented in Section 2 to the Fan-Hasan ONB-II multiplier. The resulting architecture is presented in Fig. 5.

Fig. 5. Recombination of the reconstruction block of Fan-Hasan ONB-II multiplier



5.2 Recombination using vector symmetry

Considering the fact that in (9) the vectors A and $\rho(A)$ are the reverse of each other, we can join the component vector formation of these two vectors. In order to do that we will use some symmetric property of the CVF function. We first begin with some basic properties of the coordinate reversing order.

Lemma 2. *Let V and V' be two vectors of length m. Then the following assertion holds*

i. $\rho(V+V') = \rho(V)+\rho(V')$.

ii. $\rho(\begin{bmatrix} V \\ V' \end{bmatrix}) = \begin{bmatrix} \rho(V') \\ \rho(V) \end{bmatrix}$

The CVF function computes the component representation of vector V of size n. In the two-way split approach, CVF$_2$ is computed recursively as follows

$$\mathrm{CVF}_2(V) = \begin{cases} V, \text{ if } V \quad \text{has size } 1, \\ [\mathrm{CVF}_2(V_1),\, \mathrm{CVF}_2(V_0+V_1),\, \mathrm{CVF}_2(V_0)], \quad \text{if } V = \begin{bmatrix} V_0 \\ V_1 \end{bmatrix} \text{ and } V_0 \text{ and } V_1 \text{ are of size } n/2. \end{cases} \tag{10}$$

The following lemma state that the coordinate reversing order commute with the CVF$_2$ function.

Lemma 3. *Let V be a vector of size n, and let $\rho(V)$ be the vector which has the coordinates of V in the reverse order. Let $CVF_2(V)$ be the two-way split approach for the component vector formation of V. Then*

$$CVF_2(\rho(V)) = \rho(CVF_2(V)) \tag{11}$$

Proof: We can prove it by induction on the size of V. It is clear from (10) which defines the function CVF$_2$ that the lemma is true when V has size $n=1$.

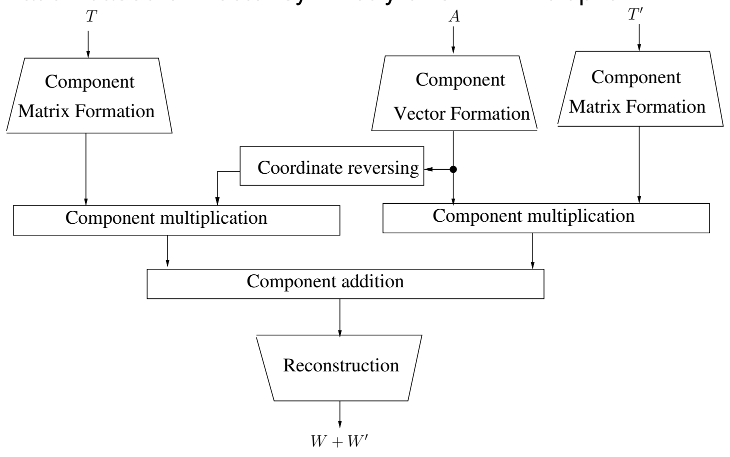
- Let $n = 2$ then
$$V = \begin{bmatrix} v_0 \\ v_1 \end{bmatrix} \text{ and } \rho(V) = \begin{bmatrix} v_1 \\ v_0 \end{bmatrix}$$
if we apply the CVF_2 formula (10) we get
$$\begin{aligned} \text{CVF}_2(V) &= [v_1, v_0 + v_1, v_0], \\ \text{CVF}_2(\rho(V)) &= [v_0, v_0 + v_1, v_1]. \end{aligned}$$
In other words, we have $\text{CVF}_2(\rho(V)) = \rho(\text{CVF}_2(V))$.

- Let $n = 2^s$ and we assume that equation (11) is true for $s - 1$. We split V and $\rho(V)$ in two parts of size 2^{s-1}
$$V = \begin{bmatrix} V_0 \\ V_1 \end{bmatrix} \text{ and } \rho(V) = \begin{bmatrix} \rho(V_1) \\ \rho(V_0) \end{bmatrix}.$$
Now, we apply the CVF_2 function defined in (10) to these decompositions of V and $\rho(V)$, we obtain
$$\begin{aligned} \text{CVF}_2(V) &= [\text{CVF}_2(V_1), \text{CVF}_2(V_0 + V_1), \text{CVF}_2(V_0)], \\ \text{CVF}_2(\rho(V)) &= [\text{CVF}_2(\rho(V_0)), \text{CVF}_2(\rho(V_0) + \rho(V_1)), \text{CVF}_2(\rho(V_1))]. \end{aligned}$$
Using Lemma 2, we get that $\rho(V_0) + \rho(V_1) = \rho(V_0 + V_1)$, and then using the induction hypothesis we have
$$\begin{aligned} \text{CVF}_2(\rho(V_0)) &= \rho(\text{CVF}_2(V_0)), \\ \text{CVF}_2(\rho(V_0) + \rho(V_1)) &= \text{CVF}_2(\rho(V_1 + V_0)) = \rho(\text{CVF}_2(V_1 + V_0)), \\ \text{CVF}_2(\rho(V_1)) &= \rho(\text{CVF}_2(V_1)). \end{aligned}$$
Finally, the statement $ii.$ of Lemma 2 provides that
$$\begin{aligned} \text{CVF}_2(\rho(V)) &= [\rho(\text{CVF}_2(V_0)), \rho(\text{CVF}_2(V_1 + V_0)), \rho(\text{CVF}_2(V_1))] \\ &= \rho([\text{CVF}_2(V_1), \text{CVF}_2(V_0 + V_1), \text{CVF}_2(V_0)]), \end{aligned}$$
which proves that $\text{CVF}_2(\rho(V)) = \rho(\text{CVF}_2(V))$.

□

The previous lemma implies that we do not need to compute both $\text{CVF}_2(A)$ and $\text{CVF}_2(\rho(A))$ in the architecture of Fig. 5. Indeed, we can compute $\text{CVF}_2(A)$ and then we get $\text{CVF}_2(\rho(A))$ by reversing the bits of $\text{CVF}_2(A)$. We recombine the architecture of Fig. 6 by removing the CVF block corresponding to $\text{CVF}_2(\rho(A))$, and adding a *Coordinate Reversing* block with input $\text{CVF}_2(A)$ and thus output $\text{CVF}_2(\rho(A))$ (Lemma 3).

Fig. 6. Block Recombination based on vector symmetry of ONB-II multiplier



5.3 Block recombination based on matrix symmetry

Here we show that the symmetric structure of the Toeplitz matrix T' involved in ONB-II multiplier can be used to reduce the space complexity of the multiplier. The main result we will need is given in the following subsection. It states that the CMF function has some commutative property with the transposition of a matrix.

5.3.1 Commutative property of matrix transposition and CMF

Let us first recall that the recursive definition of CMF$_2$ is the function which computes the two-way split component representation of an $n \times n$ Toeplitz matrix T

$$\mathrm{CMF}_2(T) = \begin{cases} T, \quad \text{if } T \text{ has size } 1, \\ [\mathrm{CMF}_2(T_1+T_0), \mathrm{CMF}_2(T_1), \mathrm{CMF}_2(T_2+T_1)], \quad \text{if } T = \begin{bmatrix} T_1 & T_0 \\ T_2 & T_1 \end{bmatrix}. \end{cases} \tag{12}$$

The main result is the following lemma.

Lemma 4. *We consider a Toeplitz matrix T and its corresponding transposed matrix T^t. Then we have*

$$CMF_2(T^t) = \rho(CMF_2(T)), \tag{13}$$

where $\rho()$ is the function which reverses the order of a vector.

Proof: We prove it by induction on $n = 2^s$.

- For $n = 2$ we have

$$T = \begin{bmatrix} t_1 & t_0 \\ t_2 & t_1 \end{bmatrix} \text{ and } T^t = \begin{bmatrix} t_1 & t_2 \\ t_0 & t_1 \end{bmatrix}$$

if we apply the CMF$_2$ recursive formula to T and T^t we get

$$\begin{array}{rcl} \mathrm{CMF}_2(T) & = & [t_1+t_0, t_1, t_2+t_1], \\ \mathrm{CMF}_2(T^t) & = & [t_1+t_2, t_1, t_0+t_1]. \end{array}$$

In other words we have $\mathrm{CMF}_2(T^t) = \rho(\mathrm{CMF}_2(T))$.

- Let us now consider $n = 2^s$ and we suppose that equation (13) holds for $n = 2^{s-1}$. We split T, i.e.,

$$T = \begin{bmatrix} T_1 & T_0 \\ T_2 & T_1 \end{bmatrix},$$

we then remark that T^t satisfies

$$T^t = \begin{bmatrix} T_1^t & T_2^t \\ T_0^t & T_1^t \end{bmatrix}.$$

Now we apply the recursive expression of CMF$_2$ given in (12) to T and T', we obtain

$$\begin{array}{rcl} \mathrm{CMF}_2(T) & = & \left[\ \mathrm{CMF}_2(T_1+T_0),\quad \mathrm{CMF}_2(T_1),\quad \mathrm{CMF}_2(T_2+T_1)\ \right], \\ \mathrm{CMF}_2(T^t) & = & \left[\ \mathrm{CMF}_2(T_1^t+T_2^t),\quad \mathrm{CMF}_2(T_1^t),\quad \mathrm{CMF}_2(T_0^t+T_1^t)\ \right]. \end{array}$$

Now using $T_1^t + T_0^t = (T_1+T_0)^t$ and $T_1^t + T_2^t = (T_1+T_2)^t$ and the induction hypothesis, we obtain the three following identities

$$\begin{array}{rcl} \mathrm{CMF}_2(T_1^t+T_0^t) & = & \rho(\mathrm{CMF}_2(T_1+T_0)), \\ \mathrm{CMF}_2(T_1^t) & = & \rho(\mathrm{CMF}_2(T_1)), \\ \mathrm{CMF}_2(T_1^t+T_2^t) & = & \rho(\mathrm{CMF}_2(T_1+T_2)). \end{array}$$

We now replace each of them in the previous expression of CMF$_2(T^t)$, and we use statement $ii.$ of Lemma 2 and we finally get

$$\begin{array}{rcl} \mathrm{CMF}_2(T^t) & = & \left[\ \rho(\mathrm{CMF}_2(T_1+T_2)),\quad \rho(\mathrm{CMF}_2(T_1)),\quad \rho(\mathrm{CMF}_2(T_1+T_0))\ \right] \\ & = & \rho(\left[\ \mathrm{CMF}_2(T_1+T_0),\quad \mathrm{CMF}_2(T_1),\quad \mathrm{CMF}_2(T_1+T_2)\ \right]) \\ & = & \rho(\mathrm{CMF}_2(T)). \end{array}$$

This ends the proof.

□

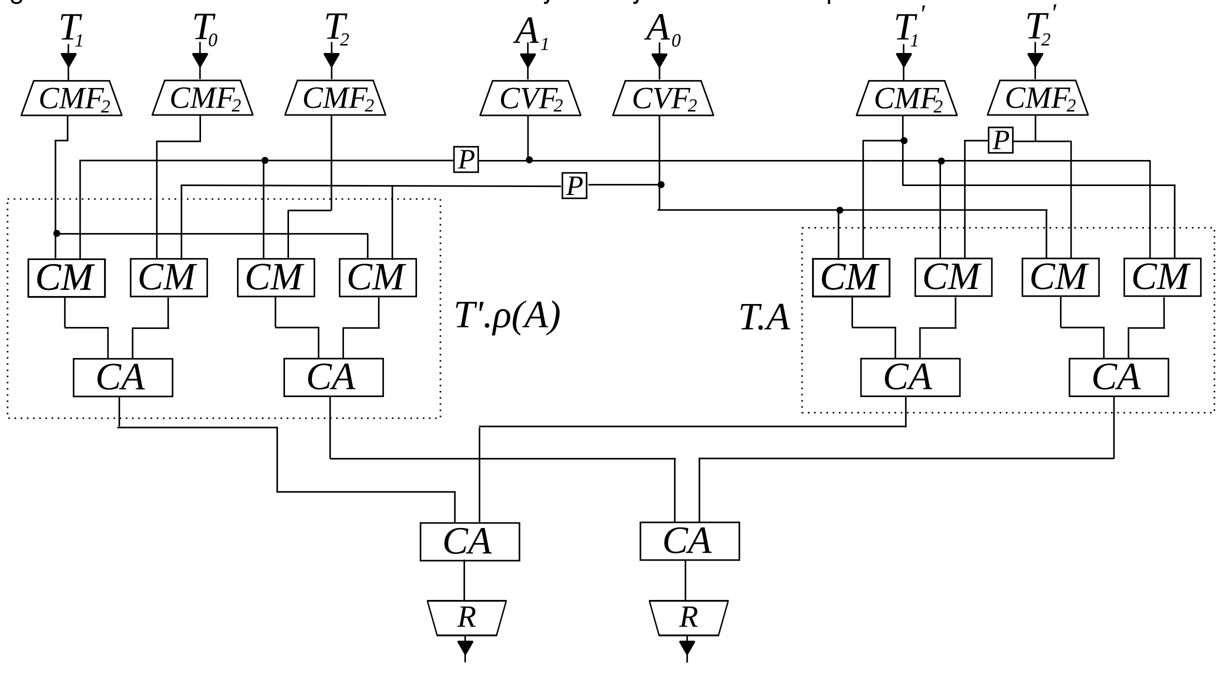
5.3.2 Recombination based on commutative property of matrix transposition

We would like to use the previous result to reduce some space complexity in the CMF computations in the ONB-II multiplier. We remark that the matrix on the right side of (3), which is the two-TMVPs-and-add expression of the multiplication in ONB-II, is symmetric. Consequently we rewrite (3) by decomposing the matrix in four sub-matrices and the vectors in two sub-vectors. We remark that the sub-matrix on the top right of T' is the transpose of the sub-matrix in the bottom left of T'. We then obtain

$$\begin{aligned} C &= \begin{bmatrix} T_1 & T_0 \\ T_2 & T_1 \end{bmatrix} \cdot \begin{bmatrix} \rho(A_1) \\ \rho(A_0) \end{bmatrix} + \begin{bmatrix} T_1' & (T_2')^t \\ T_2' & T_1' \end{bmatrix} \cdot \begin{bmatrix} A_0 \\ A_1 \end{bmatrix} \\ &= \begin{bmatrix} T_1 \cdot \rho(A_1) + T_0 \cdot \rho(A_0) \\ T_2 \cdot \rho(A_1) + T_1 \cdot \rho(A_0) \end{bmatrix} + \begin{bmatrix} T_1' \cdot A_0 + (T_2')^t \cdot A_1 \\ T_2' \cdot A_0 + T_1' \cdot A_1 \end{bmatrix}. \end{aligned}$$

We finally design an architecture which performs the CVF$_2$ of A_0 and A_1 and CMF$_2$ of T_0, T_1, T_2, T_1' and T_2'. Then using the results of the previous subsection we get the CVF$_2$ of $\rho(A_0)$ and $\rho(A_1)$ by reversing CVF$_2(A_0)$ and CVF$_2(A_1)$ respectively. In Fig. 7 the reverse ordering is schematized through a box P. Using Lemma 4 we obtain the CMF of $(T_2')^t$ by reversing CMF$_2(T_2')$. Then we can perform the different multiplications in component representation, and then perform component addition and reconstruction. The resulting architecture is depicted in Fig. 7.

Fig. 7. Block Recombination based on matrix symmetry of ONB-II multiplier



5.4 Second recombination using a second matrix symmetry

We would like to perform the recombination based on matrix symmetry on the matrix T of (3). The first matrix-vector product is as follows

$$T \cdot A = \begin{bmatrix} b_n & b_n & b_{n-1} & \dots & b_4 & b_3 & b_2 \\ b_{n-1} & b_n & b_n & \dots & b_5 & b_4 & b_3 \\ b_{n-2} & b_{n-1} & b_n & \dots & b_6 & b_5 & b_4 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ b_3 & b_4 & b_5 & \dots & b_n & b_n & b_{n-1} \\ b_2 & b_3 & b_4 & \dots & b_{n-1} & b_n & b_n \\ b_1 & b_2 & b_3 & \dots & b_{n-2} & b_{n-1} & b_n \end{bmatrix} \cdot \begin{bmatrix} a_n \\ a_{n-1} \\ a_{n-2} \\ \vdots \\ a_3 \\ a_2 \\ a_1 \end{bmatrix}$$

The matrix is not symmetric, consequently we cannot apply directly the method presented in the previous section. We use a strategy similar to the one used for ONB of type-I. Let us split T in four parts

$$T \cdot A = \begin{bmatrix} T_1 & T_0 \\ T_2 & T_1 \end{bmatrix} \cdot \begin{bmatrix} \rho(A_1) \\ \rho(A_0) \end{bmatrix}$$

We consider the matrix vector product $T_0 \cdot \rho(A_0)$ and we rewrite it as

$$T_0 \cdot \rho(A_0) = \begin{bmatrix} b_{n/2+1} & b_{n/2} & b_{n/2-1} & \dots & b_4 & b_3 & b_2 \\ b_{n/2+2} & b_{n/2+1} & b_{n/2} & \dots & b_5 & b_4 & b_3 \\ b_{n/2+3} & b_{n/2+2} & b_{n/2+1} & \dots & b_6 & b_5 & b_4 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ b_{n-2} & b_{n-3} & b_{n-4} & \dots & b_{n/2+1} & b_{n/2} & b_{n/2-1} \\ b_{n-1} & b_{n-2} & b_{n-3} & \dots & b_{n/2+2} & b_{n/2+1} & b_{n/2} \\ b_n & b_{n-1} & b_{n-2} & \dots & b_{n/2+3} & b_{n/2+2} & b_{n/2+1} \end{bmatrix} \cdot \begin{bmatrix} a_{n/2} \\ a_{n/2-1} \\ \vdots \\ a_3 \\ a_2 \\ a_1 \end{bmatrix}$$

$$= a_{n/2} \begin{bmatrix} b_{n/2+1} \\ b_{n/2-1} \\ b_{n/2-2} \\ \vdots \\ b_{n-2} \\ b_{n-1} \\ b_n \end{bmatrix} + \begin{bmatrix} b_{n/2} & b_{n/2-1} & \dots & b_3 & b_2 & b_1 \\ b_{n/2+1} & b_{n/2} & \dots & b_4 & b_3 & b_2 \\ b_{n/2+2} & b_{n/2+1} & \dots & b_5 & b_4 & b_3 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ b_{n-3} & b_{n-4} & \dots & b_{n/2} & b_{n/2-1} & b_{n/2-2} \\ b_{n-2} & b_{n-3} & \dots & b_{n/2+1} & b_{n/2} & b_{n/2-1} \\ b_{n-1} & b_{n-2} & \dots & b_{n/2+2} & b_{n/2+1} & b_{n/2} \end{bmatrix} \cdot \begin{bmatrix} a_{n/2-1} \\ a_{n/2-2} \\ \vdots \\ a_3 \\ a_2 \\ a_1 \\ 0 \end{bmatrix}.$$

So now we see that the latter $n/2 \times n/2$ matrix is equal to the transposed matrix of T_2

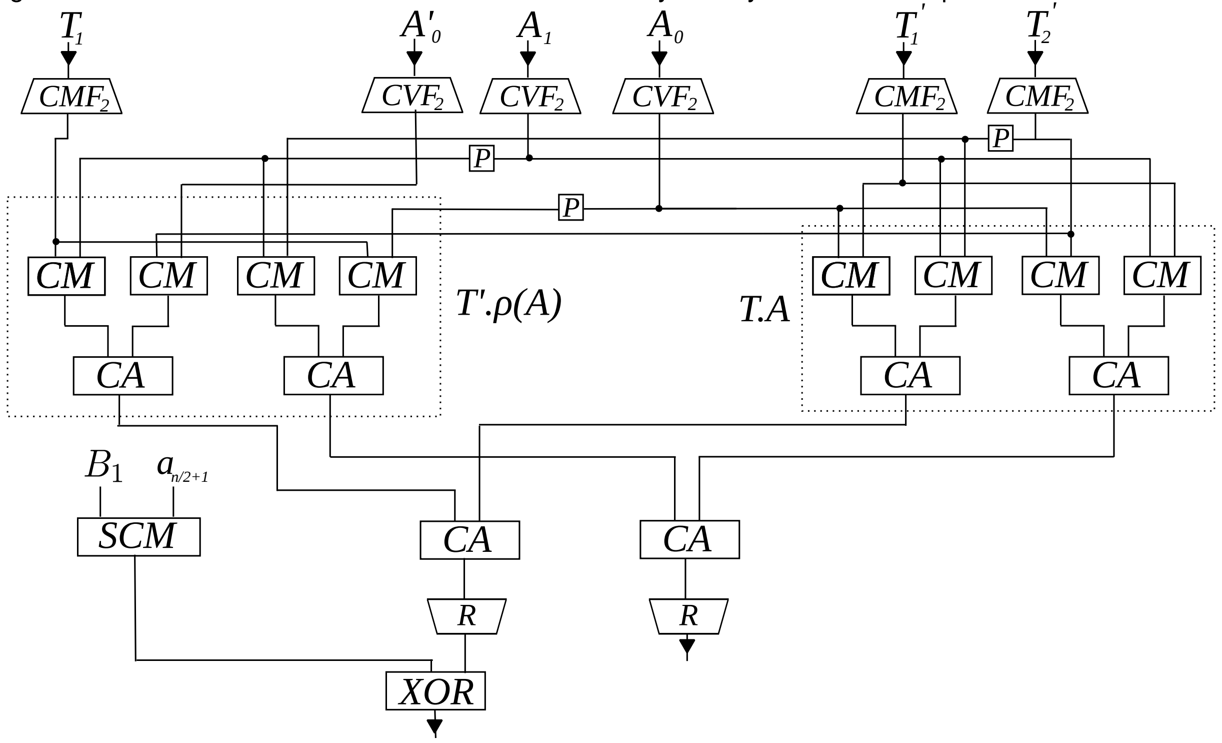
$$T_2 = \begin{bmatrix} b_{n/2} & b_{n/2+1} & b_{n/2+2} & \dots & b_{n-2} & b_{n-1} \\ b_{n/2-1} & b_{n/2} & b_{n/2+1} & \dots & b_{n-3} & b_{n-2} \\ b_{n/2-2} & b_{n/2-1} & b_{n/2} & \dots & b_{n-4} & b_{n-3} \\ \vdots & \vdots & \vdots & \dots & \vdots & \vdots \\ b_3 & b_4 & b_5 & \dots & b_{n/2+3} & b_{n/2+2} \\ b_2 & b_3 & b_4 & \dots & b_{n/2} & b_{n/2+1} \\ b_1 & b_2 & b_3 & \dots & b_{n/2-1} & b_{n/2} \end{bmatrix}.$$

We remark also that T_2 is equal to the transposed matrix of T_2'. So finally if we put these new expressions for T_2 and T_0 in (9) we get

$$\begin{aligned} C &= \begin{bmatrix} T_1 & T_0 \\ T_2 & T_1 \end{bmatrix} \cdot \begin{bmatrix} \rho(A_1) \\ \rho(A_0) \end{bmatrix} + \begin{bmatrix} T_1' & (T_2')^t \\ T_2' & T_1' \end{bmatrix} \cdot \begin{bmatrix} A_0 \\ A_1 \end{bmatrix} \\ &= \begin{bmatrix} T_1 \cdot \rho(A_1) + a_{n/2} B_1 + T_2' \cdot A_0' \\ T'^t_2 \cdot \rho(A_1) + T_1 \cdot \rho(A_0) \end{bmatrix} + \begin{bmatrix} T_1' \cdot A_0 + (T_2')^t \cdot A_1 \\ T_2' \cdot A_0 + T_1' \cdot A_1 \end{bmatrix}. \end{aligned} \tag{14}$$

Now, we can perform the following modifications on the architecture of Fig. 7: first we remove the CMF$_2$ block corresponding to T_0 and T_2, we put a permutation box which reverses the order of CMF$_2(T_2')$ and this provides CMF$_2(T_2^t)$. We also need to compute CVF$_2(V_1')$ and then modify the connections. The resulting architecture is shown in Fig. 8.

Fig. 8. Block Recombination based on a second matrix symmetry of ONB-II multiplier



Complexity evaluation. We assume here that n is a power of 2. In order to evaluate the complexity of this architecture we evaluate the contribution of each block in terms of the number of XOR and AND gates

$$\begin{aligned} \mathcal{S}_{2,\oplus} &= 3S_{2,\oplus}^{CMF}(n/2) + 3S_{2,\oplus}^{CVF}(n/2) + 6S_{2,\oplus}^{CA}(n/2) + 2S_{2,\oplus}^{R}(n/2) + n/2, \\ \mathcal{S}_{2,\otimes} &= 6S_{2,\otimes}^{CM}(n/2) + \tfrac{n}{2}. \end{aligned}$$

Using the block complexities of Table 3 we finally obtain that

$$\begin{array}{rcl} \mathcal{S}_{2,\oplus} & = & \frac{41}{6}n^{\log_2(3)} - \frac{15}{2}n + \frac{3}{2}, \\ \mathcal{S}_{2,\otimes} & = & 2n^{\log_2(3)} + \frac{n}{2}. \end{array}$$

Remark 4. The architecture given in Fig. 8 can be designed when n can is divisible by 2. Indeed, under this assumption we can express (9) as (14). Then, when $n/2$ is not divisible by 2, we can enlarge each $n/2 \times n/2$ Toeplitz matrix and vector to a size of the form $2^k 3^l$. Then we can apply the two-way and three-way formulas of Fan-Hasan to perform CMF$_2$, CVF$_2$, CM, CA and reconstruction.

6 Recombination of three-way split ONB-II multiplier

In this section we present a recombination process for the three-way split approach for multiplication in ONB of type II. This process follows the same different steps used in the two-way split case. We go back to the expression (3) of the multiplication in ONB of type II of two elements through a two-TMVPs-and-add

$$C = T \cdot \rho(A) + T' \cdot A.$$

We can apply directly the recombination of the two-TMVPs-and-add which has been depicted in Fig.2. Then we apply symmetry properties for the vector and then for the matrices. Before that we need to state the following commutative properties of CVF$_3$ (resp. CMF$_3$) and the reverse bit ordering.

6.1 Commutative property for vector and matrix

We first need to define a permutation σ_n of component representation. It is a permutation in the sense that it reorders the coordinates in a specific order.

Definition 1. Let $W = [W_0, W_1, W_2, W_3, W_4, W_5]$ be a bit array of size $n^{\log_3(6)}$ where W_i has size $\frac{n^{\log_3(6)}}{6}$. We define $\sigma_n(W)$ recursively as follows

$$\sigma_n(W) = \begin{cases} W, \text{ if } W \text{ has size } 1, \\ [\sigma_{n/3}(W_2), \sigma_{n/3}(W_1), \sigma_{n/3}(W_0), \sigma_{n/3}(W_5), \sigma_{n/3}(W_4), \sigma_{n/3}(W_3)], \text{ otherwise.} \end{cases}$$

We recall that function CVF$_3$ is defined recursively as follows

$$\mathrm{CVF}_3(V) = \begin{cases} V, \text{ if } V \text{ has size } 1, \\ [\mathrm{CVF}_3(V_2), \mathrm{CVF}_3(V_1), \mathrm{CVF}_3(V_0), \mathrm{CVF}_3(V_1+V_2), \\ \qquad \mathrm{CVF}_3(V_0+V_2), \mathrm{CVF}_3(V_0+V_1)], \text{ otherwise.} \end{cases}$$

We then prove the following commutative property on the reverse bit ordering and the CVF$_3$ function.

Lemma 5. *Let V be a vector of size n, and let $\rho(V)$ the vector which has the coordinates of V in the reverse order. Let $CVF_3(V)$ be the three-way component vector formation of V. Then we have*

$$CVF_3(\rho(V)) = \sigma_n \left(CVF_3(V)\right) \tag{15}$$

where σ_n has been previously defined in Definition 1.

Proof: We prove it by induction on the size of V. It is clear that the lemma is true when V has size 1.

- Let $n = 3$ and let

$$V = \begin{bmatrix} v_0 \\ v_1 \\ v_2 \end{bmatrix} \text{ and } \rho(V) = \begin{bmatrix} v_2 \\ v_1 \\ v_0 \end{bmatrix}.$$

If we apply the CVF$_3$ formula to V and $\rho(V)$ we obtain that

$$\begin{aligned} \mathrm{CVF}_3(V) &= [v_2, v_1, v_0, v_1 + v_2, v_2 + v_0, , v_1 + v_0], \\ \mathrm{CVF}_3(\rho(V)) &= [v_0, v_1, v_2, v_1 + v_0, v_0 + v_2, v_1 + v_2]. \end{aligned}$$

Then we can check that $\mathrm{CVF}_3(\rho(V)) = \sigma_3(\mathrm{CVF}_3(V))$.

- Let $n = 3^s$ and assume that (11) is true for $n = 3^i$ when $i < s$. We split V and $\rho(V)$ in three parts of size 3^{s-1}

$$V = \begin{bmatrix} V_0 \\ V_1 \\ V_2 \end{bmatrix}, \quad \rho(V) = \begin{bmatrix} \rho(V_2) \\ \rho(V_1) \\ \rho(V_0) \end{bmatrix}$$

where $\rho(V_2)$, $\rho(V_1)$ and $\rho(V_0)$ are the reverse vectors of V_2, V_1 and V_0, respectively. Now we have

$$\begin{aligned} \mathrm{CVF}_3(V) &= [\mathrm{CVF}_3(V_2), \mathrm{CVF}_3(V_1), \mathrm{CVF}_3(V_0), \\ &\qquad \mathrm{CVF}_3(V_1 + V_2), \mathrm{CVF}_3(V_0 + V_2), \mathrm{CVF}_3(V_0 + V_1)], \\ \mathrm{CVF}_3(\rho(V)) &= [\mathrm{CVF}_3(\rho(V_0)), \mathrm{CVF}_3(\rho(V_1)), \mathrm{CVF}_3(\rho(V_2)), \\ &\qquad \mathrm{CVF}_3(\rho(V_1 + V_0)), \mathrm{CVF}_3(\rho(V_0 + V_2)), \mathrm{CVF}_3(\rho(V_1 + V_2))]. \end{aligned}$$

For each i the vector $\rho(V_i))$ has size $\frac{n^{\log_3(6)}}{6}$ and for each i and j the vector $\rho(V_i + V_j)$ has also size $\frac{n^{\log_3(6)}}{6}$. Then we can apply the induction hypothesis to their corresponding CVF$_3$ computation. We obtain

$$\begin{aligned} \mathrm{CVF}_3(\rho(V)) &= [\sigma_{n/3}(\mathrm{CVF}_3(V_0)), \sigma_{n/3}(\mathrm{CVF}_3(V_1)), \sigma_{n/3}(\mathrm{CVF}_3(V_2)), \\ &\qquad \sigma_{n/3}(\mathrm{CVF}_3(V_1 + V_0)), \sigma_{n/3}(\mathrm{CVF}_3(V_0 + V_2)), \sigma_{n/3}(\mathrm{CVF}_3(V_1 + V_2))]. \end{aligned}$$

But, using now the recursive definition of σ_n we obtain

$$\begin{aligned} \mathrm{CVF}_3(\rho(V)) &= \sigma_n\left([\mathrm{CVF}_3(V_2), \mathrm{CVF}_3(V_1), \mathrm{CVF}_3(V_0), \mathrm{CVF}_3(V_1 + V_2), \mathrm{CVF}_3(V_0 + V_2), \mathrm{CVF}_3(V_1 + V_0)]\right) \\ &= \sigma_n(\mathrm{CVF}_3(V)). \end{aligned}$$

This ends the proof.

□

Now we would like to have a similar property concerning the CMF$_3$ of a transposed matrix. In order to do that we will need the following recursive definition of the function CMF$_3$

$$\mathrm{CMF}_3(T) = \begin{cases} T \text{ if } T \text{ has size } 1, \\ [\mathrm{CMF}_3(T_0+T_1+T_2),\ \mathrm{CMF}_3(T_1+T_2+T_3),\ \mathrm{CMF}_3(T_2+T_3+T_4), \\ \qquad \mathrm{CMF}_3(T_1),\ \mathrm{CMF}_3(T_2),\ \mathrm{CMF}_3(T_3)] \text{ otherwise.} \end{cases}$$

We then prove the following commutative property on the reverse bit ordering and the CMF$_3$ function.

Lemma 6. *Let T be an $n \times n$ matrix where n is a power of 3, and let T^t be the transposed matrix of T. Let CMF$_3(T)$ be the three-way component matrix formation of T. We have that*

$$CMF_3(T^t) = \sigma_n\left(CMF_3(T)\right) \tag{16}$$

where σ_n has been previously defined in Definition 1.

Proof: We prove it by induction on the size of V. It is clear that the Lemma is true when T has size 1. Let us prove it for $n > 1$.

- Let $n = 3$ and let

$$T = \begin{bmatrix} t_2 & t_1 & t_0 \\ t_3 & t_2 & t_1 \\ t_4 & t_3 & t_2 \end{bmatrix} \text{ and } T^t = \begin{bmatrix} t_2 & t_3 & t_4 \\ t_1 & t_2 & t_3 \\ t_0 & t_1 & t_2 \end{bmatrix},$$

  if we apply the CMF$_3$ formula we get

$$\begin{aligned} \mathrm{CMF}(T) &= [t_0+t_1+t_2,\ t_1+t_2+t_3,\ t_2+t_3+t_4,\ t_1,\ t_2,\ t_3], \\ \mathrm{CMF}(T^t) &= [t_4+t_3+t_2,\ t_1+t_2+t_3,\ t_2+t_1+t_0,\ t_3,\ t_2,\ t_1]. \end{aligned}$$

  Then we can easily check that $\mathrm{CMF}(T^t) = \sigma_3(\mathrm{CMF}(T))$.

- Let $n = 3^s$ and assume that (16) is true for $n = 3^i$ when $i < s$. We split T and T^t in 9 sub-matrices of size 3^{s-1}, we get

$$T = \begin{bmatrix} T_2 & T_1 & T_0 \\ T_3 & T_2 & T_1 \\ T_4 & T_3 & T_2 \end{bmatrix} \text{ and } T^t = \begin{bmatrix} T_2^t & T_3^t & T_4^t \\ T_1^t & T_2^t & T_3^t \\ T_0^t & T_1^t & T_2^t \end{bmatrix}$$

  where T_i^t are the transpose of T_i. Now using the recursive definition of CMF$_3$ we get

$$\begin{aligned} \mathrm{CMF}_3(T) &= [\mathrm{CMF}_3(T_0+T_1+T_2),\ \mathrm{CMF}_3(T_1+T_2+T_3),\ \mathrm{CMF}_3(T_2+T_3+T_4), \\ &\qquad \mathrm{CMF}_3(T_1),\ \mathrm{CMF}_3(T_2),\ \mathrm{CMF}_3(T_3)], \\ \mathrm{CMF}_3(T^t) &= [\mathrm{CMF}_3(T_2^t+T_3^t+T_4^t),\ \mathrm{CMF}_3(T_1^t+T_2^t+T_3^t),\ \mathrm{CMF}_3(T_0^t+T_1^t+T_2^t), \\ &\qquad \mathrm{CMF}_3(T_3^t),\ \mathrm{CMF}_3(T_2^t),\ \mathrm{CMF}_3(T_1^t)]. \end{aligned}$$

We now apply the induction hypothesis on the previous expression of $\mathrm{CMF}_3(T^t)$ we get

$$\mathrm{CMF}_3(T^t) = [\sigma_{n/3}(\mathrm{CMF}_3(T_2+T_3+T_4)),\ \sigma_{n/3}(\mathrm{CMF}_3(T_1+T_2+T_3)),\ \sigma_{n/3}(\mathrm{CMF}_3(T_0+T_1+T_2)),\ \sigma_{n/3}(\mathrm{CMF}_3(T_3)),\ \sigma_{n/3}(\mathrm{CMF}_3(T_2)),\ \sigma_{n/3}(\mathrm{CMF}_3(T_1))].$$

Finally, using we use the recursive definition of σ_n and we obtain

$$\begin{aligned}\mathrm{CMF}_3(T^t) &= \sigma_n[\mathrm{CMF}_3(T_0+T_1+T_2),\ \mathrm{CMF}_3(T_1+T_2+T_3),\ \mathrm{CMF}_3(T_2+T_3+T_4),\\ &\qquad \mathrm{CMF}_3(T_1),\ \mathrm{CMF}_3(T_2),\ \mathrm{CMF}_3(T_3)]\\ &= \sigma_n(\mathrm{CMF}_3(T)).\end{aligned}$$

This ends the proof.

□

6.2 Recombination of ONB-II multiplier based on commutative properties

The commutative properties given in the previous subsection can be applied to recombine the two-TMVPs-and-add architecture (Fig. 2) for the special case of (3). The recombination works as follows

- *Recombination using vector symmetry.* Using Lemma 5 we can replace the CVF_3 computation of $\rho(A)$ by a permutation box P applied to $\mathrm{CVF}_3(A)$. The box P performs the permutation σ_n, it just changes the order of the bits and thus do not require any gates.
- *Recombination using matrix symmetry.* In order to apply Lemma 6 we first need to decompose each TMVP in three-way. We get

$$C = \begin{bmatrix} T_2 & T_1 & T_0 \\ T_3 & T_2 & T_1 \\ T_4 & T_3 & T_2 \end{bmatrix} \cdot \begin{bmatrix} \rho(A_2) \\ \rho(A_1) \\ \rho(A_0) \end{bmatrix} + \begin{bmatrix} T'_2 & T'_1 & T'_0 \\ T'_3 & T'_2 & T'_1 \\ T'_4 & T'_3 & T'_2 \end{bmatrix} \cdot \begin{bmatrix} A_0 \\ A_1 \\ A_2 \end{bmatrix}$$

  Now, since the matrix T' is symmetric, we have $T'_1 = {T'_3}^t$ and $T'_0 = {T'_4}^t$, and the previous equation becomes

$$C = \begin{bmatrix} T_2 & T_1 & T_0 \\ T_3 & T_2 & T_1 \\ T_4 & T_3 & T_2 \end{bmatrix} \cdot \begin{bmatrix} \rho(A_2) \\ \rho(A_1) \\ \rho(A_0) \end{bmatrix} + \begin{bmatrix} T'_2 & {T'_3}^t & {T'_4}^t \\ T'_3 & T'_2 & {T'_3}^t \\ T'_4 & T'_3 & T'_2 \end{bmatrix} \cdot \begin{bmatrix} A_0 \\ A_1 \\ A_2 \end{bmatrix} \qquad (17)$$

  Using Lemma 6 we know that the component matrix formation of ${T'_3}^t$ can be computed as the σ_n permutation of $\mathrm{CMF}_3(T'_3)$. Similarly ${T'_4}^t$ is equal to the σ_n permutation of $\mathrm{CMF}_3(T'_4)$. Finally if we design a multiplier based on (17) and using the previous recombination previously mentioned, then we replace the two CMF blocks corresponding to T'_1 and T'_0 by two permutation blocks.
- The final recombination is based on the following lemma.

Lemma 7. *Let T be the matrix on the left side of* (17) *and T' the matrix on the right side. Then the Toeplitz matrix vector product $T\cdot\rho(A)$ can be done as*

$$T\cdot\rho(A)=\begin{bmatrix} T_2\cdot\rho(A_2)+a_{2n/3}B_1+T_3'\cdot A_1'+T_4'\cdot A_0' \\ {T_3'}^t\cdot\rho(A_2)+T_2\cdot\rho(A_1)+a_{n/3}B_1+T_3^t\cdot A_0' \\ {T_4'}^t\cdot\rho(A_2)+{T_3'}^t\cdot\rho(A_1)+T_2\cdot\rho(A_0)\end{bmatrix}$$

where the vectors B_1 is the first column of T_1 and A_1' and A_0' are as follows

$$A_1'=\begin{bmatrix} a_{2n/3-1}\\ a_{2n/3-2}\\ \vdots \\ a_{n/3+2}\\ a_{n/3+1}\\ a_{n/3}\end{bmatrix} \text{ and } A_0'=\begin{bmatrix} a_{n/3-1}\\ a_{n/3-2}\\ \vdots \\ a_2\\ a_1\\ 0\end{bmatrix}$$

Finally we can rewrite the previous expression (17) of C as

$$C=\begin{bmatrix} T_2\cdot\rho(A_2)+a_{2n/3}B_1+T_3'\cdot A_1'+T_4'\cdot A_0' \\ {T_3'}^t\cdot\rho(A_2)+T_2\cdot\rho(A_1)+a_{n/3}B_1+T_3^t\cdot A_0' \\ {T_4'}^t\cdot\rho(A_2)+{T_3'}^t\cdot\rho(A_1)+T_2\cdot\rho(A_0)\end{bmatrix}+\begin{bmatrix} T_2'\cdot A_0+{T_3'}^t\cdot A_1+{T_4'}^t\cdot A_2 \\ T_3'\cdot A_0+T_2'\cdot A_1+{T_3'}^t\cdot A_2 \\ T_4'\cdot A_0+T_3'\cdot A_1+T_2'\cdot A_2\end{bmatrix} \quad (18)$$

We can thus perform some further simplification in the two-TMVPs-and-add architecture. Indeed, we can remove the CMF$_3$ computation of T_4, T_3, T_1 and T_0. They are replaced by σ_n permutation of CMF(T_3') and CMF(T_4'). On other hand we need to compute additional CVF$_3$ corresponding to the two new vectors A_1' and A_0' of length $n/3$.

Using these different recombinations, we can obtain the final architecture. A block diagram of the architecture involves a lot of details and is not necessary for the reader to understand how this architecture works: hence we do not present it in this paper.

Complexity evaluation. We evaluate the complexity of the architecture under the assumption that n is a power of 3. The number of CMF$_3$ and CVF$_3$ blocks is equal to the number of distinct matrices and vectors in (18). The number of component multiplication is equal to the number of TMVPs of size $n/3$. The number of component addition and reconstruction can then be evaluated.

We obtain the following contribution of each block in the number of AND gates and XOR gates of the architecture

$$\begin{aligned}\mathcal{S}_{3,\oplus} &= 4\mathcal{S}_{3,\oplus}^{CMF}(n/3)+5\mathcal{S}_{3,\oplus}^{CVF}(n/3)+15\mathcal{S}_{3,\oplus}^{CA}(n/2)+3\mathcal{S}_{3,\oplus}^{R}(n/3)+\tfrac{2n}{3},\\ \mathcal{S}_{3,\otimes} &= 18\mathcal{S}_{3,\otimes}^{CM}(n/3)+\tfrac{2n}{3}.\end{aligned}$$

Using Table 3, we have

$$\begin{array}{rcl}\mathcal{S}_{3,\oplus} & = & \frac{83}{15}n^{\log_3(6)} - \frac{55n}{9} + \frac{4}{5},\\ \mathcal{S}_{3,\otimes} & = & 3n^{\log_3(6)} + \frac{2n}{3}.\end{array}$$

When n is not a power of 3 but at least divisible by 3 we would use hybrid formulas (combining two-way and three way formulas of Fan and Hasan [2]) to design the different blocks (CMF, CVF,...) of the architecture.

7 Complexity comparison and conclusion

In this section, we compare the proposed multiplication schemes with a number of similar schemes recently published in the literature. To this end, we start our comparison in terms of the number of two-input AND and XOR gates that we would need for hardware implementation in fully bit parallel format. We should add that for a given scheme, our AND and XOR gate counts correspond to the number of modulo two additions and multiplications, respectively, and hence the sum of the two counts is the scheme's arithmetic complexity. For the purpose of comparison at architectural level, sometimes the sum is considered to be a measure of the space requirements of a bit parallel multiplier. In hardware technologies, where XOR and AND gate sizes are not the same, a more accurate measure would be a weighted sum of the gate counts. In our comparison, in addition to gate counts we consider computation time in terms of delays through AND and XOR gates, i.e., D_A and D_X, respectively.

In Table 4, we compare subquadratic space complexity field multiplication schemes that use type I optimal normal bases. In particular, we consider the two-way and the three-way split based schemes presented in this article and those in [4]. For the two-way split, this paper and [4] have the same gate delay with almost equal number of gates– AND and XOR combined. For the three-way split, the gate delays are the same, but the total gate count of the scheme of this paper is slightly lower ($\approx ((1+4.8)-(1.5+3.73))(1+4.8) = 9.8\%$ for large n). In design environments, where an XOR gate is larger than an AND gate, the gate counts of the three-way split scheme of this paper are likely to translate into an even better space reduction. For example, if an XOR gate is twice the size of an AND gate, then the space requirements for the scheme presented in this article would be reduced by about $((1+2\times4.8)-(1.5+2\times3.73))/(1+2\times4.8) = 15.5\%$.

Our comparison of type II ONB field multiplication schemes is given in Table 5. Here, we report gate counts and delays of the two-way and the three-way split schemes presented in [3], [1] and in this article. For the scheme in [1], we assume that the Karatsuba algorithm is applied in the way as reported in [4], which has the least gate delay, i.e., multiplication of two polynomials of degree n has a delay of $D_A + 2(\log_2 n + 1)D_X$ (respectively, $D_A + 3(\log_3 n + 1)D_X$) for the two-way (respectively, three-way) split.

As it be seen in Table 5, for the two-way split and sufficiently large n, the total gate count of [3] is about 46.1% (resp. 33.6%) more than that of [1] (resp. this article). On the other hand, the gate delay of

[3] is about 50% less than that of [1] and is the same as that of this article. In other words, the scheme of this paper requires about 20% more gates, but is about twice faster than that of [1]. As a result, among the three two-way split based schemes considered in Table 5, the one presented in this article has the least gate count and delay product (i.e., area-time product), and hence it would perform the most number of field multiplications per unit area-time. A similar attribute also applies to the three-way split based scheme presented in this article.

TABLE 4

Space and time complexities of ONB I multipliers

Method	Split	# AND	# XOR	Delay
[3]	2	$n^{\log_2 3} + n$	$5.5n^{\log_2 3} - 4n + 0.5$	$D_A + (2\log_2(n) + 1)D_X$
This paper	2	$1.33n^{\log_2(3)} + 1.5n$	$4.67n^{\log_2(3)} - 4n + 1$	$D_A + (2\log_2(n) + 2)D_X$
[3]	3	$n^{\log_3 6} + n$	$4.8n^{\log_3 6} - 3n + 0.2$	$D_A + (3\log_3(n) + 1)D_X$
This paper	3	$1.5n^{\log_2(3)} + 1.66n$	$3.73n^{\log_3(6)} - 4.77n + 0.6$	$D_A + (3\log_3(n) + 2)D_X$

TABLE 5

Space and time complexities of ONB II multipliers

Method	Split	# AND	# XOR	Delay
[3]	2	$2n^{\log_2 3}$	$11n^{\log_2 3} - 12n + 1$	$D_A + (2\log_2(n) + 1)D_X$
[1]	2	$n^{\log_2(3)}$	$6n^{\log_2(3)} - 8n + 2 + (2n + 2)\log_2(n/2)$	$D_A + (4\log_2(n) + 2)D_X$
This paper	2	$2n^{\log_2(3)} + 0.5n$	$6.83n^{\log_2(3)} - 7.5n + 1.5$	$D_A + (2\log_2(n) + 1)D_X$
[3]	3	$2n^{\log_3 6}$	$9.6n^{\log_3 6} - 10n + 0.2$	$D_A + (3\log_3(n) + 1)D_X$
[1]	3	$n^{\log_3(6)}$	$4.8n^{\log_3(6)} - 7.3n + 2 + (2n + 2)\log_2(n/2)$	$D_A + (3\log_3(n) + 2\log_2(n) + 2)D_X$
This paper	3	$3n^{\log_3(6)} + 0.66n$	$5.53n^{\log_3(6)} - 6.11n + 0.8$	$D_A + (3\log_3(n) + 1)D_X$

REFERENCES

[1] D. J. Bernstein and T. Lange. Type-II Optimal Polynomial Bases. In *WAIFI 2010*, volume 6087 of *LNCS*, pages 41–61, 2010.

[2] H. Fan and M. A. Hasan. A New Approach to Sub-quadratic Space Complexity Parallel Multipliers for Extended Binary Fields. *IEEE Trans. Computers*, 56(2):224–233, 2007.

[3] H. Fan and M. A. Hasan. Subquadratic Computational Complexity Schemes for Extended Binary Field Multiplication Using Optimal Normal Bases. *IEEE Trans. Computers*, 56:1435–1437, 2007. This was previously published as a CACR 2006-26 technical report.

[4] H. Fan, J. Sun, M. Gu, and K.-Y. Lam. Overlap-free Karatsuba-Ofman Polynomial Multiplication Algorithm. *IET Information Security*, 4, March 2010.

[5] M. A. Hasan, N. Méloni, A. H. Namin, and C. Negre. Block Recombination Approach for Subquadratic Space Complexity Binary Field Multiplication based on Toeplitz Matrix-Vector Product. Technical Report CACT 2010-12, CACR, May 2010.

[6] M. A. Hasan, M. Wang, and V. K. Bhargava. A Modified Massey-Omura Parallel Multiplier for a Class of Finite Fields. *IEEE Trans. Computers*, 42(10):1278–1280, 1993.

[7] C. K. Koc and B. Sunar. Low-Complexity Bit-Parallel Canonical and Normal Basis Multipliers for a Class of Finite Fields. *IEEE Trans. Computers*, 47(3):353–356, 1998.

[8] J. L. Massey and J. K. Omura. Computational method and apparatus for finite field arithmetic. US patent number 4,587,627.

[9] R. C. Mullin, S. A.Vanstone, and R. M. Wilson. Optimal normal bases in GF(p^n). *Discrete Applied Mathematics*, 22(2):149–161, 1989.

[10] J. von zur Gathen, A. Shokrollahi, and J. Shokrollahi. Efficient multiplication using type 2 optimal normal bases. In *WAIFI '07*, LNCS, pages 55–68, 2007.