Fast Bit-Parallel Binary Multipliers Based on Type-I Pentanomials

José L. Imaña

Abstract—In this paper, a fast implementation of bit-parallel polynomial basis (PB) multipliers over the binary extension field $GF(2^m)$ generated by type-I irreducible pentanomials is presented. Explicit expressions for the coordinates of the multipliers and a detailed example are given. Complexity analysis shows that the multipliers here presented have the lowest delay in comparison to similar bit-parallel PB multipliers found in the literature based on this class of irreducible pentanomials. In order to prove the theoretical complexities, hardware implementations over Xilinx FPGAs have also been performed. Experimental results show that the approach here presented exhibits the lowest delay with a balanced $Area \times Time$ complexity when it is compared with similar multipliers.

Index Terms—Multipliers, bit-parallel, $GF(2^m)$, polynomial basis, pentanomials

1 INTRODUCTION

EFFICIENT VLSI implementations of high-speed multipliers over binary extension fields $GF(2^m)$ are highly desirable for several applications, such as cryptography, digital signal processing or coding theory [1]. Elements in $GF(2^m)$ are mainly represented in polynomial basis (PB) because it provides more freedom on hardware optimizations for arithmetic operations. The efficiency of their hardware implementations is measured in terms of the number of 2-input gates (AND, XOR) and of the gate delays (T_A, T_X) of the circuit. Many approaches and architectures have been proposed to perform PB multipliers [2], [3], [4], [5], [6]. The complexity of the multiplier mainly depends on the irreducible polynomial $f(y)$ selected for the finite field. For hardware implementations, trinomials [7], [8], [9] and pentanomials are normally used. PB multiplication requires a multiplication of polynomials followed by a modular reduction. Efficient bit-parallel multipliers can be implemented using a *product matrix* that combine the above two steps together [10], [11], [12], [13], [14]. A new PB multiplication method based on the decomposition of a product matrix was used in [15]. This method introduced the functions $\mathbf{S_i}$ and $\mathbf{T_i}$ given by sum of terms $x_k = (a_k b_k)$ and $z_i^j = (a_i b_j + a_j b_i)$, where $a_i, b_i \in GF(2)$ are the coefficients of $A, B \in GF(2^m)$. The coefficients of the product can be computed as the sum of that functions. The above method was applied in [15] to *type I irreducible pentanomials*, where *groups* of shared subexpressions were determined in order to reduce the area complexity of the multiplier. In [16], the sum of products given in the $\mathbf{S_i}$ and $\mathbf{T_i}$ functions were splitted into sums of 2^j product terms that can be implemented as binary trees of XOR gates with depth j. The sum in pairs of binary trees with the same depth yields a reduction of the number of XOR levels needed to compute the product coefficients. Furthermore, the use of binary trees of XOR gates can minimize power consumption in comparison to the use of linear arrays of XORs [17]. The multiplication approach given in [16] was applied to *type II irreducible pentanomials* in the form $f(y) = y^m + y^{n+2} + y^{n+1} + y^n + 1$.

• *J. L. Imaña is with the Department of Computer Architecture and Systems Engineering, Faculty of Physics, Complutense University, Madrid 28040, Spain. E-mail: jluimana@ucm.es.*

Manuscript received 11 May 2017; revised 25 Nov. 2017; accepted 26 Nov. 2017. Date of publication 0 . 0000; date of current version 0 . 0000.
(Corresponding author: José L. Imaña.)
Recommended for acceptance by W. Liu.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TC.2017.2778730

In this paper, a new fast bit-parallel $GF(2^m)$ polynomial basis multiplier is presented, where the splitting approach in [16] has been applied to general *type I irreducible pentanomials* and where the expressions of the product coefficients given in [15] for these pentanomials have been simplified in order to obtain high-speed multipliers. *Type I irreducible pentanomials* $f(y) = y^m + y^{n+1} + y^n + y + 1$, where $2 \le n \le \lfloor m/2 \rfloor - 1$, are very important because they are abundant (there are 807 different m values in the interval $[8, 1000]$ such that a type I irreducible pentanomial of degree m exists) and they are used in important applications. For example, arithmetic used in the *Advanced Encryption Standard* (AES) is based on the binary extension field $GF(2^8)$ generated by type I irreducible pentanomial $f(y) = y^8 + y^4 + y^3 + y + 1$. Furthermore, the three finite fields $m \in \{163, 233, 283\}$ from the five recommended by National Institute of Standards and Technology (NIST) for Elliptic Curve Digital Signature Algorithm (ECDSA) can be constructed using such pentanomials. The bit-parallel PB multiplier here presented has the lowest delay known to date for similar PB multipliers based on this type of irreducible pentanomials. In order to prove the theoretical complexities, hardware implementations over Xilinx FPGAs have also been performed. NIST and SECG (Standards for Efficient Cryptography Group) recommended $GF(2^m)$ multipliers have been described in VHDL and post-place and route implementation results in Xilinx Artix-7 have been reported. Experimental results show that the approach here presented exhibits the lowest delay with a balanced $Area \times Time$ complexity when it is compared with similar multipliers.

The paper is organized as follows. Section 2 provides notation and mathematical background. Type I irreducible pentanomials are introduced in Section 3, where new reduced expressions for multiplication are given. Section 4 describes the new multiplier, gives an example of multiplication and analyses the theoretical complexity. Comparisons with other similar multipliers are given in Section 5. Hardware implementation results are presented in Section 6. Finally, conclusions are given in Section 7.

2 BACKGROUND

Let $f(y) = \sum_{i=0}^{m} f_i y^i$ be a monic irreducible polynomial of degree m over $GF(2)$. The elements of the binary extension field $GF(2^m)$ can be represented in the *polynomial basis* $\{1, x, \ldots, x^{m-1}\}$, where x is a root of the irreducible generating polynomial $f(y)$. Any element $A \in GF(2^m)$ is represented in PB as $A = \sum_{i=0}^{m-1} a_i x^i$, where $a_i's \in GF(2)$ are the coefficients of A. In order to compute the coefficients of the product $C = A \cdot B$, a new method was used in [15]. This method introduced the functions $\mathbf{S_i}$ and $\mathbf{T_i}$ given by the sum of terms $x_k = (a_k b_k)$ and $z_i^j = (a_i b_j + a_j b_i)$, where $a_i, b_i \in GF(2)$ are the coefficients of A and B, respectively. These functions are implemented as *binary trees* of 2-input XOR gates with a lower level of 2-input AND gates (corresponding to the $a_i b_j$ products). The product $C = A \cdot B$ can be computed as the sum of these functions. The expressions for $\mathbf{S_i}$ $(1 \le i \le m)$ and $\mathbf{T_i}$ $(0 \le i \le m-2)$ with $\varsigma = \lfloor i/2 \rfloor$ and $\gamma = (\lceil m/2 \rceil + \lfloor i/2 \rfloor)$, are [16]

$$\mathbf{S_i} = x_\varsigma + \sum_{h=0}^{\varsigma-1} z_h^{i-h-1}, \quad \mathbf{T_i} = x_\gamma + \sum_{j=1}^{\eta-(i+1)} z_{i+j}^{m-j}, \tag{1}$$

where $x_\varsigma = a_\varsigma b_\varsigma$ only appears for i *odd* and x_γ only appears for (m and i *even*) or for (m and i *odd*). In this case, $\eta = \gamma$. Otherwise, i.e., for (m *even* and i *odd*) or for (m *odd* and i *even*), the term x_γ does not appear and the value of $\eta = (\lceil m/2 \rceil + \lceil i/2 \rceil)$. For example, for $GF(2^5)$ the terms $\mathbf{S_i}$ and $\mathbf{T_i}$ are as follows: $\mathbf{S_1} = x_0 = a_0 b_0$, $\mathbf{S_2} = z_0^1 = (a_0 b_1 + a_1 b_0)$, $\mathbf{S_3} = x_1 + z_0^2 = a_1 b_1 + (a_0 b_2 + a_2 b_0)$, $\mathbf{S_4} = (a_0 b_3 + a_3 b_0) + (a_1 b_2 + a_2 b_1)$, $\mathbf{S_5} = a_2 b_2 + (a_0 b_4 + a_4 b_0) + (a_1 b_3 + a_3 b_1)$,

TABLE 1
Coordinates c_i of the Product for the Pentanomial $f(y) = y^{13} + y^4 + y^3 + y + 1$

c_0	$\mathbf{S_1}$	$\mathbf{T_0}$	$\mathbf{T_9}$			$\mathbf{T_{10}}$				
c_1	$\mathbf{S_2}$	$\mathbf{T_1}$	~~$\mathbf{T_{10}}$~~			$\mathbf{T_{11}}$			$\mathbf{T_0}$	$\mathbf{T_9}$	~~$\mathbf{T_{10}}$~~
c_2	$\mathbf{S_3}$	$\mathbf{T_2}$	~~$\mathbf{T_{11}}$~~						$\mathbf{T_1}$	$\mathbf{T_{10}}$	~~$\mathbf{T_{11}}$~~
c_3	$\mathbf{S_4}$	$\mathbf{T_3}$				$\mathbf{T_0}$	$\mathbf{T_9}$	$\mathbf{T_{10}}$	$\mathbf{T_2}$	$\mathbf{T_{11}}$	
c_4	$\mathbf{S_5}$	$\mathbf{T_4}$	$\mathbf{T_0}$	$\mathbf{T_9}$	~~$\mathbf{T_{10}}$~~	$\mathbf{T_1}$	~~$\mathbf{T_{10}}$~~	$\mathbf{T_{11}}$	$\mathbf{T_3}$		
c_5	$\mathbf{S_6}$	$\mathbf{T_5}$	$\mathbf{T_1}$	$\mathbf{T_{10}}$	~~$\mathbf{T_{11}}$~~	$\mathbf{T_2}$	~~$\mathbf{T_{11}}$~~		$\mathbf{T_4}$		
c_6	$\mathbf{S_7}$	$\mathbf{T_6}$	$\mathbf{T_2}$	$\mathbf{T_{11}}$		$\mathbf{T_3}$			$\mathbf{T_5}$		
c_7	$\mathbf{S_8}$	$\mathbf{T_7}$	$\mathbf{T_3}$			$\mathbf{T_4}$			$\mathbf{T_6}$		
c_8	$\mathbf{S_9}$	$\mathbf{T_8}$	$\mathbf{T_4}$			$\mathbf{T_5}$			$\mathbf{T_7}$		
c_9	$\mathbf{S_{10}}$	$\mathbf{T_9}$	$\mathbf{T_5}$			$\mathbf{T_6}$			$\mathbf{T_8}$		
c_{10}	$\mathbf{S_{11}}$	$\mathbf{T_{10}}$	$\mathbf{T_6}$			$\mathbf{T_7}$			$\mathbf{T_9}$		
c_{11}	$\mathbf{S_{12}}$	$\mathbf{T_{11}}$	$\mathbf{T_7}$			$\mathbf{T_8}$			$\mathbf{T_{10}}$		
c_{12}	$\mathbf{S_{13}}$		$\mathbf{T_8}$			$\mathbf{T_9}$			$\mathbf{T_{11}}$		

$\mathbf{T_0} = (a_1b_4 + a_4b_1) + (a_2b_3 + a_3b_2)$, $\mathbf{T_1} = a_3b_3 + (a_2b_4 + a_4b_2)$, $\mathbf{T_2} = (a_3b_4 + a_4b_3)$, $\mathbf{T_3} = a_4b_4$.

3 TYPE I IRREDUCIBLE PENTANOMIALS

Type I irreducible pentanomials were defined in [14] as $f(y) = y^m + y^{n+1} + y^n + y + 1$, for $2 \le n \le \lfloor m/2 \rfloor - 1$. These pentanomials are very important because they are abundant and they are used in a wide number of applications. For example, the specific type I pentanomial $f(y) = y^8 + y^4 + y^3 + y + 1$ is used in the *Advanced Encryption Standard*.

Polynomial basis multiplication for type I irreducible pentanomials was studied in [15], where expressions of the product coefficients were computed. In these expressions, *groups* $\mathbf{G_i^j}$ of subexpressions given as sums of j terms $\mathbf{T_k}$ were also found. These j-terms groups $\mathbf{G_i^j}$ can be shared among different coefficients leading to a reduction of area complexity of the multiplier. In this work, it is observed that the common groups found in [15] can be simplified in order to reduce the delay of the multiplier. The simplification is shown in the following example with $(m, n) = (13, 3)$.

3.1 $GF(2^{13})$ Multiplier for $f(y) = y^{13} + y^4 + y^3 + y + 1$

The product $C = A \cdot B$ in $GF(2^{13})$ generated by the type I irreducible pentanomial with parameters (m, n) = (13,3) can be computed using the expressions given in [15]. The coefficients c_i of the product are $c_0 = \mathbf{S_1} + \mathbf{G_0^3}$, $c_1 = \mathbf{S_2} + \mathbf{G_0^6}$, $c_2 = \mathbf{S_3} + \mathbf{G_0^5}$, $c_3 = \mathbf{S_4} + \mathbf{G_0^3} + \mathbf{G_2^3}$, $c_4 = \mathbf{S_5} + \mathbf{G_1^2} + \mathbf{G_0^6}$, $c_5 = \mathbf{S_6} + \mathbf{G_2^2} + \mathbf{G_0^5}$, $c_6 = \mathbf{S_7} + \mathbf{G_3^2} + \mathbf{G_2^3}$, $c_7 = \mathbf{S_8} + \mathbf{G_4^2} + \mathbf{G_1^2}$, $c_8 = \mathbf{S_9} + \mathbf{G_5^2} + \mathbf{G_2^2}$, $c_9 = \mathbf{S_{10}} + \mathbf{G_6^2} + \mathbf{G_3^2}$, $c_{10} = \mathbf{S_{11}} + \mathbf{G_7^2} + \mathbf{G_4^2}$, $c_{11} = \mathbf{S_{12}} + \mathbf{G_8^2} + \mathbf{G_5^2}$, $c_{12} = \mathbf{S_{13}} + \mathbf{T_{11}} + \mathbf{G_6^2}$. In the above coefficients, the 2-terms groups are given by the expressions [15] $\mathbf{G_0^2} = (\mathbf{T_2} + \mathbf{T_{11}})$, $\mathbf{G_1^2} = (\mathbf{T_3} + \mathbf{T_4})$, $\mathbf{G_2^2} = (\mathbf{T_4} + \mathbf{T_5})$, $\mathbf{G_3^2} = (\mathbf{T_5} + \mathbf{T_6})$, $\mathbf{G_4^2} = (\mathbf{T_6} + \mathbf{T_7})$, $\mathbf{G_5^2} = (\mathbf{T_7} + \mathbf{T_8})$, $\mathbf{G_6^2} = (\mathbf{T_8} + \mathbf{T_9})$, $\mathbf{G_7^2} = (\mathbf{T_9} + \mathbf{T_{10}})$, $\mathbf{G_8^2} = (\mathbf{T_{10}} + \mathbf{T_{11}})$, the 3-terms groups are given by $\mathbf{G_0^3} = (\mathbf{T_0} + \mathbf{G_7^2})$, $\mathbf{G_1^3} = (\mathbf{T_1} + \mathbf{G_8^2})$, $\mathbf{G_2^3} = (\mathbf{T_3} + \mathbf{G_0^2})$, the 5-terms group is $\mathbf{G_0^5} = (\mathbf{G_0^2} + \mathbf{G_1^3})$, and the 6-terms group is $\mathbf{G_0^6} = (\mathbf{G_0^3} + \mathbf{G_1^3})$. The coefficients of this multiplier are given in Table 1, where a c_i coordinate is the sum of the $\mathbf{S_l}$ and $\mathbf{T_p}$ terms in the ith row. In Table 1, the above $\mathbf{G_i^j}$ groups are not represented and only individual terms $\mathbf{T_k}$ are shown. It can also be observed that there are several $\mathbf{T_i}$ terms that are cancelled in some rows.

3.2 New General Expressions for the Multiplier

In a similar way to that seen in the previous example, the coordinates of the product $C = A \cdot B$ in PB for general type I pentanomials $f(y) = y^m + y^{n+1} + y^n + y + 1$, with $2 \le n < \lfloor m/2 \rfloor - 1$, are given in Table 2, where $z = m - n$. From the table, it can be observed that several $\mathbf{T_i}$ terms are cancelled, therefore reducing the complexity of the multiplier.

The new general reduced expressions for the coordinates are also given in Table 3. In this table, the coefficients have been divided into eight sections (named from Ⓐ to Ⓗ), depending on the terms $\mathbf{T_i}$ involved and on the number of $\mathbf{S_i}$ and $\mathbf{T_i}$ terms in the sums for the coefficients. The number of terms in sections Ⓐ, Ⓑ, Ⓒ, Ⓓ, Ⓔ, Ⓕ, Ⓖ and Ⓗ is 4, 5, 4, 7, 7, 6, 5 and 4, respectively. It can be observed that coefficients in sections Ⓓ and Ⓔ present the maximum number of terms (seven). Furthermore, from equation (1), the term $\mathbf{T_0}$ is given by the addition of $\eta - 1$ terms z_i^j and the term x_γ (if it exists), i.e., it performs the sum of the maximum number of z_i^j terms and therefore it presents the highest delay. From (1), the complexity of $\mathbf{T_i}$ terms decreases when subindex i increases, so the next most complex $\mathbf{T_i}$ term is $\mathbf{T_1}$. It must be noted that the coefficient c_{n+1} (in section Ⓔ) has the maximum number of terms (seven) and it includes the two most complex terms $\mathbf{T_0}$ and $\mathbf{T_1}$ in its sum. Therefore, c_{n+1} is the most complex coefficient and it will be used in following sections to determine the delay of the new multiplier.

4 NEW MULTIPLIER FOR TYPE I IRREDUCIBLE PENTANOMIALS

As shown in Table 3, the coefficients of the product $C = A \cdot B$ in PB can be computed as the addition of functions $\mathbf{S_i}$ and $\mathbf{T_i}$ that are given in (1) by sum of terms $x_k = (a_kb_k)$ and $z_i^j = (a_ib_j + a_jb_i)$. However, the monolithic construction of $\mathbf{S_i}$ and $\mathbf{T_i}$ terms can represent a problem if low-delay implementations are required. For example, for $GF(2^5)$, functions $\mathbf{T_1} = x_3 + z_2^4 = (a_3b_3 + (a_2b_4 + a_4b_2))$ and $\mathbf{T_3} = x_4 = a_4b_4$ are defined. The addition $\mathbf{T_1} + \mathbf{T_3} =$

TABLE 2
Coefficients c_i of the Product for Type I Pentanomial $f(y) = y^m + y^{n+1} + y^n + y + 1$ with $2 \le n < \lfloor m/2 \rfloor - 1$

c_0	$\mathbf{S_1}$	$\mathbf{T_0}$	$\mathbf{T_{z-1}}$			$\mathbf{T_z}$				
c_1	$\mathbf{S_2}$	$\mathbf{T_1}$	~~$\mathbf{T_z}$~~			$\mathbf{T_{z+1}}$			$\mathbf{T_0}$	$\mathbf{T_{z-1}}$	~~$\mathbf{T_z}$~~
$\vdots$	$\vdots$	$\vdots$	$\vdots$			$\vdots$			$\vdots$	$\vdots$	$\vdots$
c_{n-2}	$\mathbf{S_{n-1}}$	$\mathbf{T_{n-2}}$	~~$\mathbf{T_{m-3}}$~~			$\mathbf{T_{m-2}}$			$\mathbf{T_{n-3}}$	$\mathbf{T_{m-4}}$	~~$\mathbf{T_{m-3}}$~~
c_{n-1}	$\mathbf{S_n}$	$\mathbf{T_{n-1}}$	~~$\mathbf{T_{m-2}}$~~						$\mathbf{T_{n-2}}$	$\mathbf{T_{m-3}}$	~~$\mathbf{T_{m-2}}$~~
c_n	$\mathbf{S_{n+1}}$	$\mathbf{T_n}$				$\mathbf{T_0}$	$\mathbf{T_{z-1}}$	$\mathbf{T_z}$	$\mathbf{T_{n-1}}$	$\mathbf{T_{m-2}}$	
c_{n+1}	$\mathbf{S_{n+2}}$	$\mathbf{T_{n+1}}$	$\mathbf{T_0}$	$\mathbf{T_{z-1}}$	~~$\mathbf{T_z}$~~	$\mathbf{T_1}$	~~$\mathbf{T_z}$~~	$\mathbf{T_{z+1}}$	$\mathbf{T_n}$		
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$		
c_{2n-2}	$\mathbf{S_{2n-1}}$	$\mathbf{T_{2n-2}}$	$\mathbf{T_{n-3}}$	$\mathbf{T_{m-4}}$	~~$\mathbf{T_{m-3}}$~~	$\mathbf{T_{n-2}}$	~~$\mathbf{T_{m-3}}$~~	$\mathbf{T_{m-2}}$	$\mathbf{T_{2n-3}}$		
c_{2n-1}	$\mathbf{S_{2n}}$	$\mathbf{T_{2n-1}}$	$\mathbf{T_{n-2}}$	$\mathbf{T_{m-3}}$	~~$\mathbf{T_{m-2}}$~~	$\mathbf{T_{n-1}}$	~~$\mathbf{T_{m-2}}$~~		$\mathbf{T_{2n-2}}$		
c_{2n}	$\mathbf{S_{2n+1}}$	$\mathbf{T_{2n}}$	$\mathbf{T_{n-1}}$	$\mathbf{T_{m-2}}$		$\mathbf{T_n}$			$\mathbf{T_{2n-1}}$		
c_{2n+1}	$\mathbf{S_{2n+2}}$	$\mathbf{T_{2n+1}}$	$\mathbf{T_n}$			$\mathbf{T_{n+1}}$			$\mathbf{T_{2n}}$		
$\vdots$	$\vdots$	$\vdots$	$\vdots$			$\vdots$			$\vdots$		
c_{m-2}	$\mathbf{S_{m-1}}$	$\mathbf{T_{m-2}}$	$\mathbf{T_{z-3}}$			$\mathbf{T_{z-2}}$			$\mathbf{T_{m-3}}$		
c_{m-1}	$\mathbf{S_m}$		$\mathbf{T_{z-2}}$			$\mathbf{T_{z-1}}$			$\mathbf{T_{m-2}}$		

TABLE 3
New Reduced Expressions for the Coefficients of the Product

$c_0 = \mathbf{S_1} + \mathbf{T_0} + \mathbf{T_{z-1}} + \mathbf{T_z}$;	A
if $n \geq 3$, for $i = 1 \dots n-2$: $c_i = \mathbf{S_{i+1}} + \mathbf{T_{i-1}} + \mathbf{T_i} + \mathbf{T_{z+(i-2)}} + \mathbf{T_{z+i}}$;	B
$c_{n-1} = \mathbf{S_n} + \mathbf{T_{n-2}} + \mathbf{T_{n-1}} + \mathbf{T_{m-3}}$;	C
$c_n = \mathbf{S_{n+1}} + \mathbf{T_0} + \mathbf{T_{n-1}} + \mathbf{T_n} + \mathbf{T_{z-1}} + \mathbf{T_z} + \mathbf{T_{m-2}}$;	D
if $n \geq 3$, for $i = n+1 \dots 2n-2$: $c_i = \mathbf{S_{i+1}} + \mathbf{T_{i-(n+1)}} + \mathbf{T_{i-n}} + \mathbf{T_{i-1}} + \mathbf{T_i} + \mathbf{T_{i+z-(n+2)}} + \mathbf{T_{i+z-n}}$;	E
$c_{2n-1} = \mathbf{S_{2n}} + \mathbf{T_{n-2}} + \mathbf{T_{n-1}} + \mathbf{T_{2n-2}} + \mathbf{T_{2n-1}} + \mathbf{T_{m-3}}$; $c_{2n} = \mathbf{S_{2n+1}} + \mathbf{T_{n-1}} + \mathbf{T_n} + \mathbf{T_{2n-1}} + \mathbf{T_{2n}} + \mathbf{T_{m-2}}$;	F
if $m \geq 2n+3$, for $i = 2n+1 \dots m-2$: $c_i = \mathbf{S_{i+1}} + \mathbf{T_{i-(n+1)}} + \mathbf{T_{i-n}} + \mathbf{T_{i-1}} + \mathbf{T_i}$;	G
$c_{m-1} = \mathbf{S_m} + \mathbf{T_{z-2}} + \mathbf{T_{z-1}} + \mathbf{T_{m-2}}$;	H

TABLE 4
$\mathbf{S_i}$ and $\mathbf{T_i}$ Functions for $GF(2^{13})$

	2^3	2^2	2^1	2^0	binary
$\mathbf{S_1}$ $\mathbf{T_{11}}$				x_0 x_{12}	0001
$\mathbf{S_2}$ $\mathbf{T_{10}}$			z_0^1 z_{11}^{12}		0010
$\mathbf{S_3}$ $\mathbf{T_9}$			z_0^2 z_{10}^{12}	x_1 x_{11}	0011
$\mathbf{S_4}$ $\mathbf{T_8}$		$(z_0^3 + z_1^2)$ $(z_9^{12} + z_{10}^{11})$			0100
$\mathbf{S_5}$ $\mathbf{T_7}$		$(z_0^4 + z_1^3)$ $(z_8^{12} + z_9^{11})$		x_2 x_{10}	0101
$\mathbf{S_6}$ $\mathbf{T_6}$		$(z_1^4 + z_2^3)$ $(z_8^{11} + z_9^{10})$	z_0^5 z_7^{12}		0110
$\mathbf{S_7}$ $\mathbf{T_5}$		$(z_1^5 + z_2^4)$ $(z_7^{11} + z_8^{10})$	z_0^6 z_6^{12}	x_3 x_9	0111
$\mathbf{S_8}$ $\mathbf{T_4}$	$(z_0^7 + z_1^6 + z_2^5 + z_3^4)$ $(z_5^{12} + z_6^{11} + z_7^{10} + z_8^9)$				1000
$\mathbf{S_9}$ $\mathbf{T_3}$	$(z_0^8 + z_1^7 + z_2^6 + z_3^5)$ $(z_4^{12} + z_5^{11} + z_6^{10} + z_7^9)$			x_4 x_8	1001
$\mathbf{S_{10}}$ $\mathbf{T_2}$	$(z_1^8 + z_2^7 + z_3^6 + z_4^5)$ $(z_4^{11} + z_5^{10} + z_6^9 + z_7^8)$		z_0^9 z_3^{12}		1010
$\mathbf{S_{11}}$ $\mathbf{T_1}$	$(z_1^9 + z_2^8 + z_3^7 + z_4^6)$ $(z_3^{11} + z_4^{10} + z_5^9 + z_6^8)$		z_0^{10} z_2^{12}	x_5 x_7	1011
$\mathbf{S_{12}}$ $\mathbf{T_0}$	$(z_2^9 + z_3^8 + z_4^7 + z_5^6)$ $(z_3^{10} + z_4^9 + z_5^8 + z_6^7)$	$(z_0^{11} + z_1^{10})$ $(z_1^{12} + z_2^{11})$			1100
$\mathbf{S_{13}}$	$(z_2^{10} + z_3^9 + z_4^8 + z_5^7)$	$(z_0^{12} + z_1^{11})$		x_6	1101

$((a_3b_3 + (a_2b_4 + a_4b_2)) + a_4b_4)$, where terms in brackets indicate that they must be added previously to the XOR with the other terms, results in a 3-level binary tree of 2-input XOR gates. However, the addition $\mathbf{T_1} + \mathbf{T_3}$ involves the XOR of four product terms. This sum could be implemented with a 2-level complete binary tree of XOR gates if the additions could be done in a separate way, i.e., if the product a_3b_3 could be first XORed with the term a_4b_4 and then perform the XOR with $(a_2b_4 + a_4b_2)$ in the form $\mathbf{T_1} + \mathbf{T_3} = ((a_3b_3 + a_4b_4) + (a_2b_4 + a_4b_2))$.

In [16], a new approach was given by considering the functions $\mathbf{S_i}$ and $\mathbf{T_i}$ as an addition of $\mathbf{S_i^j}$ and $\mathbf{T_i^j}$ terms, respectively, in such a way that $\mathbf{S_i} = s_\rho^i \mathbf{S_i^\rho} + \dots + s_0^i \mathbf{S_i^0}$ and $\mathbf{T_i} = t_\rho^i \mathbf{T_i^\rho} + \dots + t_0^i \mathbf{T_i^0}$ for $GF(2^m)$, with $s_j^i, t_j^i \in GF(2)$ and $\rho = \lfloor log_2 m \rfloor$. The initial terms $\mathbf{S_i^j}$ and $\mathbf{T_i^j}$ represent the sum of 2^j products $a_k b_l$, so they can be implemented as j-level complete binary trees of 2-input XOR gates. The addition of two terms $\mathbf{S_i^j}$ and $\mathbf{T_i^j}$ with the same superscript j results in a new 2-input XOR in level $j+1$ that represents a $(j+1)$-level binary tree. If the sum of $\mathbf{S_i}$ and $\mathbf{T_i}$ functions is done grouping the additions of terms with the same j-level $\mathbf{S_i^j}$ and $\mathbf{T_i^j}$, starting with lower levels, then the number of XOR levels needed to compute the coefficients of the product can be reduced. The 0-level initial terms $\mathbf{S_i^0}$ and $\mathbf{T_i^0}$ should be first added in pairs to give rise to a new XOR in level 1, that in turn should be XORed with other 1-level term to give rise to a new 2-level binary tree and so on. If there is only one j-level term (or there is an unpaired j-level term), then it should be XORed with a $(j+1)$-level term in order to have a new $(j+2)$-level tree.

It can be noted that vectors $(s_\rho^i, \dots, s_0^i)_2$ and $(t_\rho^i, \dots, t_0^i)_2$ are given by the binary representations of the subindex i for $\mathbf{S_i}$ and of the value $m-1-i$ for $\mathbf{T_i}$, respectively [16]. Furthermore, common terms appearing in several coefficients can be shared in order to reduce the number of XORs. These common terms correspond to the sums $(\mathbf{S_i} + \mathbf{S_{i+1}})$ and $(\mathbf{T_i} + \mathbf{T_{i+1}})$ that involve the additions $(\mathbf{S_i^l} + \mathbf{S_{i+1}^l})$ and $(\mathbf{T_i^l} + \mathbf{T_{i+1}^l})$, respectively, for different levels l determined by the binary representations of i (for $\mathbf{S_i}$) and $m-1-i$ (for $\mathbf{T_i}$). The notation $\mathbf{T_{i,j}^{l+1}} = (\mathbf{T_i^l} + \mathbf{T_j^l})$ and $\mathbf{ST_{i,j}^{l+1}} = (\mathbf{S_i^l} + \mathbf{T_j^l})$ can be used to represent the addition of two terms in level l to yield a new term in level $l+1$. From Table 3, it can be observed that only common additions $(\mathbf{T_i} + \mathbf{T_{i+1}})$ can be found (with $i = 0, \dots, m-4$ for *even* m, and with $i = 0, \dots, m-4$ for *odd* m) [16]. The following algorithm for multiplication using the above approach was given in [16]:

1) Compute $\mathbf{S_i^j}$ and $\mathbf{T_i^j}$ terms using (1).
2) For each level $l = 0 \dots \rho$, create common terms $\mathbf{T_{i,i+1}^{l+1}}$.
3) For each coefficient of the multiplier:
   a) For each level $l = 0 \dots \rho$:
      * Share common terms $\mathbf{T_{i,i+1}^{l+1}}$.
      * Sum $\mathbf{U^l}$ terms in pairs to create $\mathbf{U^{l+1}}$ terms.
      * If $\exists$ a non-paired $\mathbf{U^l_-}$ term, consider it as $\mathbf{U^{l+1}_-}$.
   b) While the number of $\mathbf{U^l_-}$ terms ≥ 2:
      * Sum $\mathbf{U^l_-}$ terms in pairs to create $\mathbf{U^{l+1}_-}$ terms.
      * If $\exists$ a non-paired $\mathbf{U^l_-}$ term, consider it as $\mathbf{U^{l+1}_-}$.

where $\mathbf{U^l_-}$ denotes $\mathbf{T_i^l}$, $\mathbf{S_i^l}$, $\mathbf{T_{i,j}^l}$, $\mathbf{S_{i,j}^l}$ or $\mathbf{ST_{i,j}^l}$ terms at level l. In the next section, the representation introduced in [16] is applied to the new reduced expressions given in Table 3 for the type I pentanomial multiplier with $(m, n) = (13, 3)$.

4.1 Type I Pentanomial Multiplier for $(m, n) = (13, 3)$

Let us consider the product $C = A \cdot B$ in $GF(2^{13})$ generated by type I pentanomial $f(y) = y^{13} + y^4 + y^3 + y + 1$. Using equation (1), $\mathbf{S_i}$ and $\mathbf{T_i}$ functions are given in Table 4 where $\mathbf{S_i}$ and $\mathbf{T_i}$ are the XOR of the x_k and z_i^j terms given in their rows. In this table, the columns labeled as 2^0, 2^1, 2^2 and 2^3 represent the number of product terms $a_h b_l$ involved in each column. For example, $\mathbf{S_{11}} = x_5 + z_0^{10} + (z_1^9 + z_2^8 + z_3^7 + z_4^6)$, where x_5 involves $1 = 2^0$ product term (a_5b_5), the term z_0^{10} involves the XOR of $2 = 2^1$ terms $(a_0b_{10} + a_{10}b_0)$ and $(z_1^9 + z_2^8 + z_3^7 + z_4^6)$ is the sum of $8 = 2^3$ product terms $((a_1b_9 + a_9b_1) + (a_2b_8 + a_8b_2) + (a_3b_7 + a_7b_3) + (a_4b_6 + a_6b_4))$. Term $\mathbf{S_{11}}$ can then be represented by $\mathbf{S_{11}} = s_3^{11}\mathbf{S_{11}^3} + s_2^{11}\mathbf{S_{11}^2} + s_1^{11}\mathbf{S_{11}^1} + s_0^{11}\mathbf{S_{11}^0} = 1.\mathbf{S_{11}^3} + 0.\mathbf{S_{11}^2} + 1.\mathbf{S_{11}^1} + 1.\mathbf{S_{11}^0}$ where $\mathbf{S_{11}^3}$, $\mathbf{S_{11}^2}$, $\mathbf{S_{11}^1}$ and $\mathbf{S_{11}^0}$ stand for terms with 2^3, 2^2, 2^1 and 2^0 product terms, respectively. In this case, the not null terms $\mathbf{S_{11}^0} = x_5$, $\mathbf{S_{11}^1} = z_0^{10}$ and $\mathbf{S_{11}^3} = (z_1^9 + z_2^8 + z_3^7 + z_4^6)$. It can be observed that the binary vector $(s_3^{11}, s_2^{11}, s_1^{11}, s_0^{11})_2 = (1, 0, 1, 1)_2 = 11_{10}$. This representation is given in the column labeled *binary* in Table 4, where it can be observed that the binary vector $(s_3^i, s_2^i, s_1^i, s_0^i)_2$ for $\mathbf{S_i}$ matches with the binary vector $(t_3^j, t_2^j, t_1^j, t_0^j)_2$ for $\mathbf{T_j}$, with $j = m-1-i$. For example, the term $\mathbf{T_1}$ corresponds with the binary vector $(1011)_2$ that is the binary representation of the value $13-1-1 = 11$ (in this example with $m = 13$). This fact is represented in Table 4 including terms $\mathbf{S_i}$ and $\mathbf{T_{m-1-i}}$ in a row with the same binary representation.

The space complexity of the multiplier can be reduced if common terms that appear in several coefficients are shared. In Table 4, consecutive $\mathbf{S_i}$ and $\mathbf{T_i}$ terms having $\mathbf{S_i^j}$ and $\mathbf{T_i^j}$ terms with the same

TABLE 5
Coefficients of the Product for $GF(2^{13})$

c_0	$\mathbf{S_1} + \mathbf{T_0} + \mathbf{T_9} + \mathbf{T_{10}}$	$((\mathbf{ST^1_{1,9}} + \mathbf{T^2_0}) + \mathbf{T^2_{9,10}}) + \mathbf{T^3_0};$	(A)
c_1	$\mathbf{S_2} + \mathbf{T_0} + \mathbf{T_1} + \mathbf{T_9} + \mathbf{T_{11}}$	$((\mathbf{S^1_2} + \mathbf{T^2_0}) + \mathbf{T^3_0}) + (((\mathbf{T^1_{1,9}} + \mathbf{T^1_1}) + (\mathbf{T^1_9} + \mathbf{T^0_{11}})) + \mathbf{T^3_1});$	(B)
c_2	$\mathbf{S_3} + \mathbf{T_1} + \mathbf{T_2} + \mathbf{T_{10}}$	$((\mathbf{ST^1_{3,1}} + \mathbf{S^1_3}) + (\mathbf{T^2_{1,2}} + \mathbf{T^1_{10}})) + \mathbf{T^4_{1,2}};$	(C)
c_3	$\mathbf{S_4} + \mathbf{T_0} + \mathbf{T_2} + \mathbf{T_3} + \mathbf{T_9} + \mathbf{T_{10}} + \mathbf{T_{11}}$	$(\mathbf{ST^3_{4,0}} + \mathbf{T^3_0}) + (((\mathbf{T^1_{3,9}} + \mathbf{T^1_2}) + \mathbf{T^2_{9,10}}) + \mathbf{T^4_{2,3}});$	(D)
c_4	$\mathbf{S_5} + \mathbf{T_0} + \mathbf{T_1} + \mathbf{T_3} + \mathbf{T_4} + \mathbf{T_9} + \mathbf{T_{11}}$	$(((\mathbf{ST^1_{5,1}} + \mathbf{T^1_1}) + \mathbf{T^3_1}) + (\mathbf{ST^3_{5,0}} + \mathbf{T^3_0})) + ((\mathbf{T^1_{3,9}} + (\mathbf{T^1_9} + \mathbf{T^0_{11}})) + \mathbf{T^4_{3,4}});$	(E)
c_5	$\mathbf{S_6} + \mathbf{T_1} + \mathbf{T_2} + \mathbf{T_4} + \mathbf{T_5} + \mathbf{T_{10}}$	$((((\mathbf{T^1_{1,5}} + \mathbf{S^1_6}) + \mathbf{S^2_6}) + \mathbf{T^2_{1,2}}) + \mathbf{T^4_{1,2}}) + ((\mathbf{T^2_{5,10}} + \mathbf{T^2_5}) + \mathbf{T^3_4});$	(F)
c_6	$\mathbf{S_7} + \mathbf{T_2} + \mathbf{T_3} + \mathbf{T_5} + \mathbf{T_6} + \mathbf{T_{11}}$	$((\mathbf{ST^2_{7,2}} + \mathbf{S^2_7}) + \mathbf{T^3_2}) + ((((\mathbf{ST^1_{7,3}} + \mathbf{T^1_{5,11}}) + \mathbf{T^2_{5,6}}) + \mathbf{T^3_3}) + \mathbf{T^3_{5,6}});$	
c_7	$\mathbf{S_8} + \mathbf{T_3} + \mathbf{T_4} + \mathbf{T_6} + \mathbf{T_7}$	$(\mathbf{S^3_8} + \mathbf{T^4_{3,4}}) + (((\mathbf{T^1_{3,7}} + \mathbf{T^1_6}) + \mathbf{T^2_6}) + \mathbf{T^2_7});$	
c_8	$\mathbf{S_9} + \mathbf{T_4} + \mathbf{T_5} + \mathbf{T_7} + \mathbf{T_8}$	$(((\mathbf{ST^1_{9,5}} + \mathbf{T^2_5}) + \mathbf{T^3_4}) + \mathbf{S^3_9}) + ((\mathbf{T^0_7} + \mathbf{T^1_5}) + \mathbf{T^3_{7,8}});$	
c_9	$\mathbf{S_{10}} + \mathbf{T_5} + \mathbf{T_6} + \mathbf{T_8} + \mathbf{T_9}$	$((\mathbf{S^1_{10}} + \mathbf{T^2_{5,6}}) + \mathbf{S^3_{10}}) + (((\mathbf{T^1_{5,9}} + \mathbf{T^1_9}) + \mathbf{T^2_8}) + \mathbf{T^3_{5,6}});$	(G)
c_{10}	$\mathbf{S_{11}} + \mathbf{T_6} + \mathbf{T_7} + \mathbf{T_9} + \mathbf{T_{10}}$	$(((\mathbf{S^0_{11}} + \mathbf{S^1_{11}}) + \mathbf{T^2_6}) + \mathbf{S^3_{11}}) + (((\mathbf{T^1_{7,9}} + \mathbf{T^1_6}) + \mathbf{T^2_7}) + \mathbf{T^2_{9,10}});$	
c_{11}	$\mathbf{S_{12}} + \mathbf{T_7} + \mathbf{T_8} + \mathbf{T_{10}} + \mathbf{T_{11}}$	$((((\mathbf{T^1_{7,11}} + \mathbf{T^1_{10}}) + \mathbf{S^2_{12}}) + \mathbf{T^3_{7,8}}) + \mathbf{S^3_{12}});$	
c_{12}	$\mathbf{S_{13}} + \mathbf{T_8} + \mathbf{T_9} + \mathbf{T_{11}}$	$((\mathbf{S^0_{13}} + \mathbf{T^1_9}) + \mathbf{T^1_{9,11}}) + (\mathbf{ST^3_{13,8}} + \mathbf{S^3_{13}});$	(H)

level j can be observed. For example, $\mathbf{S_6}$ and $\mathbf{S_7}$ have 1-level terms $\mathbf{S^1_6}$ and $\mathbf{S^1_7}$ and 2-level terms $\mathbf{S^2_6}$ and $\mathbf{S^2_7}$. The same applies to $\mathbf{T_5}$ and $\mathbf{T_6}$, with ($\mathbf{T^1_5}$, $\mathbf{T^2_5}$) and ($\mathbf{T^1_6}$, $\mathbf{T^2_6}$) terms, respectively. The sum $\mathbf{S_6} + \mathbf{S_7}$ (and $\mathbf{T_5} + \mathbf{T_6}$) then implies the additions $\mathbf{S^1_6} + \mathbf{S^1_7}$ ($\mathbf{T^1_5} + \mathbf{T^1_6}$) and $\mathbf{S^2_6} + \mathbf{S^2_7}$ ($\mathbf{T^2_5} + \mathbf{T^2_6}$) that give rise to 2-level and 3-level binary trees of XOR gates, respectively. Therefore, the groups ($\mathbf{S_6} + \mathbf{S_7}$) and ($\mathbf{T_5} + \mathbf{T_6}$) can reduce the complexity. The groups for this example are represented in Table 4 by shadowed cells with same color. The $\mathbf{S}$ groups are $(\mathbf{S_2}, \mathbf{S_3})$, $(\mathbf{S_4}, \mathbf{S_5})$, $(\mathbf{S_6}, \mathbf{S_7})$, $(\mathbf{S_8}, \mathbf{S_9})$, $(\mathbf{S_{10}}, \mathbf{S_{11}})$ and $(\mathbf{S_{12}}, \mathbf{S_{13}})$, while that the $\mathbf{T}$ groups are $(\mathbf{T_1}, \mathbf{T_2})$, $(\mathbf{T_3}, \mathbf{T_4})$, $(\mathbf{T_5}, \mathbf{T_6})$, $(\mathbf{T_7}, \mathbf{T_8})$ and $(\mathbf{T_9}, \mathbf{T_{10}})$.

Using Tables 1 and 3, the coefficients of the product for this $GF(2^{13})$ multiplier are given in Table 5, where the previous $\mathbf{T}$ groups are shadowed. From Table 5, it can be observed that the group $(\mathbf{T_9} + \mathbf{T_{10}})$ appears in three coefficients (c_0, c_3 and c_{10}) while that $(\mathbf{T_1} + \mathbf{T_2})$, $(\mathbf{T_3} + \mathbf{T_4})$, $(\mathbf{T_5} + \mathbf{T_6})$ and $(\mathbf{T_7} + \mathbf{T_8})$ are found in two coefficients. Therefore, only one of each of these groups must be implemented. The number of $\mathbf{T^j_i}$ terms in each group determines the number of XOR gates that can be reduced. From Table 4, it can be observed that the group $(\mathbf{T_1} + \mathbf{T_2})$ involves the addition of the two terms $(\mathbf{T^1_1} + \mathbf{T^1_2})$ and $(\mathbf{T^3_1} + \mathbf{T^3_2})$, therefore requiring 2 XOR gates. Likewise, $(\mathbf{T_3} + \mathbf{T_4})$, $(\mathbf{T_5} + \mathbf{T_6})$, $(\mathbf{T_7} + \mathbf{T_8})$ and $(\mathbf{T_9} + \mathbf{T_{10}})$ require 1, 2, 1, and 1 XOR gates, respectively. In addition, $(\mathbf{T_9} + \mathbf{T_{10}})$ can be found in three different coefficients, so the number of XOR gates that can be reduced will be $2 \cdot 1 = 2$. Therefore, the number of XOR gates that can be reduced by sharing is 2 + 1 + 2 + 1 + 2 = 8 XOR. General expressions for the computation of the number of XOR gates that can be reduced due to sharing of groups are given in Section 4.2. Using the algorithm for multiplication previously given [16] and using the $\mathbf{S^j_i}$ and $\mathbf{T^j_i}$ terms given in Table 4, the coefficients of the product are shown in the third column of Table 5. The precedence of the sums of terms in Table 5 is represented with parenthesis.

As stated in Section 3, coefficient c_{n+1} is the most complex one for Type I pentanomials. For $GF(2^{13})$, this coefficient corresponds with c_4, which implementation is given in Fig. 1. Using Table 5, it requires the addition of 7 terms (including the most complex ones $\mathbf{T_0}$ and $\mathbf{T_1}$), so it determines the maximum delay of the multiplier. The initial $\mathbf{S^l_i}$, $\mathbf{T^l_i}$ terms and the $\mathbf{T^l_{i,j}}$, $\mathbf{ST^l_{i,j}}$ terms given in Table 5 are represented in Fig. 1 by black and gray circles, respectively. For c_4, the initial terms are $\mathbf{S_5} = \mathbf{S^0_5} + \mathbf{S^2_5}$, $\mathbf{T_0} = \mathbf{T^2_0} + \mathbf{T^3_0}$, $\mathbf{T_1} = \mathbf{T^0_1} + \mathbf{T^1_1} + \mathbf{T^3_1}$, $\mathbf{T_3} = \mathbf{T^0_3} + \mathbf{T^3_3}$, $\mathbf{T_4} = \mathbf{T^3_4}$, $\mathbf{T_9} = \mathbf{T^0_9} + \mathbf{T^1_9}$ and $\mathbf{T_{11}} = \mathbf{T^0_{11}}$, while that terms $\mathbf{ST^1_{5,1}} = \mathbf{S^0_5} + \mathbf{T^0_1}$, $\mathbf{ST^3_{5,0}} = \mathbf{S^2_5} + \mathbf{T^2_0}$, $\mathbf{T^1_{3,9}} = \mathbf{T^0_3} + \mathbf{T^0_9}$ and $\mathbf{T^4_{3,4}} = \mathbf{T^3_3} + \mathbf{T^3_4}$ are also represented. In Fig. 1, levels of XOR binary trees are represented by horizontal dashed lines and $\mathbf{S_i}$ and $\mathbf{T_i}$ functions are represented by ellipses enclosing their $\mathbf{S^j_i}$ and $\mathbf{T^j_i}$ terms, respectively. Furthermore, a gray square in level l represents a non-paired term in level $l-1$ that must be considered as l-level term in order to be XORed with another term in level l (for example, the non-paired term $\mathbf{T^0_{11}}$ is considered as a 1-level term to sum it with $\mathbf{T^1_9}$). The sharing group $\mathbf{T^4_{3,4}}$ are also represented in the figure with a double gray circle.

Time complexity of this $GF(2^{13})$ multiplier can be computed taking into account that the most complex coefficient c_4 requires a 6-level binary XOR tree, so the delay is given by $T_{\rm A} + 6T_{\rm X}$. The $T_{\rm A}$ delay corresponds to the 0-level a_ib_j products of the coefficients of A and B. For area complexity, the number of 2-input AND and XOR gates must be computed. The number of AND gates is given by the products a_ib_j, with $i, j \in [0, m-1]$, and for $GF(2^m)$ multipliers is m^2 [16]. Therefore, the number of AND gates is 169 for the $GF(2^{13})$ multiplier. The number of XOR gates can be computed as the sum of XOR gates in the initial $\mathbf{S^j_i}$ and $\mathbf{T^j_i}$ terms (as given in Table 4) plus the number of new XOR gates generated in the coefficients (as given in Table 5) minus the number of XOR gates due to shared groups. The $\mathbf{S^j_i}$ and $\mathbf{T^j_i}$ terms perform the XOR of 2^j product terms, so the number of XORs is $2^j - 1$. In this example, there are 7 $\mathbf{S^0_i}$ terms and 6 $\mathbf{S^1_i}$, $\mathbf{S^2_i}$ and $\mathbf{S^3_i}$ terms, so the number of XOR gates in the initial $\mathbf{S^j_i}$ terms will be $7 \cdot (2^0 - 1) + 6 \cdot (2^1 - 1) + 6 \cdot (2^2 - 1) + 6 \cdot (2^3 - 1) = 66$ XOR. There are also 6 $\mathbf{T^0_i}$ and $\mathbf{T^1_i}$ terms and 5 $\mathbf{T^2_i}$ and $\mathbf{T^3_i}$ terms, so the number of XORs in the initial $\mathbf{T^j_i}$ terms is $6 \cdot (2^0 - 1) + 6 \cdot (2^1 - 1) + 5 \cdot (2^2 - 1) + 5 \cdot (2^3 - 1) = 56$ XOR. The number of new XORs generated in the coefficients due to the addition of $\mathbf{S^j_i}$ and $\mathbf{T^j_i}$ terms is found to be 110 (see Table 5). The number of XORs due to the shared groups were previously computed (8 XOR). Therefore, the total number of XOR gates of this multiplier is 66 + 56 + 110 − 8 = 224 XOR.

4.2 Complexity Analysis of the New Multiplier

4.2.1 Time Complexity

In Section 3.2 was found that c_{n+1} is the most complex coefficient, so it is used to determine the delay of the new multiplier. To do that, the complexity of $\mathbf{S_i}$ and $\mathbf{T_i}$ terms must be determined. As shown in Section 4.1, the number of initial terms $\mathbf{S^j_i}$ and $\mathbf{T^j_i}$ are given by the binary representations of the subindex i for $\mathbf{S_i}$ and by the value $m-1-i$ for $\mathbf{T_i}$, respectively [16]. Therefore, the equivalence (only in relation to the number of terms) $\mathbf{T_i} \equiv \mathbf{S_{m-1-i}}$ can be used to determine the number of $\mathbf{T^j_i}$ terms in $\mathbf{T_i}$. This equivalence determines that, for c_{n+1}, $\mathbf{T_0} \equiv \mathbf{S_{m-1}}$, $\mathbf{T_1} \equiv \mathbf{S_{m-2}}$, $\mathbf{T_n} \equiv \mathbf{S_{z-1}}$, $\mathbf{T_{n+1}} \equiv \mathbf{S_{z-2}}$, $\mathbf{T_{z-1}} \equiv \mathbf{S_n}$ and $\mathbf{T_{z+1}} \equiv \mathbf{S_{n-2}}$, where $z = m - n$, so

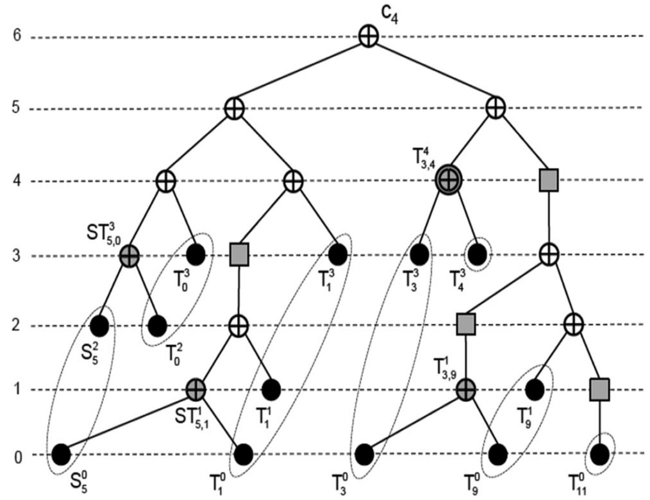

Fig. 1. Implementation of coefficient c_4 for $GF(2^{13})$.

$c_{n+1} \equiv \mathbf{S_{n+2}} + \mathbf{S_{m-1}} + \mathbf{S_{m-2}} + \mathbf{S_{z-1}} + \mathbf{S_{z-2}} + \mathbf{S_n} + \mathbf{S_{n-2}}$. In order to determine the binary representation of these subindexes, the expression $q = \sum_{i=0}^{\lfloor log_2 q \rfloor} (\lfloor q/2^i \rfloor mod 2) \cdot 2^i$ giving the binary configuration of a number q can be used [16]. The value $\lfloor q/2^i \rfloor \ mod \ 2$ determines if the binary representation of q has a 1 in the position with weight 2^i, representing that $\mathbf{S_q}$, $\mathbf{T_{m-1-q}}$ have a term $\mathbf{S_q^i}$, $\mathbf{T_{m-1-q}^i}$ that is the addition of 2^i product terms and that is implemented with a binary XOR tree of depth i. The depth of the binary XOR tree implementing c_{n+1} can be determined by first computing the total number of terms in $\lfloor log_2 m \rfloor$-level. The initial levels are $0, 1, \ldots, \lfloor log_2 m \rfloor$. For a given level i, the number of new XOR terms created in level $i+1$ due to the sum in pairs of i-level terms is $\lceil M_i/2 \rceil$, where M_i denotes the number of terms in level i. For instance, in Fig. 1 there are five 0-level terms ($\mathbf{S_5^0}$, $\mathbf{T_1^0}$, $\mathbf{T_3^0}$, $\mathbf{T_9^0}$, $\mathbf{T_{11}^0}$) and their sum results in the three 1-level terms $\mathbf{ST_{5,1}^1}$, $\mathbf{T_{3,9}^1}$ and $\mathbf{T_{11}^0}$ (considered as 1-level term to be XORed to $\mathbf{T_9^1}$). The number of initial terms $\mathbf{S_i^j}$ and $\mathbf{T_i^j}$ in level j, denoted as μ_j, is given by the binary representations of the subindexes of the $\mathbf{S_i}$ terms included in $c_{n+1} \equiv \mathbf{S_{n+2}} + \mathbf{S_{m-1}} + \mathbf{S_{m-2}} + \mathbf{S_{z-1}} + \mathbf{S_{z-2}} + \mathbf{S_n} + \mathbf{S_{n-2}}$. Denoting $q_j^* = \lfloor q/2^j \rfloor \ mod \ 2$, then μ_j can be computed as [16] $\mu_j = (m-1)_j^* + (m-2)_j^* + (z-1)_j^* + (z-2)_j^* + (n+2)_j^* + n_j^* + (n-2)_j^*$. Using this expression, the number of initial terms for the most complex coefficient c_4 given in Section 4.1 can be computed. The number of initial terms in level 3, for example, will be $\mu_3 = (12)_3^* + (11)_3^* + (9)_3^* + (8)_3^* + (5)_3^* + (3)_3^* + (1)_3^* = 1+1+1+1+0+0+0 = 4$ corresponding to $\mathbf{T_0^3} \equiv \mathbf{S_{12}^3}$, $\mathbf{T_1^3} \equiv \mathbf{S_{11}^3}$, $\mathbf{T_3^3} \equiv \mathbf{S_9^3}$, $\mathbf{T_4^3} \equiv \mathbf{S_8^3}$ (black circles in Fig. 1).

The total number of terms $M_{\lfloor log_2 m \rfloor}$ in the $\lfloor log_2 m \rfloor$-level is the addition of initial terms $\mu_{\lfloor log_2 m \rfloor}$ plus the terms in that level created by the XOR of terms in lower levels. Using the property of *modulo* operation $\lceil q \rceil + n = \lceil q+n \rceil$, with n integer, and having into account that the total number of terms in level i is $M_i = \mu_i + \lceil \mu_{i-1}/2 \rceil$, then it can be proved [16] that the terms created in level $\lfloor log_2 m \rfloor$ due to the addition in pairs of terms in level $\lfloor log_2 m \rfloor - 1$ is $\lceil (\sum_{i=0}^{\lfloor log_2 m \rfloor - 1} 2^i \mu_i)/2^{\lfloor log_2 m \rfloor} \rceil$. Therefore, the total number of terms in $\lfloor log_2 m \rfloor$-level will be the sum of $\mu_{\lfloor log_2 m \rfloor}$ plus the above expression, i.e., $M_{\lfloor log_2 m \rfloor} = \lceil (\sum_{i=0}^{\lfloor log_2 m \rfloor} 2^i \mu_i)/2^{\lfloor log_2 m \rfloor} \rceil$. In order to compute this expression, the number μ_j of initial terms in level j should be known. This number was previously given for c_{n+1}. Using the fact that *mod* operator is defined by $x \ mod \ y = x - y \lfloor x/y \rfloor$, for real $x, y \ (y \neq 0)$, then $q_j^* = \lfloor q/2^j \rfloor - 2 \cdot \lfloor \lfloor q/2^j \rfloor /2 \rfloor = \lfloor q/2^j \rfloor - 2 \cdot \lfloor q/2^{j+1} \rfloor$. Therefore, it can be proved that [16]

$$M_{\lfloor log_2 m \rfloor} = \left\lceil \frac{4m+n-6}{2^{\lfloor log_2 m \rfloor}} \right\rceil. \quad (2)$$

The number of XOR levels needed to compute the coefficient c_{n+1} will be $\lfloor log_2 m \rfloor + \lceil log_2 M_{\lfloor log_2 m \rfloor} \rceil$, so the highest delay of the multiplier based on type I pentanomials is

$$T_A + \left(\lfloor log_2 m \rfloor + \left\lceil log_2 \left\lceil \frac{4m+n-6}{2^{\lfloor log_2 m \rfloor}} \right\rceil \right\rceil \right) T_X. \quad (3)$$

4.2.2 Area Complexity

From (1), the number of AND gates in $\mathbf{S_i}$ and $\mathbf{T_i}$ are i and $m-i-1$, respectively. Therefore, the total number of AND gates of the multiplier is m^2. In order to compute the number of XOR gates of the multiplier, the number of XORs ① given by $\mathbf{S_i}$ and $\mathbf{T_i}$ must be determined. From (1), these values are $i-1$ and $m-i-1$, respectively, and therefore ① is $(m-1)^2$. The number ② of XOR gates used for the addition of $\mathbf{S_i}$ and $\mathbf{T_i}$ terms in the product coefficients of Table 3 must also be computed. Functions $\mathbf{S_i}$ appear only once in Table 3 while that $\mathbf{T_i}$ terms appear several times. Therefore, the number of XORs in (1) determines the XORs of $\mathbf{S_i^j}$ and $\mathbf{T_i^j}$ terms, the XORs used for the addition of all $\mathbf{S_i^j}$ terms of $\mathbf{S_i}$ and the XORs given in one sum of $\mathbf{T_i^j}$ terms of $\mathbf{T_i}$. If a term $\mathbf{T_i}$ appears p_i times in Table 3, then the other $p_i - 1$ occurrences are taken into account determining the number $\mathbf{\Theta}_i$ of XORs needed for the addition of the $\mathbf{T_i^j}$ terms and multiplying it by $p_i - 1$. Therefore, the XOR gates ③ given by $\sum_{i=0}^{m-2} (p_i - 1) \cdot \mathbf{\Theta}_i$ must also be computed. Finally, the number ④ of XORs given by shared groups $(\mathbf{T_i}, \mathbf{T_j})$ should also be determined. The total number of XOR gates of the multiplier will then be ① + ② + ③ − ④ [16]. In Appendix the following values have been computed:

- The number ② of XORs needed for the sum of $\mathbf{S_i}$ and $\mathbf{T_i}$ terms in product coefficients is $4m + 2n - 3$.
- The number ③ of XOR gates can be given as $\sum_{i=0}^{m-2} (p_i - 1) \cdot \mathbf{\Theta}_i = 3 \cdot \Upsilon_{m-1} + 2 \cdot \Upsilon_n + \Psi_n$. In this expression, the number of XOR gates Ψ_n needed for the sum of the $\mathbf{S_n^j}$ terms of $\mathbf{S_n}$ is $\Psi_n = H_n - 1$ [16], where H_n is the *Hamming Weight* of n, and $\Upsilon_h = \sum_{i=1}^{h} \Psi_i = \sum_{i=1}^{h} H_i - h$.
- The number ④ of XOR gates given by shared groups $(\mathbf{T_i}, \mathbf{T_j})$ in the product coefficients is $(\sum_{i=\kappa,\kappa+2,\ldots}^{\iota} H_i) + H_{n-1}^{\dagger} = \Delta_n + H^{\dagger}$, where $\kappa = 2\lfloor n/2 \rfloor$, $\iota = (m-2)$ for *even* m and $(m-3)$ for *odd* m, and † represents that H only appears for *odd* n.

Therefore, the number of XOR gates of the multiplier given by ① + ② + ③ − ④ will be

$$m^2 - m + 3\Sigma_{m-1} + 2\Sigma_n + H_n - \Delta_n - H^{\dagger}, \quad (4)$$

where $\Sigma_h = \sum_{i=1}^{h} H_i$. Using (4), for the example given in Section 4.1 with $m = 13$, $n = 3$, the values $\Sigma_{12} = 22$, $\Sigma_3 = 4$, $H_3 = 2$, $\Delta_3 = 7$ and $H_2 = 1$. Applying these values to equation (4) we obtain $169 - 13 + 3 \cdot 22 + 2 \cdot 4 + 2 - 7 - 1 = 224$ XOR gates, matching the result given in Section 4.1.

5 Comparison with Other Multipliers

In Table 6 theoretical complexities of the multiplier here proposed are compared with the best results known to date for bit-parallel PB multipliers over $GF(2^m)$ generated by type I irreducible pentanomials. Simulations done with *Maple* have proven that the delay of our multiplier is less than or equal to the best delay given in [11] and [15], i.e., $\lfloor log_2 m \rfloor + \lceil log_2 \lceil (4m+n-6)/2^{\lfloor log_2 m \rfloor} \rceil \rceil \leq 3 + \lceil log_2 (m-1) \rceil$. From these results, it was found that for the 807 different values of the field size $m \in [8, 1000]$ for which a type I pentanomial exists, the proposed multiplier has the smallest delay in 762 different values of m. Furthermore, among the 1974 (m, n) combinations with $m \in [8, 1000]$ for which type I pentanomials exist, there are 187 and 1787 different pairs (m, n) for which the proposed multiplier has equal and less delay, respectively, than the multipliers given in [11]

TABLE 6
Complexities of Bit-Parallel PB Multipliers for Type I Pentanomial $f(y) = y^m + y^{n+1} + y^n + y + 1$

	#AND	#XOR	Delay
[13]	m^2	$m^2 + 2m - 3$	$T_A + (6 + \lceil log_2 m \rceil)T_X$
[18]	$\frac{3m^2+2m-1}{4}$	$\frac{3m^2+24m+8n+\delta}{4}$	$T_A + (3 + \lceil log_2(m+1) \rceil)T_X$
[14]	m^2	$m^2 + m + 2n$	$T_A + (3 + \lceil log_2(m) \rceil)T_X$
[11]	m^2	$m^2 + m$	$T_A + (3 + \lceil log_2(m-1) \rceil)T_X$
[15]	m^2	$m^2 + m - 1$	$T_A + (3 + \lceil log_2(m-1) \rceil)T_X$
This work	m^2	$m^2 - m + 3\Sigma_{m-1} + 2\Sigma_n + H_n - \Delta_n - H^{\dagger}$	$T_A + \left(\lfloor log_2 m \rfloor + \left\lceil log_2 \left\lceil \frac{4m+n-6}{2^{\lfloor log_2 m \rfloor}} \right\rceil \right\rceil\right)T_X$

$\delta = 21$ *(odd n), 17 (even n).* † = *term included for* odd n.

and [15]. With respect to area complexity, the proposed multiplier presents equal number of AND gates (except for [18]) and a higher number of XOR gates. This increased number is due to the splitting of functions $\mathbf{S_i}$ and $\mathbf{T_i}$ into $\mathbf{S_i^j}$ and $\mathbf{T_i^j}$ terms, respectively. In Table 7 the complexities of bit-parallel PB multipliers for NIST recommended $GF(2^m)$, with $m \in \{163, 233, 283\}$, for which type I irreducible pentanomials exist are presented. It can be observed that the multiplier here proposed presents the lowest delay among the different analyzed methods. These reductions range from 8.3 percent for $GF(2^{283})$ to 9.1 percent for $GF(2^{163})$ and $GF(2^{233})$ with respect to the best delays found in the literature.

6 Hardware Implementations

In order to further compare the new approach with other similar methods, bit-parallel $GF(2^m)$ PB multipliers based on type I irreducible pentanomials have been described in VHDL, synthesized and implemented on Xilinx FPGA Artix-7 XC7A200T-FFG1156. Experimental results are those reported by Xilinx ISE 14.7 using XST synthesizer. Furthermore, same pin assignments and *speed high* optimizations have been part of the design methodology. Experimental post-place and route results are given in Table 8 for multipliers based on type I irreducible pentanomials for SECG [20] recommended finite fields $GF(2^m)$, with (m, n) = (113, 8), (113, 24), (113, 40), (131, 59), and for NIST (163, 59). Area complexity is expressed in Table 8 in terms of the used number of *LUTs* and *Slices*, and time results (in nanoseconds) represent the minimum time needed for performing one $GF(2^m)$ multiplication. The $A \times T$ metrics express area by time delay in $Slices \times ns$ in order to compare the area and delay (less is better). From the experimental results, it can be observed that the new multiplier here proposed exhibits the lowest delay among the different methods. Moreover, the new approach presents the best $Area \times Time$ values in three of the five implemented multipliers, therefore also showing a restrained area usage in comparison with other methods.

TABLE 7
Complexities of Bit-Parallel PB Multipliers Using Type I Pentanomials for Three Recommended NIST Fields

	#AND	#XOR	Delay	(m, n)
[13]	26,569	26,892	$T_A + 14T_X$	
[18]	20,008	21,028	$T_A + 11T_X$	
[14]	26,569	26,850	$T_A + 11T_X$	(163, 59)
[11]	26,569	26,732	$T_A + 11T_X$	
[15]	26,569	26,731	$T_A + 11T_X$	
This work	26,569	28,280	$T_A + 10T_X$	
[13]	54,289	54,752	$T_A + 14T_X$	
[18]	40,833	42,170	$T_A + 11T_X$	
[14]	54,289	54,572	$T_A + 11T_X$	(233, 25)
[11]	54,289	54,522	$T_A + 11T_X$	
[15]	54,289	54,521	$T_A + 11T_X$	
This work	54,289	56,471	$T_A + 10T_X$	
[13]	80,089	80,652	$T_A + 15T_X$	
[18]	60,208	61,888	$T_A + 12T_X$	
[14]	80,089	80,490	$T_A + 12T_X$	(283, 59)
[11]	80,089	80,372	$T_A + 12T_X$	
[15]	80,089	80,371	$T_A + 12T_X$	
This work	80,089	83,068	$T_A + 11T_X$	

7 Conclusion

In this paper, a new fast bit-parallel $GF(2^m)$ polynomial basis multiplier for type I irreducible pentanomials has been presented. Efficient implementations of high-speed multipliers over binary extension fields are highly desirable for several important applications. Furthermore, type I irreducible pentanomials are abundant and they are used in applications such as the AES. In this work, explicit expressions for the coordinates of the proposed multiplier are given. These expressions are implemented as the addition in pairs of binary trees of XOR gates with the same depth, leading to a reduction of delay. Moreover, the use of binary trees can minimize power consumption in comparison to the use of linear arrays of XOR gates. A detailed multiplication example has been also given. Theoretical complexity analysis has shown that the proposed multiplier presents the lowest delay among the best results known to date for similar

TABLE 8
Comparison of Hardware Implementations for $GF(2^m)$ Multipliers

	LUTS	*Slices*	*Time(ns)*	$A \times T$	(m, n)
[10]	5,554	2,882	24.74	71300.68	
[19]	5,515	2,851	23.18	66086.18	
[14]	5,434	2,718	22.89	62215.02	(113, 8)
[11]	5,427	2,571	21.23	54582.33	SECG
[15]	5,735	2,446	21.33	52173.18	
This work	5,501	2,354	20.56	48398.24	
[10]	5,529	2,727	21.99	59966.73	
[19]	5,528	2,824	22.24	62805.76	
[14]	5,436	2,406	21.32	51295.92	(113, 24)
[11]	5,435	2,546	22.15	56393.90	SECG
[15]	5,653	2,363	20.79	49126.77	
This work	5,460	2,466	20.34	50158.44	
[10]	5,533	2,662	22.10	58830.20	
[19]	5,524	2,746	22.60	62059.60	
[14]	5,455	2,488	21.13	52571.44	(113, 40)
[11]	5,431	2,508	21.15	53044.20	SECG
[15]	5,548	2,571	21.33	54839.43	
This work	5,481	2,459	20.37	50089.83	
[10]	7,423	2,743	23.07	63281.01	
[19]	7,383	2,671	23.76	63462.96	
[14]	7,308	2,168	20.95	45419.60	(131, 59)
[11]	7,286	2,287	22.96	52509.52	SECG
[15]	7,392	2,318	20.86	48353.48	
This work	7,341	2,185	20.33	44421.05	
[10]	11,412	3,852	24.29	93565.08	
[19]	11,364	4,060	24.19	98211.40	
[14]	11,290	3,664	21.79	79838.56	(163, 59)
[11]	11,304	3,730	22.62	84372.60	NIST
[15]	11,471	3,209	22.78	73101.02	
This work	11,320	3,532	21.33	75337.56	

multipliers based on this type of irreducible pentanomials. Simulation results have proven that for the 1,974 (m,n) combinations, with $m \in [8, 1000]$ and $2 \leq n \leq \lfloor m/2 \rfloor - 1$, for which type I irreducible pentanomials exist, there are 187 and 1,787 different pairs (m,n) for which the proposed multiplier has equal and less delay, respectively, than the best results found in the literature. Furthermore, for NIST recommended finite fields $GF(2^m)$ with $(m,n) = (163, 59)$, $(233, 25)$ and $(283, 59)$, the multiplier here proposed presents a reduction of the delay ranging from 8.3 to 9.1 percent with respect to the best results known to date. In order to prove the theoretical complexities, hardware implementations over Xilinx FPGAs have also been performed. NIST and SECG $GF(2^m)$ multipliers have been described in VHDL and post-place and route implementation results in Artix-7 have been reported. Experimental results have shown that the proposed multiplier exhibits the lowest delay with a balanced $Area \times Time$ complexity when compared with similar multipliers.

Appendix

Area Complexity

In order to determine the number of XOR gates of the multiplier, the quantities ②, ③ and ④ must be computed.

The coefficients in Table 3 have been divided into eight sections. The number of $\mathbf{S_i}$ and $\mathbf{T_i}$ terms in each section was given in Section 3.2. The XOR gates in the product coefficients are the following: 3 in section Ⓐ; $4(n-2)$ in Ⓑ; 3 and 6 in Ⓒ and Ⓓ, respectively; $6(n-2)$ in Ⓔ; 10 in Ⓕ; $4(m-2n-2)$ in Ⓖ and finally 3 in Ⓗ. Therefore, the number ② of XOR gates needed for the addition of $\mathbf{S_i}$ and $\mathbf{T_i}$ terms in the product coefficients is $4m + 2n - 3$.

The number p_i of times each $\mathbf{T_i}$ appears in Table 3 must be determined to compute the number ③ of XOR gates. There are $z-1$ terms $(\mathbf{T_0}, \ldots, \mathbf{T_{z-2}})$ that appear 4 times, $n-2$ terms $(\mathbf{T_{z+1}}, \ldots, \mathbf{T_{m-2}})$ that appear 6 times and $\mathbf{T_{z-1}}$ appears 7 times. One occurrence of $\mathbf{T_i}$ terms are already included in ①, so the number of XORs due to the above terms appearing 3, 5 and 6 times, respectively, must be determined. If we define $\Phi_{a,b} = \sum_{i=a}^{b} \Theta_i$, where Θ_i is the number of XORs needed for the sum of the $\mathbf{T_i^j}$ terms for $\mathbf{T_i}$, then the number ③ of XORs is $\sum_{i=0}^{m-2}(p_i - 1) \cdot \Theta_i = 3 \cdot \Phi_{0,m-2} + 2 \cdot \Phi_{z-1,m-2} + \Theta_{z-1}$. Using the equivalence $\mathbf{T_i} \equiv \mathbf{S_{m-1-i}}$ and denoting $\Upsilon_h = \sum_{i=1}^{h} \Psi_i$, then ③ can be computed as $\sum_{i=0}^{m-2}(p_i - 1) \cdot \Theta_i = 3 \cdot \Upsilon_{m-1} + 2 \cdot \Upsilon_n + \Psi_n$. The XORs Ψ_i needed for the addition of $\mathbf{S_i^j}$ terms in $\mathbf{S_i}$ can be determined using the number of 1's in the binary configuration of i [16]. For example, $\mathbf{S_{11}}$ in Table 4 can be written as $\mathbf{S_{11}} = \mathbf{S_{11}^3} + \mathbf{S_{11}^1} + \mathbf{S_{11}^0}$ and therefore 2 XORs are needed to perform the additions of $\mathbf{S_{11}^j}$ terms. Binary configuration of subindex 11 is $(1,0,1,1)_2$, with three 1's, so the number of XOR gates Ψ_{11} will be its *Hamming Weight* H_{11} minus 1. Therefore, $\Psi_i = H_i - 1$ and $\Upsilon_h = \sum_{i=1}^{h} \Psi_i = \sum_{i=1}^{h} H_i - h$.

Table 3 is used to compute the number of XOR gates given by shared groups $(\mathbf{T_i}, \mathbf{T_j})$. It can be found that for *even* n, there are $\lfloor z/2 \rfloor$ groups $(\mathbf{T_i}, \mathbf{T_{i+1}})$, $i \in [0, z-2]$ for *even* m and $i \in [1, z-2]$ for *odd* m, that appear in two coefficients. For *odd* n, the group $(\mathbf{T_{z-1}}, \mathbf{T_z})$ appears in three coefficients while that $\lceil z/2 \rceil - 1$ groups $(\mathbf{T_i}, \mathbf{T_{i+1}})$, $i \in [0, z-3]$ for *even* m and $i \in [1, z-3]$ for *odd* m, appear in two coefficients. For these groups, the term with highest subindex gives the XORs to be shared [16]. The XORs represented by the above groups are therefore given by the Hamming Weight of binary representation of the lowest subindex i of $\mathbf{S_i}$ for each group. The quantities to be computed are $(H_n + H_{n+2} + \cdots + H_{m-2})$ for *even* n and m, $(H_n + H_{n+2} + \cdots + H_{m-3})$ for *even* n and *odd* m, $(H_{n-1} + H_{n+1} + \cdots + H_{m-2}) + H_{n-1}$ for *odd* n and *even* m, and $(H_{n-1} + H_{n+1} + \cdots + H_{m-3}) + H_{n-1}$ for *odd* n and m. Therefore the number ④ of XORs given by the shared groups will be $(\sum_{i=\kappa,\kappa+2,\ldots}^{\iota} H_i) + H_{n-1}^{\dagger} = \Delta_n + H^{\dagger}$, where $\kappa = 2\lfloor n/2 \rfloor$, $\iota = (m-2)$ for *even* m and $(m-3)$ for *odd* m, and † represents that H only appears for *odd* n.

Acknowledgments

This work has been supported by the EU (FEDER) and the Spanish MINECO, under grants TIN 2015-65277-R and TIN2012-32180.

References

[1] J. Lin, J. Sha, Z. Wang, and L. Li, "Efficient decoder design for nonbinary quasicyclic LDPC codes," *IEEE Trans. Circuits Syst. I Reg. Papers*, vol. 57, no. 5, pp. 1071–1082, May 2010.

[2] K. Kobayashi and N. Takagi, "A combined circuit for multiplication and inversion in $GF(2^m)$," *IEEE Trans. Circuits Syst. II Exp. Briefs*, vol. 55, no. 11, pp. 1144–1148, Nov. 2008.

[3] B. Sunar and Ç.K. Koç, "Mastrovito multiplier for all trinomials," *IEEE Trans. Comput.*, vol. 48, no. 5, pp. 522–527, May 1999.

[4] H. Wu, "Bit-parallel finite field multiplier and squarer using polynomial basis," *IEEE Trans. Comput.*, vol. 51, no. 7, pp. 750–758, Jul. 2002.

[5] H. Fan, "A chinese remainder theorem approach to bit-parallel $GF(2^m)$ polynomial basis multipliers for irreducible trinomials," *IEEE Trans. Comput.*, vol. 65, no. 2, pp. 343–352, Feb. 2016.

[6] H. Wu, "Bit-parallel polynomial basis multiplier for new classes of finite fields," *IEEE Trans. Comput.*, vol. 57, no. 8, pp. 1023–1031, Aug. 2008.

[7] A. Halbutogullari and Ç. K. Koç, "Mastrovito multiplier for general irreducible polynomials," *IEEE Trans. Comput.*, vol. 49, no. 5, pp. 503–518, May 2000.

[8] H. Fan and Y. Dai, "Fast bit parallel $GF(2^m)$ multiplier for all trinomials," *IEEE Trans. Comput.*, vol. 54, no. 4, pp. 485–490, Apr. 2005.

[9] P. K. Meher, "Systolic and super-systolic multipliers for finite field $GF(2^m)$ based on irreducible trinomials," *IEEE Trans. Circuits Syst. I Reg. Papers*, vol. 55, no. 4, pp. 1031–1040, May 2008.

[10] E. D. Mastrovito, "VLSI architectures for multiplication over finite fields $GF(2^m)$," in *Proc. 6th Int'l Conf. Appl. Algebra Algebraic Algorithms Error-Correcting Codes*, Jul. 1988, pp. 297–309.

[11] A. Reyhani-Masoleh and M. A. Hasan, "Low complexity bit parallel architectures for polynomial basis multiplication over $GF(2^m)$," *IEEE Trans. Comput.*, vol. 53, no. 8, pp. 945–959, Aug. 2004.

[12] J. L. Imaña, R. Hermida and F. Tirado, "Low complexity bit-parallel polynomial basis multipliers over binary fields for special irreducible pentanomials," *Integration*, vol. 46, pp. 197–210, 2013.

[13] T. Zhang and K. K. Parhi, "Systematic design of original and modified mastrovito multipliers for general irreducible polynomials," *IEEE Trans. Comput.*, vol. 50, no. 7, pp. 734–749, Jul. 2001.

[14] F. Rodríguez-Henríquez and Ç.K. Koç, "Parallel multipliers based on special irreducible pentanomials," *IEEE Trans. Comput.*, vol. 52, no. 12, pp. 1535–1542, Dec. 2003.

[15] J. L. Imaña, R. Hermida, and F. Tirado, "Low complexity bit-parallel multipliers based on a class of irreducible pentanomials," *IEEE Trans. VLSI Syst.*, vol. 14, no. 12, pp. 1388–1393, Dec. 2006.

[16] J. L. Imaña, "High-speed polynomial basis multipliers over $GF(2^m)$ for special pentanomials," *IEEE Trans. Circuits Syst. I Reg. Papers*, vol. 63, no. 1, pp. 58–69, Jan. 2016.

[17] L. Song and K. K. Parhi, "Low-energy digit-serial/parallel finite field multipliers," *J. VLSI Signal Process.*, vol. 19, pp. 149–166, 1998.

[18] S.-M. Park, K.-Y. Chang, D. Hong, and C. Seo, "New efficient bit-parallel polynomial basis multiplier for special pentanomials," *Integration*, vol. 47, pp. 130–139, 2014.

[19] B. Rashidi, R. R. Farashahi, and S. M. Sayedi, "Efficient implementation of low time complexity and pipelined bit-parallel polynomial basis multiplier over binary finite fields," *Int. J. Inform. Security*, vol. 7, no. 2, pp. 101–114, Jul. 2015.

[20] Certicom Research, "SEC 2: Recommended Elliptic Curve Domain Parameters", Standards for Efficient Cryptography". Version 1.0, Sep. 2000.