Integration, the VLSI Journal 75 (2020) 114–121

Contents lists available at ScienceDirect

Integration, the VLSI Journal

journal homepage: www.elsevier.com/locate/vlsi

High-throughput architecture for post-quantum DME cryptosystem

José L. Imaña [a,*], Ignacio Luengo [b]

[a] Department of Computer Architecture and Automation, Complutense University, 28040, Madrid, Spain
[b] Department of Algebra, Geometry and Topology, Complutense University, 28040, Madrid, Spain

ARTICLE INFO

Keywords:
Post-quantum cryptography
Multivariate public-key cryptosystem
DME
Finite field
Field-Programmable Gate Array (FPGA)
Pipelined
High-throughput

ABSTRACT

Quantum computers have the potential to solve difficult mathematical problems efficiently, therefore meaning an important threat to Public-Key Cryptography (PKC) if large-scale quantum computers are ever built. The goal of Post-Quantum Cryptography (PQC) is to develop cryptosystems that are secure against both classical and quantum computers. DME is a new proposal of quantum-resistant PKC algorithm that was presented for NIST PQC Standardization competition in order to set the next-generation of cryptography standards. DME is a multivariate public key, signature and Key Encapsulation Mechanism (KEM) system based on a new construction of the central maps, that allows the polynomials of the public key to be of an arbitrary degree. In this paper, a high-throughput pipelined architecture of DME is presented and hardware implementations over Xilinx FPGAs have been performed. Experimental results show that the architecture here presented exhibits the lowest execution time and highest throughput when it is compared with other PQC multivariate implementations given in the literature.

1. Introduction

The rapid development of quantum computing constitutes a significant threat to modern Public-Key Cryptography (PKC). The use of Shor's algorithm [1] with potential powerful quantum computers could easily break the two most widely used public key cryptosystems, namely, RSA and Elliptic Curve Cryptography (ECC), based on integer factorization and discrete logarithm problems. For this reason, Post-Quantum Cryptography (PQC) [2] based on alternative mathematical features has become a fundamental research topic due to its resistance against quantum computers. The National Institute of Standards and Technology (NIST) has even opened a call for proposals of quantum-resistant PKC algorithms in order to standardize one or more PQC algorithms. Cryptographic systems that appear to be extremely difficult to break with large quantum computers are *hash-based cryptography* [3], *lattice-based cryptography* [4], *code-based cryptography* [5], and *multivariate-quadratic cryptography* [6]. Furthermore, efficient hardware implementations are required for these alternative quantum-resistant cryptosystems [7–9].

Multivariate Public-Key Cryptosystems (MPKCs) are cryptosystems for which the public key is a set of polynomials $P(X) = (p_1, \ldots, p_m)$ in variables $X = (x_1, \ldots, x_n)$ where all the variables and coefficients are in a finite field. Their security relies on the difficulty of the problem of solving a set of multivariable quadratic polynomial equations over $GF(q)$, which is in general NP-hard. Different schemes of MPKCs have been proposed in the literature [10–12]. TTS (*Tame Transformation Signature*) schemes, such as *amended* TTS (*am*TTS) [13] and *enhanced* TTS (*en*TTS) [14] were based on the *Tame Transformation Method* (TTM) [15]. The Oil-Vinegar family of MPKCs consists of three families, named balanced Oil-Vinegar, unbalanced Oil-Vinegar (UOV) and Rainbow [16], that is a multilayer construction using UOV at each layer. Rainbow has great potential in terms of its efficiency and applications in ubiquitous computing [16]. Furthermore, Rainbow is a candidate for the NIST PQC Standardization Process that has moved on to the second round of the competition.

DME [17] is a new multivariate proposal of quantum-resistant public key, signature and Key Encapsulation Mechanism (KEM) system based on a double exponentiation with matrix exponents that was presented for NIST PQC Standardization competition. This new PKC system uses a new construction of the central maps, that allow the polynomials of the public key to be of an arbitrary degree. In order to get a reasonable size for the public key, a small number of variables and special non-dense linear maps must be used at both ends of the composition. DME-(m, n, e), with parameters m, n and e, is very new and it has not yet been completely studied. Although DME-(3,2,48) presented to NIST could not move on to the second round due to security issues, it is believed that a different selection of parameters such as (4,2,48) can

* Corresponding author.
E-mail addresses: jluimana@ucm.es (J.L. Imaña), iluengo@ucm.es (I. Luengo).

https://doi.org/10.1016/j.vlsi.2020.07.002
Received 29 April 2020; Received in revised form 2 July 2020; Accepted 4 July 2020
Available online 2 August 2020

make it secure against standard attacks. Several candidates presented to NIST PQC competition were based on previous versions of the same algorithms, so the study of DME would be very important for the possibility to present it to future competitions. Furthermore, DME is better than other PQC candidates in aspects such as the encrypted message size, exhibiting the lowest size of encrypted text for the same security level. For these reasons, DME and its hardware implementations have important research values. In this paper, a high-throughput pipelined architecture for DME-(3,2,48), the reference implementation presented for the NIST PQC competition, is proposed. An important characteristic is that this architecture is valid for any selection of DME parameters. Hardware implementations of DME-(3,2,48) over Xilinx FPGA have been performed. Experimental results show that the architecture here presented exhibits the lowest execution time and highest throughput when it is compared with similar PQC implementations given in the literature. This high performance makes the study and implementation of DME of great interest.

The paper is organized as follows. Section 2 provides the basis of the new DME cryptosystem. Section 3 gives the parameters used for DME-(3,2,48), the reference implementation presented for the NIST PQC proposal. The new pipelined architecture for DME is presented in Section 4. FPGA implementations and experimental results are given in Section 5, where performance comparison with other PQC multivariate implementations are also given. Finally, conclusions are described in Section 6.

2. DME cryptosystem

Let $\mathbb{F}_q = GF(q)$ be a finite field with $q = 2^e$ elements and let n, m, with $2 \leq n < m$, be fixed integers. DME cryptosystem [17], denoted as DME-(m,n,e), is based on a polynomial map $P : \mathbb{F}_q^{nm} \to \mathbb{F}_q^{nm}$ where P is the composition of five maps, $P = L_3 \circ G_2 \circ L_2 \circ G_1 \circ L_1$, according to (1).

$$\begin{array}{ccccccc} \mathbb{F}_q^{nm} & \xrightarrow{L_1} & (\mathbb{F}_{q^n})^m & \xrightarrow{G_1} & (\mathbb{F}_{q^n})^m & & \\ P\big(& & & & & \big\downarrow L_2 & \\ \mathbb{F}_q^{mn} & \xleftarrow{L_3} & (\mathbb{F}_{q^m})^n & \xleftarrow{G_2} & (\mathbb{F}_{q^m})^n & & \end{array} \tag{1}$$

The map $L_1 = \widetilde{\pi}_1 \circ \widetilde{L}_1 \circ \widetilde{l}$ is given by the composition of three linear $\mathbb{F}_q$-isomorphism according to the diagram (2).

$$\begin{array}{ccccccc} \mathbb{F}_q^{nm} & \xrightarrow{\widetilde{l}} & (\mathbb{F}_q^n)^m & \xrightarrow{\widetilde{L}_1} & (\mathbb{F}_q^n)^m & \xrightarrow{\widetilde{\pi}_1} & (\mathbb{F}_{q^n})^m \\ & & & \underset{L_1}{\searrow} & & & \end{array} \tag{2}$$

The isomorphism $\widetilde{l}$ is obtained by grouping the components of the input x in m vectors according to its index, i.e., $\widetilde{l}(x_1, \ldots, x_{nm}) = (\underline{x}_1, \ldots, \underline{x}_m)$, where $\underline{x}_i = (x_{i1}, \ldots, x_{in})$. The isomorphism $\widetilde{L}_1 = (L_{11}, \ldots, L_{1m})$ is defined by its components $L_{1i} : \mathbb{F}_q^n \to \mathbb{F}_q^n$ given by $L_{1i}(\underline{x}_i) = \underline{x}_i A_{1i}$, where $A_{1i} \in \mathscr{M}_{n\times n}(\mathbb{F}_q)$ and $\det(A_{1i}) \neq 0$. Finally, the map $\widetilde{\pi}_1 = (\pi_1, \ldots, \pi_1)$ is defined by the $\mathbb{F}_q$-linear isomorphism $\pi_1 : \mathbb{F}_q^n \to \mathbb{F}_{q^n}$, $\pi_1(v_1, \ldots, v_n) = \alpha_1 v_1 + \cdots + \alpha_n v_n$, where $\{\alpha_1, \ldots, \alpha_n\}$ is a fixed $\mathbb{F}_q$ basis of $\mathbb{F}_{q^n}$.

The $\mathbb{F}_q$-linear isomorphism $L_2 = \widetilde{\pi}_2 \circ \widetilde{L}_2 \circ M \circ \widetilde{\pi}_1^{-1}$ is given by the composition according to the diagram (3).

$$\begin{array}{ccccccc} (\mathbb{F}_{q^n})^m & \xrightarrow{\widetilde{\pi}_1^{-1}} & (\mathbb{F}_q^n)^m & \xrightarrow{M} & (\mathbb{F}_q^m)^n & \xrightarrow{\widetilde{L}_2} & (\mathbb{F}_q^m)^n \\ & \underset{L_2}{\searrow} & & & & & \big\downarrow \widetilde{\pi}_2 \\ & & & & & & (\mathbb{F}_{q^m})^n \end{array} \tag{3}$$

The *mixing* isomorphism M transforms the m vectors of $\mathbb{F}_q^n$ in n vectors of $\mathbb{F}_q^m$ in such a way that the components of $\underline{x}_1, \ldots, \underline{x}_n$ are placed in the first n components of $\underline{x}'_1, \ldots, \underline{x}'_n$ and the components of $\underline{x}_{m-n+1}, \ldots, \underline{x}_m$ are placed in the last $m - n$ components of $\underline{x}'_1, \ldots, \underline{x}'_n$ [17]. The isomorphism $\widetilde{L}_2 = (L_{21}, \ldots, L_{2n})$ is defined (in a similar way as $\widetilde{L}_1$) by its components $L_{2i} : \mathbb{F}_q^m \to \mathbb{F}_q^m$ with $L_{2i}(\underline{x}_i) = \underline{x}_i A_{2i}$, where $A_{2i} \in \mathscr{M}_{m\times m}(\mathbb{F}_q)$ and $\det(A_{2i}) \neq 0$. The map $\widetilde{\pi}_1^{-1}$ is the inverse of $\widetilde{\pi}_1$ and $\widetilde{\pi}_2 = (\pi_2, \ldots, \pi_2)$ is defined by the isomorphism $\pi_2 : \mathbb{F}_q^m \to \mathbb{F}_{q^m}$, where $\pi_2(v_1, \ldots, v_m) = \alpha_1 v_1 + \cdots + \alpha_m v_m$, where $\{\alpha_1, \ldots, \alpha_m\}$ is a fixed $\mathbb{F}_q$ basis of $\mathbb{F}_{q^m}$.

The $\mathbb{F}_q$-linear isomorphism $L_3 = \widetilde{e}^{-1} \circ \widetilde{L}_3 \circ \widetilde{\pi}_2^{-1}$ is given by the composition according to the diagram (4).

$$\begin{array}{ccccccc} (\mathbb{F}_{q^m})^n & \xrightarrow{\widetilde{\pi}_2^{-1}} & (\mathbb{F}_q^m)^n & \xrightarrow{\widetilde{L}_3} & (\mathbb{F}_q^m)^n & \xrightarrow{\widetilde{e}^{-1}} & (\mathbb{F}_q)^{mn} \\ & \underset{L_3}{\searrow} & & & & & \end{array} \tag{4}$$

The isomorphism $\widetilde{L}_3 = (L_{31}, \ldots, L_{3n})$ is defined by its components $L_{3i} : \mathbb{F}_q^m \to \mathbb{F}_q^m$ with $L_{3j}(\underline{x}'_j) = \underline{x}'_j A_{3j}$, where $A_{3j} \in \mathscr{M}_{m\times m}(\mathbb{F}_q)$ and $\det(A_{3j}) \neq 0$. The map $\widetilde{\pi}_2^{-1}$ is the inverse of $\widetilde{\pi}_2$ and the map $\widetilde{e}^{-1}$ is the inverse of the isomorphism $\widetilde{e}$ that is obtained by grouping the components of x in n vectors according to its index, i.e., $\widetilde{e}(x_1, \ldots, x_{nm}) = (\underline{x}_1, \ldots, \underline{x}_n)$, where $\underline{x}_j = (x_{j1}, \ldots, x_{jm})$.

The exponential map $G_1 : (\mathbb{F}_{q^n})^m \to (\mathbb{F}_{q^n})^m$ is built as $G_1(u_1, \ldots, u_m) = \left(\left(u_1^{a_{11}} \cdot \ldots \cdot u_m^{a_{1m}}\right), \ldots, \left(u_1^{a_{m1}} \cdot \ldots \cdot u_m^{a_{mm}}\right)\right)$, where $A = (a_{ij}) \in \mathbb{Z}^{m\times m}$ is invertible modulo $q^n - 1$ (this is equivalent to $\gcd(\det(A), q^n - 1) = 1$) and the entries a_{ij} are either zero or powers of two (up to $q^n - 1$).

The exponential map $G_2 : (\mathbb{F}_{q^m})^n \to (\mathbb{F}_{q^m})^n$ is built as $G_2(w_1, \ldots, w_n) = ((w_1^{b_{11}} \cdot \ldots \cdot w_n^{b_{1n}}), \ldots, (w_1^{b_{n1}} \cdot \ldots \cdot w_n^{b_{nn}}))$, where $B = (b_{ij}) \in \mathbb{Z}^{n\times n}$ is invertible modulo $q^m - 1$ (equivalent to $\gcd(\det(B), q^m - 1) = 1$) and the entries b_{ij} are either zero or powers of two (up to $q^m - 1$).

If $\underline{x} = (x_{11}, \ldots, x_{nm}) \in \mathbb{F}_q^{nm}$ are the initial coordinates, then the composition of the five maps L_1, G_1, L_2, G_2, and L_3 allows the computation of the components of $P(\underline{x})$ as polynomials $P_i \in \mathbb{F}_q[x_{11}, \ldots, x_{nm}]$. The *secret key* consists of the three maps L_1, L_2 and L_3, which are chosen to be invertible. The *public key* consists of the nm polynomials in nm variables that define P. The security of the cryptosystem relies on the conjecture that it is very hard to recover the maps L_1, L_2 and L_3 from the polynomial expression of P, and that it is hard to compute $P^{-1}(y)$ for a given $y \in \mathbb{F}_q^{nm}$ without access to L_1, L_2 and L_3 [17].

3. DME-(3,2,48)

In this paper, a high-throughput pipelined implementation of DME-(3,2,48), with $m = 3$, $n = 2$, and $q = 2^e = 2^{48}$, is presented. These settings correspond with the reference implementation given in the NIST proposal [18].

The finite field $\mathbb{F}_{2^{48}} = GF(2^{48})$ is generated by the *type I irreducible pentanomial* [19] $f(y) = y^{48} + y^{28} + y^{27} + y + 1$ over $GF(2)$. The elements of the binary extension field $GF(2^{48})$ can be represented in the *polynomial basis* (PB) $\{1, x, \ldots, x^{m-1}\}$, where x is a root of the irreducible generating polynomial $f(y)$. Any element $C \in GF(2^{48})$ is represented in PB as $C = \sum_{i=0}^{47} c_i x^i$, where $c_i's \in GF(2)$ are the coefficients of C. The composite fields $\mathbb{F}_{(2^{48})^2} = GF((2^{48})^2)$ and $\mathbb{F}_{(2^{48})^3} = GF((2^{48})^3)$ are generated by the irreducible polynomials $f_2(T) = T^2 + aT + b$ and $f_3(S) = S^3 + cS^2 + dS + e$ over $\mathbb{F}_{2^{48}}$, respectively, where the elements $a, b, c, d, e \in GF(2^{48})$ were randomly chosen and given as [18]
$a = 1 + x + x^3 + x^4 + x^8 + x^9 + x^{13} + x^{19} + x^{20} + x^{21} + x^{22} + x^{23} + x^{24} + x^{25} + x^{26} + x^{29} + x^{34} + x^{36} + x^{38} + x^{43}$,
$b = 1 + x^2 + x^3 + x^7 + x^8 + x^9 + x^{14} + x^{16} + x^{17} + x^{18} + x^{21} + x^{22} + x^{23} + x^{24} + x^{26} + x^{27} + x^{30} + x^{31} + x^{35} + x^{37} + x^{38} + x^{39} + x^{40} + x^{43} + x^{45} + x^{46} + x^{47}$, $c = x + x^2 + x^3 + x^5 + x^8 + x^{11} + x^{12} + x^{13} + x^{14} + x^{15} + x^{17} + x^{19} + x^{20} + x^{22} + x^{24} + x^{26} + x^{29} + x^{30} + x^{32} + x^{33} + x^{34} + x^{36} + x^{37} + x^{38} + x^{39}$,
$d = 1 + x + x^2 + x^3 + x^4 + x^7 + x^9 + x^{10} + x^{12} + x^{14} + x^{15} + x^{16} + x^{17} + x^{20} + x^{21} + x^{25} + x^{30} + x^{31} + x^{32} + x^{33} + x^{37} + x^{38} + x^{41} + x^{44} + x^{45} + x^{46}$, and $e = x + x^2 + x^5 + x^6 + x^8 +$

$x^9 + x^{11} + x^{12} + x^{13} + x^{14} + x^{15} + x^{17} + x^{19} + x^{20} + x^{23} + x^{24} + x^{25} + x^{26} + x^{32} + x^{35} + x^{38} + x^{39} + x^{42} + x^{46} + x^{47}$.

The exponential maps $G_1 : (\mathbb{F}_{(2^{48})^2})^3 \rightarrow (\mathbb{F}_{(2^{48})^2})^3$ and $G_2 : (\mathbb{F}_{(2^{48})^3})^2 \rightarrow (\mathbb{F}_{(2^{48})^3})^2$ are induced by the matrices A and B, respectively, that are given as follows

$$A = \begin{pmatrix} 2^{a_{11}} & 2^{a_{12}} & 0 \\ 2^{a_{21}} & 0 & 2^{a_{23}} \\ 0 & 2^{a_{32}} & 2^{a_{33}} \end{pmatrix}, B = \begin{pmatrix} 2^{b_{11}} & 2^{b_{12}} \\ 2^{b_{21}} & 2^{b_{22}} \end{pmatrix} \quad (5)$$

where $0 \leq a_{11}, a_{12}, a_{21}, a_{23}, a_{32}, a_{33} < 96$ and $0 \leq b_{11}, b_{12}, b_{21}, b_{22} < 144$ were chosen in such a way that $\gcd(\det(A), 2^{96} - 1) = 1$ and $\gcd(\det(B), 2^{144} - 1) = 1$, and that the modular inverses of A and B (modulo $2^{96} - 1$ and $2^{144} - 1$, respectively) have the maximum number of non-zero bits [18]. Therefore, the chosen values have been $a_{11} = 24, a_{12} = 59, a_{21} = 21, a_{23} = 28, a_{32} = 29, a_{33} = 65$ and $b_{11} = 50, b_{12} = 24, b_{21} = 7, b_{22} = 88$. The *secret key* consists of the three maps L_1, L_2 and L_3. L_1 is given by the matrices $A_{11}, A_{12}, A_{13} \in \mathscr{M}_{2\times2}(\mathbb{F}_{2^{48}})$, L_2 is given by $A_{21}, A_{22} \in \mathscr{M}_{3\times3}(\mathbb{F}_{2^{48}})$ and L_3 is given by $A_{31}, A_{32} \in \mathscr{M}_{3\times3}(\mathbb{F}_{2^{48}})$, where their coefficients are randomly chosen. DME requires an initial padding step of the input in order to fit 36 bytes (6 inputs with 48 bits). With the above parameters, the *public key* has 2304 bytes (6 polynomials with 64 monomials each and with 48-bit coefficients), the *secret key* has 288 bytes (48-bit coefficients of the L_1, L_2 and L_3 matrices) and the *ciphertext* has 36 bytes (6 outputs with 48 bits each).

4. Pipelined architecture of DME

The pipelined architecture presented for DME-(3,2,48) is given in Fig. 1, where $\mathbf{L_1}$, $\mathbf{G_1}$, $\mathbf{L_2}$, $\mathbf{G_2}$ and $\mathbf{L_3}$ are implemented as combinational modules with 288-bit inputs and outputs. The communication between these modules are performed through 288-bit pipeline registers. The architectures of $\mathbf{L_1}$, $\mathbf{G_1}$, $\mathbf{L_2}$, $\mathbf{G_2}$ and $\mathbf{L_3}$ are described in next subsections.

4.1. Module L_1

The map $L_1 : (\mathbb{F}_{2^{48}})^6 \rightarrow (\mathbb{F}_{(2^{48})^2})^3$ is implemented with the module $\mathbf{L_1}$ that has six inputs $(x_1, x_2, x_3, x_4, x_5, x_6)$, with $x_i \in GF(2^{48})$, and three outputs (v_1, v_2, v_3), with $v_i \in GF((2^{48})^2)$.

The isomorphism $\widetilde{l}$ groups the inputs in three vectors in the form $\widetilde{l}(x_1, x_2, x_3, x_4, x_5, x_6) = (\underline{x}_1, \underline{x}_2, \underline{x}_3)$, where $\underline{x}_1 = (x_1, x_2)$, $\underline{x}_2 = (x_3, x_4), \underline{x}_3 = (x_5, x_6) \in \mathbb{F}^2_{2^{48}}$. The isomorphism $\widetilde{L}_1 = (L_{11}, L_{12}, L_{13})$ is defined by its components $L_{1i} : \mathbb{F}^2_{2^{48}} \rightarrow \mathbb{F}^2_{2^{48}}$ given by $L_{1i}(\underline{x}_i) = \underline{x}_i A_{1i}$, where $A_{1i} \in \mathscr{M}_{2\times2}(\mathbb{F}_{2^{48}})$. For example, $L_{11}(\underline{x}_1)$ is given by

$$L_{11}(\underline{x}_1) = L_{11}(x_1, x_2) = (x_1, x_2) \cdot \begin{pmatrix} a_{11} & a_{12} \\ a_{13} & a_{14} \end{pmatrix} = (x_1 a_{11} + x_2 a_{13}, x_1 a_{12} + x_2 a_{14}) = (v_{11}, v_{12}) \quad (6)$$

where $v_{11} = (x_1 a_{11} + x_2 a_{13})$ and $v_{12} = (x_1 a_{12} + x_2 a_{14})$ are elements of the finite field $\mathbb{F}_{2^{48}}$. The products and additions involved in (6) are operations over $GF(2^{48})$. In this paper, the bit-parallel polynomial basis multiplier over $GF(2^{48})$ generated by the type I irreducible pentanomial $f(y) = y^{48} + y^{28} + y^{27} + y + 1$ given in Ref. [19] has been used, and addition is performed by bitwise XOR operation. Finally, the map $\widetilde{\pi}_1 = (\pi_1, \pi_1, \pi_1)$ is defined by $\pi_1 : \mathbb{F}^2_{2^{48}} \rightarrow \mathbb{F}_{(2^{48})^2}$, with $\pi_1(v_{i1}, v_{i2}) = v_{i1} + v_{i2}\tau = v_i$, $i \in \{1, 2, 3\}$, where $\{1, \tau\}$ is the PB of the composite finite field $\mathbb{F}_{(2^{48})^2} = GF((2^{48})^2)$ generated by the irreducible polynomial $f_2(T) = T^2 + aT + b$ over $\mathbb{F}_{2^{48}}$, being τ a root of $f_2(T)$. For example, $v_1 = v_{11} + v_{12}\tau = (x_1 a_{11} + x_2 a_{13}) + (x_1 a_{12} + x_2 a_{14})\tau \in \mathbb{F}_{(2^{48})^2}$.

4.2. Module G_1

The exponential map $G_1 : (\mathbb{F}_{(2^{48})^2})^3 \rightarrow (\mathbb{F}_{(2^{48})^2})^3$ is implemented with the module $\mathbf{G_1}$ that has three inputs (v_1, v_2, v_3) and three outputs (h_1, h_2, h_3), where $v_i, h_i \in GF((2^{48})^2)$. G_1 is induced by the A matrix given in (5), in such a way that $G_1(v_1, v_2, v_3) = (h_1, h_2, h_3)$ is given by

$$\begin{aligned} h_1 &= v_1^{2^{24}} \cdot v_2^{2^{59}} = h_{11} + h_{12}\tau \\ h_2 &= v_1^{2^{21}} \cdot v_3^{2^{28}} = h_{21} + h_{22}\tau \\ h_3 &= v_2^{2^{29}} \cdot v_3^{2^{65}} = h_{31} + h_{32}\tau \end{aligned} \quad (7)$$

where $\{1, \tau\}$ is the PB of the composite field $\mathbb{F}_{(2^{48})^2}$ generated by the irreducible polynomial $f_2(T) = T^2 + aT + b$ over $\mathbb{F}_{2^{48}}$, with τ a root of $f_2(T)$. The operations involved in (7) are the multiplication $(v_i^{2^k} \cdot v_j^{2^l})$ and the k-squared exponentiation $(v_i^{2^k})$ of $\mathbb{F}_{(2^{48})^2}$ elements. These operations are described in the following subsections.

4.2.1. Multiplication over $\mathbb{F}_{(2^{48})^2}$

The product $p = u_1 \cdot u_2$, with $p, u_1, u_2 \in \mathbb{F}_{(2^{48})^2}$ generated by $f_2(T) = T^2 + aT + b$ over $\mathbb{F}_{2^{48}}$, with τ a root of $f_2(T)$, can be done as follows

$$\begin{aligned} p = p_1 + p_2\tau = u_1 \cdot u_2 = (u_{11} + u_{12}\tau) \cdot (u_{21} + u_{22}\tau) = \\ u_{11}u_{21} + u_{11}u_{22}\tau + u_{12}u_{21}\tau + u_{12}u_{22}\tau^2 = \\ u_{11}u_{21} + (u_{11}u_{22} + u_{12}u_{21})\tau + u_{12}u_{22}(a\tau + b) = \\ (u_{11}u_{21} + bu_{12}u_{22}) + (u_{11}u_{22} + u_{12}u_{21} + au_{12}u_{22})\tau \end{aligned} \quad (8)$$

where $\tau^2 = a\tau + b$. It must be noted that the operations involved in (8) are products and additions over $GF(2^{48})$, implemented using the multiplier given in Ref. [19] and with bitwise XORs, respectively. The architecture for the multiplication over $\mathbb{F}_{(2^{48})^2}$ is given in Fig. 2, where $\boxtimes$ and $\oplus$ stand for multiplication and addition over $GF(2^{48})$, respectively.

4.2.2. k-squared exponentiation over $\mathbb{F}_{(2^{48})^2}$

In order to give an expression for the computation of $v_i^{2^k}$, with $v_i \in \mathbb{F}_{(2^{48})^2}$, we can first compute the expression for v_i^2 as follows:

$$\begin{aligned} v_i^2 = (v_{i1} + v_{i2}\tau)^2 = v_{i1}^2 + \cancel{v_{i1}v_{i2}\tau} + \cancel{v_{i1}v_{i2}\tau} + v_{i2}^2\tau^2 = \\ v_{i1}^2 + v_{i2}^2(a\tau + b) = (v_{i1}^2 + v_{i2}^2 b) + (av_{i2}^2)\tau \end{aligned} \quad (9)$$

where we use the facts that we are working with finite fields of characteristic 2 and that we use the irreducible polynomial $f_2(T) = T^2 + aT + b$ with root τ, so $\tau^2 = a\tau + b$. The computation of successive squares gives

$$\begin{aligned} v_i^{2^1} &= (v_{i1}^2 + v_{i2}^2 b) + (av_{i2}^2)\tau \\ v_i^{2^2} &= (v_{i1}^4 + v_{i2}^4(b^2 + ba^2)) + (a^3 v_{i2}^4)\tau \\ v_i^{2^3} &= (v_{i1}^8 + v_{i2}^8(b^4 + b^2a^4 + a^6 b)) + (a^7 v_{i2}^8)\tau \\ v_i^{2^4} &= (v_{i1}^{16} + v_{i2}^{16}(b^8 + b^4a^8 + a^{12}b^2 + a^{14}b)) + (a^{15}v_{i2}^{16})\tau \\ &\cdots \end{aligned} \quad (10)$$

from where the following general expression can be given for the computation of the k-squared exponentiation of v_i

$$v_i^{2^k} = (v_{i1}^{2^k} + v_{i2}^{2^k}\Phi_{ab}(k)) + (a^{2^k-1}v_{i2}^{2^k})\tau \quad (11)$$

The function $\Phi_{ab}(k)$ in (11) can be deduced from (10) and is given by the expression

$$\Phi_{ab}(k) = b^{2^{k-1}} + \sum_{i=1}^{k-2} a^{(\sum_{j=1}^{i} 2^{k-j})} b^{2^{k-1-i}} + a^{(2^k-2)}b \quad (12)$$

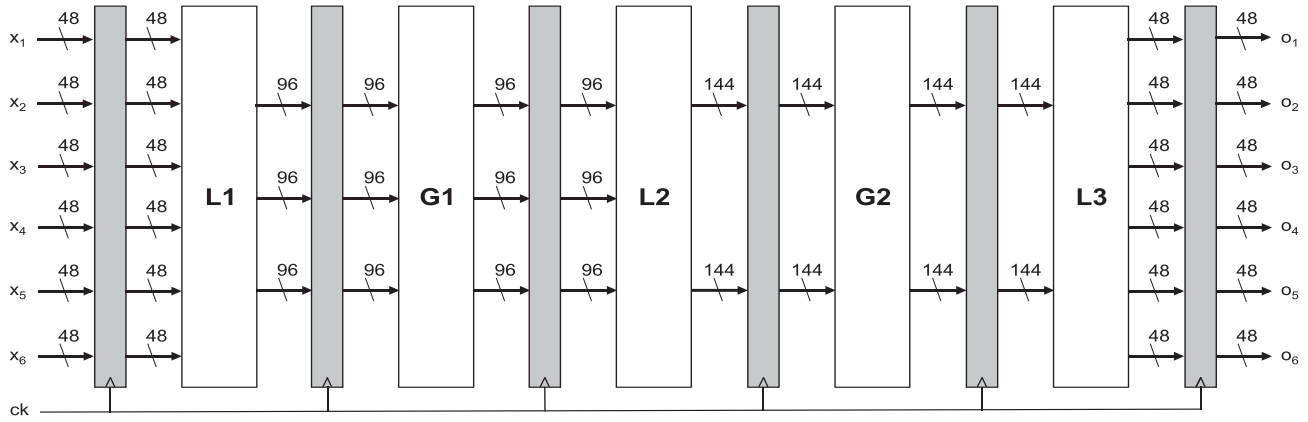

Fig. 1. Pipelined architecture of DME-(3,2,48).

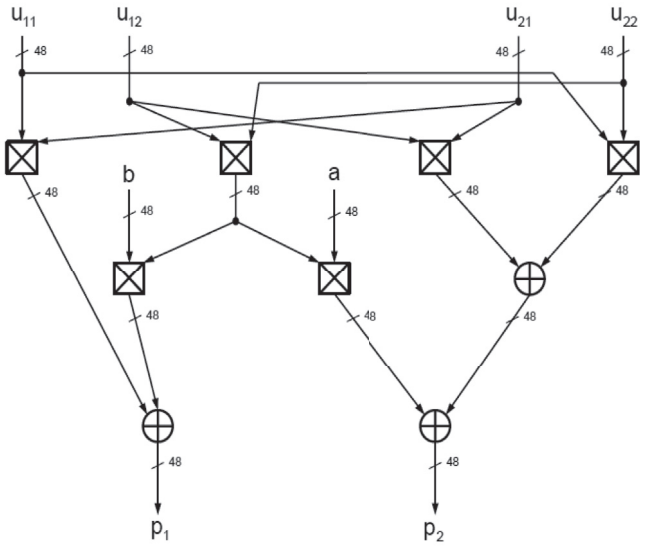

Fig. 2. Multiplication over $\mathbb{F}_{(2^{48})^2}$.

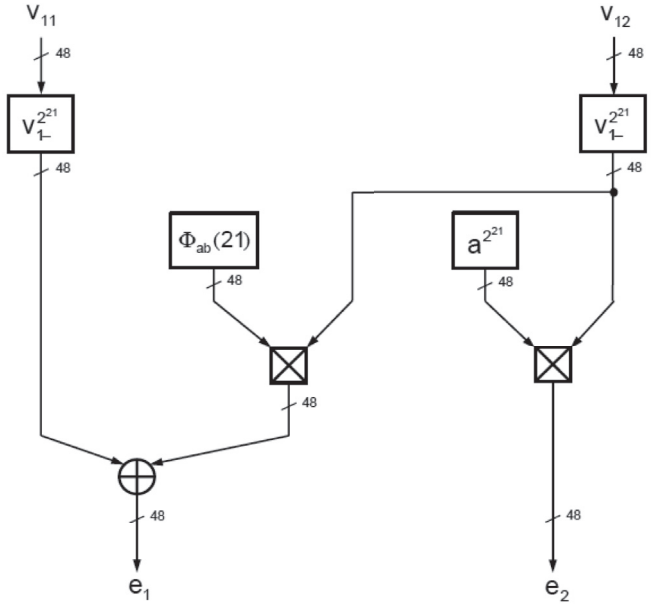

Fig. 3. Architecture for $v_1^{2^{21}}$ over $\mathbb{F}_{(2^{48})^2}$.

Using the fact that the sum of the first k terms of a geometric series is $\sum_{h=0}^{k-1} 2^h = 2^k - 1$, then it can be proven that $\sum_{j=1}^{i} 2^{k-j} = (2^k - 1) - \sum_{h=0}^{k-i-1} 2^h = 2^k - 2^{k-i}$, so the general expression for the function $\Phi_{ab}(k)$ is given by

$$\Phi_{ab}(k) = b^{2^{k-1}} + \sum_{i=1}^{k-2} a^{(2^k - 2^{k-i})} b^{2^{k-1-i}} + a^{(2^k-2)} b \tag{13}$$

The expression in (11) requires the computation of $\Phi_{ab}(k)$ given in (13) and a^{2^k-1}, where a is a constant randomly chosen given in Ref. [18]. These values were precomputed with *Maple* environment for the specific values used in (7), $k = 21, 24, 28, 29, 59, 65$, and stored as constants for the implementation. The computation of $v_i^{2^k}$ in (11) also requires the computation of $v_{i-}^{2^k}$, with $v_{i-} \in GF(2^{48})$. This k-squared exponentiation were performed by successive squaring of v_{i-} using the method given in Ref. [19]. As an example, the architecture for the computation of $v_1^{2^{21}} = e_1 + e_2\tau$ over $\mathbb{F}_{(2^{48})^2}$ using (11) is given in Fig. 3, where $\boxtimes$ and $\oplus$ stand for multiplication and addition over $GF(2^{48})$, respectively.

4.3. Module $\mathbf{L_2}$

The map $L_2 : (\mathbb{F}_{(2^{48})^2})^3 \rightarrow (\mathbb{F}_{(2^{48})^3})^2$ is implemented with the module $\mathbf{L_2}$ that has three inputs (h_1, h_2, h_3), with $h_i \in GF((2^{48})^2)$, and two outputs (k_1, k_2), where $k_i \in GF((2^{48})^3)$.

The map $\widetilde{\pi}_1^{-1} = (\pi_1^{-1}, \pi_1^{-1}, \pi_1^{-1})$ is the inverse of $\widetilde{\pi}_1$, in such a way that $\pi_1^{-1} : \mathbb{F}_{(2^{48})^2} \rightarrow \mathbb{F}_{2^{48}}^2$, with $\pi_1^{-1}(h_i) = \pi_1^{-1}(h_{i1} + h_{i2}\tau) = (h_{i1}, h_{i2})$, $i \in \{1, 2, 3\}$, where $\{1, \tau\}$ is the PB of the composite field $\mathbb{F}_{(2^{48})^2}$ generated by $f_2(T) = T^2 + aT + b$ over $\mathbb{F}_{2^{48}}$, with τ a root of $f_2(T)$. The *mixing* isomorphism $M : (\mathbb{F}_{2^{48}}^2)^3 \rightarrow (\mathbb{F}_{2^{48}}^3)^2$ transforms the three vectors of $\mathbb{F}_{2^{48}}^2$ in two vectors of $\mathbb{F}_{2^{48}}^3$ as follows

$$\begin{pmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \\ h_{31} & h_{32} \end{pmatrix} \overset{M}{\Longrightarrow} \begin{pmatrix} h_{11} & h_{12} & h_{31} \\ h_{21} & h_{22} & h_{32} \end{pmatrix} \tag{14}$$

where $\underline{h}_1 = (h_{11}, h_{12}, h_{31})$ and $\underline{h}_2 = (h_{21}, h_{22}, h_{32})$. The isomorphism $\widetilde{L}_2 = (L_{21}, L_{22})$ is defined by $L_{2i} : \mathbb{F}_{2^{48}}^3 \rightarrow \mathbb{F}_{2^{48}}^3$ with $L_{2i}(\underline{h}_i) = \underline{h}_i A_{2i}$ and

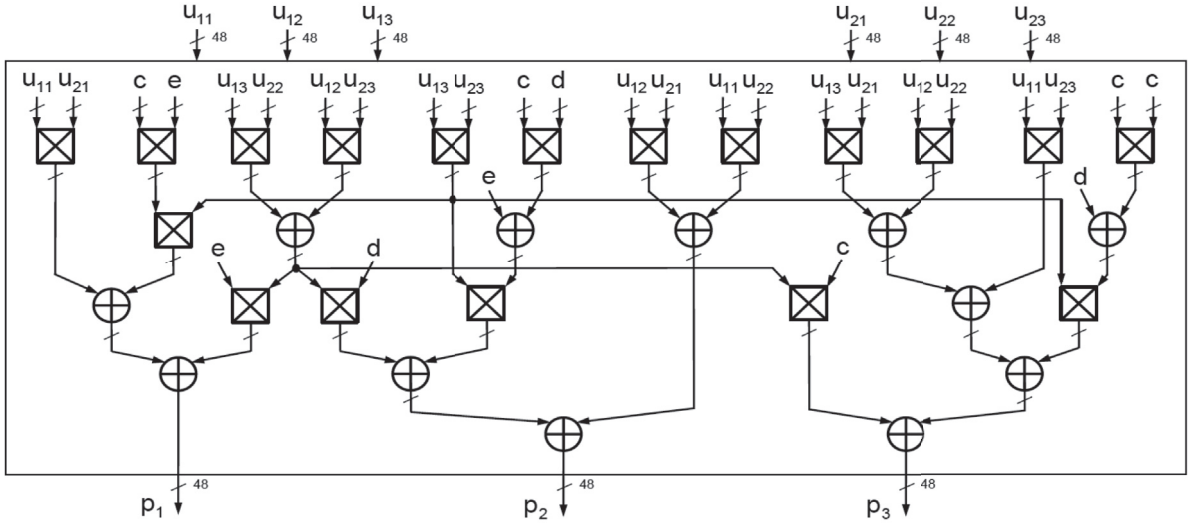

Fig. 4. Multiplication over $\mathbb{F}_{(2^{48})^3}$.

$A_{2i} \in \mathscr{M}_{3\times 3}(\mathbb{F}_{2^{48}})$. For example, $L_{21}(\underline{h}_1) = (k_{11}, k_{12}, k_{13})$ is computed by

$$L_{21}(\underline{h}_1) = (h_{11}, h_{12}, h_{31}) \cdot \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{14} & a_{15} & a_{16} \\ a_{17} & a_{18} & a_{19} \end{pmatrix} \tag{15}$$

where $k_{11} = (h_{11}a_{11} + h_{12}a_{14} + h_{31}a_{17})$, $k_{12} = (h_{11}a_{12} + h_{12}a_{15} + h_{31}a_{18})$ and $k_{13} = (h_{11}a_{13} + h_{12}a_{16} + h_{31}a_{19})$ are elements of the finite field $\mathbb{F}_{2^{48}}$. The products and additions involved in (15) are operations over $GF(2^{48})$. Finally, the map $\widetilde{\pi}_2 = (\pi_2, \pi_2)$ is defined by the isomorphism $\pi_2 : \mathbb{F}_{2^{48}}^3 \to \mathbb{F}_{(2^{48})^3}$, with $\pi_2(k_{i1}, k_{i2}, k_{i3}) = k_{i1} + k_{i2}\nu + k_{i3}\nu^2 = k_i$, $i \in \{1, 2\}$, where $\{1, \nu, \nu^2\}$ is the polynomial basis of the composite field $\mathbb{F}_{(2^{48})^3} = GF((2^{48})^3)$ generated by the irreducible polynomial $f_3(S) = S^3 + cS^2 + dS + e$ over $\mathbb{F}_{2^{48}}$, being ν a root of $f_3(S)$. For example, $k_1 = k_{11} + k_{12}\nu + k_{13}\nu^2 \in \mathbb{F}_{(2^{48})^3}$.

4.4. Module G_2

The exponential map $G_2 : (\mathbb{F}_{(2^{48})^3})^2 \to (\mathbb{F}_{(2^{48})^3})^2$ is implemented with the module $\mathbf{G_2}$ that has two inputs (k_1, k_2) and two outputs (w_1, w_2) with $k_i, w_i \in GF((2^{48})^3)$. G_2 is induced by the B matrix given in (5), in such a way that $G_2(k_1, k_2) = (w_1, w_2)$ is given by

$$\begin{aligned} w_1 &= k_1^{2^{50}} \cdot k_2^{2^{24}} = w_{11} + w_{12}\nu + w_{13}\nu^2 \\ w_2 &= k_1^{2^7} \cdot k_2^{2^{88}} = w_{21} + w_{22}\nu + w_{23}\nu^2 \end{aligned} \tag{16}$$

where $\{1, \nu, \nu^2\}$ is the PB of $\mathbb{F}_{(2^{48})^3}$ generated by $f_3(S) = S^3 + cS^2 + dS + e$ over $\mathbb{F}_{2^{48}}$, with ν a root of $f_3(S)$.

The operations involved in (16) are the multiplication $(k_i^{2^r} \cdot k_j^{2^s})$ and the r-squared exponentiation $(k_i^{2^r})$ of $\mathbb{F}_{(2^{48})^3}$ elements. These operations are described in the following subsections.

4.4.1. Multiplication over $\mathbb{F}_{(2^{48})^3}$

The product $p = u_1 \cdot u_2$, with $p, u_1, u_2 \in \mathbb{F}_{(2^{48})^3}$ generated by $f_3(S) = S^3 + cS^2 + dS + e$ over $\mathbb{F}_{2^{48}}$, with ν a root of $f_3(S)$, can be done as

$$\begin{aligned} &p = p_1 + p_2\nu + p_3\nu^2 = u_1 \cdot u_2 = \\ &(u_{11} + u_{12}\nu + u_{13}\nu^2) \cdot (u_{21} + u_{22}\nu + u_{23}\nu^2) = u_{11}u_{21} + \\ &(u_{11}u_{22} + u_{12}u_{21})\nu + (u_{11}u_{23} + u_{12}u_{22} + u_{13}u_{21})\nu^2 + \\ &(u_{12}u_{23} + u_{13}u_{22})\nu^3 + u_{13}u_{23}\nu^4 = \\ &[u_{11}u_{21} + e(u_{12}u_{23} + u_{13}u_{22}) + ceu_{13}u_{23}] + \\ &[(u_{11}u_{22} + u_{12}u_{21}) + d(u_{12}u_{23} + u_{13}u_{22}) + \\ &(cd + e)u_{13}u_{23}]\nu + [(u_{11}u_{23} + u_{12}u_{22} + u_{13}u_{21}) + \\ &c(u_{12}u_{23} + u_{13}u_{22}) + (c^2 + d)u_{13}u_{23}]\nu^2 \end{aligned} \tag{17}$$

where $\nu^3 = c\nu^2 + d\nu + e$ and $\nu^4 = (c^2 + d)\nu^2 + (cd + e)\nu + ce$.

It must be noted that the arithmetic operations involved in (17) are multiplications and additions over $GF(2^{48})$, implemented using the multiplier given in Ref. [19] and with bitwise XORs, respectively. The architecture for the multiplication over $\mathbb{F}_{(2^{48})^3}$ is given in Fig. 4, where $\boxtimes$ and $\oplus$ stand for multiplication and addition over $GF(2^{48})$, respectively.

4.4.2. r-squared exponentiation over $\mathbb{F}_{(2^{48})^3}$

In order to give an expression for the computation of $k_i^{2^r}$, with $k_i = (k_{i1} + k_{i2}\nu + k_{i3}\nu^2) \in \mathbb{F}_{(2^{48})^3}$, we can first compute the expression for k_i^2 as follows:

$$\begin{aligned} k_i^2 &= k_{i1}^2 + \cancel{k_{i1}k_{i2}\nu} + \cancel{k_{i1}k_{i3}\nu^2} + \cancel{k_{i2}k_{i1}\nu} + k_{i2}^2\nu^2 + \\ &\cancel{k_{i2}k_{i3}\nu^3} + \cancel{k_{i3}k_{i1}\nu^2} + \cancel{k_{i3}k_{i2}\nu^3} + k_{i3}^2\nu^4 = [k_{i1}^2 + cek_{i3}^2] \\ &+ [(cd + e)k_{i3}^2]\nu + [k_{i2}^2 + (c^2 + d)k_{i3}^2]\nu^2 \end{aligned} \tag{18}$$

where we use the facts that we are working with finite fields of characteristic 2 and that we use the irreducible polynomial $f_3(S) = S^3 + cS^2 + dS + e$ with ν a root of $f_3(S)$, so $\nu^3 = c\nu^2 + d\nu + e$ and $\nu^4 = (c^2 + d)\nu^2 + (cd + e)\nu + ce$.

In order to compute the r-squared exponentiation $k_i^{2^r}$, it must be noted the following modular reduction property. The computation of successive powers of ν gives:

$$\begin{aligned} \nu^3 &= c\nu^2 + d\nu + e \\ \nu^4 &= [c^2 + d]\nu^2 + [cd + e]\nu + ce \\ \nu^5 &= [c^3 + e]\nu^2 + [(c^2 + d)d + ce]\nu + [(c^2 + d)e] \\ \nu^6 &= [c^4 + c^2d + d^2]\nu^2 + [c^3d + c^2e]\nu + [(c^3 + e)e] \\ &\cdots \end{aligned} \tag{19}$$

from where it can be observed that if $\nu^i = \eta\nu^2 + \psi\nu + \omega$, then

$$\nu^{i+1} = [\eta c + \psi]\nu^2 + [\eta d + \omega]\nu + [\eta e] \tag{20}$$

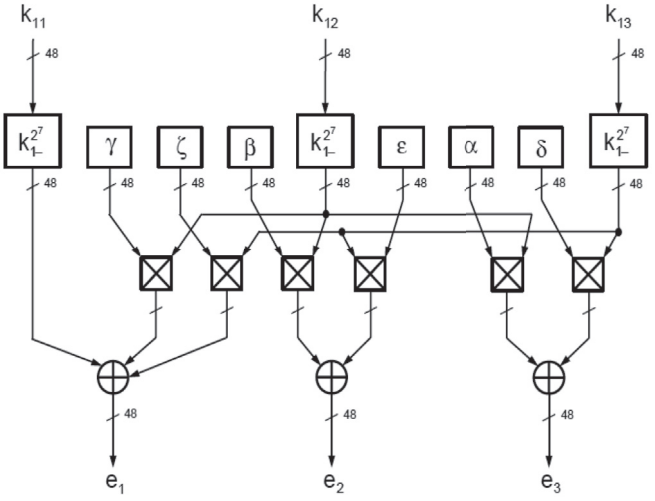

Fig. 5. Architecture for $k_1^{2^7}$ over $\mathbb{F}_{(2^{48})^3}$.

Furthermore, it can also be observed that if the j-squared exponentiation $v^{2^j} = \alpha v^2 + \beta v + \gamma$, then

$$v^{2^{j+1}} = [\beta^2 + \alpha^2(c^2 + d)]v^2 + [\alpha^2(cd + e)]v + [\gamma^2 + \alpha^2 ce] = \delta v^2 + \epsilon v + \zeta \quad (21)$$

Using the expressions (20) and (21), the successive squarings of k_i can be computed. For example, $k_i^{2^2}$ is given by

$$k_i^{2^2} = [k_{i1}^4 + cek_{i2}^4 + (c^2e^2 + ce(c^4 + d^2))k_{i3}^4] + [(cd + e)k_{i2}^4 + ((cd + e)(c^4 + d^2))k_{i3}^4]v + [(c^2 + d)k_{i2}^4 + (c^6 + e^2 + d(c^4 + d^2))k_{i3}^4]v^2 \quad (22)$$

The computation of successive squarings as in (22) let us to give a general expression for the computation of $k_i^{2^r}$. Let $v^{2^r} = \alpha v^2 + \beta v + \gamma$ and $v^{2^{r+1}} = \delta v^2 + \epsilon v + \zeta$ (21). Then the r-squared exponentiation $k_i^{2^r}$ is given by the expression

$$k_i^{2^r} = [k_{i1}^{2^r} + \gamma k_{i2}^{2^r} + \zeta k_{i3}^{2^r}] + [\beta k_{i2}^{2^r} + \epsilon k_{i3}^{2^r}]v + [\alpha k_{i2}^{2^r} + \delta k_{i3}^{2^r}]v^2 \quad (23)$$

where $\delta = \beta^2 + \alpha^2(c^2 + d)$, $\epsilon = \alpha^2(cd + e)$ and $\zeta = \gamma^2 + \alpha^2 ce$.

The computation of $k_1^{2^7}, k_2^{2^{24}}, k_1^{2^{50}}$ and $k_2^{2^{88}}$ given in (16) can be done using (23), where the values α, β, γ and δ, ϵ, ζ must be computed for $v^{2^7}, v^{2^{24}}, v^{2^{50}}, v^{2^{88}}$ and $v^{2^8}, v^{2^{25}}, v^{2^{51}}, v^{2^{89}}$, respectively. In order to speed up the implementation, these values were precomputed using (21) with *Maple* environment and stored as constants. The computation of $k_i^{2^r}$ in (23) also requires the computation of $k_{i-}^{2^r}$, with $k_{i-} \in GF(2^{48})$. This r-squared exponentiation were performed by successive squaring of k_{i-} using the method given in Ref. [19]. As an example, the architecture for the computation of $k_1^{2^7} = e_1 + e_2 v + e_3 v^2$ over $\mathbb{F}_{(2^{48})^3}$ using (23) is given in Fig. 5, where $\boxtimes$ and $\oplus$ stand for multiplication and addition over $GF(2^{48})$, respectively.

4.5. Module L_3

Module $\mathbf{L_3}$ implements the map $L_3 : (\mathbb{F}_{(2^{48})^3})^2 \to (\mathbb{F}_{2^{48}})^6$, so it has two inputs (w_1, w_2) with $w_i \in GF((2^{48})^3)$ and six outputs $(o_1, o_2, o_3, o_4, o_5, o_6)$ with $o_i \in GF(2^{48})$. The inputs (w_1, w_2) to $\mathbf{L_3}$ are the outputs of $\mathbf{G_2}$.

The map $\widetilde{\pi}_2^{-1} = (\pi_2^{-1}, \pi_2^{-1})$ is the inverse of $\widetilde{\pi}_2$, in such a way that $\pi_2^{-1} : \mathbb{F}_{(2^{48})^3} \to \mathbb{F}_{2^{48}}^3$, with $\pi_2^{-1}(w_i) = \pi_2^{-1}(w_{i1} + w_{i2}v + w_{i3}v^2) = (w_{i1}, w_{i2}, w_{i3}) = \underline{w}_i$, $i \in \{1, 2\}$, where $\{1, v, v^2\}$ is the PB of the composite field $\mathbb{F}_{(2^{48})^3}$ generated by $f_3(S) = S^3 + cS^2 + dS + e$ over $\mathbb{F}_{2^{48}}$, being v a root of $f_3(S)$. The isomorphism $\widetilde{L}_3 = (L_{31}, L_{32})$ is defined by its components $L_{3i} : \mathbb{F}_{2^{48}}^3 \to \mathbb{F}_{2^{48}}^3$ given by $L_{3i}(\underline{w}_i) = \underline{w}_i A_{3i} = \underline{o}_i$, where $A_{3i} \in \mathscr{M}_{3\times3}(\mathbb{F}_{2^{48}})$. For example, $L_{31}(\underline{w}_1) = (o_1, o_2, o_3) = \underline{o}_1$ is computed by

$$L_{31}(\underline{w}_1) = (w_{11}, w_{12}, w_{31}) \cdot \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{14} & a_{15} & a_{16} \\ a_{17} & a_{18} & a_{19} \end{pmatrix} \quad (24)$$

where $o_1 = (w_{11}a_{11} + w_{12}a_{14} + w_{31}a_{17})$, $o_2 = (w_{11}a_{12} + w_{12}a_{15} + w_{31}a_{18})$ and $o_3 = (w_{11}a_{13} + w_{12}a_{16} + w_{31}a_{19})$ are elements of $\mathbb{F}_{2^{48}}$. The products and additions involved in (24) are operations over $GF(2^{48})$. Finally, the map $\widetilde{e}^{-1}$ is the inverse of the isomorphism $\widetilde{e}$ that groups the six inputs in two vectors, so $\widetilde{e}^{-1}(\underline{o}_1, \underline{o}_2) = (o_1, o_2, o_3, o_4, o_5, o_6)$, with $\underline{o}_1 = (o_1, o_2, o_3)$ and $\underline{o}_2 = (o_4, o_5, o_6)$.

Table 1
Comparison of clock cycles.

Scheme	Clock cycles
*en*TTS(28,20) [14]	16,000
ECC-163 [20]	4050
ECC-163 [21]	3308
Rainbow(42,24) [13]	3150
UOV(60,20) [13]	2260
ECC-163 [22]	1371
Rainbow(42,24) [23]	804
UOV(30,10) [13]	630
Rainbow(42,24) [7]	198
*am*TTS(34,24) [13]	195
*en*TTS(28,20) [13]	162
Rainbow(42,24) [24]	148
*en*TTS(28,20) [25]	90
DME-(3,2,48)	5

5. FPGA implementation and performance comparison

The pipelined architecture for DME-(3,2,48) has been compared in Table 1 with other multivariate public-key cryptosystems, with respect to the number of clock cycles needed for performing the computations. In Table 1, efficient architectures for Elliptic-Curve Point Multiplication in $GF(2^{163})$ (ECC-163) have also been included for comparison, where parameters (n, m) for UOV, Rainbow, *am*TTS and *en*TTS correspond with n- and m-bytes signature and message sizes, respectively. In Table 1 it can be observed that the architecture for DME-(3,2,48) only requires five clock cycles in comparison with other multivariate schemes, which require a minimum of 90 clock cycles (*en*TTS). It must also be noted that the pipelined architecture is valid for any DME-(m, n, e).

The architecture previously presented for DME-(3,2,48) has been described in VHDL (*VHSIC-Hardware Description Language*), where the modules $\mathbf{L_1}$, $\mathbf{G_1}$, $\mathbf{L_2}$, $\mathbf{G_2}$ and $\mathbf{L_3}$ were modeled as combinational logic with 288-bit inputs/outputs using 288-bit interleaved pipeline registers. The design was synthesized and implemented using Xilinx ISE 14.7 tool on Artix-7 XC7A200T-FFG1156 FPGA device. Furthermore, *speed high* optimizations have been part of the design methodology. Experimental post-place and route results of our implementation are given in Table 2, where FPGA results reported in the literature for other multivariate schemes have also been included for comparison. In order to determine the frequency of our pipelined implementation, time restrictions were imposed to the synthesis tool that determined a minimum clock period of 15.0 ns and, therefore, a clock frequency of 67 MHz. In Table 2, T (μs) represent the execution time for each scheme and the Area × T metrics (less is better) expresses area by execution time (in slices × miliseconds) in order to compare the area and delay. It must be

Table 2
FPGA implementation comparison.

Scheme	Device	F (MHz)	T (μs)	Area (slices)	Throughput (Mb/s)	Area × T (slices × ms)
ECC-163 [20]	V2	100	41.000	3416	3.98	140.06
ECC-163 [20]	V4	197	21.000	4080	7.93	85.68
ECC-163 [21]	VE	48	68.900	–	2.36	447.90
ECC-163 [22]	V5	250	5.500	6150	29.64	33.82
Rainbow(42,24) [13]	V3	80	7.780	4123	43.18	32.08
Rainbow(42,24) [13]	V5	200	5.600	2000	60.00	11.20
Rainbow(42,24) [7]	St-II	50	3.960	–	84.85	–
Rainbow(42,24) [24]	V7	200	0.610	5878	550.82	3.58
UOV(60,20) [13]	V3	80	14.620	9821	32.82	143.60
UOV(60,20) [13]	V5	200	5.850	5334	82.05	31.20
UOV(30,10) [13]	V3	80	4.190	3060	57.31	12.80
UOV(30,10) [13]	V5	200	1.670	1585	143.28	2.70
*am*TTS(34,24) [13]	V3	80	2.440	3139	111.57	7.70
*am*TTS(34,24) [13]	V5	200	0.970	1659	278.97	1.60
*en*TTS(28,20) [13]	V3	80	2.020	3060	110.62	6.20
*en*TTS(28,20) [13]	V5	200	0.810	1585	276.54	1.20
*en*TTS(28,20) [25]	St	100	0.900	–	248.90	–
DME-(3,2,48)	A7	67	0.075	24,562	3840.00	1.84

VE = Xilinx Virtex-E, V2 = Virtex-2, V3 = Virtex-3, V4 = Virtex-4, V5 = Virtex-5, V7 = Virtex-7, A7 = Artix-7, St = Altera Stratix, St-II = Stratix II.

noted that the Artix-7 XC7A200T-FFG1156 used for the implementation has a limited number of 500 input/output user pins, so the architecture given in Fig. 1 had to be slightly modified. In order to include the complete architecture in only one FPGA, the outputs obtained from the $\mathbf{L_3}$ module were loaded into a 288-bit shift-register (controlled by a small FSM) in such a way that the results $o_i \in GF(2^{48})$, with $i = \{1, \ldots, 6\}$, were obtained in additional clock cycles (one per cycle). The results reported in Table 2 do not include these additional cycles because they are due to the limited number of pins.

Experimental results show that DME-(3,2,48) can perform a complete computation in 75 ns, one magnitude order (87.7%) faster than Rainbow(42,24) [24], the fastest scheme given in the literature. This result is specially important because Rainbow is one of the candidates to the NIST PQC Standardization Process that has moved on to the second round of the competition. Furthermore, the throughput of our implementation is 3.84 Gbps, again one magnitude order (597%) higher than Rainbow(42,24) [24], the best multivariate throughput result given in Table 2. A comparison of throughputs is shown in Fig. 6, where the best results for the different schemes considered in Table 2 are included. Due to the pipelined architecture and combinational modeling of the different DME modules, the number of slices used for the implementation is the highest one among the other schemes. Anyway, the Area × T metrics (less is better) expresses a very good value of 1.84 slices × miliseconds, that is the third best result among the other implementations given in Table 2.

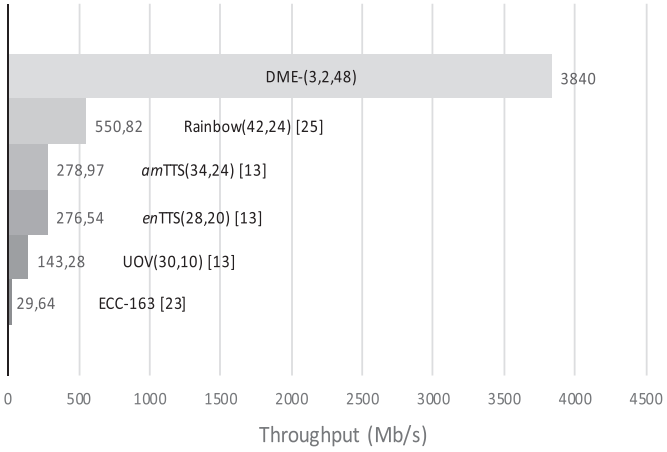

Fig. 6. Throughput for most relevant schemes given in Table 2.

6. Conclusion

DME-(m, n, e) is a new proposal of quantum-resistant multivariate PKC algorithm. In this paper, a pipelined architecture of DME-(3,2,48) has been presented and hardware implementations over Xilinx FPGAs have been performed. The proposed pipelined architecture, that is valid for any DME-(m, n, e) cryptosystem, only requires five clock cycles for performing computations. Experimental results show that the architecture here presented exhibits the lowest execution time and highest throughput when it is compared with other multivariate PQC implementations. Specifically, DME-(3,2,48) is 87.7% faster and presents a throughput 597% higher than Rainbow(42,24), the best multivariate implementation found in the literature. These results are specially important because Rainbow is one of the candidates to the NIST PQC competition that has moved on to the second round. Furthermore, the Area × T metrics for DME-(3,2,48) presents the third best result among the other multivariate schemes here considered.

CRediT author statement

José L. Imaña: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Writing - original draft. **Ignacio Luengo:** Conceptualization, Formal analysis, Writing - review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work has been supported by the Spanish MINECO and CM under grants S2018/TCS-4423, TIN 2015-65277-R and RTI2018-093684-B-I00.

References

[1] P. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM J. Comput. 26 (5) (1997) 1484–1509.
[2] J. Bernstein, J. Buchmann, E. Dahmen, Post-quantum Cryptography, Springer, 2009.

[3] S. Rohde, T. Eisenbarth, E. Dahmen, J. Buchmann, C. Paar, Fast hash-based signatures on constrained devices, in: CARDIS 2008, vol. LNCS-5189, 2008, pp. 104–117.
[4] J. Hoffstein, J. Pipher, J. Silverman, Ntru: a ring-based public key cryptosystem, in: Algorithmic Number Theory, Proc. Third Int'l Symp., ANTS-III, 1998, pp. 267–288.
[5] R. McEliece, A Public-Key Cryptosystem Based on Algebraic Coding Theory, Deep Space Network Progress Report, Jet Propulsion Lab., California Inst. Technology, 1978.
[6] J. Ding, D. Schmidt, Multivariate public key cryptosystems, Adv. Inf. Secur. 25 (2006).
[7] S. Tang, H. Yi, J. Ding, H. Chen, G. Chen, High-speed hardware implementation of rainbow signature on fpgas, in: Fourth Int'l. Conf. On Post-Quantum Cryptography, PQCrypto 2011, vol. LNCS-7071, 2011, pp. 228–243.
[8] W. Dai, W. Whyte, Z. Zhang, Optimizing polynomial convolution for ntruencrypt, IEEE Trans. Comput. 67 (11) (2018) 1572–1583.
[9] S. Balasubramanian, H. Carter, A. Bogdanov, A. Rupp, Fast multivariate signature generation in hardware: the case of rainbow, in: Proc. 16th Int'l Symp. on Field-Programmable Custom Computing Machines, FCCM 08, 2008, pp. 281–282.
[10] H. Imai, T. Matsumoto, Algebraic methods for constructing asymmetric cryptosystems, in: Applied Algebra, Algebraic Algorithms, and Error-Correcting Codes, Proc. Third Int'l Conf., AAECC-3, vol. LNCS-29, 1985, pp. 108–119.
[11] A. Shamir, Efficient signature schemes based on birational permutations, in: Advances in Cryptology, CRYPTO 1993, vol. LNCS-773, 1993, pp. 1–12.
[12] J. Patarin, Hidden field equations (hfe) and isomorphisms of polynomials (ip): two new families of asymmetric algorithms, in: Advances in Cryptology, EUROCRYPT 1996, vol. LNCS-1070, 1996, pp. 33–48.
[13] A. Bogdanov, T. Eisenbarth, A. Rupp, C. Wolf, Time-area optimized public-key engines: Mq-cryptosystems as replacement for elliptic curves? in: Proc. Int'l Workshop on Cryptographic Hardware and Embedded Systems, CHES 2008, vol. LNCS-5154, 2008, pp. 45–61.
[14] B.-Y. Yang, C.-M. Cheng, B.-R. Chen, J.-M. Chen, Implementing minimized multivariate pkc on low-resource embedded systems, in: SPC 2006, vol. LNCS-3934, 2006, pp. 73–88.
[15] T. Moh, A public key system with signature and master key functions, Commun. Algebra 27 (5) (1999) 2207–2222.
[16] J. Ding, D. Schmidt, Rainbow, a new multivariable polynomial signature scheme, in: Applied Cryptography and Network Security, ACNS 2005, vol. LNCS-3531, 2005, pp. 164–175.
[17] I. Luengo, Dme: a Public Key, Signature and Kem System Based on Double Exponentiation with Matrix Exponents, Tech. rep., 2017.
[18] I. Luengo, M. Avendaño, M. Marco, DME a Public Key, Signature and KEM System Based on Double Exponentiation, NIST proposal, 2017, https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Round-1-Submissions.
[19] J.L. Imaña, Fast bit-parallel binary multipliers based on type-i pentanomials, IEEE Trans. Comput. 67 (6) (2018) 898–904.
[20] B. Ansari, M. Hasan, High-performance of elliptic curve scalar multiplication, IEEE Trans. Comput. 57 (11) (2008) 1443–1453.
[21] C. Shu, K. Gaj, T. El-Ghazawi, Low latency elliptic curve cryptography accelerators for nist curves on binary fields, in: Proc. IEEE Int'l Conf. on Field-Programmable Technology, FPT05, 2005.
[22] G. Sutter, J.-P. Deschamps, J.L. Imaña, Efficient elliptic curve point multiplication using digit-serial binary operations, IEEE Trans. Ind. Electron. 60 (1) (2013) 217–225.
[23] S. Balasubramanian, A. Bogdanov, A. Rupp, A. Ding, J. Carter, Fast multivariate signature generation in hardware: the case of rainbow, in: Proc. 16th Int'l Symposium on Field-Programmable Custom Computing Machines, 2008, pp. 281–282.
[24] A. Ferozpuri, K. Gaj, High-speed fpga implementation of the nist round 1 rainbow signature scheme, in: Proc. Int'l Conf. on ReConFigurable Computing and FPGAs, ReConFig2018, 2018, pp. 1–8.
[25] H. Yi, Z. Nie, High-speed hardware architecture for implementations of multivariate signature generations on fpgas, EURASIP J. Wirel. Commun. Netw. (93) (2018).