

VLSI Architecture of Euclideanized BM Algorithm for Reed-Solomon Code

HUANG-CHI CHEN^{1,2}, YU-WEN CHANG³ AND REY-CHUE HWANG¹

¹*Department of Electrical Engineering*

I-Shou University

Kaohsiung, 840 Taiwan

²*Department of Management Information System*

Far East University

Tainan, 744 Taiwan

³*Department of Electronic Communication Engineering*

National Kaohsiung Marine University

Kaohsiung, 840 Taiwan

In 1988, Eastman showed that the Euclideanized Berlekamp-Massey (BM) algorithm can be used to eliminate the calculation of discrepancies and the divisions of the finite field elements to find the errata locator polynomial in a Reed-Solomon (RS) decoder. However, the separate computations of the errata locator and the errata evaluator polynomials are still needed in Eastman's decoder. In this paper, a modified decoding algorithm based on the idea of Eastman is presented. It is derived to solve the errata locator and the errata evaluator polynomials simultaneously without performing the operations of polynomial division and field element inversion. Moreover, the weights used to represent the discrepancies at each iteration can be directly extracted from the coefficient. Therefore, the proposed algorithm saves many controlling circuits and provides a modular VLSI architecture with parallel. As a consequence, it is simple and easy to implement. And the decoding complexity of the algorithm proposed by Eastman can be further reduced.

Keywords: error correcting code, Reed-Solomon code, Berlekamp's key equation, Euclidean algorithm, VLSI, architecture

1. INTRODUCTION

Reed-Solomon codes for correcting both errors and erasures are used extensively in space communication links [1], compact disk (CD), audio system [2], high definition television (HDTV) [3], digital versatile disk (DVD) [4]. It is well known that Forney [5] defined an errata locator polynomial using what are now called Forney syndromes to correct both errors and erasures for RS codes. Blahut [6] showed that the errata locator polynomial can be obtained directly by initializing the Berlekamp-Massey (BM) algorithm with the erasure locator polynomial.

It is well known that the Berlekamp-Massey (BM) algorithm or the Euclidean algorithm can be used to solve the Berlekamp's key equation [9]. Based on the extended Euclidean algorithm [8, 10], a direct manner is employing to obtain the weight coefficients used to perform the linear combination of polynomials. The algorithm in [10], the

Received September 19, 2007; revised March 13 & June 26, 2008; accepted September 25, 2008.
Communicated by Liang-Gee Chen.

linear combination to produce zero coefficients of the error evaluator polynomial is always performed at a fixed location and the weight coefficients can be extracted from the fixed $((d - 1)$ -th) position. Therefore, the architecture in [10] will be regularity. The previous approaches [7, 11], the Euclideanized BM or reformulated BM algorithm is adopting to eliminate the calculations of discrepancies and finite-field inversions. The algorithm in [11] is presented to correct error only for RS decoder and the discrepancy used to perform the update of polynomials is always extracted from the fixed (zero-th) position.

A Euclideanized BM algorithm proposed by Eastman given in [7] was the first one to simplify the BM algorithm for correcting both errors and erasures of RS codes. The key idea of this decoding algorithm is to expand the BM algorithm in a Euclidean context in order to eliminate the computation of the discrepancy at each iteration needed in the conventional BM algorithm. The advantage is that the computations of the discrepancy and the finite field element inversion are completely avoided. However, the separate computations of the errata locator and the errata evaluator polynomials are still required. It means that after the errata locator polynomial is obtained, one more multiplication operation used to find the errata evaluator polynomial is needed. Moreover, a two-dimensional systolic array implementation is adopted by Eastman [7]. Thus, some of the following circuits such as the multiplication operation block, time delay unit are needed for the design of the decoder. Therefore, it will increase the complexity of the hardware implementation.

In this paper, the modified algorithm is presented to improve Eastman's decoder [7]. It means that the errata locator and the errata evaluator polynomials can be obtained simultaneously by initializing the Euclideanized BM algorithm with the erasure locator polynomial and the Forney syndromes. Thus the decoding algorithm has lower delay. Furthermore, the multiplication operation block and time delay unit needed in [7] can be completely avoided. On the other hand, the hardware implementation is much simpler because of the parallel structure. Consequently, these facts lead to a substantial reduction of the complexity of this new RS decoder from the previous approach [7].

Addition, compare with the previous approaches [10], the proposed algorithm is used to correct both error and erasure for RS decoder in this paper. It is presented to improve the Euclideanization of the BM algorithm developed by Eastman in [7], which algorithm derives from BM algorithm. Even though the extended version of the Euclidean algorithm showed by Sugiyama is adopted to solve Berlekamp's key equation, the weight coefficients can be extracted from different position and the linear combination to produce zero coefficients is performed in different location. Therefore, the proposed algorithm and the hardware implementation are not the same as the previous approaches [10].

Addition, compare with the previous approaches [11], the proposed algorithm is used to correct both error and erasure for RS decoder in this paper. It is presented to improve the Euclideanization of the BM algorithm developed by Eastman in [7], which algorithm derives from BM algorithm in an extended Euclidean, the weight coefficients can be extracted from different position and the linear combination to produce zero coefficients is performed in different location. Therefore, the proposed algorithm and the hardware implementation are not the same as the previous approaches [11].

There are two main computation units for the VLSI architecture of the proposed algorithm: the computation units of the errata evaluator polynomial (EE) and the errata

locator polynomial (EL). Each unit is constructed using only one simple type of processing element (PE). Therefore, this architecture is much simpler, modular and regular, and as such can be easily configured for various applications.

This paper is structured as follows. Section 2 introduces the time domain decoder for RS codes. In section 3, a brief review of Eastman's algorithm is presented. Next, the modified decoding algorithm to improve Eastman's decoder is derived in section 4. This proposed decoding algorithm has been verified by a software simulation using C++ program, which is illustrated in section 5. Section 6 shows the VLSI architecture design of the proposed algorithm. Finally, the conclusions are given in section 7.

2. TIME DOMAIN DECODER FOR RS CODES

Let C be a (n, k) RS code with minimum distance d over $GF(2^m)$, where $n = 2^m - 1$ is the block length, k is the number of m -bit message symbols and $d - 1$ is the number of parity symbols. Now denote the codeword polynomial, the error polynomial, and the erasure polynomial by $c(x) = \sum_{i=0}^{n-1} c_i x^i$, $e(x) = \sum_{i=0}^{n-1} e_i x^i$, and $\tau(x) = \sum_{i=0}^{n-1} \tau_i x^i$, respectively. Then the received polynomial can be expressed as $r(x) = \sum_{i=0}^{n-1} r_i x^i = c(x) + e(x) + \tau(x)$. The RS decoder is utilized to compute the error polynomial $e(x)$ and erasure polynomial $\tau(x)$. Once these polynomials are found, the corrected codeword $\hat{c}(x)$ can be obtained by subtracting $e(x)$ and $\tau(x)$ from the received polynomial $r(x)$. Suppose that v errors and s erasures occur in the received word $r(x)$ and assume that $s + 2v \leq d - 1$. Then, the maximum number of errors that can be corrected is $t = \lfloor (d - 1 - s)/2 \rfloor$, where $\lfloor x \rfloor$ denotes the greatest integer less than or equal to x .

The syndromes, which are computed from the received polynomial $r(x)$ can be expressed as

$$\begin{aligned} S_j &= r(\alpha^j) = c(\alpha^j) + e(\alpha^j) + \tau(\alpha^j) = e(\alpha^j) + \tau(\alpha^j) \\ &= \sum_{i=1}^v (Y_i \cdot X_i^j) + \sum_{k=1}^s (W_k \cdot Z_k^j), \text{ for } 0 \leq j \leq d - 2 \end{aligned} \quad (1)$$

where α is a primitive element in $GF(2^m)$ and Y_i , X_i , W_k , and Z_k are elements of $GF(2^m)$. In Eq. (1), Y_i and X_i are the i th unknown error value and unknown error location, respectively. Also, W_k and Z_k are the k th unknown error value and known error location, respectively.

The syndrome polynomial has the form

$$S(x) = \sum_{j=0}^{d-2} S_j x^j \quad (2)$$

where $S_j \in GF(2^m)$. If $S(x) = 0$, then $r(x)$ is a codeword and no further decoding is necessary. Otherwise, the RS decoder is needed to determine the error polynomial and erasure polynomial from the received polynomial $r(x)$.

Next, let $\rho(x)$ be the error locator polynomial with zeros at the inverse error locations, namely,

$$\rho(x) = \prod_{i=1}^{\nu} (1 + X_i x) = \sum_{i=0}^{\nu} \rho_i x^i \quad (3)$$

where $\rho_i \in GF(2^m)$, $\rho_0 = 1$, and $\deg\{\rho(x)\} = \nu$.

Also, let $\sigma(x)$ be the erasure locator polynomial with zeros at the inverse erasure locations. That is,

$$\sigma(x) = \prod_{j=1}^s (1 + Z_j x) = \sum_{j=0}^s \sigma_j x^j \quad (4)$$

where $\sigma_j \in GF(2^m)$, $\sigma_0 = 1$, and $\deg\{\sigma(x)\} = s$.

Finally, the errata locator polynomial $\Lambda(x)$ is computed as

$$\Lambda(x) = \rho(x) \cdot \sigma(x) = \sum_{i=0}^{s+\nu} \Lambda_i x^i \quad (5)$$

where $\Lambda_i \in GF(2^m)$ for $0 \leq i \leq s + \nu$, $\Lambda_0 = 1$, and $\deg\{\Lambda(x)\} = s + \nu$.

The Forney syndrome polynomial, introduced by Forney [5], is defined by

$$T(x) \equiv S(x) \cdot \sigma(x) \pmod{x^{d-1}} = \sum_{j=0}^{d-2} T_j x^j, \quad (6)$$

where $T_j \in GF(2^m)$ for $0 \leq j \leq d - 2$ is called the Forney syndromes.

Then the errata evaluator polynomial $\Omega(x)$, which is related to $S(x)$ and $\Lambda(x)$, is given as

$$\Omega(x) \equiv \Lambda(x) \cdot S(x) \pmod{x^{d-1}}, \quad (7)$$

where $\deg\{\Omega(x)\} = s + \nu - 1$.

This is known to be the Berlekamp's key equation. The crucial problem of the RS decoder is to find both $\Lambda(x)$ and $\Omega(x)$ from Eq. (7) under the condition $\deg\{\Omega(x)\} < \lfloor (d - 1 + s)/2 \rfloor$.

Once $\Lambda(x)$ and $\Omega(x)$ are obtained, the Chien search can be applied to find the roots of $\Lambda(x)$ which are the inverse locations of errors and erasures. And the error values Y_j and erasure values W_k can be computed from $\Omega(x)$ using Forney's algorithm, namely,

$$Y_i = \frac{\Omega(X_i^{-1})}{-X_i^{-1} \Lambda'(X_i^{-1})}, \quad \text{for } 1 \leq i \leq \nu, \quad (8)$$

and

$$W_k = \frac{\Omega(Z_k^{-1})}{-Z_k^{-1} \Lambda'(Z_k^{-1})}, \quad \text{for } 1 \leq k \leq s, \quad (9)$$

where $\Lambda'(Z_k^{-1})$ is the derivative of $\Lambda'(x)$ with respect to x , evaluated at $x = Z_k^{-1}$.

3. A BRIEF REVIEW OF EASTMAN'S ALGORITHM

In this section, Eastman's algorithm used to correct the RS codes is presented. First, recall that the extended version of the Euclidean algorithm is a recursive procedure to find the polynomials $r(x)$, $f(x)$, and $g(x)$, for given two known polynomials $m(x)$ and $n(x)$ with $\deg(m(x)) \geq \deg(n(x))$, such that

$$r(x) = f(x) \cdot [m(x)] + g(x) \cdot [n(x)] = \text{GCD}(m(x), n(x)). \quad (10)$$

At each iteration j , the relationship of these polynomials is maintained as

$$r^{(j)}(x) = f^{(j)}(x) \cdot [m(x)] + g^{(j)}(x) \cdot [n(x)] \quad (11)$$

where j denotes an iteration counter.

If the superscript j in Eq. (11) is replaced by a and b , one obtains the two equations as

$$r^{(a)}(x) = f^{(a)}(x) \cdot [m(x)] + g^{(a)}(x) \cdot [n(x)], \quad (12)$$

$$r^{(b)}(x) = f^{(b)}(x) \cdot [m(x)] + g^{(b)}(x) \cdot [n(x)]. \quad (13)$$

The expanded BM algorithm will preserve these Eqs. (12) and (13) and maintains the same operations in the recursive procedure. Now, choose $m(x)$ and $n(x)$ as

$$m(x) = 1 \quad (14)$$

$$n(x) = x \cdot S(x) = x \cdot \sum_{j=1}^{2t} S_j x^{j-1} = \sum_{j=1}^{2t} S_j x^j.$$

The corresponding polynomials for $r(x)$, $f(x)$, and $g(x)$ of the two initial Eqs. (12) and (13) as follows:

$$r^{(a)}(x) = 1, f^{(a)}(x) = 1, \text{ and } g^{(a)}(x) = 0 \quad (15)$$

and

$$r^{(b)}(x) = x \cdot S(x), f^{(b)}(x) = 0, \text{ and } g^{(b)}(x) = 1. \quad (16)$$

From an algebraic point of view, it is not difficult to verify, the degree of $f^{(b)}(x)$ is less than or equal to the degree of $g^{(b)}(x)$ and the polynomials $f^{(b)}(x)$ and $g^{(b)}(x)$ have the same operations in the recursive procedure. Then, one always has

$$g_i^{(b)} = 0, \text{ for all } i > l \quad (17)$$

and

$$f_i^{(b)} = 0, \text{ for all } i > l \quad (18)$$

where l denotes the degree of the $g^{(b)}(x)$.

At the j th iteration, let $h(x) = g^{(b)}(x) \cdot S(x)$. Since $j \geq l$, one has

$$h_j = \sum_{i=0}^j g_i^{(b)} \cdot S_{j-i} = \sum_{i=0}^l g_i^{(b)} \cdot S_{j-i} \quad (19)$$

and, by Eq. (11), one has

$$r_j^{(b)} = f_j^{(b)} \cdot [1] + h_j = 0 \cdot [1] + h_j = \sum_{i=0}^l g_i^{(b)} \cdot S_{j-i} = d_j \quad (20)$$

where d_j denotes, as usually, the discrepancy at the j th iteration needed in the conventional BM algorithm. Thus, the polynomial $r(x)$ can be used to omit the computation of the discrepancy at each iteration of the BM algorithm. In other words, the discrepancy at the j th iteration is equal to the coefficient r_j of x^j in $r^{(b)}(x)$. This is a basic idea of Eastman, which is to expand the BM algorithm in a Euclidean context in order to avoid the computation of the discrepancy.

4. THE PROPOSED ALGORITHM

The advantage of the Eastman's algorithm is that the computations of the discrepancy and the finite field element inversion needed in the conventional BM algorithm are completely avoided. However, the separate computations of the errata locator and the errata evaluator polynomials are still required. In other word, one more multiplication operation is needed to obtain the errata evaluator polynomial.

Now, based on the idea of Eastman as given above, a modified Euclideanized BM algorithm is proposed to obtain the errata locator and the errata evaluator polynomials simultaneously as follow.

In 1975, Sugiyama showed that the extended version of the Euclidean algorithm can be used to solve Berlekamp's key equation for RS codes. To adapt the Euclidean algorithm, Berlekamp's key Eq. (7) can be re-expressed as

$$\Omega(x) \equiv \Theta(x) \cdot [x^{d-1}] + \Lambda(x) \cdot [S(x)]. \quad (21)$$

The errata locator polynomial $\Lambda(x)$ and the errata evaluator polynomial $\Omega(x)$ can be obtained for solving the Eq. (21). To adopt the operations of equations, the polynomials $r(x)$, $f(x)$, and $g(x)$ in Eqs. (11) to (13) and Eqs. (15) to (16) are now replaced by $\Omega(x)$, $\Theta(x)$, and $\Lambda(x)$, respectively. From Eqs. (11) and (21), the $m(x)$ and $n(x)$ are changed as

$$\begin{aligned} m(x) &= x^{d-1} \\ n(x) &= S(x) = \sum_{j=0}^{2t-1} S_j x^j. \end{aligned} \quad (22)$$

By initializing the Euclideanized BM algorithm with the erasure locator polynomial and the Forney syndromes, the errata locator and the errata evaluator polynomials can be obtained simultaneously. Thus, the corresponding polynomials of these two initial Eqs. (12) and (13) are defined as

$$\Omega^{(a)}(x) = x^{d-1}, \Theta^{(a)}(x) = 1, \Lambda^{(a)}(x) = 0 \quad (23)$$

and

$$\Omega^{(b)}(x) = T(x), \Theta^{(b)}(x) = 0, \Lambda^{(a)}(x) = \sigma(x). \quad (24)$$

Assume at the j th iteration, the two known equation are

$$\Omega^{(a)}(x) = \Theta^{(a)}(x) \cdot x^{d-1} + \Lambda^{(a)}(x) \cdot S(x), \quad (25)$$

$$\Omega^{(b)}(x) = \Theta^{(b)}(x) \cdot x^{d-1} + \Lambda^{(b)}(x) \cdot S(x). \quad (26)$$

Then, the third equation can be obtained by taking the direct linear combination of Eqs. (25) and (26) as

$$\Omega^{(c)}(x) = \Theta^{(c)}(x) \cdot x^{d-1} + \Lambda^{(c)}(x) \cdot S(x). \quad (27)$$

In Eq. (27), the operations of the linear combination are given by

$$\begin{aligned} \Omega^{(c)}(x) &= \gamma \cdot \Omega^{(a)}(x) + \delta \cdot \Omega^{(b)}(x) \\ \Theta^{(c)}(x) &= \gamma \cdot \Theta^{(a)}(x) + \delta \cdot \Theta^{(b)}(x) \\ \Lambda^{(c)}(x) &= \gamma \cdot \Lambda^{(a)}(x) + \delta \cdot \Lambda^{(b)}(x) \end{aligned} \quad (28)$$

where the field elements δ and γ , which are the weights of the linear combination, represent the old discrepancy and the new discrepancy, respectively. At each iteration, the elements δ and γ can be obtained by extracting the coefficients of polynomials $\Omega^{(a)}(x)$ and $\Omega^{(b)}(x)$, respectively in Eq. (28).

After computing Eq. (28), there are two possibilities according to the value of γ . If $\gamma \neq 0$, one will perform the updated operation as follows:

$$\begin{aligned} \Omega^{(a)}(x) &\leftarrow \Omega^{(b)}(x), \Lambda^{(a)}(x) \leftarrow \Lambda^{(b)}(x) \\ \hat{l} &\leftarrow (j - \hat{l}), \delta \leftarrow \gamma \end{aligned} \quad (29)$$

and

$$\Omega^{(b)}(x) \leftarrow \Omega^{(c)}(x), \Lambda^{(b)}(x) \leftarrow \Lambda^{(c)}(x). \quad (30)$$

On the other hand, if $\gamma = 0$, the updated operation Eq. (29) will be omitted to avoid obtaining zero polynomials. To control this flow, in a similar way to the usual BM algorithm, one introduces the binary signal sw to switch the updated conditions; that is, $sw = (\gamma = 0) \vee (j \leq 2\hat{l})$, where \hat{l} denotes an auxiliary index. If $j = d - 1 - s$, then the quantities $\Omega(x) = \Omega^{(c)}(x)$ and $\Lambda(x) = \Lambda^{(c)}(x)$ are the errata evaluator polynomial and the errata locator polynomial, respectively. The detailed steps of this modified algorithm are given explicitly by:

Algorithm

Step 1: Initial setting:

$$\begin{aligned} \text{(a)} \quad \Omega^{(a)}(x) &= x^{d-1}, \Lambda^{(a)}(x) = 0, \Omega^{(b)}(x) = T(x), \Lambda^{(b)}(x) = \sigma(x), \\ j &= 0, l = s, \delta = 1, \gamma = 0, \hat{l} = 0. \end{aligned} \quad (31)$$

Step 2: Compute the followings:

$$\begin{aligned} \text{(a)} \quad j &= j + 1, \\ \Omega^{(a)}(x) &\leftarrow \Omega^{(a)}(x) \cdot x, \Lambda^{(a)}(x) \leftarrow \Lambda^{(a)}(x) \cdot x. \end{aligned} \quad (32)$$

$$(b) \gamma = \Omega_{j-1+s}^{(b)}. \quad (33)$$

$$(c) \begin{aligned} \Omega^{(c)}(x) &= \gamma \cdot \Omega^{(a)}(x) + \delta \cdot \Omega^{(b)}(x), \\ \Lambda^{(c)}(x) &= \gamma \cdot \Lambda^{(a)}(x) + \delta \cdot \Lambda^{(b)}(x). \end{aligned} \quad (34)$$

(d) If $j \neq d - 1 - s$, go to (e); otherwise, stop.

(e) $sw = (\gamma = 0) \vee (j \leq 2 \cdot \hat{l})$.

If $sw \neq 0$, go to (f); otherwise,

$$\begin{aligned} \Omega^{(a)}(x) &\leftarrow \Omega^{(b)}(x), \Lambda^{(a)}(x) \leftarrow \Lambda^{(b)}(x), \\ \hat{l} &\leftarrow (j - \hat{l}), \delta \leftarrow \gamma. \end{aligned} \quad (35)$$

$$(f) \Omega^{(b)}(x) \leftarrow \Omega^{(c)}(x), \Lambda^{(b)}(x) \leftarrow \Lambda^{(c)}(x). \quad (36)$$

Step 3: Go to step 2.

5. PROGRAM IMPLEMENTATION AND SIMULATION RESULTS

To point out that the computation of the discrepancy at each iteration can be omitted in above algorithm in section 4, the following is an example of an RS code over $GF(2^4)$.

Example: Let α be the generator of the multiplicative group of nonzero elements of $GF(2^4)$ satisfying $\alpha^4 + \alpha + 1 = 0$. Consider a (15, 9) RS code over $GF(2^4)$ with a minimum distance $d = 7$. In this case, $s = 1$ erasures and $v = 2$ errors occurring in the received word satisfy the condition $s + 2v \leq d - 1$. The generator polynomial of this (15, 9) RS code is defined by

$$g(x) = \prod_{i=0}^5 (x - \alpha^i) = x^6 + \alpha^{10}x^5 + \alpha^{14}x^4 + \alpha^4x^3 + \alpha^6x^2 + \alpha^9x + \alpha^6.$$

Let the codeword vector, which is a multiple of $g(x)$, be the following vector

$$c = (\alpha^{10}, \alpha^9, \alpha^8, \alpha^{11}, \alpha^3, \alpha^7, \alpha^{14}, \alpha^{13}, \alpha^4, \alpha^3, \alpha^8, \alpha^{11}, \alpha^{10}, \alpha^0, \alpha^{10}).$$

Assume the erasure vector is as follow $\tau = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, \alpha^7, 0, 0)$. And, the error vector is $e = (0, 0, 0, 0, 0, \alpha^{11}, 0, 0, 0, 0, 0, 0, 0, \alpha^{13}, 0)$ and the errata vector is $\tilde{e} = \tau + e = (0, 0, 0, 0, 0, \alpha^{11}, 0, 0, 0, 0, 0, 0, 0, \alpha^7, \alpha^{13}, 0)$. Then, the received vector is

$$r = c + \tilde{e} = (\alpha^{10}, \alpha^9, \alpha^8, \alpha^{11}, \alpha^3, \alpha^8, \alpha^{14}, \alpha^{13}, \alpha^4, \alpha^3, \alpha^8, \alpha^{11}, \alpha^6, \alpha^6, \alpha^{10}).$$

By Eq. (1) (resp. Eq. (4)), the syndrome polynomial (resp. the erasure locator polynomial) is computed as $S(x) = \alpha^1x^5 + \alpha^0x^4 + \alpha^9x^3 + \alpha^5x^2 + \alpha^8x + \alpha^3$ (resp. $\sigma(x) = \alpha^2x + \alpha^0$). Then, the Forney syndrome polynomial is computed by Eq. (6) as

$$T(x) = \alpha^5x^5 + \alpha^{12}x^4 + \alpha^0x^3 + \alpha^0x^2 + \alpha^4x + \alpha^3.$$

The initial conditions and the detailed data at each iteration are shown in Table 1. The quantities j and \hat{l} , denote the iteration counter and the auxiliary index, respectively.

Table 1. Detail data at each iteration for example.

j	\hat{l}	δ	γ	$\hat{\delta}$	d_j	$\Omega(x)$								$\Lambda(x)$							
						7	6	5	4	3	2	1	0	4	3	2	1	0			
0	0	0	-	0	-	$\Omega^{(a)}(x)$	0											$\Lambda^{(a)}(x)$			
						$\Omega^{(b)}(x)$			5	12	0	0	4	3				2	0	$\Lambda^{(b)}(x)$	
						$\Omega^{(c)}(x)$														$\Lambda^{(c)}(x)$	
1	0	0	4	0	4	$\Omega^{(a)}(x)$	0											$\Lambda^{(a)}(x)$			
						$\Omega^{(b)}(x)$			5	12	0	0	4	3				2	0	$\Lambda^{(b)}(x)$	
						$\Omega^{(c)}(x)$			5	12	0	0	4	3				2	0	$\Lambda^{(c)}(x)$	
2	1	4	0	4	0	$\Omega^{(a)}(x)$	5	12	0	0	4	3				2	0	$\Lambda^{(a)}(x)$			
						$\Omega^{(b)}(x)$			5	12	0	0	4	3				2	0	$\Lambda^{(b)}(x)$	
						$\Omega^{(c)}(x)$			8	4	1		13	7			2	13	4	$\Lambda^{(c)}(x)$	
3	1	4	1	4	1	$\Omega^{(a)}(x)$	5	12	0	0	4	3			2	0		$\Lambda^{(a)}(x)$			
						$\Omega^{(b)}(x)$			8	4	1		13	7			2	13	4	$\Lambda^{(b)}(x)$	
						$\Omega^{(c)}(x)$			13	10		4	2	11		3	11	2	8	$\Lambda^{(c)}(x)$	
4	2	1	10	1	10	$\Omega^{(a)}(x)$	8	4	1		13	7			2	13	4	$\Lambda^{(a)}(x)$			
						$\Omega^{(b)}(x)$			13	10		4	2	11		3	11	2	8	$\Lambda^{(b)}(x)$	
						$\Omega^{(c)}(x)$						4	6	12		6	9	0	9	$\Lambda^{(c)}(x)$	
5	2	1	-	1	-	$\Omega^{(a)}(x)$	8	4	1		13	7		2	13	4		$\Lambda^{(a)}(x)$			
						$\Omega^{(b)}(x)$						4	6	12		6	9	0	9	$\Lambda^{(b)}(x)$	
						$\Omega^{(c)}(x)$						5	7	13		7	10	1	10	$\Lambda^{(c)}(x)$	
Normalized						$\Omega(x)$							10	12	3		12	0	6	0	$\Lambda(x)$

Both of them are positive integers. The field element δ (resp. γ), which plays the role of the old discrepancy (resp. the new discrepancy) is obtained from directly extracting the coefficient of x^j in the polynomial $\Omega^{(a)}(x)$ (resp. $\Omega^{(b)}(x)$) at each iteration. In Table 1, we list two extra field elements $\hat{\delta}$ and d_j , which are also the old discrepancy and the new discrepancy, respectively. And they are obtained using the computation of Eq. (20). For all of the other entries in Table 1, each integer i denotes the exponential of the field element α^i in $GF(2^4)$.

At the iteration $j = 1$ st, the new discrepancy is as

$$\gamma = \Omega_{j-1+s}^{(b)} = \Omega_1^{(b)} = \alpha^4,$$

$$d_1 = \sum_{i=0}^l \Lambda_i^{(b)} \cdot S_{j-i} = \alpha^0 \cdot \alpha^8 + \alpha^2 \cdot \alpha^3 = \alpha^8 + \alpha^5 = \alpha^4.$$

At $j = 2$ nd iteration, the new discrepancy is as

$$\gamma = \Omega_{j-1+s}^{(b)} = \Omega_2^{(b)} = \alpha^0,$$

$$d_2 = \sum_{i=0}^l \Lambda_i^{(b)} \cdot S_{j-i} = \alpha^0 \cdot \alpha^5 + \alpha^2 \cdot \alpha^8 = \alpha^5 + \alpha^{10} = \alpha^0.$$

At $j = 3$ rd iteration, the new discrepancy is as

Now, the proposed decoding method has been verified using a C++ program, which is implemented to correct s erasures and v errors satisfying $s + 2v \leq 16$ for the (255, 239) RS code over $GF(2^8)$. Table 2 shows the computation times in milliseconds for the proposed algorithm. During the simulation, the erasure and error patterns are randomly generated over all possible locations and magnitudes for the (255, 239) RS code. Each number inside this table is averaged over 100 computations of finding the errata locator and errata evaluator polynomials.

6. VLSI ARCHITECTURE DESIGN FOR THE PROPOSED ALGORITHM

The proposed algorithm described in section 4 is mapped to a parallel and regular VLSI architecture as given in Fig. 1. Due to the fact that both the errata evaluator and

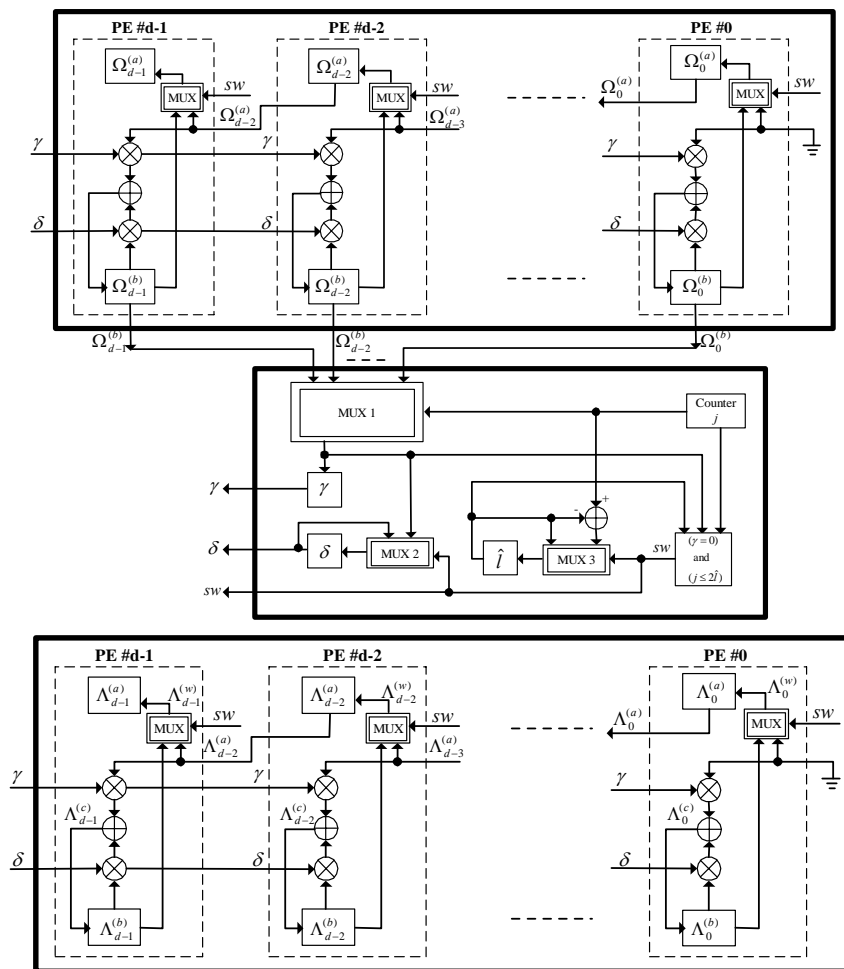


Fig. 1. The VLSI architecture for the proposed algorithm in section 4 to find the errata locator and the errata evaluator polynomials.

errata locator polynomials will be obtained simultaneously, there are two major recursive computation units: the errata evaluator polynomial computation unit (EE) and errata locator polynomial computation unit (EL). Each unit with one-dimensional parallel structure is constructed using only one simple type of processing element (PE). In the proposed architecture, the number of PE is $2d$ used to obtain the errata evaluator and errata locator polynomials simultaneously. Besides these two major portions, there is one controlling unit.

First, the initial condition $\Omega^{(a)}(x) = x^{d-1}$, $\Omega^{(b)}(x) = T(x)$ and $\Lambda^{(a)}(x) = 0$, $\Lambda^{(b)}(x) = \sigma(x)$ is fed into the EE block and EL block, respectively. And, the parameters $j = 0$, $\delta = 1$, $\gamma = 0$, and $\hat{l} = 0$ are fed into the controlling unit. The controlling signal sw is a 1-bit signal which is used to control the multiplexer to determine whether or not the polynomials $\Omega^{(a)}(x)$ and $\Lambda^{(a)}(x)$ and the parameters δ and \hat{l} are updated; see step 2(e) of algorithm in section 4. All other signals in this controlling unit are m -bit field elements. A register transfer language (RTL) of the proposed algorithm is the following.

T_0 : $\Omega^{(a)} \leftarrow x^{d-1}$, $\Omega^{(b)} \leftarrow T(x)$, $\Lambda^{(a)} \leftarrow 0$, $\Lambda^{(b)} \leftarrow \sigma(x)$, $j \leftarrow 0$, $\delta \leftarrow 1$, $\gamma \leftarrow 0$, $\hat{l} \leftarrow 0$, and $sw \leftarrow 0$.

T_1 : $j \leftarrow j + 1$, $\gamma \leftarrow \Omega_{j-1+s}^{(b)}$ and simultaneously $\Omega_{i+1}^{(a)} \leftarrow \Omega_i^{(a)}$ for $0 \leq i \leq d-2$.

T_2 : Compute $\Omega_i^{(c)} \leftarrow \gamma \cdot \Omega_i^{(a)} + \delta \cdot \Omega_i^{(b)}$ and $\Lambda_i^{(c)} \leftarrow \gamma \cdot \Lambda_i^{(a)} + \delta \cdot \Lambda_i^{(b)}$ for $0 \leq i \leq d-1$.

T_3 : If $j = d-1-s$ stop. Otherwise, $sw \leftarrow (\gamma = 0) \vee (j \leq 2 \cdot \hat{l})$. If $sw = 0$, $\Omega_i^{(a)} \leftarrow \Omega_i^{(b)}$, $\Lambda_i^{(a)} \leftarrow \Lambda_i^{(b)}$ for $0 \leq i \leq d-1$, $\delta \leftarrow \gamma$, and $\hat{l} \leftarrow (j - \hat{l})$.

T_4 : $\Omega_i^{(b)} \leftarrow \Omega_i^{(c)}$, $\Lambda_i^{(b)} \leftarrow \Lambda_i^{(c)}$ for $0 \leq i \leq d-1$, then go to T_1 .

7. CONCLUSION

In Eastman's decoder [7], a two-dimensional systolic array implementation is adopted to obtain the errata locator polynomial. The number of processing element is $d^2 - d$ in Eastman's architecture. After the errata locator polynomial is obtained, one more multiplication operation is needed to compute the errata evaluator polynomial by Eq. (7). Thus, the hardware implement developed by Eastman is not only larger but also slower.

In this paper, a decoding algorithm given in section 4 is proposed to reduce the complexity of algorithm developed by Eastman [7]. Through the proposed algorithm, not only the errata locator and the errata evaluator polynomials can be obtained simultaneously, but also the computations of the polynomial division, discrepancy, and finite field element inversion were all avoided. Moreover, the computation units of the EE and the EL are constructed using the same simple type of processing element (PE), which number is $2d$. Consequently, these features reduced the complexity of the decoding algorithm, resulting in a simple and modular architecture.

REFERENCES

1. Telemetry Channel Coding, Recommendation for Space Data System Standard, 101.0-B-3, Blue Book, Issue 3, May 1992.

2. S. B. Wicker and V. K. Bhargava, *Reed-Solomon Codes and their Application*, IEEE Press, New York, 1994.
3. C. Basile, "The U.S. HDTV standard the grand," *IEEE Spectrum*, Vol. 32, 1995, pp. 36-45.
4. H. C. Chang, C. B. Shung, and C. Y. Lee, "A reed-Solomon product-code (RS-PC) decoder chip for DVD applications," *IEEE Journal of Solid-State Circuits*, Vol. 36, 2001, pp. 229-238.
5. J. R. Forney, "On decoding BCH codes," *IEEE Transactions on Information Theory*, Vol. IT-11, 1965, pp. 549-557.
6. R. E Blahut, *Theory and Practice of Error Control Codes*, Reading Mass, Addison Wesley, MA, 1984.
7. W. L. Eastman, "Euclideanization of the Berlekamp-Massey algorithm," in *Proceedings of the Tactical Communication Conference*, Vol. 1, 1988, pp. 295-303.
8. T. K. Truong, J. H. Jeng, and T. C. Cheng, "A new decoding algorithm for correcting both erasures and errors of reed-Solomon codes," *IEEE Transactions on Communications*, Vol. 51, 2003, pp. 381-388.
9. Y. Sugiyama, M. Kasahara, S. Hirasawa, and T. Namekawa, "A method for solving key equation for decoding Goppa codes," *Information and Control*, Vol. 27, 1975, pp. 87-99.
10. Y. W. Chang, T. K. Truong, and J. H. Jeng, "VLSI architecture of modified Euclidean algorithm for reed-Solomon code," *Information Sciences*, Vol. 155, 2003, pp. 139-150.
11. D. V. Sarwate and N. R. Shanbhag, "High-speed architecture for reed-Solomon decoders," *IEEE Transactions on VLSI Systems*, Vol. 9, 2001, pp. 641-655.



Huang-Chi Chen (陳煌琦) received M.S. degrees from Department of Electrical Engineering of I-Shou University, Kaohsiung Country, Taiwan, in 1998. He is currently a Lecturer of Department of Management Information System of Far-East University, and he entered the Ph.D. program in Department of Electrical Engineering of I-Shou University in 2006. His research interests include error correct codes, image process and machine learning.



Yu-Wen Chang (張玉雯) received the Ph.D. degree in Electrical Engineering from I-Shou University, Kaohsiung County, Taiwan, in year 2006. She joined the faculty of the Department of Electronic Communication Engineering, National Kaohsiung Marine University in year 2007. Her research interests include error correct codes, image process, VLSI, and fuzzy control.



Rey-Chue Hwang (黃瑞初) received the Ph.D. degree in Electrical Engineering from Southern Methodist University, Dallas, Texas, U.S.A. in year 1993. He joined the faculty of the Department of Electrical Engineering, I-Shou University at the same year. Dr. Hwang received the outstanding teaching award of ISU in year 2000. He was the co-chair of the IEEE CIS Tainan Chapter in 2005-2007. He also served as the member of a council for the Taiwanese Fuzzy System Association in 2005-2006. Currently, he serves as the Chair of the IEEE CIS Tainan Chapter. He is also the director of General Educational Center of I-Shou University now. His research interests include artificial neural networks, intelligent systems, and fuzzy control. He has published over 140 scientific papers in international and national journals or referred conference proceedings.