PAPER

Design and Implementation of a Low-Complexity Reed-Solomon Decoder for Optical Communication Systems

Ming-Der SHIEH[†a)], *Member and* **Yung-Kuei LU**[†], *Student Member*

SUMMARY A low-complexity Reed-Solomon (RS) decoder design based on the modified Euclidean (ME) algorithm proposed by Truong is presented in this paper. Low complexity is achieved by reformulating Truong's ME algorithm using the proposed polynomial manipulation scheme so that a more compact polynomial representation can be derived. Together with the developed folding scheme and simplified boundary cell, the resulting design effectively reduces the hardware complexity while meeting the throughput requirements of optical communication systems. Experimental results demonstrate that the developed RS(255, 239) decoder, implemented in the TSMC 0.18 μm process, can operate at up to 425 MHz and achieve a throughput rate of 3.4 Gbps with a total gate count of 11,759. Compared to related works, the proposed decoder has the lowest area requirement and the smallest area-time complexity.

key words: *channel decoder, modified euclidean algorithm, optical communication, Reed-Solomon codes, VLSI architectures*

1. Introduction

Reed-Solomon (RS) codes, which have an excellent capability of correcting burst errors, are widely applied in storage and digital communication systems. In general, a syndrome-based RS decoder consists of three main blocks: the syndrome computation (SC) unit, the key equation solver (KES) unit, and the Chien search and error evaluation (CSEE) unit. The SC unit computes the syndrome polynomial from the received word. The error locator and error evaluator polynomials are then determined by solving the key equation in the KES unit. The two polynomials are passed to the CSEE unit to find the error locations and values by using Chien search and Forney algorithms, respectively. Together with a first-in first-out (FIFO) buffer that compensates for the accumulated latency in the three units, the received word is corrected according to the computed error locations and values when it is being read out of the decoder.

Solving the key equation to find the error locator and error evaluator polynomials is recognized as the most critical part in the design of RS decoders. Two main decoding algorithms are commonly adopted to solve the key equation: the modified Euclidean (ME) algorithms [1]–[7] and the Berlekamp-Massey (BM) algorithms [8]–[13]. Conventionally, architectures based on the BM algorithm for solving the key equation have the advantage of low computational complexity. Compared with the BM algorithm, the ME algorithm usually leads to a more regular architecture with a higher computational complexity and hardware requirement. Recently, Truong et al. presented a fast ME algorithm [2], denoted as the TME algorithm hereafter, which greatly reduces complexity by eliminating the need for degree computation and comparison.

Since the RS(255,239) code can provide about 5.5 dB coding gain for error correction and reduce the bit error rate from 10^{-4} to 10^{-15}, it is adopted for the submarine fiber-optic system [14]. To meet the requirements of optical systems, an RS(255,239) decoder design employing pipelined multipliers to reduce the critical path delay and parallel architecture to obtain a higher throughput rate was presented in [3]. However, the architecture is not cost-effective and requires a larger number of clock cycles to solve the key equation. In recent years, three area-efficient folded architectures developed using the ME algorithm have been presented [4]–[6]. In [4], Lee employed the pipelined recursive method to obtain a high throughput rate. The basic idea of the design in [5] is to use a pre-computation method to eliminate the idle time of the KES unit; the hardware cost is reduced by employing folding techniques. Based on Lee's work, the authors in [6] removed the need for the degree computation circuit and shortened the critical path delay.

In this paper, an area-efficient RS decoder is presented based on the TME algorithm. The decoder can operate at high data rates with a critical path delay of $T_{\text{mult}} + T_{\text{ff}}$, where T_{mult} and T_{ff} denote the delays of the finite-field multiplier (FFM) and flip-flop, respectively. To reduce hardware complexity, an efficient polynomial manipulation scheme based on our previous work [20] is presented to remove redundant information in the polynomial representation of the TME algorithm. Applying the boundary cell simplification and folding techniques, the proposed RS(255,239) decoder obtains a 36 % reduction in hardware requirement and has a better area-time complexity compared to those in [6]. Experimental results show that the proposed RS(255, 239) decoder can operate at up to 425 MHz with a total gate count of 11,759 based on the TSMC 0.18 μm process.

The rest of this paper is organized as follows. Section 2 reviews the background and defines the notation used in this work. The proposed architecture is presented in Sect. 3. In Sect. 4, the performance evaluation is given and a comparison with related studies is made. Finally, conclusions are given in Sect. 5.

Manuscript received September 2, 2010.
Manuscript revised February 19, 2011.

[†]The authors are with Department of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan, R.O.C.
a) E-mail: shiehm@mail.ncku.edu.tw

DOI: 10.1587/transinf.E94.D.1557

Copyright © 2011 The Institute of Electronics, Information and Communication Engineers

2. Background and Notation

This section gives a brief review of RS decoding and summarizes the TME algorithm presented in [2], which forms the basis of this work.

2.1 RS Decoding

A primitive RS(n, k) code is defined over GF(2^m), where $n = 2^m - 1$ and k are the codeword and information lengths, respectively, with each symbol m bits wide. The error-correcting capability of RS codes is $t = \lfloor (n-k)/2 \rfloor$, where the floor function $\lfloor x \rfloor$ denotes the greatest integer less than or equal to x. Let $C(x)$ and $R(x)$ denote the transmitted codeword polynomial and received word polynomial, respectively. Assume that e errors were induced by channel noise during transmission, and then we have

$$R(x) = C(x) + E(x) = \sum_{j=0}^{n-1} r_j x^j \tag{1}$$

where $E(x)$ represents the error polynomial. In the decoding process, the first step is to compute the syndrome:

$$s_i = R\left(\alpha^{m_0+i}\right) = E\left(\alpha^{m_0+i}\right),\ 0 \le i < 2t, \tag{2}$$

where α is the primitive element of GF(2^m) and m_0 is typically zero or one. If $2t$ syndromes are all zero, it is assumed that no errors have occurred. Otherwise, the syndrome polynomial is constructed as:

$$S(x) = \sum_{i=0}^{2t-1} s_i x^i. \tag{3}$$

Then, the error values and error locations are calculated by solving the key equation:

$$\Lambda(x) S(x) \equiv \Omega(x) \bmod x^{2t} \tag{4}$$

where the error locator polynomial $\Lambda(x)$ and error evaluator polynomial $\Omega(x)$ are defined as:

$$\Lambda(x) = \prod_{j=1}^{e} \left(1 - X_j x\right) \tag{5}$$

$$\Omega(x) = \sum_{i=1}^{e} Y_i X_i^{m_0} \prod_{j=1, j\neq i}^{e} \left(1 - X_j x\right) \tag{6}$$

The inverse of the roots of $\Lambda(x), X_1, X_2, \ldots, X_e$, are called error locators; i.e., if error locations are $x^{i_1}, x^{i_2}, \ldots, x^{i_e}$, then $X_1 = \alpha^{i_1}, X_2 = \alpha^{i_2}, \ldots, X_e = \alpha^{i_e}$. Corresponding to these error locations, $Y_1, Y_2, \ldots, Y_e$ in $\Omega(x)$ are the error values. Once $\Lambda(x)$ and $\Omega(x)$ are calculated, the decoder can use the Chien search [15], conducted by checking whether $\Lambda(\alpha^{-j}) = 0$ for each $0 \le j \le n-1$, to find error locators X_i. Error values can be determined using the Forney algorithm [16]:

$$Y_i = -\frac{\Omega(X_i^{-1})}{X_i^{m_0-1}\Lambda'(X_i^{-1})} = -\left.\frac{x^{m_0}\Omega(x)}{x\Lambda'(x)}\right|_{x=\alpha^{-j}} \tag{7}$$

where $\Lambda'(x)$ is the formal derivative of $\Lambda(x)$. The calculated error values are subtracted from the corresponding erroneous symbols to complete the decoding process.

2.2 TME Algorithm

The proposed architecture is based on the TME algorithm [2]. For completeness, the TME algorithm is briefly reviewed below. Based on the Euclidean algorithm, Truong et al. proposed a decoding algorithm that uses the mechanism of the inverse-free BM algorithm [12] to avoid polynomial division and the computation of discrepancy in conventional Euclidean and BM algorithms, respectively. The TME algorithm is stated as follows:

TME Algorithm
Initialization:

$\Omega^{(a)}(x) = x^{2t}, \Omega^{(b)}(x) = S(x), \Lambda^{(a)}(x) = 0, \Lambda^{(b)}(x) = 1,$
$k = 0, l = 0;$

TME.1:

$$\Omega^{(b)}(x) = x\Omega^{(b)}(x), \Lambda^{(b)}(x) = x\Lambda^{(b)}(x); \tag{8}$$

$$\Omega^{(c)}(x) = u\Omega^{(a)}(x) + v\Omega^{(b)}(x) \tag{9}$$
$$\Lambda^{(c)}(x) = u\Lambda^{(a)}(x) + v\Lambda^{(b)}(x)$$

If $u \neq 0$ and $2l \le k$ **then**

$$\Omega^{(a)}(x) = \Omega^{(b)}(x), \Lambda^{(a)}(x) = \Lambda^{(b)}(x), l = k + 1 - l; \tag{10}$$

$$\Omega^{(b)}(x) = \Omega^{(c)}(x), \Lambda^{(b)}(x) = \Lambda^{(c)}(x); \tag{11}$$

TME.2:
$k = k + 1$. **If** $k \le 2t - 1$, **then** go to **TME.1**.
Output: $\Omega^{(b)}(x), \Lambda^{(b)}(x)$.

The variables u and v are the $(2t)$-th coefficients, i.e., leading coefficients, of $\Omega^{(b)}(x)$ and $\Omega^{(a)}(x)$, respectively. The superscripts a/b denote the previous/current states of polynomials. The TME algorithm produces a zero coefficient at the x^{2t} term of $\Omega^{(c)}(x)$ by performing (8) and (9) in each iteration. Together with the control mechanism of $u \neq 0$ and $2l \le k$ in (10), the TME algorithm removes the need for the degree computation and comparison in conventional ME algorithms [1]. After $2t$ iterations, the desired $\Omega^{(b)}(x)$ and $\Lambda^{(b)}(x)$ are obtained simultaneously, and the results can be directly used in the Chien search and the Forney algorithm for finding the error locators and the error values [2].

Based on the TME algorithm, Truong et al. presented a regular KES architecture [7]. This architecture can operate at high data rates with a critical path delay of $T_{\text{mult}} + T_{\text{add}} + T_{\text{ff}}$, where T_{add} is the delay of the finite-field adder (FFA). However, the main drawback of this architecture is its high hardware requirement of $4t + 2$ basic cells for a t-error-correcting RS code. In the following, we will show how to

reduce the number of required basic cells in [7] from $4t + 2$ to $3t$ with the proposed polynomial manipulation scheme.

3. Cost-Effective RS Decoder

In the TME algorithm, as the iterations proceed, we observe that the length of consecutive zero coefficients of $\Omega^{(b)}(x)$ grows incrementally, starting from the constant term. These zero coefficients can be discarded because they are not used in later iterations. That is, redundant zero coefficients can be removed for efficient hardware implementation. Moreover, assume that both the SC and the CSEE units of the pipelined designs are operated in a serial manner. This implies that 255 clock cycles are needed to complete their tasks for an RS(255,239) code. In contrast, the KES unit takes $2t = 16$ clock cycles to perform the TME algorithm. Compared with the SC and the CSEE units, the KES unit has a large amount of idle time; therefore, the folding technique can be employed to improve the utilization of the KES unit.

3.1 SC Unit

The $2t$ syndromes s_i computed based on the received word polynomial $R(x)$ can be derived using Horner's rule as:

$$s_i = \left(\ldots\left(\left(r_{n-1}\alpha^i + r_{n-2}\right)\alpha^i + r_{n-3}\right)\ldots\right)\alpha^i + r_0, \qquad (12)$$

One register, one constant FFM, one multiplexer, and one FFA are needed to calculate each syndrome. Figure 1 shows the basic cell and the overall architecture of the SC unit. At the end of n clock cycles, the $2t$ syndromes are obtained and then sent into the KES unit for finding $\Lambda(x)$ and $\Omega(x)$. Note that a constant FFM has much smaller complexity than a full FFM does.

3.2 KES Unit

3.2.1 Proposed Polynomial Manipulation Scheme

To facilitate hardware implementation, it is assumed that except the first leading zero coefficient of the polynomials, the following zero coefficients, if any, are reserved for the next iteration. That is, only one leading zero coefficient is removed in each iteration. For example, if $\Omega^{(b)}(x) = 0x^{16} + 0x^{15} + x^{14} + \ldots + \alpha x + \alpha^2$ is produced by performing (8), (9), and (11) in the k-th iteration, the degree of $\Omega^{(b)}(x)$ is regarded as 15 for the next iteration.

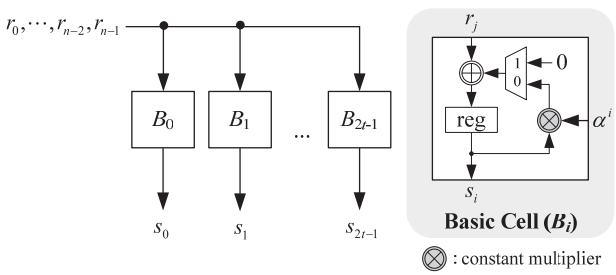


Fig. 1 SC unit and its i-th syndrome cell (B_i).

Let the notations $\underline{\Omega}(x)$ and $\underline{\Lambda}(x)$ denote the error evaluator and error locator polynomials in the conventional ME algorithm [1], respectively, and $\deg(F(x))$ represent the degree of $F(x)$. Note that $\underline{\Omega}(x)$ and $\underline{\Lambda}(x)$, respectively, can be used instead of $\Omega(x)$ and $\Lambda(x)$ defined in (4) to obtain the error locations and the corresponding error values when performing the Chien search and the Forney algorithm. Before describing the proposed approach, we first introduce two features that holds during the iterations of the conventional ME algorithm: (i) As stated in [17], the sum of $\deg(\underline{\Omega}^{(a)}(x))$ and $\deg(\underline{\Lambda}^{(b)}(x))$ remains $2t$ when performing the conventional ME algorithm; (ii) in study [1], it was shown that the sum of $\deg(\underline{\Omega}^{(a)}(x))$ and $\deg(\underline{\Omega}^{(b)}(x))$ is $4t - 1 - k$ at the beginning of iteration k.

Based on features (i) and (ii), we deduce $\deg(\underline{\Omega}^{(b)}(x)) - \deg(\underline{\Lambda}^{(b)}(x)) = 2t - 1 - k$. Since $\deg(\Omega^{(b)}(x)) = 2t - 1$ and $\deg(\Lambda^{(b)}(x)) = k$ at the beginning of the k-th iteration in the TME algorithm for $0 \le k \le 2t$, we can further derive $\deg(\Omega^{(b)}(x)) - \deg(\underline{\Omega}^{(b)}(x)) = \deg(\Lambda^{(b)}(x)) - \deg(\underline{\Lambda}^{(b)}(x))$. This implies that the left-shift operation in (8) has the same effect on both $\Omega^{(b)}(x)$ and $\Lambda^{(b)}(x)$ during iterations of the TME algorithm, i.e., $\Omega^{(b)}(x) = x^{p(k)}\underline{\Omega}^{(b)}(x)$ and $\Lambda^{(b)}(x) = x^{p(k)}\underline{\Lambda}^{(b)}(x)$ for $p(k) \ge 0$. Thus, $\Omega^{(b)}(x)$ consists of $p(k)$ consecutive zero coefficients, starting from the constant term, at the beginning of the k-th iteration. The $p(k)$ terms can be treated as redundant information, which can be removed with no information loss. Moreover, $\deg(\underline{\Lambda}^{(b)}(x))$ is no more than t during each iteration because it is a non-decreasing value and is equal to t at the end of $2t$ iterations [1]. Since $\deg(\underline{\Lambda}^{(b)}(x)) = \deg(\Lambda^{(b)}(x)) - p(k)$ which indicates $p(k) \ge k - t$, the number of redundant zero coefficients of $\Omega^{(b)}(x)$ are more than $k - t$ in the k-th iteration. Therefore, the following equation can be derived:

$$\deg(\underline{\Omega}^{(b)}(x)) = \deg(\Omega^{(b)}(x)) - p(k) \le 3t - 1 - k. \qquad (13)$$

Knowing that there are more than $k - t$ consecutive zero coefficients of $\Omega^{(b)}(x)$, a new polynomial $\Psi^{(*)}(x)$ is defined as the concatenation of the two polynomials $\Omega^{(*)}(x)$ and $\Lambda^{(*)}(x)$:

$$\Psi^{(*)}(x) = \Omega^{(*)}(x) \cdot x^{t+1} + \Lambda^{(*)}(x) \qquad (14)$$

The notation $\Psi^{(*)}(x)$ is used to represent $\Psi^{(a)}(x)$ or $\Psi^{(b)}(x)$ where appropriate. Since the degree of $\Lambda^{(b)}(x)$ is equal to k, the reformulated Eq. (14) ensures that the useful data of $\Omega^{(b)}(x)$, i.e., $\underline{\Omega}(x)$, cannot be overwritten by those of $\Lambda^{(b)}(x)$ during iterations. Therefore, the concatenated polynomial $\Psi(x)$ is enough to find $\underline{\Omega}(x)$ and $\underline{\Lambda}(x)$, which is helpful for reducing the area complexity of the hardware implementation. Note that the degrees of $\Psi^{(a)}(x)$ and $\Psi^{(b)}(x)$ are $3t + 1$ and $3t$, respectively, because $\deg(\Omega^{(a)}(x)) = 2t$ and $\deg(\Omega^{(b)}(x)) = 2t - 1$.

3.2.2 Proposed Folding ME Algorithm

Since the degree of the defined polynomial $\Psi(x)$ is restricted

to $3t + 1$, the corresponding architecture can be constructed using $3t + 2$ basic cells. The basic cell design is shown in Fig. 2, in which Ψ_i denotes the coefficient at the x^i term of $\Psi(x)$; u and v are the leading coefficients of $\Psi^{(b)}(x)$ and $\Psi^{(a)}(x)$, respectively. The control signal w is used to determine the previous state $\Psi^{(a)}(x)$ for the next iteration. Like the $(2t)$-th coefficient of $\Omega^{(c)}(x)$ in (9), i.e., $\Omega^{(c)}_{2t}$, the coefficient $\Psi^{(c)}_{3t+1}$ is always reduced to zero; thus, there is no need to store this value. As a result, the circuit used to update $\Psi^{(b)}_{3t+1}$ is removed and the basic cell employed for updating $\Psi^{(a)}_{3t+1}$ can be further simplified and merged into the control block.

Moreover, from (8)–(11), (14), and the initial setting of the TME algorithm, the constant term $\Psi^{(a)}_0$ is always equal to zero at each iteration, and $\Psi^{(b)}_0$ remains 0 except for its initial value. Hence, one register can be used instead of a basic cell to store and update $\Psi^{(a)}_0$ and $\Psi^{(b)}_0$. As a result, the number of required basic cells is reduced from $3t + 2$ to $3t$.

To derive the low-complexity KES architecture for t-error-correcting RS codes, a folding factor f is chosen and two variables, g and p such that $gf + p = 3t$, where $p < f$ are defined. In this manner, the $3t$ basic cells can be divided into either g or $g + 1$ groups, depending on whether p is zero. After applying the defined concatenated polynomial $\Psi(x)$ in (14) and the folding technique, the proposed folding ME algorithm, denoted as the FME algorithm, is expressed as:

FME Algorithm

Initialization:

$\Psi^{(a)}(x) = x^{3t+1}, \Psi^{(b)}(x) = S(x)x^{t+1} + 1, k = 0, l = 0;$

FME.1:

$$\Psi^{(b)}(x) = x\Psi^{(b)}(x); \tag{15}$$

For $i = 1$ to f

/ * $h = 0, 1, \ldots, g - 1$ (if $p = 0$) or g (if $p \neq 0$) * /

$$\Psi^{(c)}_{hf+i} = u\Psi^{(a)}_{hf+i} + v\Psi^{(b)}_{hf+i}; \tag{16}$$

If $u \neq 0$ and $2l \leq k$ **then**

$$\Psi^{(a)}_{hf+i} = \Psi^{(b)}_{hf+i}; \tag{17}$$

$$\Psi^{(b)}_{hf+i} = \Psi^{(c)}_{hf+i}; \tag{18}$$

End

If $u \neq 0$ and $2l \leq k$ **then**

$$l = k + 1 - l; \tag{19}$$

FME.2

$k = k + 1$. **If** $k \leq 2t - 1$, **then** go to **FME.1**.

Output: $\Psi^{(b)}_i$ for $t \leq i \leq 3t$.

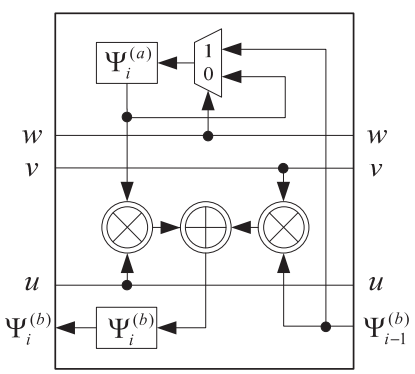


Fig. 2 Basic cell of the KES unit.

In the FME algorithm, the variables u and v denote the leading coefficients $\Psi^{(b)}_{3t+1}$ and $\Psi^{(a)}_{3t+1}$, respectively. At the end of the FME algorithm, the coefficients of the output $\Psi^{(b)}_i$ for $t \leq i \leq 2t$ and $2t + 1 \leq i \leq 3t$, respectively, represent the desired error locator and error evaluator polynomials, which are the same as the useful information of $\Lambda^{(b)}(x)$ and $\Omega^{(b)}(x)$ derived from the TME algorithm. The implementation of an area-efficient design based on the developed approach is described below.

3.2.3 Low-Complexity KES Architecture

The architecture designed using the proposed FME algorithm is referred to as the FME architecture. For the targeted RS(255,239) code, the number of basic cells required is $3t = 24$. We choose $f = 12$ to get an area-efficient solution for the RS(255,239) decoders. Figure 3 shows the proposed folded architecture with its initial values indicated in the folded basic cells (FBC). The architecture consists of 2 FBC cells. Each FBC cell is derived by folding 12 identical basic cells so that the 12 pairs of registers in each FBC cell share the same arithmetic resources. In Fig. 3, six additional registers, denoted as D_j for $0 \leq j \leq 5$, are employed to reduce the critical path delay, and the signal *SEL* is introduced to control the folded dataflow. The critical path delay of the FME architecture is equal to T_{mult}, as indicated in Fig. 3. Note that in the first cycle of each iteration, *SEL* is set to 1 to transfer data between two FBC cells.

In each iteration, (12+2) clock cycles are needed to complete the operations. After 16 iterations, the coefficients of $\Omega^{(b)}(x)$ and $\Lambda^{(b)}(x)$ are simultaneously obtained. Therefore, the total number of clock cycles needed to calculate $\Omega^{(b)}(x)$ and $\Lambda^{(b)}(x)$ is $16 \times (12 + 2) = 224$. Since the cycle number is lower than those of the SC and CSEE units, the increased number of cycles resulting from adopting the folding scheme does not lead to performance degradation. As a result, the FME architecture can efficiently handle the data flow and operate at high speeds with a great hardware reduction as compared to conventional parallel architectures.

3.3 CSEE Unit

After obtaining the error locator polynomial $\underline{\Lambda}(x)$ and the error evaluator polynomial $\underline{\Omega}(x)$ from the KES unit, the Chien search is used to find the error locations by checking for a result of $\underline{\Lambda}(\alpha^{-j}) = 0$ for $j = 0, 1, \ldots, 254$. The Forney algorithm can then be used to evaluate the corresponding error values. Figure 4 shows the overall structure of the CSEE unit and its basic cell design. Since $x\Lambda'(x)$ in (7) is equal to the sum of the terms with odd degree in $\Lambda(x)$, its value can be evaluated by summing these odd-degree terms of $\Lambda(x)$ at $x = \alpha^{-j}$. In our design, $\underline{\Omega}(x)$ and $\underline{\Lambda}(x)$, respectively, can

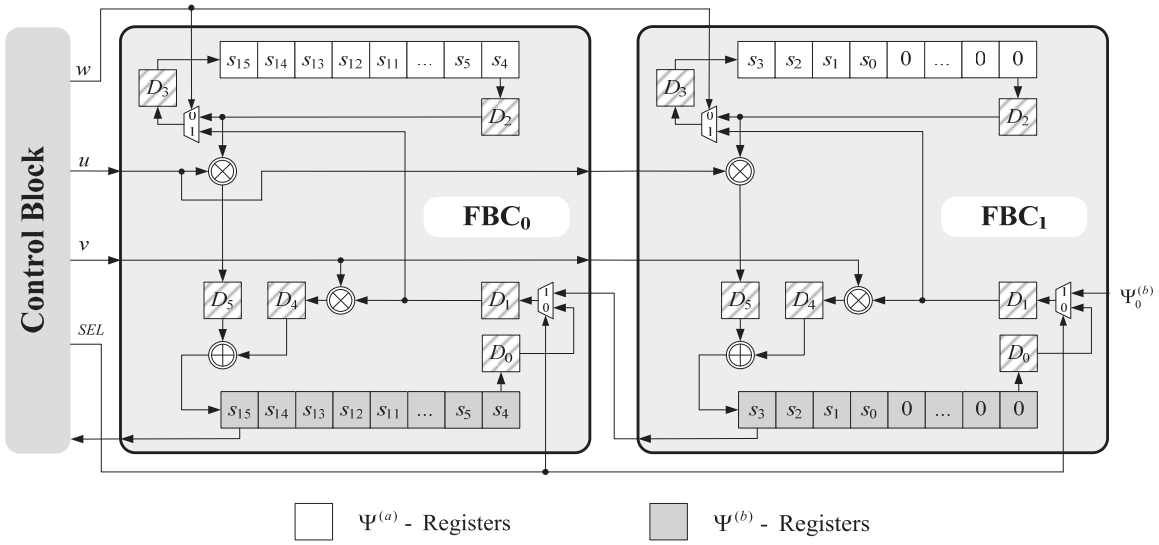


Fig. 3 Proposed FME architecture for targeted RS(255,239) decoder.

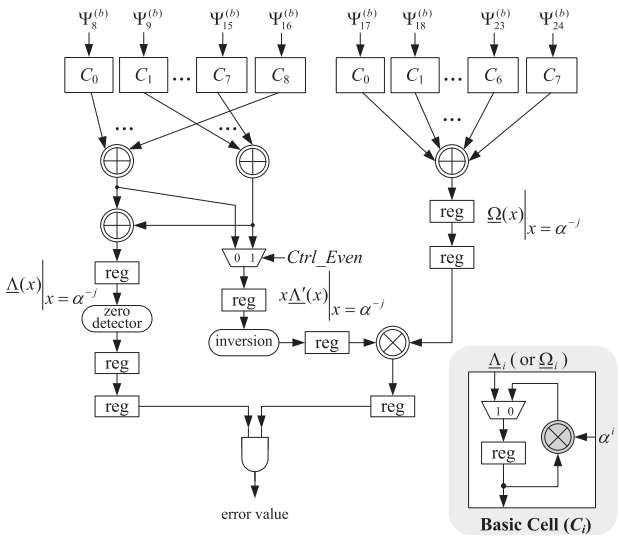


Fig. 4 CSEE unit and its basic cell.

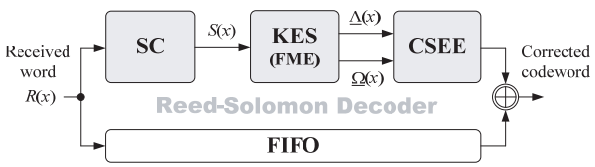


Fig. 5 Proposed RS decoder.

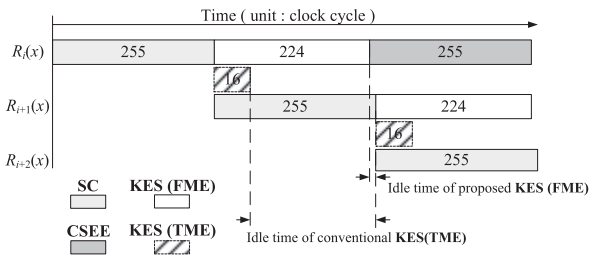


Fig. 6 Pipelining strategy with the FME architecture.

be used instead of $\Omega(x)$ and $\Lambda(x)$ in (7) to calculate the error values. The control signal *Ctrl_Even* is used to choose the odd-degree terms of $\Lambda(x)$. When the number of errors e is even, *Ctrl_Even* is set to 1; otherwise *Ctrl_Even* = 0. The inversion block in Fig. 4 can be realized with a ROM table that has a shorter delay than that of a FFM. Note that $\deg(\underline{\Lambda}(x))$ and $\deg(\underline{\Omega}(x))$ are t and $t-1$, respectively, and the two polynomials can be represented as $\underline{\Lambda}(x) = \beta x^{t-e}\Lambda(x)$ and $\underline{\Omega}(x) = \beta x^{t-e}\Omega(x)$ for $e \le t$, where β is a constant; moreover, the value of m_0 in (7) is set to 0.

3.4 Hardware Utilization

Figure 5 depicts the overall architecture of the proposed RS decoder, in which the first-in first-out (FIFO) is employed to compensate for the delays of the three main blocks. As mentioned previously, both the SC and the CSEE units of the pipelined design are operated in a serial manner. Thus, they take 255 clock cycles to complete their tasks for the targeted RS(255,239) decoder. Since the total number of clock cycles spent in the proposed KES unit is 224, which is less than codeword length 255, a 3-stage pipelined scheme with execution time of 255 clock cycles per stage are adopted. This implies that there is no throughput degradation in the proposed FME architecture. Figure 6 shows the timing relationship among the three units of the decoder. Compared with the architectures derived based on the conventional ME [1] or the TME [2] algorithms, the idle time of the developed KES unit is greatly reduced from 239 to 31 clock cycles. This means that the hardware utilization of the proposed decoder is improved by the pipelining strategy. The proposed decoder also has a low area requirement. The total latency of the proposed RS decoder is 255 + 224 + 5 = 484 clock cycles, in which the extra 5 cycles are introduced by the

pipelined registers used to shorten the critical path delay in the CSEE unit.

4. Experimental Results and Comparisons

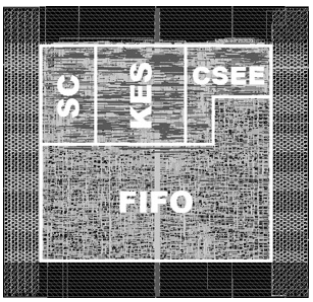
4.1 Post-Layout Simulation Results

Using the proposed FME architecture, a low-complexity RS(255,239) decoder was coded in the Verilog hardware description language and synthesized using the Design Compiler from Synopsys. The post-layout simulation shows that the proposed design can operate at up to 425 MHz with a total gate count of 11,759 when implemented in the TSMC 0.18 μm technology. The layout view of the proposed RS(255,239) decoder designed with the FIFO buffer for storing input symbols is shown in Fig. 7. The decoder has a core size of about 0.63 mm × 0.63 mm.

4.2 Complexity Analysis of the KES Unit

To verify the effectiveness of the proposed design, we employed the cell library information in [18] to analyze the resulting area and time complexities of the various KES designs for targeted RS(255,239) codes. For simplicity, the cell delay and area requirement of each cell are normalized with respect to those of the 2-input NAND gate as given in Table 1. The compiled results including the area-time (AT) complexity and improvement are shown in Table 2. We define the AT complexity as the product of the area complexity (AC) and the time complexity (TC), i.e., AT = AC×TC. The AT improvement is defined as $(AT_2 - AT_1)/AT_2$, where AT_2 and AT_1 denote the AT complexity of the related work and ours, respectively.

For a fair comparison, we used the same kind of FFM, as presented in [3], for each design. The 3-stage pipelined FFM and 2-stage FFM, respectively, need extra 23 and 15 pipelined flip-flops. The critical path delay and the area requirement of the non-pipelined FFM, employed in the proposed architecture, can be estimated as $T_{\text{mult}} = T_{\text{and2}} + 5T_{\text{xor2}}$ and $A_{\text{mult}} = 64A_{\text{and2}} + 77A_{\text{xor2}}$, respectively. Note that the widths of FFAs, registers, and multiplexers listed in Table 2 are 8 bits. Excluding the control block, the proposed FME architecture consists of 4 FFMs, 2 FFAs, 60 registers, and 4 multiplexers. In Table 2, although the designs in [3], [4], and [6] have better TC than ours because of the use of pipelined multipliers, the proposed FME architecture has the lowest AC and the smallest AT complexity. Compared with design in [5], ours has small AC and TC, respectively, because fewer registers are required for storing the coefficients of updated polynomials and a shorter critical path delay exists in the proposed design. The design in [6] takes 260 clock cycles for solving the key equation. This implies that the performance of the decoder is degraded because the decoding delay of the KES unit is more than 255. Note that the symbol T_{xor2} represents the delay time of a 2-input XOR gate, which is the same as the delay time of FFA.



Technology	TSMC 0.18 μm
Clock rate	425 MHz
Throughput	3400 Mbps
Gate count	11759 (w/o FIFO)
Core size	0.63 mm × 0.63 mm

Fig. 7 Layout view of the proposed RS decoder.

Table 1 Normalized delay and area of employed standard cells.

Cell	2-input NAND	2-input AND	2-input OR	2-input XOR	3-input XOR	2-input XNOR	2-to-1 MUX	DFF
delay ratio	1	3.41	4	3.83	7.54	3.81	3.37	7.43
area ratio	1	1.33	1.33	2.67	6	2.67	2.67	5.67

4.3 Comparison with Related Works

Table 3 lists the performances of various RS(255,239) decoder designs. To consider the scaling effect of fabrication

Table 3 Performance comparison with various RS(255,239) decoders.

Decoder Designs	[3](1)	[4](1)	[5](2)	[6](1)	[20](2)	Proposed(2)
Technology (nm)	130	130	180	180	180	180
# of Gates in Decoder	115500	24600	20614	18400	27271	11759
Clock rate (MHz)	770	625	400	640	375	425
Overall latency (cycles)	355	522	512	516	275	484
Throughput (Mbps)	6160	5000	3200	5022	3000	3400
TSNT (Mbps/K gate)	77.0	293.6	310.5	545.9	220.0	578.3

(1) Synthesis results.
(2) Post-layout simulation results.

Table 2 Comparison of hardware complexity, critical path delay, and latency of KES unit for RS(255,239) decoders.

KES Designs	FFMs	FFAs	REGs	MUXs	Critical path delay	Latency	AC(1)(2)	TC(1)(2)	AT(1)	ATI(1)
[3]	64(3)	64	628	322	$3T_{\text{or2}}+T_{\text{xnor2}}+T_{\text{mux2}}$ (4)$+T_{\text{ff}}$	80	63682	26.61	1694578	92.9%
[4]	4(3)	2	170	30	$3T_{\text{or2}}+T_{\text{xnor2}}+T_{\text{mux2}}+T_{\text{ff}}$	255	9898	26.61	263386	54.3%
[5]	4	2	66	4	$T_{\text{mult}}+T_{\text{xor2}}+T_{\text{mux2}}+T_{\text{ff}}$	255	4285	37.19	159359	24.5%
[6]	4(3)	2	90	24	$T_{\text{and2}}+T_{\text{xor2}}+T_{\text{xor3}}+T_{\text{ff}}$	260	6141	22.65	139614	13.8%
Proposed	4	2	60	4	$T_{\text{mult}}+T_{\text{ff}}$	224	4013	30	120390	-

(1) AC: area complexity, TC: time complexity, AT: area-time complexity, ATI: AT improvement.
(2) The values are estimated in finite field GF(2^8) constructed from the primitive polynomial $x^8+x^4+x^3+x^2+1$.
(3) Pipelined multiplier.
(4) T_{Gn} notes the delay time of a logic cell G with n inputs. Note that T_{xor2} is equal to T_{add}.

technology, we adopt the definition of technology scaled normalized throughput rate (TSNT) in [5] as

$$TSNT = (\text{Throughput rate}) \times (\text{Tech.}/90\,\text{nm})/\#\text{ of Gates}.$$

As can be seen from the table, the proposed design has the best *TSNT*. Although the designs in [3], [4], and [6] have higher throughput rates than that of the proposed decoder, they have much higher hardware cost because they use pipelined multipliers to reduce the critical path delay. From Table 3, the proposed RS decoder with the FME architecture can work at a higher speed and has lower hardware requirements than those of our pervious design in [20]. Compared with the design in [6], the proposed design reduces the hardware requirement by about 36 %. Moreover, the throughput rate of design in [6] is indeed decreased due to the reasons mentioned above. Note that the comparison of the total gate counts of decoders was conducted by excluding the effect of the FIFO employed to buffer the received input symbols.

4.4 High-Throughput Optical Communication Systems

Based on the proposed RS decoder, we can easily construct a 4-way parallel RS decoder to reach a throughput rate of $4 \times 3.4 = 13.6$ Gbps. This high-throughput RS decoder will have high-speed and area-efficient advantages than that presented in [5] to meet the demand for optical communication applications with throughput rates beyond 10 Gbps.

The targeted design in this work is an area-efficient RS(255,239) decoder which is good enough for constructing a 4-way parallel architecture to meet the requirements of optical communication systems. Therefore, the employment of pipelined multipliers was not considered in the current design. If higher throughput rates are required, the proposed FME architecture can employ the pipelined multipliers, as used in [6], to shorten its critical path delay. The proposed RS decoder with pipelined multipliers possesses a higher throughput rate than that of the design in [6] when operated at the same clock rate because the required computation cycles of the KES unit in [6] are more than 255, which results in performance degradation. The proposed design also has lower hardware requirements.

5. Conclusion

This paper presented a low-complexity RS(255,239) decoder design based on the proposed FME architecture for optical communication systems. We showed how to reformulate the TME algorithm to derive a more compact polynomial representation so that a significant reduction in hardware complexity can be obtained for VLSI implementation. Then, an efficient folded architecture was developed to further reduce the hardware requirement of the targeted RS decoder design without sacrificing its throughput. Experimental results demonstrate that the proposed decoder has the lowest area and AT complexity as compared to those of related studies. Moreover, the high-speed structure and simple control scheme of the proposed design make it suitable for high-performance implementations. The proposed RS decoder is thus a cost-effective solution for optical communications applications.

Acknowledgments

This work was supported in part by the National Science Council of R.O.C. under contract NSC 96-2221-E-006-296. Parts of this work have been presented in international symposium on circuits and systems [19].

References

[1] H.M. Shao, T.K. Truong, L.J. Deutsch, J.H. Yuen, and I.S. Reed, "A VLSI design of a pipeline Reed-Solomon decoder," IEEE Trans. Comput., vol.C-34, no.5, pp.393–402, May 1985.
[2] 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 Trans. Commun., vol.51, no.3, pp.381–388, March 2003.
[3] H. Lee, "High-speed VLSI architecture for parallel Reed-Solomon decoder," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol.11, no.2, pp.288–294, April 2003.
[4] H. Lee, "A high-speed low-complexity Reed-Solomon decoder for optical communications," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol.52, no.8, pp.461–465, Aug. 2005.
[5] H.Y. Hsu, A.Y. Wu, and J.C. Yeo, "Area-efficient VLSI design of Reed-Solomon decoder for 10GBase-LX4 optical communication systems," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol.43, no.4, pp.1019–1027, Nov. 2006.
[6] B. Yuan, Z.F. Wang, L. Li, M.L. Gao, J. Sha, and C. Zhang, "Area-efficient Reed-Solomon decoder design for optical communications," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol.56, no.6, pp.469–473, June 2009.
[7] Y.W. Chang, T.K. Truong, and J.H. Jeng, "VLSI architecture of modified Euclidean algorithm for Reed-Solomon code," Elsevier J. Inform. Sciences, vol.155, no.1-2, pp.139–150, May 2003.
[8] J.L. Massey, "Shift-register synthesis and BCH decoding," IEEE Trans. Inform. Theory, vol.IT-15, no.1, pp.122–127, Jan. 1969.
[9] A. Raghupathy and K.J.R. Liu, "Algorithm-based low-power/high-speed Reed-Solomon decoder design," IEEE Trans. Circuits Syst. II, Analog Dig. Signal Process., vol.41, no.11, pp.1254–1270, Nov. 2000.
[10] H.C. Chang, C.B. Shung, and C.Y. Lee, " A Reed-Solomon product-code (RS-PC) decoder chip for DVD applications," IEEE J. Solid-State Circuits, vol.36, no.2, pp.229–238, Feb. 2001.
[11] T. Park, "Design of the (248,216) Reed-Solomon decoder with erasure correction for Blu-ray disc," IEEE Trans. Consumer Electronics, vol.51, no.3, pp.872–878, Aug. 2005.
[12] I.S. Reed, M.T. Shih, and T.K. Truong, "VLSI design of inverse-free Berlekamp-Massey algorithm," IEE Proc.-E, vol.138, no.5, pp.295–298, Sept. 1991.
[13] M.D. Shieh, Y.K. Lu, S.M. Chung, and J.H. Chen, "Design and implementation of efficient Reed-Solomon decoders for multi-mode applications," Proc. IEEE Int. Symp. Circuits Syst. (ISCAS'2006), pp.289–292, Kos, Greece, May 2006.
[14] Telecommunication Standardization Section, International Telecom. Union, "Forward error correction for submarine systems," ITU, ITU-T Recommendation G.975, Geneva, Switzerland, Oct. 2000.
[15] S.B. Wicker and V.K. Bhargava, Reed-Solomon Codes and Their Applications, IEEE Press, New York, 1994.
[16] G. Jr. Forney, "On decoding BCH codes," IEEE Trans. Inf. Theory, vol.IT-11, no.4, pp.549–557, Oct. 1965.

[17] R.J. McEliece, The Theory of Information and Coding: A Mathematical Framework for Communication, Addison-Wesley, MA, 1977.
[18] Artisan Components, TSMC 0.18-μm Process 1.8-Volt SAGE-X^{TM} Standard Cell Library Databook, Sunnyvale, CA, 2003.
[19] Y.K. Lu, M.D. Shieh, and C.M. Wu, "Low-complexity Reed-Solomon decoder for optical communications," Proc. IEEE Int. Symp. Circuits Syst. (ISCAS'2010), pp.4173–4176, Paris, France, May 2010.
[20] Y.K. Lu and M.D. Shieh, "High-speed low-complexity architecture for Reed-Solomon decoder," IEICE Trans. Inf. Syst., vol.E93-D, no.7, pp.1824–1831, July 2010.

Ming-Der Shieh received the B.S. degree in electrical engineering from National Cheng Kung University, in 1984, the M.S. degree in electronic engineering from National Chiao Tung University, Taiwan, in 1986, and the Ph.D. degree in electrical engineering from Michigan State University, East Lansing, in 1993. From 1988 to 1989, he was an engineer at United Microelectronic Corporation, Taiwan. From 1993 to 2002, he was with the faculty of Department of Electronic Engineering, National Yunlin University of Science & Technology. He received the teaching award in 1998 and was the department chairman from 1999 to 2002. Since 2002, he has been with the Department of Electrical Engineering, National Cheng Kung University, where he is currently a professor. His research interests include VLSI design and testing, VLSI for signal processing, and digital communication. He was the program co-chair and general co-chair of Asian Test Symposium in 2004 and 2009, respectively, and the chair of Tainan Chapter of IEEE Circuits and Systems from 2009 to 2010. He is now the associate editor of IEEE Transaction on Circuits and Systems: Part I.

Yung-Kuei Lu received the B.S. and M.S. degree in electrical engineering from National Cheng Kung University, Taiwan, in 2000 and 2005 respectively. He is pursuing his Ph.D. degree in National Cheng Kung University, Taiwan, since 2005. His research interests include VLSI implementation in digital signal processing architectures and error control coding.