A High Speed Reed-Solomon decoder Chip using Inversionless Decomposed Architecture for Euclidean Algorithm

Hsie-Chia Chang, Ching-Che Chung,
Chien-Ching Lin, and Chen-Yi Lee
Department of Electronics Engineering
National Chiao Tung University
Hsinchu, Taiwan, R.O.C.
hcchang@royals.ee.nctu.edu.tw

Abstract

In this paper, a high speed Reed-Solomon (RS) decoder chip for optical communications is presented. It mainly contains one $(255,239)$ *RS decoder with* $4K$*-bit embedded memory. Due to the operation speed limitation in I/O pad, a Delay Lock Loop (DLL) circuit is also included to generate internal high-speed clock. The RS decoder features a high speed and area-efficient key equation solver using a novel inversionless* decomposed *architecture for Euclidean algorithm.*

The test chip is implemented by $0.35\mu m$ *CMOS SPQM standard cells with chip area of* $2.61mm \times 2.62mm$*. The RS decoder has the gate count of* $12.4K$*. Test results show the proposed chip can support* 2.35*-Gbps data rate while operating at* $294MHz$ *with the supply voltage of* $3.3V$*.*

1. Introduction

Among the most well-known error-correcting codes, the Reed-Solomon (RS) codes are undoubtedly the most widely used block codes in communications and storage systems. Due to increasing demand for high capacity of optical communications, the high speed and low complexity implementations of RS decoders are desirable to meet higher data rate for system-level integration. In ITU-T G.975, several $(255,239)$ RS codes are standardized to resist burst errors for optical fiber submarine cable systems. The (N,K) RS code means each codeword contains N coded symbols with K message symbols, and is capable of correcting up to $t = \lfloor \frac{N-K}{2} \rfloor$ symbol errors. For $(255,239)$ RS code, each symbol is one byte and it can correct up to 8 symbols.

The general decoding steps are illustrated in Figure 1. The syndrome calculator generates a set of syndromes from the received codeword polynomial R(x). From the syndromes, the key equation solver produces the error locator polynomial $\sigma(x)$ and the error evaluator polynomial $\Omega(x)$, which can be used by the Chien Search and the Error Value Evaluator to determine the error locations and error values, respectively.

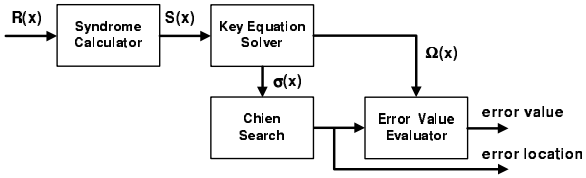

Figure 1. The Reed-Solomon decoding flowchart.

The critical part shown in Fig. 1 will be the key equation solver while implemented for high speed application. In Section 2, a novel inversionless decoding procedure for Euclidean algorithm is illustrated to not only eliminate the costly finite-field inverter (FFI) but also introduce a t-iteration decoding procedure, whereas other approaches require at most $2t$ iterations. Then the decomposed architecture of *three* finite-field multipliers (FFMs) is proposed to enhance the operation speed and reduce the hardware complexity for the key equation solver. Section 3 shows the proposed $(255,239)$ RS decoder and Section 4 describes the chip implementation and test results. Finally the conclusion is given in Section 5.

2. The Inversionless Decomposed Architecture for Euclidean Algorithm

The techniques frequently used to solve the key equation include the Berlekamp-Massey (BM) algorithm and the Euclidean algorithm. Although the BM algorithm has the lowest circuit complexity, the Euclidean algorithm can support higher operation speed due to its shorter critical path. For illustrating our modified Euclidean decoding algorithm, the key equation is written as:

$$\Omega(x) = x^{2t} \cdot Q(x) + S(x)\sigma(x) \quad (1)$$

where Q(x) is the quotient polynomial of the dividend polynomial $S(x)\sigma(x)$ and the divisor polynomial x^{2t}. Hence, the error evaluator polynomial $\Omega(x)$ can be determined by the similar process of computing the GCD polynomial of x^{2t} and $S(x)$ through the Euclidean algo-

rithm, which can be shown as follows:

$$\begin{cases} R^{(-1)}(x) &= x^{2t} \\ R^{(0)}(x) &= S(x) \\ R^{(1)}(x) &= R^{(-1)}(x) - R^{(0)}(x)\cdot Q^{(1)}(x) \\ &\vdots \\ R^{(i)}(x) &= R^{(i-2)}(x) - R^{(i-1)}(x)\cdot Q^{(i)}(x) \end{cases} \quad (2)$$

where $Q^{(i)}(x)$ is the i-th quotient polynomial and $R^{(i)}(x)$ is the i-th remainder polynomial. Each iterative step in (2) performs a polynomial division operation. While the decoding procedure stops at the degree of $R^{(i)}(x)$ less than t, $R^{(i)}(x)$ can be taken as the error evaluator polynomial $\Omega(x)$. From the extended form of Euclidean algorithm, the similar procedure except the minor difference is used to determine the error locator polynomial $\sigma(x)$. There are at most t polynomial division operations of (2) totally in the worst case with all degree of $Q^{(i)}(x)$ equaling one.

Existing architectures to implement the Euclidean algorithm in hardware were proposed by Shao and Reed [1], Song [2], and Hanho Lee [3]. These proposals all follow the decoding procedure require at least two iterations to complete each polynomial division operation of (2), and therefore the total number of iterations is at most $2t$ in the worst case. However, the decoding procedure determining both $\Omega(x)$ and $\sigma(x)$ can be modified as follows:

Initial condition:

$A^{(0)}(x) = x^{2t}$, $M^{(0)}(x) = \Omega^{(0)}(x) = S(x)$

$a^{(0)}(x) = 0$, $m^{(0)}(x) = \sigma^{(0)}(x) = 1$

For ($i = 0$ **to** t **)**

$\delta = \mathbf{deg}(A^{(i)}(x))$, $\Delta = \mathbf{deg}(M^{(i)}(x))$

If ($\mathbf{deg}(\sigma^{(i)}(x)) \le \mathbf{deg}(\Omega^{(i)}(x))$ **)**

$$q_1^{(i)} = \frac{A_\delta^{(i)}}{M_\Delta^{(i)}}$$

$$q_0^{(i)} = \begin{cases} 0 & \textbf{for} \quad \delta = \Delta \\ \frac{M_\Delta^{(i)} A_{\delta-1}^{(i)} + M_{\Delta-1}^{(i)} A_\delta^{(i)}}{M_\Delta^{(i)} M_\Delta^{(i)}} & \textbf{for} \quad \delta \neq \Delta \end{cases}$$

$$\Omega^{(i+1)}(x) = A^{(i)}(x) + x^{\delta-\Delta-1}\cdot M^{(i)}(x)\cdot q^{(i)}(x) \quad (3)$$

$$\sigma^{(i+1)}(x) = a^{(i)}(x) + x^{\delta-\Delta-1}\cdot m^{(i)}(x)\cdot q^{(i)}(x) \quad (4)$$

if ($\mathbf{deg}(\Omega^{(i+1)}(x)) < \Delta$ **)**

$A^{(i+1)}(x) = M^{(i)}$, $M^{(i+1)}(x) = \Omega^{(i+1)}(x)$

$a^{(i+1)}(x) = m^{(i)}$, $m^{(i+1)}(x) = \sigma^{(i+1)}(x)$

else

$A^{(i+1)}(x) = \Omega^{(i+1)}$, $M^{(i+1)}(x) = M^{(i)}(x)$

$a^{(i+1)}(x) = \sigma^{(i+1)}$, $m^{(i+1)}(x) = m^{(i)}(x)$

Else

$\Omega(x) = \Omega^{(i)}(x)$, $\sigma(x) = \sigma^{(i)}(x)$ **Finish**

where $q^{(i)}(x) = q_0^{(i)} + q_1^{(i)}x$ is the i-th dummy quotient polynomial, $\Omega^{(i+1)}(x)$ is the i-th dummy remainder polynomial, and $A_\delta^{(i)}$ and $M_\Delta^{(i)}$ are the leading coefficients of the i-th dummy dividend polynomial $A^{(i)}(x)$ with degree of δ and the i-th dummy divisor polynomial $M^{(i)}(x)$ with degree of Δ. Note that if the degree of $\Omega^{(i+1)}(x)$ calculated by (5) is less than the degree of $M^{(i)}(x)$, then $\Omega^{(i+1)}(x)$ and $M^{(i)}(x)$ will be the next dummy divisor and dividend polynomial respectively. Otherwise, $\Omega^{(i+1)}(x)$ with degree of at most $\delta-2$ will be assigned to the next dummy dividend polynomial and the next dummy divisor polynomial will still be $M^{(i)}(x)$. For eliminating the inverse operation, a novel inversionless decoding procedure can be shown as follows:

Initial condition:

$\widehat{A}^{(0)}(x) = x^{2t}$, $\widehat{M}^{(0)}(x) = \widehat{\Omega}^{(0)}(x) = S(x)$

$\widehat{a}^{(0)}(x) = 0$, $\widehat{m}^{(0)}(x) = \widehat{\sigma}^{(0)}(x) = 1$

For ($i = 0$ **to** t **)**

$\delta = \mathbf{deg}(\widehat{A}^{(i)}(x))$, $\Delta = \mathbf{deg}(\widehat{M}^{(i)}(x))$

If ($\mathbf{deg}(\widehat{\sigma}^{(i)}(x)) \le \mathbf{deg}(\widehat{\Omega}^{(i)}(x))$ **)**

$$\widehat{q}_1^{(i)} = \widehat{M}_\Delta^{(i)}\widehat{A}_\delta^{(i)}$$

$$\widehat{q}_0^{(i)} = \begin{cases} 0 & \textbf{for} \quad \delta = \Delta \\ \widehat{M}_\Delta^{(i)}\widehat{A}_{\delta-1}^{(i)} + \widehat{M}_{\Delta-1}^{(i)}\widehat{A}_\delta^{(i)} & \textbf{for} \quad \delta \neq \Delta \end{cases}$$

$$\widehat{\Omega}^{(i+1)}(x) = \widehat{M}_\Delta^{(i)}\widehat{M}_\Delta^{(i)}\cdot\widehat{A}^{(i)}(x) + x^{\delta-\Delta-1}\cdot\widehat{M}^{(i)}(x)\cdot\widehat{q}^{(i)}(x) \quad (5)$$

$$\widehat{\sigma}^{(i+1)}(x) = \widehat{M}_\Delta^{(i)}\widehat{M}_\Delta^{(i)}\cdot\widehat{a}^{(i)}(x) + x^{\delta-\Delta-1}\cdot\widehat{m}^{(i)}(x)\cdot\widehat{q}^{(i)}(x) \quad (6)$$

if ($\mathbf{deg}(\widehat{\Omega}^{(i+1)}(x)) < \Delta$ **)**

$\widehat{A}^{(i+1)}(x) = \widehat{M}^{(i)}$, $\widehat{M}^{(i+1)}(x) = \widehat{\Omega}^{(i+1)}(x)$

$\widehat{a}^{(i+1)}(x) = \widehat{m}^{(i)}$, $\widehat{m}^{(i+1)}(x) = \widehat{\sigma}^{(i+1)}(x)$

else

$\widehat{A}^{(i+1)}(x) = \widehat{\Omega}^{(i+1)}$, $\widehat{M}^{(i+1)}(x) = \widehat{M}^{(i)}(x)$

$\widehat{a}^{(i+1)}(x) = \widehat{\sigma}^{(i+1)}$, $\widehat{m}^{(i+1)}(x) = \widehat{m}^{(i)}(x)$

Else

$\widehat{\Omega}(x) = \widehat{\Omega}^{(i)}(x)$, $\widehat{\sigma}(x) = \widehat{\sigma}^{(i)}(x)$ **Finish**

It can be shown $\widehat{\Omega}(x)$ and $\widehat{\sigma}(x)$ can be used to find the same error locations and error values as the original $\Omega(x)$ and $\sigma(x)$ do. In the worst case with all $Q^{(i)}(x)$ in (2) equaling one, our proposal only requires one iteration to complete each polynomial division operation. And the total number of required iterations is less than t.

Following the proposed inversionless decoding procedure for Euclidean algorithm, the *decomposed* architecture, which work with individual coefficients of the polynomial instead of the entire polynomial as a whole, is proposed. For simplifying notations, $\delta - \Delta = 1$ is assumed without loss of generality and both (5) and (6) can be decomposed as:

$$\widehat{\Omega}_j^{(i+1)} = \widehat{M}_\Delta^{(i)}\widehat{M}_\Delta^{(i)}\cdot\widehat{A}_j^{(i)} + \widehat{M}_j^{(i)}\cdot\widehat{q}_0^{(i)} + \widehat{M}_{j-1}^{(i)}\cdot\widehat{q}_1^{(i)} \quad (7)$$

$$\widehat{\sigma}_k^{(i+1)} = \widehat{M}_\Delta^{(i)}\widehat{M}_\Delta^{(i)}\cdot\widehat{a}_k^{(i)} + \widehat{m}_k^{(i)}\cdot\widehat{q}_0^{(i)} + \widehat{m}_{k-1}^{(i)}\cdot\widehat{q}_1^{(i)} \quad (8)$$

where $\widehat{\Omega}_j^{(i+1)}$ and $\widehat{\sigma}_k^{(i+1)}$ correspond to the j-th and k-th coefficient of $\widehat{\Omega}^{(i+1)}(x)$ and $\widehat{\sigma}^{(i+1)}(x)$, respectively. If

$\widehat{M}_{\Delta}^{(i)}\widehat{M}_{\Delta}^{(i)}$, $\widehat{q}_0^{(i)}$ and $\widehat{q}_1^{(i)}$ can be calculated in advance, only *three* finite-field multiplications will be needed simultaneously to compute each $\widehat{\Omega}_j^{(i+1)}$ or $\widehat{\sigma}_k^{(i+1)}$. Hence, the inversionless decomposed architecture for Euclidean algorithm shown above suggests a 3-FFM implementation of the key equation solver.

It can be shown that the amount of required cycles in our decomposed architecture is less than $2t^2+2t$. Since the throughput is limited by syndrome calculator, which takes N cycles to finish, the proposed architecture slows down the Euclidean algorithm will not slow down the decoding speed if N is larger than $2t^2+2t$.

3. The RS Decoder Architecture

Although more cycles are used in the inversionless decomposed architecture for the key equation solver, the code size of our proposed $(255,239)$ RS decoder is large enough that any speed penalty can be overcome. Fig. 2 illustrates the three-stage pipelining used in our approach, where the Chien search and the error value evaluator are both executed at the third stage. Note that the *sync* signal represents the head of each codeword. After all symbols of an entire codeword received, the syndromes are carried out and put to the next stage. All design considerations are described in more detail in following subsections.

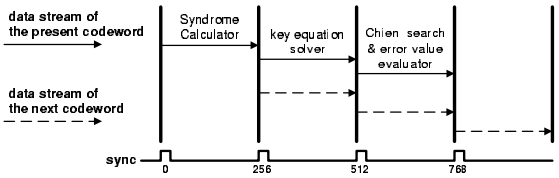

Figure 2. The pipelining diagram of our proposed $(255,239)$ RS decoder.

3.1. The syndrome calculator

By definition, the syndrome polynomial is $S(x)=S_1+S_2x+\ldots+S_{2t}x^{2t-1}$, where $S_i=R(\alpha^i)$ with the received polynomial $R(x)=R_0+R_1x+R_2x^2+\cdots+R_{254}x^{254}$ in the $(255,239)$ RS code. Fig. 3 shows the syndrome calculator cell for calculating the i-th syndrome,S_i.

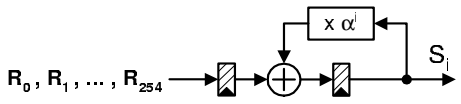

Figure 3. The syndrome calculator cell.

In Fig. 3, the partial syndrome is multiplied with α^i and accumulated with the received symbol at each cycle. After all the received symbols are processed, the accumulated result is the i-th syndrome, S_i. Note that the FFMs we implemented in syndrome calculator are all constant-variable FFMs, whose complexity and path delay are significantly lower than that of variable-variable FFMs.

3.2. The key equation solver

Existing proposals require $6t\sim 8t$ FFMs to implement Euclidean algorithm [1]–[3], whereas the proposed inversionless decomposed architecture leads to a 3-FFM architecture. Fig. 4 illustrates the key equation solver with decomposed architecture and the branch labeling corresponds to a particular time instance for computing $\Omega_j^{(i+1)}$. Since the computation process of $\sigma^{(i+1)}(x)$ is similar to that of $\Omega^{(i+1)}(x)$, the same hardware can be reconfigured to calculate $\sigma_k^{(i+1)}$.

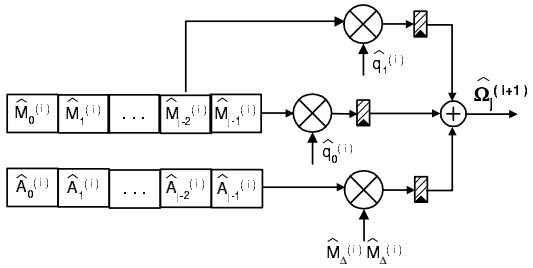

Figure 4. The key equation solver architecture.

It is evident that $2t+2$ registers are used to store the coefficients of $A^{(i)}(x)$ and $m^{(i)}(x)$, as well as the other $2t$ registers are used for storing the coefficients of $a^{(i)}(x)$ and $M^{(i)}(x)$ in the i-th iteration of our proposed decomposed architecture. Therefore, our proposal requires $4t+2$ registers totally while other architectures proposed from [1]–[3] requiring $6t\sim 8t$ registers.

3.3. Chien search

In (N,K) RS decoding algorithm, Chien search is used to check whether the error locator polynomial $\sigma(x)$ equals zero or not while $x=\alpha^{-n}$, $n=0,1,\ldots,N-1$. Fig. 5(a) shows the circuit of the Chien search cell C_i and the structure of the Chien search module with eight cells is illustrated in Fig. 5(b).

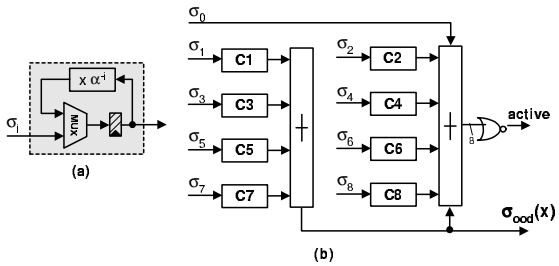

Figure 5. (a)The Chien search cell, C_i. (b)The Chien search module.

The finite-field adder (FFA) in the right hand side of Fig. 5(b) calculates the value of $\sigma(x)$ and the NOR gate is used to check if the final sum is zero or not. Note that $\sigma_{odd}(x)=\sigma_1x+\sigma_3x^3+\ldots+\sigma_{t_{odd}}x^{t_{odd}}$ is used to calculate the error value in error value evaluator, where t_{odd} represents the largest odd number less than or equal to t,

3.4. The error value evaluator

In Forney algorithm, the error value can be expressed as:

$$e_l = \frac{\Omega(\beta_l{}^{-1})}{\sigma'(\beta_l{}^{-1})} = \frac{\Omega(\beta_l{}^{-1}) \cdot \beta_l^{-1}}{\sigma_{odd}(\beta_l{}^{-1})} \quad (9)$$

where β_l^{-1} indicates the root of $\sigma(x)$, for $l = 1, \ldots, t$. Fig. 6 shows the error value evaluator. Note that all cells $C1 \sim C8$ are identical to the Chien search cell except the only difference of the loaded coefficients $\Omega_0 \sim \Omega_8$ instead of $\sigma_1 \sim \sigma_8$. While calculating the error value in parallel with the computation of Chien search, the three extra registers, as shown in Fig. 6, are used to shorten the critical path.

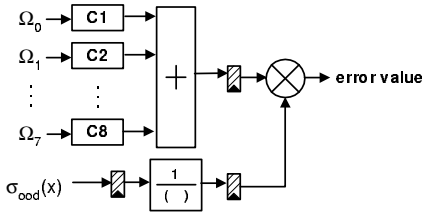

Figure 6. The error value evaluator structure.

4. Chip Implementation

Due to the operation speed limitation in I/O pad, a four times Delay Lock Loop (DLL) circuit is utilized to generate the internal high-speed clock. Fig. 7 shows the overall structure of our RS decoder chip. Since the internal clock *RS_clk* has four times the rate of the input clock *Ref_clk*, there are four bytes processed in each cycle and a multiplexer is required to distribute the 4-byte *Data_in* to the $(255, 239)$ RS decoder. In addition, the internal clock *MEM_clk* with the same rate of *Ref_clk* is provided to ensure the identical clock phase of data access between memory and RS decoder.

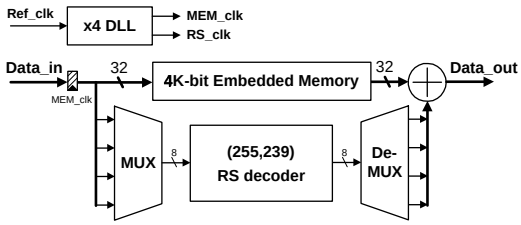

Figure 7. The RS decoder chip architecture.

The chip is designed by the standard cell library in a TSMC 0.35μm SPQM CMOS process. The chip size is $2.61mm \times 2.62mm$ with a RS core size of $0.93mm \times 0.95mm$ and The total gate counts is about $17.1K$, including $12.4K$ for the $(255, 239)$ RS decoder, $4.2K$ for the DDL circuit and $0.4K$ for the memory controller. The critical part of our design is the FFM, whose path delay is $T_{AND2} + T_{XOR8} + T_{XOR4}$ and approximate 2ns after synthesized by the standard library. Fig. 8 shows the die micrograph. As compared with the high speed RS decoder proposed by [3], which has the gate count of $43.1K$ in 0.16μm CMOS technology, our proposal can achieve about 70% reduction of circuit complexity.

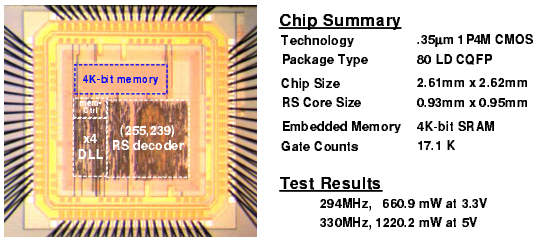

Figure 8. The die micrograph of the RS decoder chip.

While simulated at 3.3 voltage by EPIC TimeMill, our proposed RS decoder chip can work successfully with input clock period of 11.6ns, indicating the internal clock rate should be about 340MHz. After manufacturing, test results emulated by IMS-100 shows the test chip can only operate at the internal clock rate of 294MHz under the supply voltage of 3.3V. However, while the supply voltage increases to 5V, the operation speed can be raised up to 330MHz.

5. Conclusion

In this paper we propose the design and implementation of an area-efficient and high speed $(255, 239)$ RS decoder chip for optical communications. Based on the novel inversionless decomposed architecture for Euclidean algorithm, an area-efficient 3-FFM hardware structure with $4t + 2$ registers for key equation solver is proposed. As compared with other approaches, our simplification can amount to a 70% hardware reduction for the entire RS decoder.

The proposed chip solution is implemented in a 0.35μm CMOS SPQM standard cells. Test results by IMS-100 show that 2.35-Gbps data rate can be achieved by operating at 294MHz internal clock rate with the supply voltage of 3.3V.

6. Acknowledge

The authors are grateful to the support from the National Science Council of Taiwan, R.O.C., under the grant *NSC* 90-2218-*E*-009-035. They also appreciate the Chip Implementation Center for manufacturing the test chip.

[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 Transaction on Computers*, vol. c-34, pp. 393–403, 1985.

[2] L. Song, "An efficient architecture for implementing the modified Euclidean algorithm," *9th NASA Symposium on VLSI Desigm*, 2000.

[3] Hanho Lee, "Modified Euclidean algorithm block for high-speed Reed-Solomon decoder," *IEE Electronics Letters*, pp. 903–904, July 2001.