논문 2012-49TC-5-3

나눗셈회로가 필요없는 치엔머신의 최적설계

(Optimizing the Chien Search Machine without using Divider)

안 형 근[*]

(Hyeong-Keon An)

본논문을 통해, 리드솔로몬 복호기에서 매우 복잡한 나누기회로를 사용않고 , 오류위치를 찾아내는 치엔기기의 최적설계기법을 제시했다. 최적화는 매우간단한 제곱/4제곱회로를 사용하고, 병렬처리를 통해 가능했다. 이법은 현대 디지털 통신및/가전기기 대부분에 응용되질수 있다.

Abstract

In this paper, we show new method to find the error locations of received Reed-Solomon code word . New design is much faster and has much simpler logic circuit than the former design method. This optimization was possible by very simplified square/X^4 calculating circuit ,parallel processing and not using the very complex Divider. The Reed Solomon decoder using this new Chien Machine can be applicated for data protection of almost all digital communication and consumer electronic devices[7].

Keywords: Reed-Solomon(RS), Decoder, GF(2^4), Square computing, Digital, Chien search , Divider

I. Introduction

Reed-Solomon Encoder and Decoder are commonly used in data transmission and storage applications, such as broadcast equipment, wireless LANs, cable modems, xDSL, satellite commu nications, microwave networks, and digital TV. In this paper , we show how to optimize the Chien search machine of RS codec. In abstract algebra, the Chien search, named after R. T. Chien, is a fast algorithm for determining roots of polynomials defined over a finite field. The most typical use of the Chien search is in finding the roots of error-locator polynomials encountered in decoding Reed-Solomon codes and BCH codes.

In this paper, we propose new method to optimize the arithmatic logic unit for the Chien Machine[6]. To do so, huge GF(2^8) multiplier is replaced by much simpler GF(2^4) multiplier , squaring and X^4 finding circuits. Also very complicated divider is removed. Equation 1 shows the nth order error locator polynomial whose solutions are v error locations in 1 RS code word. In section Ⅱ we present the flow steps to optimize the circuit to calculate the coefficients of error locator polynomial (equation (1)). In section Ⅲ, we present how to optimize the processor structure . In section Ⅳ, we showed the application of the new optimized processor to the 4 error cases in 1 RS code word. In section Ⅴ, we showed step by step example to find 4 error locations in the received code word using the processor.

Finally, in section Ⅵ we make a concluding remark and future works to improve the design method of Reed Solomon decoder.

[*] 정회원, 동명대학교
(Tong Myung University)
접수일자: 2012년2월21일, 수정완료일: 2012년5월12일

II. Optimizing the processor for calculating the coefficients of Error locator polynomial

Error locator polynomial is as Equation (1).

$$x^{v}+\sigma_1 x^{v-1}+\cdots+\sigma_{v-1}x+\sigma_v=0 \tag{1}$$

$$A_n=\begin{bmatrix} S_1 & S_2 & \cdots & S_n \\ S_2 & S_3 & \cdots & S_{n+1} \\ \vdots & \cdots & & \vdots \\ S_n & S_{n+1} & \cdots & S_{2n-1} \end{bmatrix} \tag{2}$$

In Equation (2), A_n is Nth order Characteristic matrix and S_k's are kth order syndromes. Then, if there are v errors in the Reed solomon code, Coefficients of error locator polynomial are calculated as in equation (3)[2 ,4].

$$\delta_v=\begin{pmatrix}\sigma_v \\ \sigma_{v-1} \\ \sigma_{v-2} \\ \vdots \\ \sigma_1\end{pmatrix}=A_v^{-1}\begin{bmatrix}S_{v+1} \\ S_{v+2} \\ S_{v+3} \\ \vdots \\ S_{2v}\end{bmatrix} \tag{3}$$

Now we define new error locator polynomial coefficient vector as in equation (4).

$$\delta_v' = \mathrm{Det}(A_v)\ \delta_v \tag{4}$$

$$\delta_v' = \mathrm{Adj}(A_v)\bullet\begin{pmatrix}S_{v+1} \\ S_{v+2} \\ \vdots \\ S_{2v}\end{pmatrix} \tag{5}$$

and new Error locator polynomial is :

$$\mathrm{Det}(A_v)\bullet\ x^{v}+\sum_{k=1}^{v}\sigma_k' X^{v-k}=0 \tag{6}$$

By doing this, when we solve the error locator Polynomial equation, we need not use the dividing circuit, just using multiplier and adder to solve the equation (6)[5].

In this way, we don't need divider circuit to solve the new error locator polynomial coefficient values. Also for calculating equation 1 we can use X^4, X^2 circuit in addition to mul tiplier to speedup the calculation by parallel processing. In fig.1, we show X^4, X^2 circuit in $GF(2^4)$. As we see in fig.1 the circuits are very simple. Also huge $GF(2^8)$ multiplier is replaced by 3 $GF(2^4)$ multipliers and 4 $GF(2^4)$ adders as shown in fig.2.

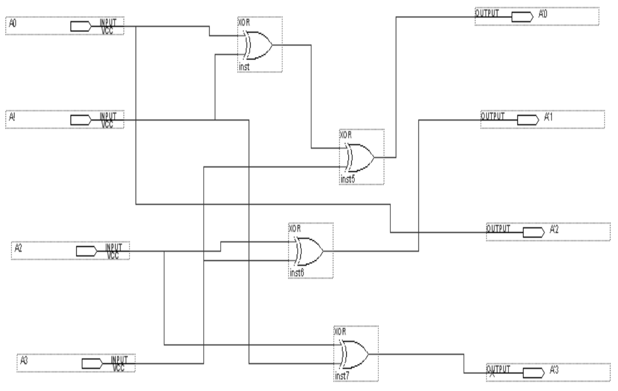


그림 1. $GF(2^4)$ 4제곱기회로
Fig. 1. X^4 circuit in $GF(2^4)$.

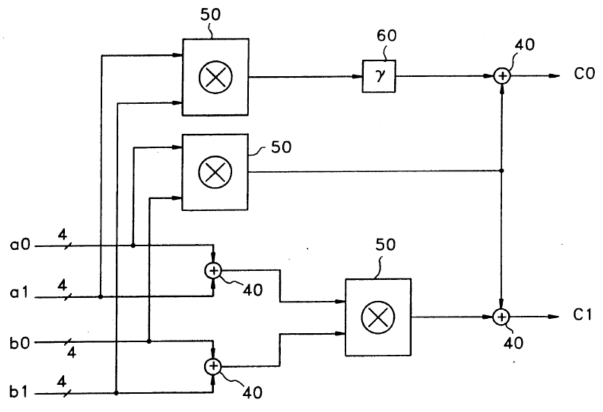


그림 2. 3개의 $GF(2^4)$승산기와 4 개의 덧셈기를 사용한 $GF(2^8)$ 승산기
Fig. 2. $F(2^8)$ multiplier calculation by 3 $GF(2^4)$ multipliers and 4 adders.

Hence to optimize the processor circuiit :

(1) We addedsimple X^4, X^2 circuits to do parallel processing to speed up the computing.

(2) We replaced the huge $GF(2^8)$ multiplier wi th 3 $GF(2^4)$ multipliers and 4 $GF(2^4)$ adders to simplify the circuit and speedup the processor in $GF(2^4)$.

(3) Total gate count of the 3 $GF(2^4)$ multipliers, 4 $GF(2^4)$ adders and X^2/X^4 is smaller than that of the huge $GF(2^8)$ multiplier[6, 8].

(4) We don't use the huge divider circuit to

simplify the machine greatly.

In Fig. 1, the X^4circuit is derived by repeatedly apply the X^2 circuit.

Ⅲ. Optimized Processor structure and operation for Chien search machine

All the GF(28) operations are converted to $GF(2^4)$ operations for optimization and for par allel processing, we need Multiplier, Squarer, X^4 circuits.

Let

$C = A \times B$, and $D= A^2$, $E = A^4$

where

$$C,D,A,B,E \in GF(2^8) \tag{7}$$

Here if

$$C= C0 + \beta C1,\ C1 \text{ and } C0 \in GF(2^4)$$

Also

$$D= D0 + \beta D1,\ D0 \text{ and } D1 \in GF(2^4)$$

and

$$E= E0 + \beta E1,\ E0 \text{ and } E1 \in GF(2^4)$$

Then

$$C0= A0B0 + A1B1\gamma$$
$$C1= A0B1+ A1B0+A1B1 \tag{8}$$

Also

$$D0= A0^2+ A1^2 \gamma ,D1= A1^2 \tag{9}$$

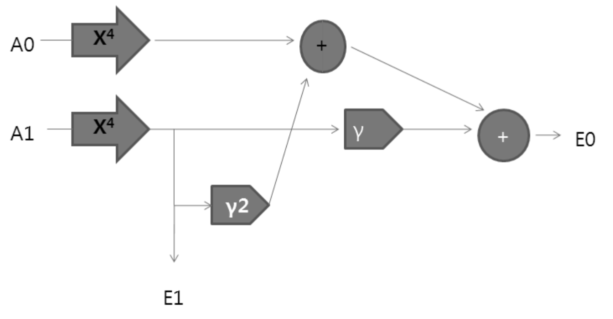


그림 3. X^4 회로
Fig. 3. X^4 circuit.

$$C= C0 + \beta(A0B0 +(A0+A1)(B0+B1)) \tag{10}$$

$$E0= A0^4+A1^4\gamma^2+A1^4\gamma$$
$$E1= A1^4 \tag{11}$$

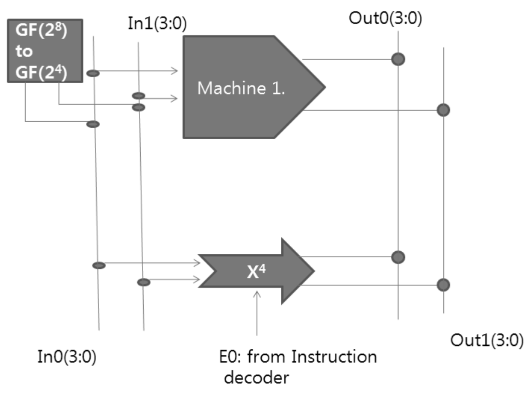
Now we show X^4 circuit in Fig. 3.

If we describe $X^4(GF(2^8))$operation in $GF(2^4)$ micro executions :

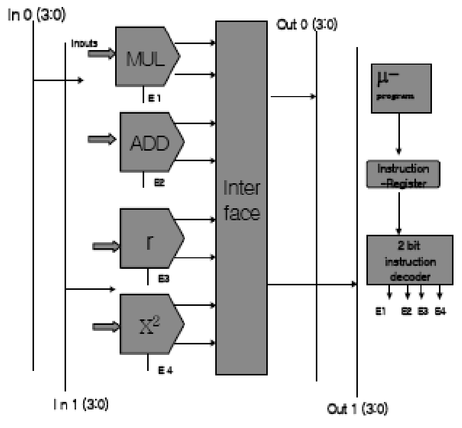
$A1^4$; E1, X^4 in $GF(2^4)$

$\gamma A1^4$; γ multiplier, $\gamma^2 A1^4$; γ multiplier

$A0^4+A1^4\gamma^2 +.A1^4\gamma$; Adder in $GF(2^4)$



(a)



(b)

그림 4. (a) GF(24)덧셈기, 승산기, ɣ승산기를 를 포함하는 최적 처리기 구조 (b) 머신 1과 버스의 연결
Fig. 4. (a) Optimized Processor Structure, which contains GF(24) adders, Multipliers, ɣ multiplier, GF(24) X^4,X^2units (b) Machine 1 connection to buses.

Now when we see equations 9, 10, 11 we see that 3 GF(2^4) multipliers,4 GF(2^4) adders, ɣ multiplier,X^4, X^2 circuits in GF(2^4) are enough to do GF(2^8) multiply, X^4,X^2 operations simultaneously. Using these GF(2^4) micro execution units, we can make faster and logically simpler processor for calculating error locations of Reed Solomon decoder[3, 8]. In Fig.4 we show the Processor structure.

Ⅳ. Application of optimized Processor to the analysis of 4 error case

Error locator polynomial for 4 error case is as in equation (12).

$$\mathrm{Det}(A_4)x^4+\sigma_1'x^3+\sigma_2'x^2+\sigma_3'x+\sigma_4'=0 \qquad (12)$$

as we already showed $X^4 \in$ GF(2^8) circuit in section 3,we here show X^2 and multiply operation using GF(2^4) execution units. So parallel processing of X^4,X^2 and multiply in GF(2^8) field is possible.

$X^2 \in$GF(2^8) operation in GF(2^4) micro executions :

$A0^2$: $X^2 \in$ GF(2^4) execution

$A1^2$: D1,$X^2 \in$ GF(2^4) execution

$A1^2$ ɣ : ɣ multiplier

$A0^2$+ $A1^2$ ɣ : GF(2^4) Adder ,D0

These micro operations are from equation (9).

Multiply in GF(2^8) using GF(2^4) microexecutions :

A1B1,A0B0 : 2 GF(2^4) multiplying

A0+A1, B0+B1 : 2 GF(2^4) adder

(A0+A1)(B0+B1) : 3rd GF(2^4) multiplying

(A0+A1)(B0+B1)+A0B0 : C1 3rd GF(2^4) adder

A1B1ɣ : ɣ multiplier

A0B0 + A1B1ɣ : GF(2^4) adder, C0

These micro operations are from equation (10).

Since GF(2^4) multiply takes much more time than GF(2^4) adding, x^2 and x^4, we can parallelly process GF(2^8) multiply, x^2, x^4 executions without computational time loss. In Fig.5 we show critical paths of GF(2^4) multiplier and we see that it is 2.5 times longer than GF(2^4) x^2 and x^4 circuit in Fig.1.

The reason why we do X^2, X^4, multiply operations

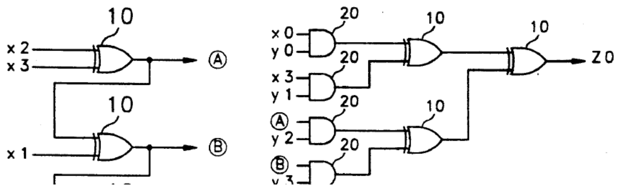


그림 5. GF(24) 승산기의 최장경로
Fig. 5. Critical path of GF(24) multiplier (x2->z0).

all in GF(2^8) field ,using components (execution all in GF(2^8) field, using execution units in GF(2^4), is: We can execute X^2, X^4, multiplier units of GF(2^4) simultaneously.

So we can greately save the Comp uting time.

Ⅳ-1. X^k (k=1,2,3,4) computing

1. Compute X^2, X^4 by GF(2^4) X^2, X^4 circuits

2. Compute X^3 by X^2 multiply X using GF(2^4) multipliers.

Ⅳ-2. Error locator polynomial coefficients computing

Now from (6)

$$\sum_{k=0}^{4} \sigma'_k X^{-k} = 0 \text{ and } \sigma'_0 = \det(A_4) \qquad (13)$$

Here we need not use the Dividing circuit.

1. for example to compute

$$\begin{aligned}\sigma_4' &= S_5\gamma_1+S_6\gamma_2+S_7\gamma_3+S_8\gamma_4 = \\ &(S_3S_5{}^2S_7+S_5{}^4+S_5S_6{}^2S_3+S_4{}^2S_7S_5) + \\ &(S_2S_6{}^3+S_3S_4S_7S_6+S_4S_5{}^2S_6+ \\ &S_2S_5S_7S_6+S_3S_5S_6{}^2+S_4{}^2S_6{}^2) + \\ &(S_2S_4S_7{}^2+S_3S_5{}^2S_7+S_3S_4S_6S_7+ \\ &S_4{}^2S_5S_7+S_2S_5S_6S_7+S_3{}^2S_7{}^2)+ \\ &(S_2S_4S_6S_8+S_4{}^3S_8+S_2S_5{}^2S_8+S_3{}^2S_6S_8)\end{aligned} \qquad (14)$$

2. Equation (14) takes about 44T, where T is GF(2^4) multiplier execution time, since $S_5{}^2$, $S_5{}^4$, S_3S_4(x^2 , x^4 ,multiply operations) are simultaneous usly done.

3. So To compute σ'_j(j= 4, 3, 2, 1,0), we need approximately 220(44X5)T . If we use old method,

60X5T_8 ($T_8$is GF(2^8) multiplier execution time). So New processor is much faster than old method [2,4].

V. Step by step example to find out 4 error locations using the Processor

Let transmitted code is (0,0, ⋯,0) (all zeroes) and received code is (α^{13} , α^8,α^5,1,0, ⋯,0) ∈ GF(2^8). Find error Loc ator Polynomial and its solutions.

<Sol>

First we get syndromes as follows.

$S_8=r(x)\mid_{x=\alpha^8}=\alpha^{13}+\alpha^{16}+\alpha^{21}+\alpha^{24}=\alpha^{181}\in$ GF(2^8)= (α^{10}, α^{12}) ∈ GF(2^4) (15)

$S_7= r(x)\mid_{x=\alpha^7}=r_0+r_1x+r_2x^2+r_3x^3\mid_{x=\alpha^7}=\alpha^{13}+\alpha^8\alpha^7+\alpha^5\alpha^{14}+\alpha^{21}=\alpha^{13}+\alpha^{15}+\alpha^{19}+\alpha^{21}=\alpha^{254}\in$ \GF(2^8) =(α^0, α^{13}) ∈ GF(2^4) (16)

Similarlily,

$S_6=\alpha^{138}\in$GF(2^8)$=\alpha+\beta\alpha^6$ =

= (α, α^6) ∈ GF(2^4) (17)

$S_5=0$ ∈GF(2^8)= (0,0) ∈ GF(2^4)=S_4

$S_3=\alpha^{109}$ ∈GF(2^8)=(0, α^3)∈GF(2^4)

$S_2=\alpha^{74}$ ∈GF(2^8)=(α^6, α^{14})∈GF(2^4)

$S_1=\alpha^{39}$ ∈GF(2^8)=(α^8, α^2)∈GF(2^4) (18)

The fourth order characteristic matrix is [5]

$$A_4=\begin{pmatrix} S_1 & S_2 & S_3 & S_4 \\ S_2 & S_3 & S_4 & S_5 \\ S_3 & S_4 & S_5 & S_6 \\ S_4 & S_5 & S_6 & S_7 \end{pmatrix}=\begin{pmatrix} \alpha^{39} & \alpha^{74} & \alpha^{109} & 0 \\ \alpha^{74} & \alpha^{109} & 0 & 0 \\ \alpha^{109} & 0 & 0 & \alpha^{138} \\ 0 & 0 & \alpha^{138} & \alpha^{254} \end{pmatrix} \quad (19)$$

From equations (19) and (3) , (4), We get :

$$\begin{pmatrix} \sigma'_4 \\ \sigma'_3 \\ \sigma'_2 \\ \sigma'_1 \end{pmatrix} = \text{Adjoint of } A_4 \cdot \begin{pmatrix} S_5 \\ S_6 \\ S_7 \\ S_8 \end{pmatrix}$$

$$= \begin{pmatrix} C_{11} & C_{21} & C_{31} & C_{41} \\ C_{12} & C_{22} & C_{32} & C_{42} \\ C_{13} & C_{23} & C_{33} & C_{43} \\ C_{14} & C_{24} & C_{34} & C_{44} \end{pmatrix}\begin{pmatrix} S_5 \\ S_6 \\ S_7 \\ S_8 \end{pmatrix} \quad (20)$$

Where C_{ij} is ith row jth column cofactor of A_4.

표 1. C_{ij} 값들
Table 1. Values of C_{ij}.

	GF(2^8)	GF(2^4)
C_{11}	α^{130}	(α,α^3)
C_{12}	α^{95}	(α^2,α^2)
C_{13}	α^{217}	(α^{14},α^{11})
C_{14}	α^{101}	(α^3 ,α)
C_{22}	α^{101}	(α^3 ,α)
C_{23}	α^{182}	(α^2, α^9)
C_{24}	α^{66}	(α^8, α^3)
C_{33}	0	(0, 0)
C_{44}	α^{72}	(α^{10},α^6)
* C_{ij}=C_{ji} (i,j= 1,2,3,4)		

Because S_5, S_4 is zero,

$C_{11}=S_6^2S_3$, $C_{12}=S_2S_6^2=C_{21}$, $C_{13}=C_{31}=S_3^2S_7$, $C_{14}=C_{41}=S_3^2S_6$, $C_{22}=S_6^2S_1+S_3^2S_7$,$C_{23}=S_2S_3S_7=C_{32}$, $C_{33}=S_1S_3S_7+S_2^2S_7$, and $C_{34}=C_{43}=S_1S_3S_6+S_2^2S_6$, $C_{44}=S_3^3$.

Table 1 shows the values of C_{ij} in GF(2^8) and GF(2^4).

From equation (20), we find equ ation (21).

$$\begin{pmatrix} \sigma'_4 \\ \sigma'_3 \\ \sigma'_2 \\ \sigma'_1 \end{pmatrix} = \begin{bmatrix} \alpha^{77} \\ \alpha^{149} \\ \alpha^{65} \\ \alpha^{146} \end{bmatrix} \in GF(2^8)$$

$$= \begin{pmatrix} 1 & \alpha^4 \\ \alpha^7 & \alpha^4 \\ \alpha^{10} & \alpha^9 \\ \alpha^{11} & \alpha^{11} \end{pmatrix} \in GF(2^4) \quad (21)$$

Also $\det(A_4)=\sum_{k=1}^{4}C_{k1}S_k=\alpha^{71}$

$\in$GF(2^8) = (1,α^3) ∈ GF(2^4) (22)

So error locator polynomial is :

$\det(A_4)\,x^4+\sigma_1'x^3+\sigma_2'x^2+\sigma_3'x+\sigma_4'=$
$\alpha^{71}x^4+\alpha^{146}x^3+\alpha^{65}x^2+\alpha^{149}x+\alpha^{77}=0$
$=\sigma(x)$ (23)

Now substitute x=1, α, α^2, α^3, α^4⋯ ,α^{254} to (23) to find table 2.

From table 2. we see that Error locations are α^0, α, α^2, α^3. This is correct !!.

표 2. 오류위치 추적표
Table 2. Error location finding table.

X	$\sigma(x) \in$ **GF(2^8)**	$\sigma(x) \in$ **GF(2^4)**
α^0 =1	0	(0,0)
α	0	(0,0)
α^2	0	(0,0)
α^3	0	(0,0)
α^4	α^{220}NonZero	(α^4,α^2)NonZero
⋮	⋮	⋮
α^{254}	(α^{210}) NonZero	(α^5,α^{13})NonZero

Ⅵ. Conclusion

In this paper, we showed that by using subfield theory, No divider circuit,parallel pr ocessing , Chien search machine can be des igned in much higher speed and simpler circu it so being resulted in optimized Chien search Processor[3, 7].

In Future, we will design the o8yptimized pro cessor to find the error value of Reed Solom on decoder using very efficient and high spe ed Galois field Divider[1].

References

[1] 최효진, 지현우, 성원용, "낸드 플래시 메모리오류 정정을 위한 병렬 BCH 복호기의 최적설계", 제16회 한국반도체학술대회, pp505~506, 2009
[2] HKAn," Design Optimization of the Arit hmatic Logic Unit Circuit for the Processor to Determine the Number of Errors in the Reed Solomon Decoder,"Jour.of KICS, pp 649~654 ,2011 Nov.
[3] Paulius Ruzgys, "Analyzing and Implementing a Reed-Solomon decoder in ADSL", 2007 June, MS Thesis,Institute ofElectronic systems, Aalborg Univ.,Denmmark.
[4] Joschi Brauchle, Ralf Koetter; A Systematic Reed Solomon Encoder with Arbitrary Parity Positions, IEEE GLO BECOM 2009 Pro-ceedings.
[5] T.K. Moon, Error Correction Coding: Math-ematical Methods and Algorithms, Hoboken, NJ: John Wiley & Sons, Inc., 2005.
[6] Hsu; Yueh-Teng, USP7984366 Efficient chien search method in reed-solomon dec oding, and recording medium, Aug.2007
[7] M. L.Cury, A.Skjellum, H.L. Ward, "Acceler-ating Reed-Solomon coding in RAID systems with GPUs", in Parallel and Distributed Processing,2008,IPDPS 2008. IEEE International Symposium on,2008
[8] Pusan Patel, "Parallel Multiplier design for Galois/Counter Mode of operation", Univ. of Wateroo Canada, Dep. of EE ,2008 MS Thesis.

저 자 소 개

안 형 근(정회원)
1979년 서울대학교 전기공학과 졸업
1981년 KAIST 전기 및 전자과 졸업
1988년 뉴욕 주립대 전기과 Ph.D
1988년~1998년 삼성전자 수석
1998년~1999년 텔슨전자 이사
2000년~현재 동명대학교 정보통신과 교수
<주관심분야: Digital System Design, LCD/OLED display,반도체>