New Efficient Design of Reed-Solomon Encoder, Which has Arbitrary Parity Positions, without Galois Field Multiplier

Hyeong-Keon An * ***Regular Member***

Abstract

In Current Digital C^3 Devices(Communication, Computer, Consumer electronic devices), Reed-Solomon encoder is essentially used.For example we should use RS encoder in DSP LSI of CDMA Mobile and Base station modem, in controller LSI of DVD Recorder and that of computer memory(HDD or SSD memory). In this paper, we propose new economical multiplierless (also without divider) RS encoder design method. The encoder has Arbitrary parity positions.

Key words : Reed-Solomon(RS), Encoder, GF(2^8), Parity, Communication, Digital, Multiplier, Divider, LSI

Ⅰ. Introduction

RS codes are systematic linear block codes, which are correcting non binary data symbols,whereas BCH codesare correecting binary data streams. Anyway it is a subset of BCH code.

These codes are specified as RS (n, k), with m bit symbols. This means that the encoder takes k data symbols of m bits each, appends n - k parity

symbols, and produces a code word of n symbols (each of m bits). Normally these parities are positioned in the left end partof each m bits code.

Reed Solomon codes are based on a specialized area of mathematics known as Galois fields (a.k.a. finite fields). RS makes use of Galois fields of the form GF (2^m), where elements of the field can be represented by m binary bits. Most RS codes of the current computer and communication devices use GF(2^8). So in this paperwe also use GF(2^8) elements[1,2].

Primitive polynomials are of interest here because they are used to define the Galois field. Apopular choice for a primitive polynomial is[3,4,8] :

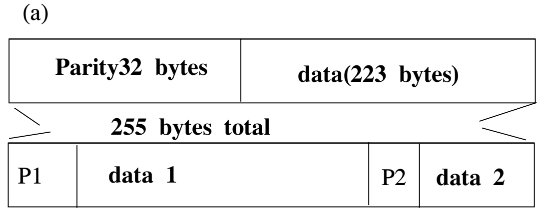
$$p(x)=x^8+x^4+x^3+x^2+1 \qquad (1)$$

Where we use GF(2^8), which is usedthroughout the Paper.

Especially our new design can put parities in any positions in aCodewordofTheRS encoder as shownin-figure 1(b). Even completely scattered paritiesa-repossible in completelyseperatedp positions,where pis the number of parities in a codeword[7,8].

In this paper chapter Iintroduce the whole paper. In chapter Ⅱ, webriefly described how the Reed Solomon encoder equationsarederived toget theparity symbols using GF(2^8) Arithmatic and Logicoper-ations. Here parities can be positioned arbitrarily in the codeword and we derive ROM which replaces GF(2^8) multiplier. In chapterIII, wepropose the way

(a)



(b) P1 5 bytes, data1 170 bytes, P2 27bytes, data2 85 bytes

Fig. 1. (a) RS(255,32) Codeword Type1 (b) RS(255,32) Codeword Type2

* 동명대학교 정보통신공학과(hkan@tu.ac.kr)
논문번호 : KICS2010-03-133, 접수일자 : 20100년 3월 31일, 최종논문접수일자 : 2010년 6월 10일

how to reduce the ROM size to make economical Si wafer of RS encoder. In chapter IV,Vanalysing the new RS encoder we compare itwith former RS Encoder, So finding what is advantageous over former method by providng examples.In chapter VI future worksandimprovements that whould be made in future will be described.

Ⅱ. Mutiplierless Derivation of Reed-Solomon Encoder Which has Arbitrary Parity Position

2.1 General RS Encoder Design Theory

Let's think about 4 parity positions (n parities case is alsoin the same way) as $\alpha^i,\alpha^j,\alpha^k,\alpha^l \in GF(2^8)$ and $i \neq j \neq k \neq l$. Then Let's define A matrix as equation (2).

$$\mathbf{A} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ \alpha^i & \alpha^j & \alpha^k & \alpha^l \\ \alpha^{2i} & \alpha^{2j} & \alpha^{2k} & \alpha^{2l} \\ \alpha^{3i} & \alpha^{3j} & \alpha^{3k} & \alpha^{3l} \end{pmatrix} \quad (2)$$

Then 4 parities are as in equations (3).

$$P_i = \frac{1}{Det(A)} \sum_{k=0}^{3} A_{(i+1)(k+1)} S_k \quad (3)$$

Where i=0,1,2,3 and A_{ij} (i,j=1,2,3,4) are cofactors of A matrix shown in Eq. (2).

We here derive RS Encoder in two cases. One is using $GF(2^8)$ operation and The other is using $GF(2^4)$ operation even though we are designing RS Encoder of $GF(2^8)$ Elements.

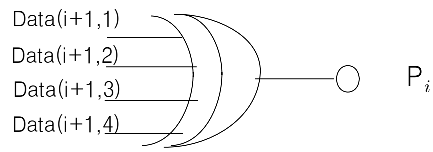
2.2 Multiplierless RS encoder design using ROM of $GF(2^8)$ elements.

In this case, equations (3) show there are 16 multiplications to get 4 parities P_i(i=0,1,2,3)s. So we compose 16 ROMs , whose address is S_k(k=0,1,2,3) which can be α^i($i \in 0,1$ ...,255), and data is$S_k \cdot A'_{ij}$ (i,j=1,2,3,4, and $A'_{ij} = \frac{A_{ij}}{Det(A)}$), Let's call the ROM as ROM_{ij}.Figure 2 shows ROM_{ij} and parity

(a) ROM_{ij} Format

Address	Data
α^0	A'_{ij}
α^1	$\alpha^1 A'_{ij}$
:	:
S_k	$S_k \cdot A'_{ij}$
:	:
α^{254}	$\alpha^{254} \cdot A'_{ij}$

(b)P_i(i=0,1,2,3) Generation



Here Data(i+1,k+1) is $S_k \cdot A'_{(i+1)(k+1)}$(k=0,1,2,3).

Fig. 2. ROM_{ij} and P_i Generation

generation. Since ROM size is 256 word by 8 bit/1 Word and we need 16 ROMs, The design is relatively inefficient to have relatively many ROMs (Former Method has 2 or 3 ALU ROMs(256X8 size)). Now P_i(i=0,1,2,3) can be got using Eq.(3). So

$$P_i = \sum_{k=0}^{3} \mathbf{A}'_{(i+1)(k+1)} S_k = \sum_{k=0}^{3} Data(i+1,k+1) \quad (4)$$

Where Data(i+1,k+1) is ROM data out of $ROM_{(i+1)(k+1)}$.

Ⅲ Multiplierless RS encoder design ,which is functioning in $GF(2^8)$ field, using ROM of $GF(2^4)$ elements

From equation (3), If C_{ik} is defined as follows

$$C_{ik} = \frac{1}{Det(A)} A_{(i+1)(k+1)} = \mathbf{A}'_{(i+1)(k+1)} \quad (5)$$

ThenP_i is expanded in GF(2^4) field as in equation(6).

$$
\begin{aligned}
P_i &= \sum_{k=0}^{3} C_{ik}S_k \quad (i=0,1,2,3) \\
&= \sum_{k=0}^{3} (C_{ik,0}+\beta C_{ik,1})(S_{k,0}+\beta S_{k,1}) \qquad (6) \\
&= \sum_{k=0}^{3} ((C_{ik,0}S_{k,0}+\gamma C_{ik,1}S_{k,1})+ \\
&\quad \beta(S_{k,1}C_{ik,0}+C_{ik,1}S_{k,0}+C_{ik,1}S_{k,1}))
\end{aligned}
$$

Here $C_{ik,0}$, $S_{k,0}$, $C_{ik,1}$, $S_{k,1} \in$GF(2^4), and $\gamma, \beta \in$ GF(2^8), also $\beta^2+\beta=\gamma$[1].

From equation (6) we need 2ROMs , one of these ROM addresses is $S_{k,0}$ or $S_{k,1}$ andcorresponding data is $C_{ik,0}$•($S_{k,0}$ or $S_{k,1}$) .The other ROM address isalso $S_{k,0}$ or $S_{k,1}$ and corresponding data is $C_{ik,1}$• ($S_{k,0}$ or $S_{k,1}$).Let's call them $C_{ik,1}$ ROM and $C_{ik,0}$ ROM. But i =0~3, andk=0~3 So there are 16 Variations of the ROM for each ($C_{ik,0}$, $C_{ik,1}$) pair, and This means we need 32 ROMs and each has 16 words X (4 bits/word). The total Size of these 32 ROMs is the same as 256 words X (8 bits/word) and is $\frac{1}{16}$ size of former GF(2^8) Multiplierless ROM

$C_{ik,1}$ ROM format

Address	Data
a^0	$C_{ik,1}$
a^1	$a^1C_{ik,1}$
:	:
$S_{k,0}$or $S_{k,1}$	($S_{k,0}$or $S_{k,1}$)$C_{ik,1}$
:	:
a^h	$a^hC_{ik,1}$
:	:
a^{14}	$a^{14}C_{ik,1}$

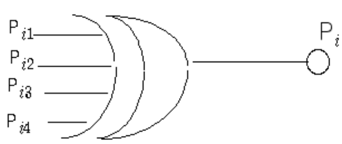
Fig. 3. GF(2^4) ROM *$C_{ik,0}$ROM(Same as$C_{ik,1}$ROM except index ik,1is changed to ik,0).

case (**II.2. Case**). So This is good. In Next section we apply this method (Multiplierless RS encoder using GF(2^4) ROM) to practical encoder design of CDP (Compact Disc Player, Digital Audio Device).

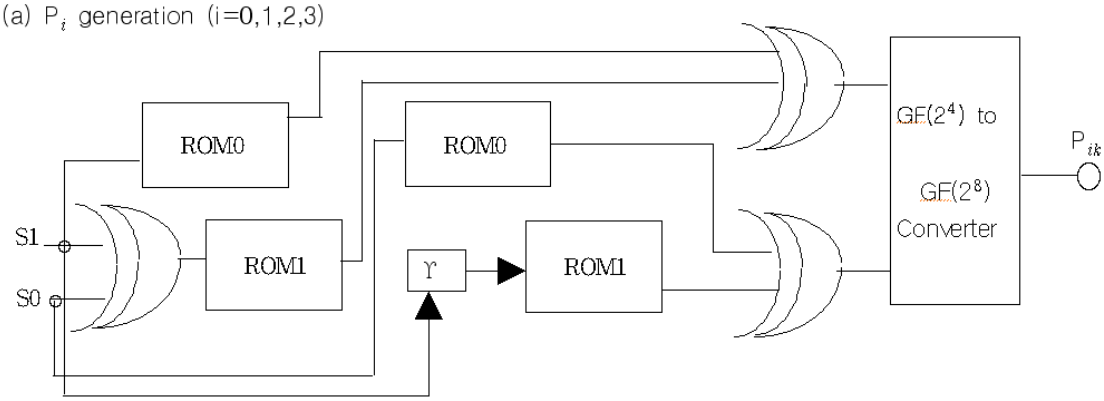
Figure 3 shows$C_{ik,1}$ ROM and $C_{ik,0}$ROM format and P_i(i=0,1,2,3) Generation. P_i is generated from P_{ik} (k=0,1,2,3) and is derived as in equation 7.

$$
\begin{aligned}
P_{ik} &= P_{ik,0}+\beta P_{ik,1} \quad (k=0,1,2,3) \\
&= (C_{ik,0}S_{k,0}+\gamma C_{ik,1}S_{k,1})+\beta(S_{k,1}C_{ik,0} \qquad (7) \\
&\quad +C_{ik,1}S_{k,0}+C_{ik,1}S_{k,1})
\end{aligned}
$$

*$P_{ik}=P_{ik,0}+\beta P_{ik,1}$, $S_k=S_{k,0}+\beta S_{k,1}$
$C_{ik}=C_{ik,0}+\beta C_{ik,1}$,and $C_{ik,0}, S_{k,0}, C_{ik,1}, S_{k,1}$
$P_{ik,0}, P_{ik,1} \in$GF(2^4) also $\beta, P_{ik}, C_{ik} \in$GF($2^8$).



(a) P_i generation (i=0,1,2,3)



*$C_{ik,0}$ROM is ROM0 and $C_{ik,1}$ROM is ROM1
Also S1 is $S_{k,1}$ and S0 is $S_{k,0}$.

(b) P_{ik} generation (k=1,2,3,4)

Fig. 4. P_i and P_{ik} generation

Where $P_{ik,0}$,$P_{ik,1}$ $\in$GF(2^4), and P_{ik}, $\beta \in$GF(2^8). figure 4 showsP_{ik} and P_i block diagram.

Ⅳ. Design of CDP Reed Solomon encoder in GF(2^8) field using Multiplierless GF(2^4)ROM.

Now For CDP(Compact Disc Player), we use GF(2^8) elements, and For Encoder, we use RS(32,28) code system. From Equation (2), for CDP Encoder, we know i=0, j=1, k=2, l=3 so from equation (5) and (6)[5], we get Equation 8.

$$[C_{ik}]=\begin{pmatrix} \alpha^{110} & \alpha^{50} & \alpha^{48} & \alpha^{104} \\ \alpha^{50} & \alpha^{30} & \alpha^{149} & \alpha^{45} \\ \alpha^{48} & \alpha^{149} & \alpha^{27} & \alpha^{44} \\ \alpha^{104} & \alpha^{45} & \alpha^{44} & \alpha^{101} \end{pmatrix} \cdot \alpha^{108} \qquad (8)$$

Through transformation into GF(2^4) We get equation (9).

$$[C_{ik}]=[C_{ik,0}] + \beta[C_{ik,1}] \qquad (9)$$

whereI,k=0,1,2,3and $[C_{ik,0}]$,$[C_{ik,1}] \in$ GF(2^4)[1].

Using VHDL simulations,equation (9) is transformed to equation (10).

$$[C_{ik,0}]=\begin{pmatrix} \alpha^{7} & \alpha^{9} & \alpha^{5} & \alpha^{3} \\ \alpha^{9} & \alpha & \alpha^{2} & \alpha^{12} \\ \alpha^{5} & \alpha^{2} & \alpha^{14} & \alpha^{12} \\ \alpha^{3} & \alpha^{12} & \alpha^{12} & \alpha^{3} \end{pmatrix} \qquad (10)$$

$$[C_{ik,1}]=\begin{pmatrix} \alpha^{6} & \alpha^{3} & \alpha^{8} & \alpha^{4} \\ \alpha^{3} & \alpha^{6} & \alpha^{7} & 0 \\ \alpha^{8} & \alpha^{7} & \alpha^{12} & \alpha^{10} \\ \alpha^{4} & 0 & \alpha^{10} & \alpha^{12} \end{pmatrix}$$

Finally composite CDP Encoder ROM(Data Fields of ROM0and ROM1 are added to make 8 bit word for each address) format is shownin figure 5. In Appendix I we show the part of the actualCDPencoder ROMwhich uses4 paritiesand GF(2^4) field[2].

Appendix II showsVHDL code for equation 10 simulation.

Using the encoder ROM and the circuit in fig.4 we can find 4 parities for the CDP. For other Digital Communication and A/V devices(ex: HDTV, CDMA mobilephone, Digital Audio Tape Recorder, MP3, etc.) we can use the same method presented in this section. Now we show the RS encoder design example[3,4].

<Example>

LetcodepolynomialC(X)beasin equation(12), then, we want to find4 parities.

$$\mathbf{C(x)= P_0+P_1X+P_2X^2+P_3X^3+\sum_{i=4}^{31} D_iX^i} \qquad (12)$$

i • k	$S_{k,0}$or $S_{k,1}$	$C_{ik,0}\cdot(S_{k,0}$ or $S_{k,1})$	$C_{ik,1}\cdot(S_{k,0}$ or $S_{k,1})$
0000 I=00, K=00 (ik=0)	0 α α^2 : α^{15}	0 $C_{0,0}\cdot\alpha$ $C_{0,0}\cdot\alpha^2$: $C_{0,0}\cdot\alpha^{15}$	0 $C_{0,1}\cdot\alpha$ $C_{0,1}\cdot\alpha^2$: $C_{0,1}\cdot\alpha^{15}$
:	:	:	:
1111 I=11, K=11 (ik=15)	0 α α^2 : α^{15}	0 $C_{15,0}\cdot\alpha$ $C_{15,0}\cdot\alpha^2$: $C_{15,0}\cdot\alpha^{15}$	0 $C_{15,1}\cdot\alpha$ $C_{15,1}\cdot\alpha^2$: $C_{15,1}\cdot\alpha^{15}$

Fig. 5. CDP Encoder ROM Format. (i • K and ($S_{k,0}$or $S_{k,1}$) is address field of the ROM .$C_{ik,0}$ •($S_{k,0}$ or $S_{k,1}$),$C_{ik,1}$ •($S_{k,0}$ or $S_{k,1}$) are data field of the ROM

WhereP_i(i=0,1,2,3) are 4parities ofCDP Player, andD_i(i=4,5,..,31)are 28 data symbols for 1code word. If $D_4=D_8=D_{12}=D_{16}=D_{20}=1$, and all theother data symbols are 0 , find 4 paritiesP_i(i=0,1,2,3) $\in$ **GF(2^8).**

<Sol> Let's assume P_i (i=0,1,2,3)=0$\in$ GF(2^8). Then

$$\begin{pmatrix} S_0 \\ S_1 \\ S_2 \\ S_3 \end{pmatrix} = \begin{pmatrix} 1 \\ \alpha^{119} \\ \alpha^{238} \\ \alpha^{17} \end{pmatrix} \in GF(2^8)$$

$$= \begin{pmatrix} 1 \\ \alpha \\ \alpha^2 \\ \alpha^{13} \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \in GF(2^4) \quad (13)$$

So frome quation (6) and the circuitin fig. 4, we find

$$\mathbf{P}_i = \sum_{k=0}^{3} \mathbf{C}_{ik}\mathbf{S}_k \ \ \textbf{(i=0,1,2,3)} = \sum_{k=0}^{3} (\mathbf{C}_{ik,0}\mathbf{S}_{k,0} + \beta(\mathbf{C}_{ik,1}\mathbf{S}_{k,0})) \quad (14)$$

Equation (14) can be calculated using the ROM in fig.5 .Also , forthis special case noticethat $S_{k,1}$ (i=0,1,2,3)=0 as shown in equation (13).Generally it isn't Zero.Nowtheresultis shown in equation (15).

$$\begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} \alpha^8 \\ \alpha \\ \alpha^5 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} \alpha^8 \\ \alpha^{10} \\ \alpha \\ 0 \end{pmatrix} \in GF(2^4) \quad (15)$$

Finallyequation(15) is converted to GF(2^8) elementsusingGF(2^4) to GF(2^8)converter[1] ,and canbe solvedbyVHDL Simulation, Whose code is shown in Appendix II.

The parities in GF(2^8) is as in Equation (16).

$$P = \begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} = \begin{pmatrix} \alpha^{214} \\ \alpha^{226} \\ \alpha^{242} \\ 0 \end{pmatrix} \in GF(2^8) \quad (16)$$

Now Let's prove they are correct parities. The syndromes are as in equation (17).

$$\begin{aligned}
&S_0=P_0+P_1X+P_2X^2+P_3X^3+\sum_{i=4}^{31} D_iX^i \mid (X=1) \\
&= P_0+P_1+P_2+P_3+\sum_{i=4}^{31} D_i=1+1=0. \\
&S_1=P_0+ \\
&P_1\alpha+P_2\alpha^2+P_3\alpha^3+\alpha^4+\alpha^8+\alpha^{12}+\alpha^{16}+\alpha^{20}= \\
&\alpha^{214}+\alpha^{227}+\alpha^{244}+\alpha^4+\alpha^8+\alpha^{12}+\alpha^{16}+\alpha^{20}= \\
&\alpha^{119}+\alpha^{119}=0. \\
&S_2=P_0+ \\
&P_1\alpha^2+P_2\alpha^4+P_3\alpha^6+\alpha^4(\alpha^4+\alpha^{12}+\alpha^{20}+\alpha^{28}+ \\
&\alpha^{36})=\alpha^{238}+\alpha^{238}=0. \\
&S_3=P_0+ \\
&P_1\alpha^3+P_2\alpha^6+P_3\alpha^9+\alpha^{12}+\alpha^{24}+\alpha^{36}+\alpha^{48}+\alpha^{60} \\
&=\alpha^{17}+\alpha^{17}=0. \in GF(2^8)
\end{aligned} \quad (17)$$

From **equation** (17),Allthesyndromes are Zeroes, and This means that the parities shown in (16) are correct parities.

V. Comparison the Method with Former Method

Our new method is different from the former Reed-Solomon encoder design in 2 respects[8].

1) J. Brauchle's RS encoder needs GF(2^8) multiplier and huge Chien machine[3], but our design doesn't need and just change the ROM contents in fig.5 to change the parity position, resulting in simpler and faster design.

2) We compute all Arithmatic and Logic operations in GF(2^4) field. After all ALU operations, we transform the results into GF(2^8) elements. As we see in table 1,GF(2^4) operations requires much less logic gates and propagation delay, hence the speed of the new encoder in this paper becomes much faster and cheaper[2].

3) Even inGF(2^4) field, the most oftenly used GF(2^4) multiplier is replaced by much faster GF(2^4) multiplying ROM. So for Multiply, we need just accessing the ROM and the output comes out resulting in the faster speed.

We summarize differences and advantages over former methods as below and in Table1[1,4].

1) The classical method uses GF(2^8) multiplier or GF(2^8) encoding Generator derived from GF(2^8) primitive polynomial[3]. Butour method doesn't use these and only GF(2^4) multiplying ROM is used.

2) Because, playing (RS decoding) and recording (encoding) can be simultaneously done, and RS decoder uses GF(2^4) multiplier and encoder uses GF(2^4) multiplying ROM, also gate counts and propagation delay is much smaller ,new encoder is much fasterand cheaper than former encoder [1,4].

Ⅵ. Conclusion

Thesedays, almost allDigitalElectronic Devices (Ex: Digital A/V devices, Communication devices (MobilePhone, Modem),Computer Memory Unit) uses RS codec for their data protection. In this case,Normally Encoding/Decoding is simultaneously performed (EX: While receiving the mobilephone Call (RS Decoding), Talking(RS Encoding) is simultaneously Performed). So If RS decoder and encoder uses same RS Multiplier, There Will be Confliction of the ALU unit using and Speed will be Greatly decreased. In this case, If we use the RS Multiplerless encoder shown here and normal RS decoder Speed won't be decreased and The Gate counts for RS Codec VLSI Implementation can be greately reduced So make the economicaldesign possible[6].

In the future we will research about the Multiplierless RS decoder design resulting in Further reduction of gate counts and speed improvement[6].

Appendix I. Part of CDP Encoder ROM

i・k	$S_{k,0}$ or $S_{k,1}$	$C_{ik,0}$($S_{k,0}$ or $S_{k,1}$)	$C_{ik,1}$($S_{k,0}$ or $S_{k,1}$)
0000 (ik=0)	0 α α^2 : α^{15}	0 α^8 α^9 : α^7	0 α^7 α^8 : α^6
⋮ ⋮	⋮ ⋮	⋮ ⋮	⋮ ⋮
1110 (ik=14)	0 α α^2 : α^{15}	0 α^5 α^6 : α^4	0 α^{13} α^{14} : α^{12}

**$C_{0,0}=\alpha^7 \cdots C_{15,0}=\alpha^4$, $C_{0,1}=\alpha^6 \cdots C_{15,1}=\alpha^{12}$

Appendix II. VHDL Code for Simulation of Equation (9)

Table 1. Comparison between new method and classical RS encoder design method

a) Reed-Solomon encoder design with multiplier

	Classical RS Encoder	Classical RS Encoder with GF(2^4) operation
GF(2^8) Multiplier implementation	137gates	123 gates
GF(2^8) Inverse Circuit implementation	798gates	147 gates

b) Reed-Solomon encoder design without multiplier

	RS Encoder by GF(2^8) operation	RS Encoder with GF(2^4) operation
ROM size forGF(2^8) mulriplier replacement	16X 256 wordsX 8 bits/word	256 wordsX 8 bits/word
GF(2^8) Inverse Circuit implementation	Same as a) case (798)	Same as a) case(147)

```
Library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_arith.all;
use IEEE.std_logic_unsigned.all;
Entity Eight_4 is
        Port (
        b0,b1,b2,b3,b4,b5,b6,b7: in std_logic;
        z0,z1,z2,z3,z4,z5,z6,z7 : out std_logic ;
  bz0,bz1,bz2,bz3,bz4,bz5,bz6,bz7 : out std_logic    );
   END Eight_4;-- Entity Ends
   Architecture Behave of Eight_4 is
      Begin
        z0 <= b0 xor b1 xor b5 ;
        z1 <= b1 xor b3 xor b5 ;
        z2 <= b2 xor b3 xor b6 ;
        z4 <= b1 xor b3 xor b5 xor b2 xor b6 xor b7;
        z5 <= b5 xor b2 xor b6 ;
        z6 <= b1 xor b3 xor b5 xor b2 xor b6 xor b4;
        z3 <= b1 xor b3 xor b4 xor b6;
        z7 <= b1 xor b3 xor b5 xor b4;
       bz0 <= b0 xor b1 xor b2 xor b6 xor b7 ;
  bz1 <= b1 xor b2 xor b5;
  bz2 <= b5 xor b3 xor b7;
  bz3 <=  b2 xor b6 xor b7;
  bz4 <= b1 xor b7  ;
  bz5 <=  b5 xor  b6 xor b7;
  bz6 <=  b3 xor b5 xor b6;
  bz7 <= b1 xor b6 xor b7 xor b4;
  End Behave;-- Architecture Ends
```

**bz's are GF(2^4) to GF(2^8) outs, and z's are GF(2^8) to GF(2^4) outs.

References

〔1〕 J. Jiang and K. Narayanan, "Iterative soft decision decoding of Reed Solomon codes based on adaptive parity check matrices," in Proc. ISIT, 2004.

〔2〕 안형근, "리드솔로몬 복호기의 에러값을 구하기위한 새로운 고속의 산술논리연산장치의 설계에 대해", 대한전자공학회논문지(통신TC편), 제46권 집4호, pp.45-51, 2009. April.

〔3〕 Joschi Brauchle, RalfKoetter; A Systematic Reed Solomon Encoder with Arbitrary Parity Positions, IEEE GLOBECOM 2009 Proceedings.

〔4〕 T.K. Moon, Error Correction Coding: Mathematical Methods and Algorithms, Hoboken, NJ: John Wiley & Sons, Inc., 2005.

〔5〕 안형근, "디지탈 오디오/비디오, 통신용 전자기기를 위한 Reed-Solomon Codec설계에 대해", pp.13-18, TC편 제 42호, 2005년 11월호, 대한전자공학회지

〔6〕 Hanho Lee; High-speed VLSI architecture for parallel Reed-Solomon decoder, IEEE transactions on VLSI Systems 2003.

〔7〕 Hyeong-Keon An, "2 Error Correcting RS Decoder Design," IDEC Conference Paper, 2004.

〔8〕 Shu Lin, Daniel J. Costello, Jr,. "Error Control Coding," Prentice-Hall, PP.240-261 (2004)

안 형 근 (Hyeong-Keon An) 정회원

1979년 서울대학교 전기공학과 학사

1981년 KAIST 전기 및 전자공학과 석사

1988년 뉴욕주립대학교 전기공학과 박사 졸업.

1988~1998 삼성전자 수석연구원

1998~1999 텔슨전자 이사

2000~현재 동명대학교 정보통신공학과 교수

<관심분야> 통신, 신호처리, 반도체, OLED/LED display 조명장치