2015 Conference on Power, Control, Communication and Computational Technologies for Sustainable Growth (PCCCTSG)
December11-12, 2015, Kurnool, Andhra Pradesh, India

FPGA based design and implementation of Reed-Solomon encoder & decoder for Error Detection and Correction

P.Parvathi
Assistant Professor,
Dept. of Electronics & Communication Engg.,
G. Pulla Reddy Engineering (Autonomous),
Kurnool, India
padma193@gmail.com

P.Rajendra Prasad,
Assistant Professor,
Dept. of Electronics & Communication Engg,
Sri Venkateswara Engineering College,
Banglore, India
rajisvec@gmail.com

***Abstract*— This paper addresses to design an R-S(Reed-Solomon) encoder and R-S decoder and verify its functionality on FPGA. An RTL (register transfer logic) code was developed for an R-S encoder that takes a 32-bit input data and produces a 48-bit output. The logic synthesis and optimization of the code was carried out using Quartus II tool. Also, an R-S decoder was developed that decodes the transmitted data.**

***Keywords*—Reed-Solomon (R.S) Encoder & Decoder, Error Detection and Correlation (EDAC), FPGA**

I. Introduction

Error Checking and Correction method can be done by identifying the error first and correcting it. Majority of errors occur due to hardware failure, disturbance or noise in transmission of data. Hamming method, RRS method, cyclic method, linear method, convolution method etc. are some of the error correction codes. Among these most efficient are Hamming codes & RS codes. Hamming codes are generally used to correct the errors present in main memory, whereas RS codes perform error correction both in main memory and peripheral devices like tapes and disk storage. Reed-Solomon (R-S) codes are made up of m-bit sequences with m>2. They are represented as R-S(n,k) for a particular encoded block with k no. of data symbols that are encoded and n indicates total no of code symbols. In general, R-S(n, k)=(2^m-1, 2^m-1-2t),with n-k=2t is the number of parity symbols. Thorough study has been made on the Reed-Solomon codes which include both encoder and decoder part. Some of the basic concepts has been learnt before going through the RS codes, they are: Finite fields, Primitive element, primitive polynomial, Galois fields, Properties of Finite fields, LFSR (linear feedback shift register) etc. S.Lin and D.J.Costello author of "Fundamentals of error control coding," explains the basics of coding, types of errors and different types of codes [1].

Bernard Sklar has proposed a tutorial paper, which explains about the basics of Reed-Solomon codes, Finite fields and its properties [2].I.S.Reed and G.Solomon has presented a paper on "Polynomial codes over certain finite fields", where he explains about the Galois fields, primitive polynomial and elements of finite fields [3]. Examples related to Reed-Solomon codes are given in [2, 9].H.Lee provided a paper intended to be principally a tutorial introduction to RS algorithm, its structure, and its analysis [4, 11].Todd K.Moon author of "Error correcting coding: mathematical methods and algorithm", explains about the various algorithm and its applications. Based on the applications, the algorithm has to be chosen and detail explanation of the algorithm with mathematical terms and equation [6]. T.R.Rao author of "Error-control coding for computer systems", explains about the type of errors that occur in the computer system, causes of error and the method of controlling these errors [15]. D.V.Sarwate explains how the performance in R-S decoder can be improved [10,17]. An RTL code was developed based on the specification. The RTL code was simulated using Quartus II, to check the correctness of the developed code. The logic synthesis and optimization of the code was carried out using Quartus II tool. The synthesized and optimized design and the RTL were verified for functionality.

II. Error Detection and Coorelation System

Due to the noise in transmission of data, 0 bit may change to 1 & vice versa. Hence data may be corrupted. Error detection and correlation should ensure a reliable delivery of data. These techniques should detect the errors and reconstruction with error-free data. An EDAC code protects the memory contents by using redundant bits. So, the redundant bits can be added to original data using error-correcting coding techniques. If any changes occur in one or more bits then Data should be corrected. This is done first locating the address of error position and applying different algorithms to make the data error free, which can be explained in Section V.

III. Reed-Solomon Encoding

The key to the RS encoding is to view symbols of the message that is to be encoded as if they are the coefficients of a polynomial. As mentioned earlier, when the R-S encoder receives an information sequence, it creates encoded blocks consisting of $n=2^m-1$ symbols, each, where m is the symbol size in bits The encoder divides the information bit sequence into message blocks of $k=n-2t$ symbols. Each message block is equivalent to a message polynomial of degree k-1, denoted as

$$m(x)=m_0+m_1x+m_2x^2+m_3x^3+\dots\dots\dots m_{k-1}x^{k-1}$$

where the coefficient m_0, m_1, m_2, ...m_{k-1} of the polynomial m(x) are the symbols of a message block. Moreover, these coefficient are elements of $GF(2^m)$. So the information sequence is mapped into an abstract polynomial by setting the coefficient equal to the symbol value.

R-S(15,11) is considered ,which is shortened to RS(12,8) as per the given specification. The result of it is shown in section VI.

R-S encoder uses GF arithmetic operation to add parity symbols. The Galois field is explained in the previous chapter. The explanation for generating parity symbols and the codeword polynomial (encoded output) are explained below. List of formulae's that is to be known in R-S(n, k) codes are:

1. Generator polynomial:

$$g(x)=g_0+g_1x^1+g_2x^2+\dots\dots\dots+g_{2t-1}x^{2t-1}+x^{2t}$$

Where g_0 ,g_1 ,.......g_{2t-1} are the coefficient from $GF(2^m)$.

2. Parity polynomial (redundant polynomial):
$p(x) = (x^{2t} *m(x)) \bmod g(x)$, where g(x) is the generator polynomial and m(x) is the message polynomial.

3. Codeword polynomial: $c(x) = x^{2t}m(x) + p(x)$.

The encoding of R-S codes is performed in systematic form. The codeword is formed by simply appending parity (or redundant) symbols to the end of the k-symbols. In particular, codeword k-symbols message block consists of k consecutive coefficients of a message polynomial, and 2t parity symbols are the coefficient (from $GF(2^m)$) of a redundant polynomial. Applying the polynomial notation, it shifts the information into the leftmost bits by multiplying by x^{2t}, leaving a codeword of the form

$$c(x) = x^{2t}m(x) + p(x) \qquad (1)$$

where c(x) is the codeword polynomial, m(x) is the message polynomial and p(x) is the redundant polynomial.

INPUT QUEUE	***CLOCK CYCLE***	***REGISTER CONTENTS***	***FEEDBACK***
$\alpha^1\ \alpha^{10}\ \alpha^{11}\ \alpha^{12}\ \alpha^1\ \alpha^2\ \alpha^3\ \alpha^4$	0	0 0 0 0	α^4
$\alpha^1\ \alpha^{10}\ \alpha^{11}\ \alpha^{12}\ \alpha^1\ \alpha^2\ \alpha^3$	1	$\alpha^{14}\ \alpha^7\ \alpha^{10}\ \alpha^2$	α^6
$\alpha^1\ \alpha^{10}\ \alpha^{11}\ \alpha^{12}\ \alpha^1\ \alpha^2$	2	$\alpha\ \alpha^4\ \alpha^2\ \ \alpha^2$	0
$\alpha^1\ \alpha^{10}\ \alpha^{11}\ \alpha^{12}\ \alpha^1$	3	$0\ \alpha^1\ \alpha^4\ \alpha^2$	α^5
$\alpha^1\ \alpha^{10}\ \alpha^{11}\ \alpha^{12}$	4	$\alpha^0\ \alpha^8\ \alpha^6\ \alpha^7$	α^2
$\alpha^1\ \alpha^{10}\ \alpha^{11}$	5	$\alpha^{12}\ \alpha^{10}\ 0\alpha^{13}$	α^4
$\alpha^1\ \alpha^{10}$	6	$\alpha^{14}\ \alpha^2\ 0\ \alpha^2$	α^4
α^1	7	$\alpha^{14}\ \alpha^1\ \alpha^4\ \alpha^2$	α^5
-	8	$\alpha^0\ \alpha^6\ \alpha^6\ \alpha^7$	-

The redundant polynomial p(x) is obtained as
$p(x)=(x2t*m(x))\bmod g(x) \qquad (2)$

So, as R-S codeword is generated using a generator polynomial, which has such property that all valid codeword are exactly divisible by the generator polynomial.

$$g(x)=g_0+g_1x^1+g_2x^2+\dots\dots\dots+g_{2t-1}x^{2t-1}+x^{2t} \qquad (3)$$

where α is a primitive element in $GF(2^m)$ and g_0, g_1, g_{2t-1} are the coefficient from $GF(2^m)$, the degree of the generator polynomial is equal to the number of parity symbols, as per the given specification

For R-S(12,8) of m=4, here n=12, and k=8 of 4–bit each (as m=4 as per given specification).and t=2.The generator polynomial for the above specification will be as $g(x) = \alpha^{10} + \alpha^3x+ \alpha^6x^2+ \alpha^{13}x^3+x^4$, which is obtained from equation 3. This generator polynomial is fixed for the above codes specification. The redundant polynomial p(x) will vary as the message polynomial m(x) varies.

For encoding the R-S (12, 8) has generated a D-flip flop as a register, Mux as a switch and a Galois addition and multiplication tables. The fig 1 shown below is the linear feedback shift register, which is used for generation the parity symbols. The encoder is a 2t tap shift register, where each register is 4-bits wide. The multiplier coefficients are fixed for R-S(12,8). The coefficients are obtained from the generator polynomial which is shown above. The Linear feedback shift register for R-S(12,8) is explained in Fig1. The parity symbols are sitting in the register, and it just remains to shift them out one by one

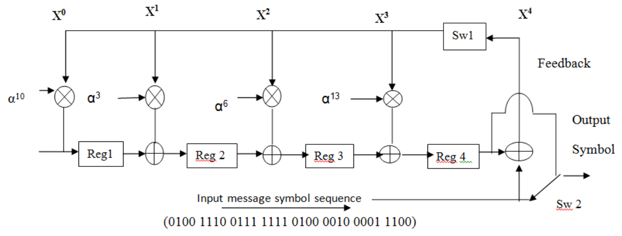


Fig: 1 Linear feedback shift register for a R-S (12,8) code.

For RS(12, 8), if consider
$m(x)=\alpha^1+\alpha^{10}x+\alpha^{11}x^2+\alpha^{12}x^3+\alpha^1x^4+\alpha^2x^5+\alpha^3x^6+\alpha^4x^7$
Where the values of α^1 =0100, α^{10}=1110, α^{11}=0111, α^{12}=1111, α^2=0010, α^3=0001, α^4=1100,.

After the eight clock cycles, the register contents are the four parity symbols α^0 α^6 α^6 α^7 as shown. The output codeword c(x) in polynomial form is given by:
$C(x)= \alpha^0+ \alpha^6x^1+ \alpha^6x^2+ \alpha^7x^3+ \alpha^1x^4+ \alpha^{10}x^5+ \alpha^{11}x^6+ \alpha^{12}x^7+ \alpha^1x^8+ \alpha^2x^9+ \alpha^3x^{10}+ \alpha^4x^{11}$ in binary form the encoded output
1000 0011 0011 1101 0100 1110 0111 1111 0100 0010 0001 1100.Hence RS encoder produces an output of 48-bit, for an input of 32-bit.

IV. Reed-Solomon Decoding

An R-S decoder was developed that decodes the transmitted data. During, syndrome computation, Decoder transfers symbols to data syndrome and it identifies the errors if detected. Based on syndromes in the finite GF(Galois field), algorithm computes error polynomial. The below figure 2 shows the block diagram of decoder block for R-S codes. The following tasks that the decoder has to perform are:

1) Compute the syndrome
2) Find the coefficient of the error-location polynomial $\sigma(x)$
3) Find the inverses of the roots of $\sigma(x)$ that is location of the errors
4) Find the values of the errors
5) Correct the received codeword with the error location and values found

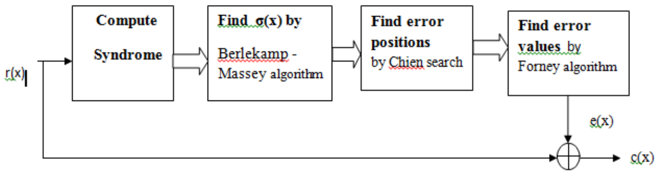

Compute Syndrome
Find σ(x) by Berlekamp - Massey algorithm
Find error positions by Chien search
Find error values by Forney algorithm
r(x)
e(x)
c(x)

Fig. 2. Architecture of a Reed-Solomon decoder

The syndrome accumulate is the first step in the R-S decoding process. This is done to detect if there are any errors in the received codeword.

After encoding a given message, the codeword polynomial

$$c(x)=c_0+c_1x+c_2x^2+\ldots\ldots\ldots+c_{n-1}x^{n-1} \quad (4)$$

then the received polynomial r(x) is expressed as

$$r(x)=r_0+r_1x+r_2x^2+\ldots\ldots+r_{n-1}x^{n-1} \quad (5)$$

Which is related to the error polynomial e(x) and the codeword polynomial c(x) as

$$r(x)=c(x)+e(x) \quad (6)$$

Where the error pattern e(x) is expressed as

$$e(x)=e_0+e_1x+e_2x^2+\ldots\ldots+e_{n-1}x^{n-1} \quad (7)$$

Any errors will result in one or more computation yielding a non-zero result. So the computation of a syndrome symbol can be described as

$$S_i=r(\alpha^i),\ i=1,2,3\ldots 2t \quad (8)$$

where α, α^2, α^3,………………. α^{2t} are the roots of g(x). For r(x) to be a valid codeword, each syndrome symbol S_i should be equal to zero, if one or more syndromes are non-zero, errors have been detected.

Let us assume that the error polynomial e(x) contains $v\leq t$ non–zero elements, which means that during transmission v errors occurred, placed at positions x^{j1}, x^{j2}, ……………….x^{jv} , where $0\leq j_1\leq j_2\ldots\ldots\ldots j_v\leq(n-1)$. Then

$$e(x)= e_{j1}x^{j1}+e_{j2}x^{j2}+\ldots\ldots+e_{jv}x^{jv} \quad (9)$$

The indices 1, 2,…v refer to the first, second, v^{th} errors, and the index j refers to the error location. Hence to correct the corrupted codeword need to determine e(x) or in other words, and to determine the error location x^{j1} and error values e_{j1}.

Let us now define the error-location number as

$$\beta_l= \alpha^{jl} \text{ , where } l=1, 2, 3, \ldots\ldots\ldots v$$

Next, the 2t syndrome symbols are obtained by substituting α^i into the received polynomial for i= 1, 2,3, ………………………………2t:

$$S_1=r(\alpha)=e_{j1}\beta_1+ e_{j2}\beta_2+ e_{j3}\beta_3+\ldots\ldots\ldots\ldots e_{jv}\beta_v$$
$$S_2=r(\alpha^2)=e_{j1}\beta^2_1+ e_{j2}\beta^2_2+ e_{j3}\beta^2_3+\ldots\ldots\ldots\ldots e_{jv}\beta^2_v$$
$$.$$
$$.$$
$$S_{2t}=r(\alpha^{2t})=e_{j1}\beta^{2t}_1+ e_{j2}\beta^{2t}_2+ e_{j3}\beta^{2t}_3+\ldots\ldots\ldots\ldots e_{jv}\beta^{2t}_v \quad (10)$$

From the above equations there are 2t unknowns (t error values and t locations), and 2t simultaneous equations. As some of the unknowns have exponents, these 2t simultaneous equations are non-linear. Solving this system of equations (10) constitutes the most computationally intensive operation in decoding R-S codes. Berlekamp-Massey algorithm for R-S decoding process. The elements involved in computation of (10) belong to a Galois field, and so the operations of addition and multiplication are also done over GF(2^m).

V. Design and Implimentation

The overall integrated module for RS encoder and decoder with 1MB of memory is shown in the below fig 3. Fig 4 shows a simple flowchart of the proposed RS encoder & decoder. An RTL (register transfer logic) code was developed for an R-S encoder that takes a 32-bit input data and produces a 48-bit output. Also, an R-S decoder was developed that decodes the transmitted data. Given data: R-S (12, 8) where m=4 and t (symbol-error correcting capability of the code) =2.

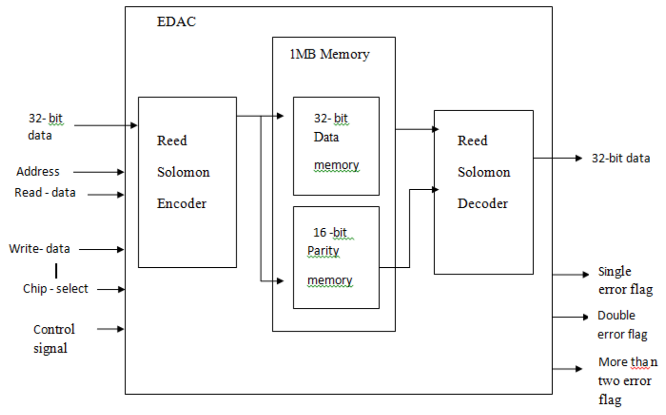


Fig. 3. Overall integrated module of EDAC.

1 Reed-Solomon encoder part:

For a given message $m(x) = \alpha+ \alpha^2x+ \alpha^3x^2+ \alpha^4x^3+ \alpha^5x^4+ \alpha^6x^5+ \alpha^7x^6+ \alpha^8x^7$, the generator polynomial, redundant polynomial and the codeword polynomial is obtained as follows. The decoder part is also explained by considering the same message polynomial. From equation 3 the generator polynomial: $g(x) = \alpha^{10}+ \alpha^3x+ \alpha^6x^2+ \alpha^{13}x+ \alpha^4$.

From equation 2 the redundant polynomial $p(x) = 0x^3+ \alpha^{11}x^2+ \alpha^4x+ \alpha^{14}$ and from equation 1 the codeword polynomial:

$c(x) = \alpha^{14}+ \alpha^4x+ \alpha^{11}x^2+ 0x^3+ \alpha^1x^4+ \alpha^2x^5+ \alpha^3x^6+ \alpha^4x^7+ \alpha^5x^8+ \alpha^6x^9+ \alpha^7x^{10}+ \alpha^8x^{11}$

2. Reed-Solomon decoder part:

Let the double symbol error be such that

$e(x) = \alpha^{12}+0x+ 0x^2+ 0x^3+ 0x^4+ 0x^5+ 0x^6+ 0x^7+ 0x^8+ 0x^9+ 0x^{10}+ \alpha^9x^{11}$

then from equation 6;

$r(x)= \alpha^5+ \alpha^4x+ \alpha^{11}x^2+ 0x^3+ \alpha^1x^4+ \alpha^2x^5+ \alpha^3x^6+ \alpha^4x^7+ \alpha^5x^8+ \alpha^6x^9+ \alpha^7x^{10}+ \alpha^{12}x^{11}$

From equation 8 the syndrome symbols are:

$S_1(\alpha)= \alpha^{14}$, $S_2(\alpha^2)= \alpha^{13}$, $S_3(\alpha^3)= 0$, $S_4(\alpha^4)= \alpha^9$

From the equations of Berlekamp-Massey algorithm the coefficient of the error-location polynomial $\sigma(x) = 1+ \alpha^{12}x+ \alpha^{11}x^2$ the error position are $\beta_1= \alpha^0$ and $\beta_2= \alpha^{11}$, finally the error values are calculated from equations of Forney algorithm and the error values are $e_1= \alpha^{12}$ and $e_2= \alpha^9$. So having found the error locations and error values finally can form the error polynomial e(x) and correct the received polynomial r(x) just by adding (with XOR operation) these two polynomials together, as shown in figure 2. Therefore the corrected codeword c(x) is as follows,

$c(x) = \alpha^{14}+ \alpha^4x+ \alpha^{11}x^2+ 0x^3+ \alpha^1x^4+ \alpha^2x^5+ \alpha^3x^6+ \alpha^4x^7+ \alpha^5x^8+ \alpha^6x^9+ \alpha^7x^{10}+ \alpha^8x^{11}$, since the message symbols constitute the rightmost k=8 symbols, the decoded message is $\alpha^1\alpha^2\alpha^3\ \alpha^4\ \alpha^5\ \alpha^6\ \alpha^7\ \alpha^8$, which is exactly the text message.

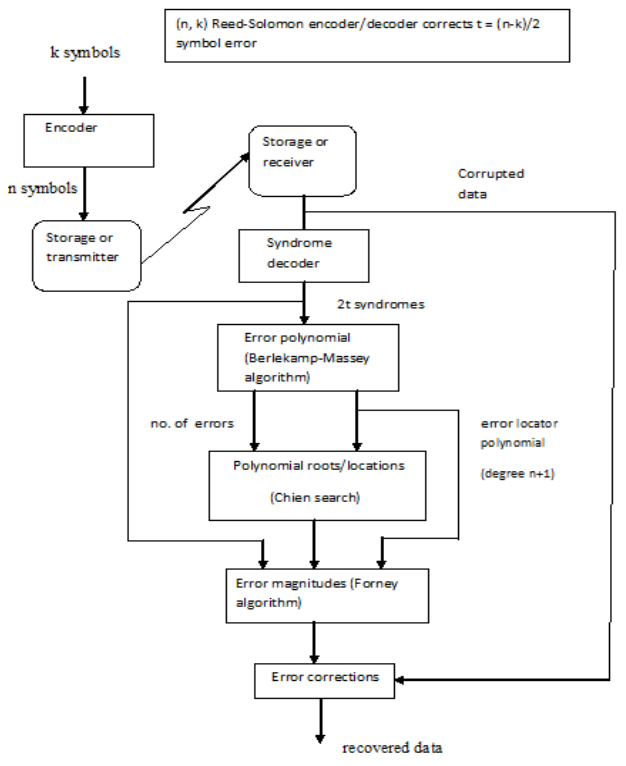

(n, k) Reed-Solomon encoder/decoder corrects t = (n-k)/2 symbol error
k symbols
Encoder
n symbols
Storage or transmitter
Storage or receiver
Corrupted data
Syndrome decoder
2t syndromes
Error polynomial (Berlekamp-Massey algorithm)
no. of errors
error locator polynomial
(degree n+1)
Polynomial roots/locations (Chien search)
Error magnitudes (Forney algorithm)
Error corrections
recovered data

Fig. 4. Data flow diagram of Reed-solomon encoder & decoder

VI. Results and Discussions

This section consists of all the possible snapshots taken during the entire process of implementing Reed-Solomon encoder and decoder. Fig 5 shows the project workspace with the Verilog code for Reed-Solomon encoder and decoder. After the successful completion of the analysis and synthesis, the code is checked for its syntax. Fig 6 shows all the signals involved in the code and its simulated waveform.

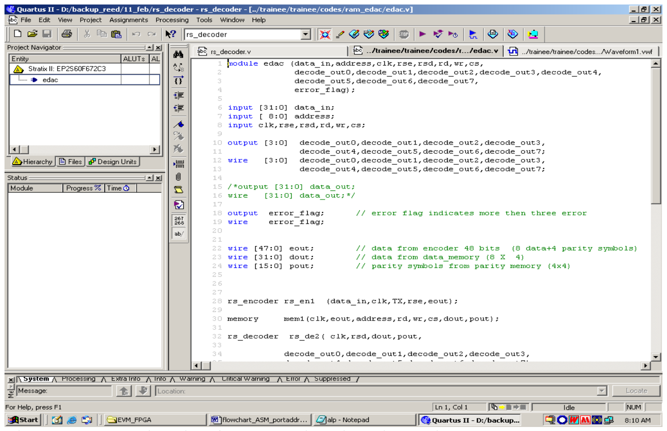
Ex if input is 1010 1101 0011 0110 1100 0001 0010 0100 then Encoder output: 1001 1100 0111 0000 0100 0010 0001 1100 0110 0011 1101 1010

Thus an RTL (register transfer logic) code was developed for an R-S encoder that takes a 32-bit input data and produces a 48-bit output. Also, an R-S decoder was developed that decodes the transmitted data. Fig15 shows the output of decoder for

Ex if input is 1001 1100 0111 0000 0100 0010 0001 1100 0110 0011 1101 1010 then the decoded output:

1010 1101 0011 0110 1100 0001 0010 0100

Berlekamp-Massey algorithm is used to find the coefficient of error polynomial Chien-search algorithm is used to find the error position. The output of it is shown in fig 8 Forney algorithm is used to find the error values. The output of it is shown in fig 8. The output of the decoder block is shown in fig 12. Fig 16 shows the 48 bit output. The below fig 13 and 14 shows the decoder block with error flag indication and with zero error flag.


Quartus II - D:/backup_reed/11_feb/rs_decoder - rs_decoder - [../trainee/trainee/codes/ram_edac/edac.v]
File Edit View Project Assignments Processing Tools Window Help
rs_decoder
Project Navigator
Entity
Stratix II: EP2S60F672C3
edac
Hierarchy
Files
Design Units
Status
Module
Progress %
Time
rs_decoder.v
../trainee/trainee/codes/r.../edac.v
../trainee/trainee/codes.../Waveform1.vwf
System
Processing
Extra Info
Info
Warning
Critical Warning
Error
Suppressed
Message:
Location:
Locate
For Help, press F1
Ln 1, Col 1
Idle
NUM
Start
EVM_FPGA
flowchart_ASM_portaddr...
alp - Notepad
Quartus II - D:/backup...
8:10 AM

```
module edac (data_in,address,clk,rse,rsd,rd,wr,cs,
             decode_out0,decode_out1,decode_out2,decode_out3,decode_out4,
             decode_out5,decode_out6,decode_out7,
             error_flag);

input [31:0] data_in;
input [ 8:0] address;
input clk,rse,rsd,rd,wr,cs;

output [3:0]  decode_out0,decode_out1,decode_out2,decode_out3,
              decode_out4,decode_out5,decode_out6,decode_out7;
wire   [3:0]  decode_out0,decode_out1,decode_out2,decode_out3,
              decode_out4,decode_out5,decode_out6,decode_out7;

/*output [31:0] data_out;
wire   [31:0] data_out;*/

output  error_flag;       // error flag indicates more then three error
wire    error_flag;


wire [47:0] eout;         // data from encoder 48 bits  (8 data+4 parity symbols)
wire [31:0] dout;         // data from data_memory (8 X  4)
wire [15:0] pout;         // parity symbols from parity memory (4x4)



rs_encoder rs_en1  (data_in,clk,TX,rse,eout);

memory     mem1(clk,eout,address,rd,wr,cs,dout,pout);

rs_decoder  rs_de2( clk,rsd,dout,pout,

           decode_out0,decode_out1,decode_out2,decode_out3,
```

Fig. 5. Integrated code of RS encoder, decoder and memory.

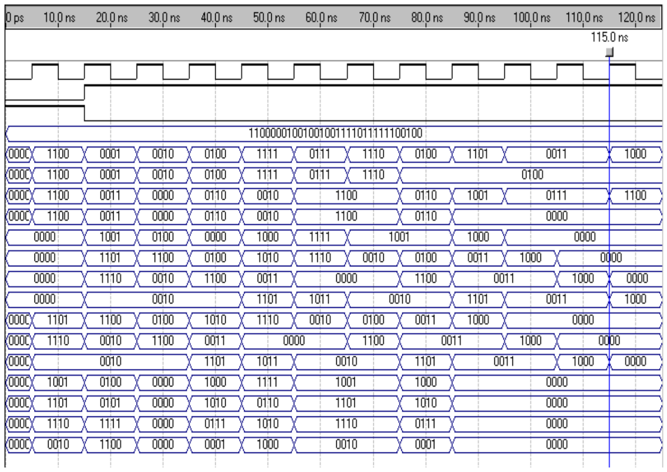
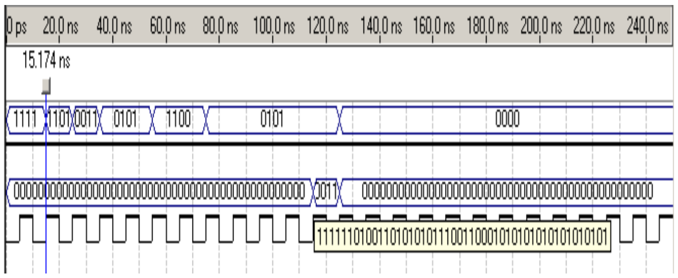


Fig. 6. Results of a linear feedback shift register of an encoder of RS (12,8)


0 ps 20.0 ns 40.0 ns 60.0 ns 80.0 ns 100.0 ns 120.0 ns 140.0 ns 160.0 ns 180.0 ns 200.0 ns 220.0 ns 240.0 ns
15.174 ns
1111 1101 0011 0101 1100 0101 0000
11111101001101010101110011000101010101010101
Fig. 7. Results of an encoder

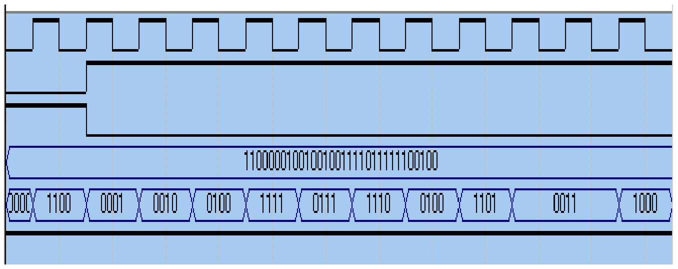
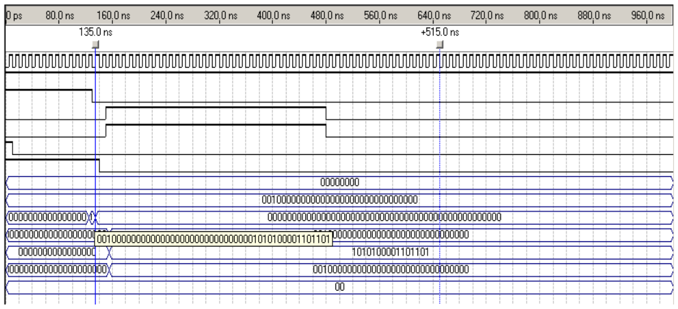


Fig. 8. Result of an encoder in serial form, for which it acts as input to the memory.


0 ps 80.0 ns 160.0 ns 240.0 ns 320.0 ns 400.0 ns 480.0 ns 560.0 ns 640.0 ns 720.0 ns 800.0 ns 880.0 ns 960.0 ns
135.0 ns
+515.0 ns
00000000
1010100001101101
00
Fig. 9. Simulation results of a memory

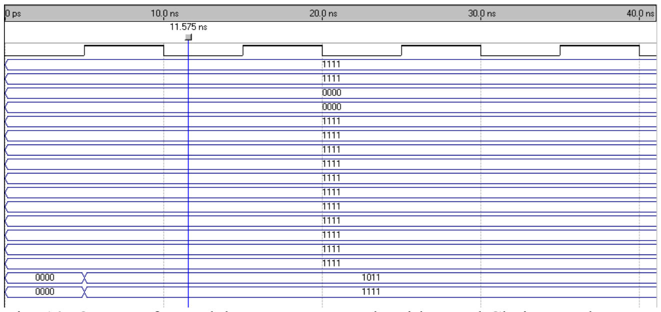
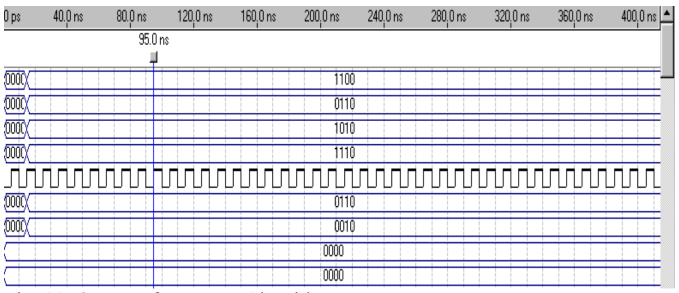
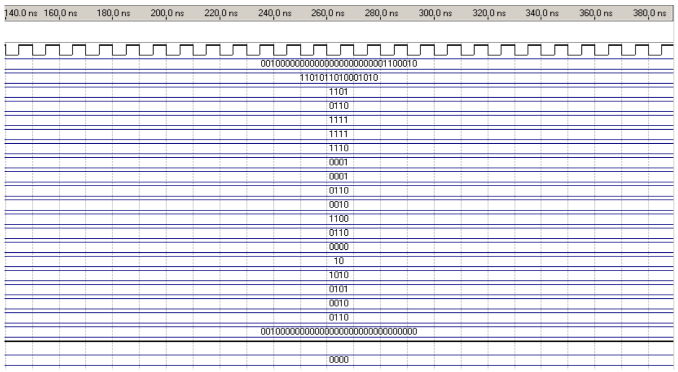
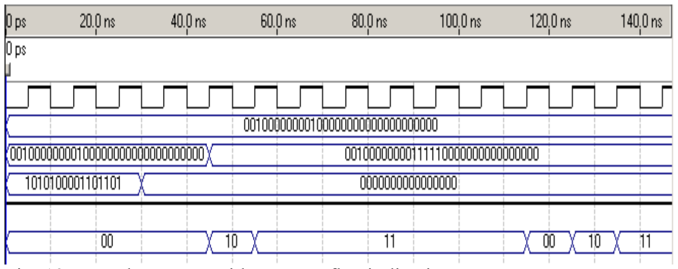
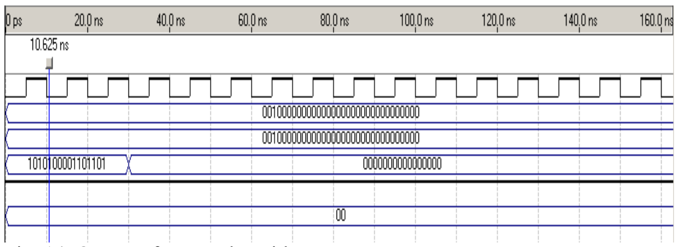


Fig. 10. Output of a Berlekamp`s Massey algorithm and Chein search


0 ps 40.0 ns 80.0 ns 120.0 ns 160.0 ns 200.0 ns 240.0 ns 280.0 ns 320.0 ns 360.0 ns 400.0 ns
95.0 ns
1100
0110
1010
1110
0110
0010
0000
0000
Fig. 11. Output of a Forney Algorithms


140.0 ns 160.0 ns 180.0 ns 200.0 ns 220.0 ns 240.0 ns 260.0 ns 280.0 ns 300.0 ns 320.0 ns 340.0 ns 360.0 ns 380.0 ns
1101011010001010
1101
0110
1111
1110
0001
0010
1100
0000
10
1010
0101
0000
Fig. 12. Total Integrated output of a decoder


0 ps 20.0 ns 40.0 ns 60.0 ns 80.0 ns 100.0 ns 120.0 ns 140.0 ns
0 ps
1010100001101101
0000000000000000
00 10 11 00 10 11
Fig. 13. Decoder output with an error flag indication


0 ps 20.0 ns 40.0 ns 60.0 ns 80.0 ns 100.0 ns 120.0 ns 140.0 ns 160.0 ns
10.625 ns
1010100001101101
0000000000000000
00
Fig. 14. Output of a Decoder with zero error

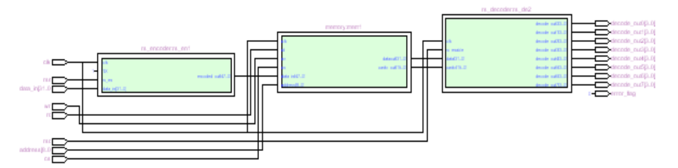


Fig. 15. RTL view of an integrated R-S encoder, memory and a decoder

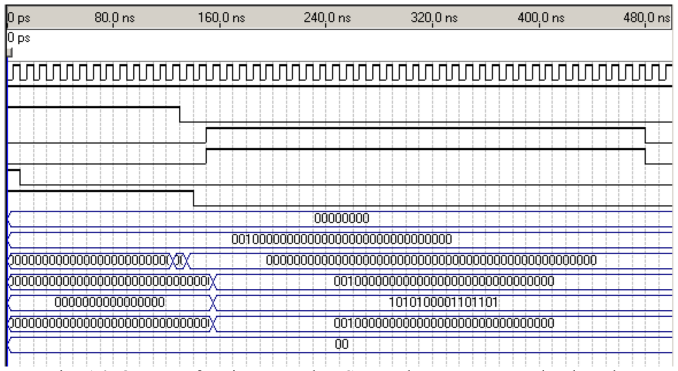


Fig. 16. Output of an integrated R-S encoder, memory and a decoder

Summary of the RS (12, 8) codes:

- Hardware Language : Verilog
- Tool used : Quartus II (Altera)

1. Simulation time : 1:05:45
2. Compilation report:
   - Total ALUs…………………………12,484
   - Total registers………………………12,484
   - Total pins……………………………240
   - Memory bits…………………………1,536
3. Analysis and Synthesis time : 00:05:41
4. Resource usage:
   - Total ALUs……………………...…..12,484
   - Total combinational function…………5674
5. ALUT usage by number of input
   - 7 input function…………………….……36
   - 6 input function………………………4455
   - 5 input function……………………..……301
   - 4 input function………………………...540
   - ≤3 input function ……………………....342
6. Total fanout ………………………………..…81,454
7. Hardware Platform: Stratix II FPGA (EP2S60F672C3)

VII. CONCLUSIONS

The overall design of an R-S encoder and R-S decoder and its functionality on FPGA is completed and its results are shown. An RTL (register transfer logic) code was developed for an R-S encoder that takes a 32-bit input data and produces a 48-bit output. Also, an R-S decoder was developed that decodes the transmitted data. The same circuit in future can be implemented as an application specific integrated circuit (ASIC). The R-S codes with higher number of data symbols can be implemented in the future and can be used in the field of communication applications.

ACCONOLEDGEMENT

The Authors have worked for this project in Bharath Electronics Ltd, Bangalore.

REFERENCES

[1] S.Lin and D.J.Costello, "Error control coding: Fundamentals and Applications", Prentice Hall: Englewood Cliffs, NJ, 1983

[2] Bernard Sklar, Reed-Solomon codes, April 2002, available from: http://www.phptr/com/articles/article.asp?p=26335&r1=1.

[3] Reed I.S. and Solomon.G, "Polynomial Codes Over Certain Finite Fields", SIAM Journal of Applied Math., Vol.8, 1960, pp. 300-304.

[4] H.Lee, "High Speed VLSI Architecture for parallel Reed-Solomon ecoder", IEEE Trans.Very Large Scale Integration (VLSI) syst.vol.11, no.2, pp. 288-294, April 2003

[5] W.Wilhelm, "A New scalable VLSI Architecture for Reed Solomon decoder", IEEE J.Solid state circuits, vol.34, no.3, Mar 1999, pp. 388-396

[6] Tood K.Moon, "Error correction coding: Mathematical methods and algorithm", John Wiley and sons Ltd 2005.

[7] S.Keneda and E.Fujiwara, "Single Byte error correcting, Double byte error detecting codes for memory system", IEEE Trans. vol-31, July 1982, pp.596-602

[8] Jorge C. Moreira, P. Guy Farrell, Essentials of Error-Control Coding", John Wiley&Sons, Ltd., Sep 2006.

[9] Wicker, "Error control system for Digital communication and storage", Prentice hall 1995

[10] Bernard Sklar, "Digital Communication: Fundamentals and Application", 2rd Edition (Prentice hall, 2001)

[11] D.V.Sarwate and N.R.Shanbhag, "High Speed architecture for Reed-Solomon decoders", IEEE Trans.Very Large Scale (VLSI) Integr.Syst. vol.9, no.5, Oct. 2001, pp 641-655

[12] Sarwate, D.V., and Shanbhag, N.R.: 'High-speed architectures for Reed-Solomon decoders', IEEE Trans. Very Large Scale Integr. (VLSI) Syst., 2001, 9, (5), pp. 641-655

[13] H.Lee, "An area-efficient Euclidean algorithm block for Reed-Solomon decoder", in IEEE Computer Society Annu. Symp. VLSI, Feb. 2003, pp 209-210

[14] T.K.Truong, J.H.Jeng , T.C.Cheng, "A New Decoding Algorithm for Correcting Both Erasures and Errors of Reed-Solomon codes", IEEE Transactions on communications, March 2003, pp.381-388.

[15] Stratix II device Handbook (All Three Volumes) (ver. 6.1, Jun 2006) and BEL documents.

[16] **"**New Scalable Decoder Architectures for Reed–Solomon Codes", Yingquan Wu, IEEE Transactions on Communications, Year: 2015, Volume: 63, Issue: 8 Pages: 2741 - 2761

[17] Yi Hua Chen; Chang Lueng Chu; Chun Chun Yeh "FPGA implementation and verification of Reed-Solomon (31, 15, 8) code in SDR system", Computer Science and Network Technology (ICCSNT), 2012 2nd International Conference on, On page(s): 465 - 468

[18] R.Koetter and A.Vardy, "Algebraic soft-decision decoding of Reed-Solomon codes", IEEE Trans.Information Theory, Vol. 49, Nov.2003, pp.2809-2825

[19] T.R.N.Rao and E.Fujiwara, "Error-control coding for computer systems", Prentice Hall, 1989.