International Journal of Soft Computing and Engineering (IJSCE)
ISSN: 2231-2307, Volume-2, Issue-6, January 2013

Design of RS (255, 251) Encoder and Decoder in FPGA

Anindya Sundar Das, Satyajit Das and Jaydeb Bhaumik

Abstract*— Detection and correction of errors in digital data is an important issue for the modern communication systems. Therefore an efficient error control code is needed to protect the digital data. In high speed communication system Reed-Solomon codes are widely used to provide error protection especially against the burst errors. Reed-Solomon codes are cyclic, non-binary codes. In this paper RS(255, 251)encoder and decoder have been designed and implemented on an FPGA platform.

Index Terms*—Reed-Solomon codes, Galois field, RS encoder, RS decoder.

I. INTRODUCTION

Error correcting codes have a wide range of applications in different fields like digital data communications, memory system design, and fault tolerant computer design among others. Reed-Solomon (RS) code is a well known non-binary linear block code. It is popularly used for error correction in many applications like storage devices (CD, DVD), wireless communications, high speed modems and satellite communications. For example RS (28, 24) and RS (32, 28) codes with interleaving are popularly used for storage in CD.

A 16 bytes error correcting RS (255, 223) code has been used for digital μ-wave radio. The code RS (255, 239) having 8 bytes error correcting capability has been recommended as a outer code in WiMax. To preserve important header information, MB-OFDM UWB adopts RS (23, 17) codes.

Basics of RS codes and a number of decoding techniques may be found in the literature [1] [2]. Several important applications of RS codes have been discussed in [3]. Parallel architecture for high-speed RS(255, 251) codec has been proposed in [4]. A high-speed architecture for Reed-Solomon decoders is proposed in [5]. The complexity of RS encoder and decoder increases with the error correcting capability of the codes. Hence, many researchers have directed their efforts to minimize the complexity of RS decoder. In VLSI system design, power consumption is of prime concern and at the same time, the silicon area should be kept as low as possible. In this scenario, our aim is to design of a FPGA based RS (255, 251) encoder and decoder having two bytes error correcting capability.

In next section, a brief introduction of Reed-Solomon code is given. In section III, RS encoder and its corresponding architecture has been described. Basic blocks of a RS decoder and corresponding architectures have been described in section IV. Synthesis result of RS (255, 251) Codec is presented in Section V and finally the paper is concluded in Section VI.

Manuscript received on January, 2013.
Anindya Sundar Das, Haldia Institute of Technology, Haldia, India.
Satyajit Das, C R Rao Advanced Institute of Mathematics, Statistics and Computer Sciences, Hyderabad, India.
Jaydeb Bhaumik, Haldia Institute of Technology, Haldia, India.

II. REED-SOLOMON CODES

Reed-Solomon codes are linear block codes and a subset of BCH codes. An RS code is based on finite fields, often called Galois fields. The number and types of errors that can be corrected depends on the characteristics of the Reed-Solomon code. RS (n, k) codes parameters are described as follows.

m - Number of bits per symbol
k- Un-coded message length in symbols
n- Codeword length in symbols
(n-k)- Number of parity check symbols
t -Number of correctable symbol errors and $2t = n-k$

Figure 1 shows a typical systematic Reed-Solomon codeword. It is systematic because the data on the left is unchanged and the parity symbols are appended with data to form the code word.

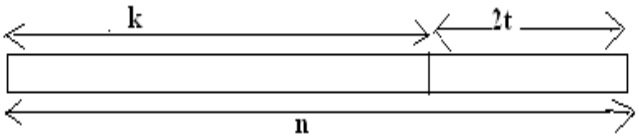

Fig1: RS(n, k) Codeword

An RS code with 8-bit symbols will use a Galois field GF(2^8). For RS (255,251) code, n = block length = 255, k = no. of un-coded massage symbols= 251, 2t = (n-k) = number of parity symbols =4, t = maximum number of errors can be corrected =2. The original massage can be recovered by employing the RS decoder provided number of errors in the received codeword is less than or equal to two.

III. RS ENCODER

In this section a brief overview of RS encoder and its corresponding architecture has been discussed. The RS encoder takes a block of symbols and adds extra parity check symbols. A RS codeword can be computed from input message symbols by employing a generator polynomial. A Generator polynomial depends on the order of the Galois field over which RS code has been defined and numbers of error to be corrected. The primitive polynomial is decided by the order of the Galois field. One of the primitive polynomial in GF(2^8) field is $x^8+x^4+x^3+x^2+1$. Let α be the primitive element in the Galois field in GF(2^8). Then the generator polynomial for t error correcting RS code is as follows.

$$g(x) = \prod_{i=1}^{2t}(x + \alpha^i) \qquad (1)$$

For a double byte error correcting code (t=2), the corresponding generator polynomial is

$$g(x) = (x + \alpha)(x + \alpha^2)(x + \alpha^3)(x + \alpha^4) \quad (2)$$

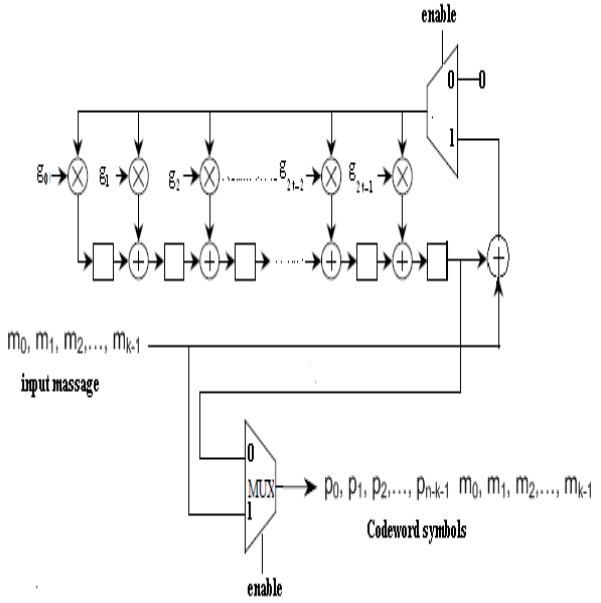

Fig2: Architecture for Reed-Solomon Encoder

Let the input message polynomial M(x) of the encoder is

$$M(x) = m_{k-1}x^{k-1} + m_{k-2}x^{k-2} + \cdots + m_1x + m_0 \quad (3)$$

The corresponding codeword polynomial is given by

$$C(x) = c_{n-1}x^{n-1} + c_{n-2}x^{n-2} + \cdots + c_1x + c_0 \quad (4)$$

This codeword is generated from the input massage polynomial by using the following formula.

$$C(x) = x^{2t}M(x) + x^{2t}M(x) \bmod g(x) \quad (5)$$

If Q(x) and P(x) are the corresponding quotient and remainder polynomial when $x^{2t}M(x)$ is divided by the g(x), then the codeword polynomial can also be expressed in another way.

$$C(x) = x^{2t}M(x) + P(x) = Q(x)g(x) \quad (6)$$

Here P(x) is the parity check polynomial. From equation (6) one can obtained

$$\frac{x^{(n-k)}M(x)}{g(x)} = Q(x) - \frac{P(x)}{g(x)} \quad (7)$$

By adding parity check polynomial P(x) with the x^{2t} M(x) polynomial, the codeword polynomial C(x) is computed, which is completely divisible by g(x).

IV. RS DECODER

The decoder processes each block and attempts to correct errors which may occur during transmission or storage. The decoder divides received codeword polynomial by the RS code generator polynomial. If the remainder is zero, then no errors are detected, else indicates the presence of errors.

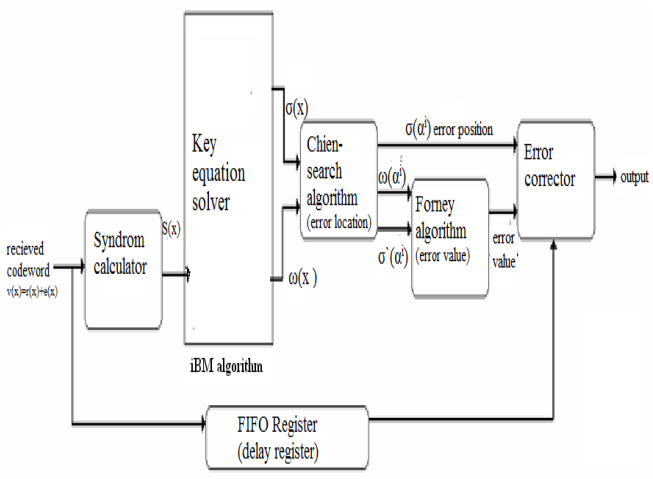

Fig3: Block Diagram of Reed-Solomon Decoder

A typical RS decoder has five stages in the decoding cycle, namely

1. Syndrome Generator
2. Key equation solver -determine error-location polynomial and error-evaluator polynomial
3. Chien search -solving the error locator polynomial
4. Forney algorithm for calculating the error magnitude
5. Error Correction

A. Syndrome Generator

Syndrome generation from received codeword is the first step of decoding process. Here the syndromes are calculated and it is decided whether there are errors in the received codeword or not.

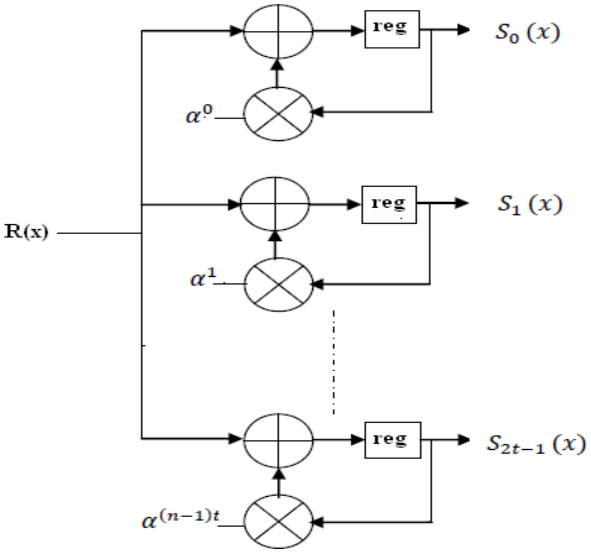

Fig4: Architecture for Syndrome Generator

The syndrome polynomial is generally represented as

$$S(x) = S_0 + S_1x + S_2x^2 + \cdots + S_{(2t-1)}x^{(2t-1)} = \sum_{i=0}^{(2t-1)} S_ix^{(i-1)} \quad (8)$$

Assume R(x) is the received codeword polynomial

$$R(x) = R_0 + R_1(x) + R_2x^2 + \cdots + R_{n-1}x^{n-1} \quad (9)$$

Then i^{th} syndrome can be expressed as follows

$$S_i = R_0 + \alpha^i(R_1 + \alpha^i(R_2 + \alpha^i(R_3 + \cdots + \alpha^i(R_{n-1}) \ldots))) \quad (10)$$

Where R_{n-1} is the first received symbol.

B. Key Equation Solver

The next step, after the computing the syndrome polynomial is to calculate the error values and their respective locations. This stage involves the solving of the 2t syndrome polynomials, formed in the previous stage. These polynomials have ν unknowns, where ν is the number of errors occurred in the received codeword. If the unknown locations are i_1, i_2... i_ν , then error polynomial can be expressed as

$$E(x) = Y_1 x^{i_1} + Y_2 x^{i_2} + \cdots + Y_\nu x^{i_\nu} \quad (11)$$

where Y_r is the magnitude of the r^{th} error at location i_r . If x_r is the field element associated with the error location i_r, then the syndrome coefficients are given by

$$S_j = \sum_{t=1}^{\nu} Y_r x_r^t \quad (12)$$

Where, j = 1,2…2t. Y_r is the error value and Xr is the error location of the r^{th} error symbol. The error locator polynomial can be determined as

$$\sigma(x) = \sigma_\nu x^\nu + \sigma_{\nu-1} x^{\nu-1} + \cdots + \sigma_1 x + \sigma_0 \quad (13)$$

The solution for the coefficients of the σ(x) is representing the error location polynomial. This σ(x) is related with the error value polynomial by a certain equation, which is known as the key equation. If the error value is defined by ω(x), the key equation is expressed as

$$S(x)\sigma(x) = \omega(x) \bmod x^{2t} \quad (14)$$

Then ω(x) can be used to solve for the error magnitudes Y_1,..... Y_ν.

These equations are solved in various way. In this paper we are using 'inversion less Berlekamp-Massey algorithm' (iBM algorithm). The algorithm is described below:[5]

Initialization

$\Lambda_0(0) = b_0(0) = 1$; $\Lambda_i(0) = b_i(0) = 0$; i = 1,…, t ;

k(0) = 0; γ(0) = 0;

input: S_i , i = 0,1,…,2t-1

for r =0 **step1 until** 2t-1 **do**

begin

step 1: $\delta(r) = S_r \Lambda_0(r) + \ldots\ldots\ldots\ldots\ldots + S_{r-t} \Lambda_t(r)$

step2: $\Lambda_i(r+1) = \gamma(r) \Lambda_i(r) - \delta(r) b_{i-1}(r)$, (i = 0, 1,…, t)

step3: if δ(r)≠0 and k(r) ≥0

then

begin

$b_i(r+1) = \Lambda_i(r)$;

$\gamma(r+1) = \delta(r)$;

k(r+1) = -k(r)-1;

end

else

begin

$b_i(r+1) = b_{i-1}(r)$;

γ(r+1) = γ(r);

k(r+1) = k(r)+1;

end

end

for i=0 **step1 until** t-1 **do**

step4: $\omega_i(2t) = S_i \Lambda_0(2t) + \ldots\ldots\ldots\ldots\ldots + S_0 \Lambda_i(2t)$

output: $\Lambda_i(2t)$, i= 0,1…,t $\omega_i(2t)$, i= 0,….,(t-1)

C. Chien-Search Algorithm

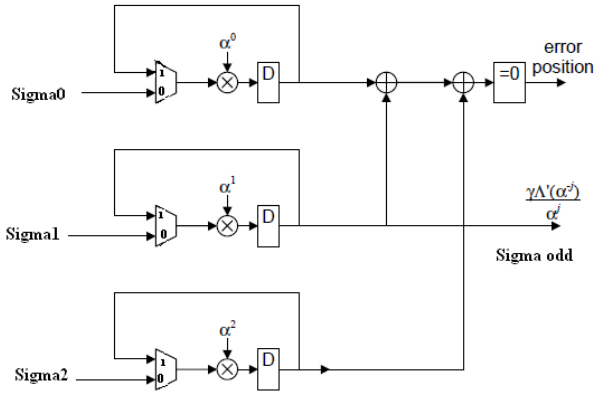

Fig5: Architecture for Chien-Search Algorithm

The Chien-search algorithm is used for evaluating the error position. The roots of the error locator polynomial are the inverse error locations of the codeword. This algorithm uses all possible input values and then checks to see if the outputs are zero. The sum for the odd values (σ_1, σ_3, σ_5……) is calculated in one side and the sum for the even values (σ_0, σ_2, σ_4, σ_6,...) is calculated in other . Then the two sums are added. If the value of the summation is zero in any clock cycle (<n) then the position of the clock cycle will determine the error position.

D. Forney Algorithm

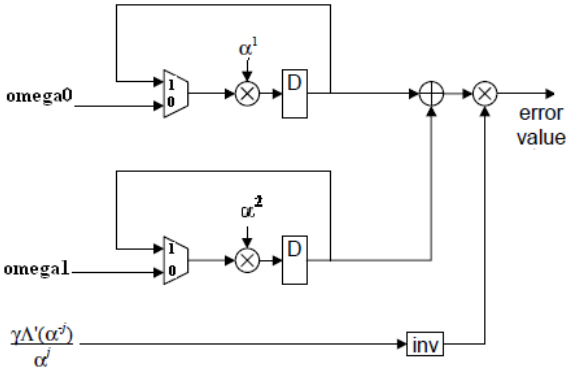

Fig6: Architecture for Forney algorithm

Forney algorithm is used for evaluating the error values. The error position and the coefficients of ω(x) is taken here as the input. It is also using Galois field multiplier as the Chien-search algorithm. The equation for the error values is given by:

$$Y_i = \frac{\omega(x_i^{-1})}{\grave{\sigma}(x_i^{-1})} \quad (15)$$

Here x_i^{-1} indicates the root as computed from the Chien-search and **σ′** (x) is the derivative of the error locator polynomial. The above equation can be written as:

$$Y_i = \frac{\omega(x_i^{-1})(x_i^{-1})}{\sigma_{odd}(x_i^{-1})} \qquad (16)$$

$\sigma_{odd}(x_i^{-1})$ can be found from the Chien-search Algorithm.

E. Error Correction

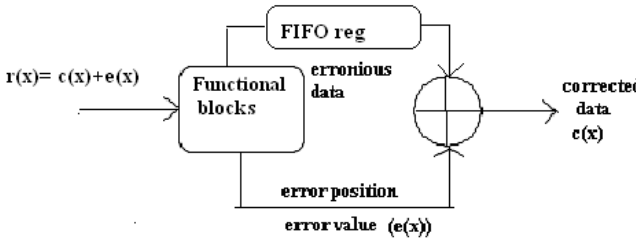

Fig7: Error Correction module

Once the error locations and magnitudes are calculated, the error corrector block takes the received code and performs XOR operation with the corresponding error magnitudes, computed at the respective error locations to obtain the corrected massage symbols.

$$C(x) = R(x) + E(x) \qquad (17)$$

Because the error symbols are generated in the reverse order of the received codeword, a FIFO register must be applied to either the received codeword or the error vector to match the order of the bytes in both vectors. The output of XOR gate is the decoder's estimated codeword.

V. SYNTHESIS RESULT

Every architectural module has been implemented in Verilog and simulated using ModelSim. The design has been synthesized by Xilinx ISE 7.1i tool. Table I gives the synthesis result of RS (255,251) encoder and decoder. The target FPGA Device was Spartan3-X3S50TQ144-5.

Fun. blocks	No. of slices	No. of FFs	No. of input LUTs	No. of IOBs	Max. freq. (MHz)	Min Period (ns)
Encoder	46	40	86	67	142.55	7.02
Syndrome calculator	56	72	104	50	180.49	5.54
Key equation solver	662	64	1167	74	102.14	9.80
Chien search block	60	32	105	66	184.38	5.42
Forney block	292	24	509	90	184.38	5.42
Error corrector	28	41	41	50	183.44	5.45

VI. CONCLUSION

The paper presents a design and implementation of RS (255, 251) double byte error correcting encoder and decoder for FPGA platform. Proposed Reed-Solomon codec has been synthesized by employing Xilinx ISE 7.1i tool. The result shows that it is very effective as it works faster and requires less hardware.

REFERENCES

[1] S. Lin and D. J. Costello, Error Control Coding: Fundamentals and Applications. Englewood Cliffs, NJ: Prentice-Hall, 1983.
[2] R. E. Blahut, Theory and Practice of Error Control Codes. Addison-Wesley, 1983
[3] S. B. Wicker and V. K. Bhargava, Reed Solomon Codes and Their Applications, IEEE Press, 1994.
[4] T. K. Matsushima, T. Matsushima, and S. Hirasawa, "Parallel architecture for high-speed Reed-Solomon codec," in Proc. IEEE Int. Telecommunication. Symposium. 1998, pp. 468–473.
[5] D. V. Sarawate and N. R. Shanbhag, "High-speed architectures for Reed-Solomon decoders," IEEE Trans. on VLSI Systems, vol. 9, no. 5, Oct. 2001, pp. 641-655.
[6] E. Mastrovito. "VLSI Architectures for Computations in Galois Fields," Ph. D. Dissertation, Dept. of Electrical Engineering, Linkoping University, Linkoping, Sweden, 1991.
[7] L. Song, K. K. Parhi, I. Kuroda. and T. Nishitani. "Hardware/software codesign of finite field datapath for low-energy Reed-Solomon codecs," IEEE Trans. on VLSI Systems, vol. 8. no. 2, Apr. 2000, pp. 160-172.
[8] C. Y. Lee, Y. H. Chen, C. W. Chiou and J. M. Lin, "Unified Parallel Systolic Multipliers over GF(2^m),"Journal of Computer Science and Technology, vol. 22, Jan. 2007, pp. 28-38.

Mr. Anindya Sundar Das is currently working as a JRF at Haldia Institute of Technology. He did his M. Tech in Electronics and Communication Engineering (specialization in Micro Electronics and VLSI Design) Under West Bengal University of Technology. He has received his B. Tech degree from Uttar Pradesh Technical University in the year of 2006 in Electronics and Instrumentation Engineering. His area of research area includes Error control coding, Optical Communication and VLSI design.

Mr. Satyajit Das is currently working as a SRF at CR Rao Advanced Institute of Mathematics, Statistics and Computer Sciences. He did his M. Tech in Electronics and Communication Engg.(specialization Micro Electronics and VLSI Design) Under West Bengal University of Technology. He has received his B. Tech degree from West Bengal University of Technology in the year of 2009 in Electronics and Communication Engg. His area of research area includes Cryptography and VLSI design.

Dr. **Jaydeb Bhaumik** is currently working as an Associate Professor in the Department of Electronics and Communication Engineering, Haldia Institute of Technology, Haldia, India. He obtained his PhD degree from G. S. Sanyal School of Telecommunications, Indian Institute of Technology Kharagpur, India in 2010. He received his B. Tech. and M. Tech degrees in Radio Physics and Electronics from University of Calcutta in 1999 and 2001 respectively. His research interests include Cryptography, Cellular Automata, Error Correcting Codes, and Digital VLSI Design. He is a member of IEEE and Cryptology Research Society of India..