International Journal of Electrical, Electronics and Data Communication, ISSN(p): 2320-2084, ISSN(e): 2321-2950
Volume-7, Issue-10, Oct.-2019, http://iraj.in

A COMPARISON OF R-S(7,3) AND R-S(15,9), EMPLOYING REED-SOLOMON ENCODER AND DECODER

[1]SANA ZARIN, [2]RAJNI GUPTA

[1]M.Tech. Student, Greater Noida Institute of Technology, Greater Noida, UP, India
[2]Assistant Professor, Greater Noida Institute of Technology, Greater Noida, UP, India
E-mail: [1]zarinsana1@gmail.com, [2]rajniec75@gmail.com

Abstract - In an era of digital communication and information technology, we all seek to transmit information in a fraction of a second. However, it's always likely that the reliability of the communication is jeopardized by noise and errors. It's because the communication system is always susceptible to errors and noise. Therefore, error detection as well as correction becomes an essential prerequisite to any efficient communication system. Here come Reed-Solomon codes that ensure smooth and hassle-free data transmission. Using VHDL code on Xilinx Spartan 6, this paper is aimed at designing, simulating and implementing R-S Encoder and R-S Decoder. It targets GF(2^3) and GF(2^4) to correct 2 burst errors and 3 burst errors in R-S(7,3) and R-S(15,9)encoder and decoder respectively.

Keywords - R-S Encoder and Decoder, Galois Field, VHDL

I. INTRODUCTION

Reed-Solomon (R-S) codes have established their credentials as non-binary codes with the capacity for correcting transmission errors effectively[12]. The codes markedly improve the quality of data transmission. R-S codes are employed in myriad applications like audio-tapes, QR codes, satellite transmission and a CD-ROM[4]. This wide range of applications requires various R-S codes. The latest digital system has to grapple with codes that use to fix errors. The hardware implementation of these codes is done through gate arrays or FPGA, by employing any of the hardware-description languages like VHDL. The users enter the desired parameters of the R-S code, such as code word length, error-correcting capability, initial root of the code of generator polynomial, the size of the Galois Field elements, and the field generator polynomial. This research work is divided into seven sections, which include this introduction. The section II of this paper deals with Reed-Solomon codes, section III is about R-S encoder, section IV sheds light on R-S decoder, section V illustrates simulation result, section VI presents conclusion ,and section VII looks at future prospectus of R-S codes.

II. REED-SOLOMON CODES

Irving S. Reed and Gustave Solomon published a paper titled "Polynomial Codes over Certain Finite Fields" in June 1960[10]. The study dealt with a new class of codes that were meant for fixing errors. Today, we know these codes as Reed-Solomon codes. R-S codes, in fact, have revolutionized the telecommunication industry. They were first used in the making of compact discs[11]. When we encode the data, it comes about in the form of polynomial; this actually is the basis of R-S codes. These codes are based on an algebraic theorem that posits that k distinct points are in a unique position. It is a subclass of non-binary BCH codes and is linear block code[2]. It varies from a binary encoder in that it works on the several bits instead of separate bits. It has highest efficient use of redundancy. It is possible to adjust block length and symbol size[1]. A finite field, which we usually call Galois Field, is the basis of R-S code. R-S(n,k) codes in which 'n' represent number of code symbols and 'k' represents information symbols. The number of parity symbols is n-k=2t and t is the number of error correction.

$n-k=2t$ (1)

Symbol size, m, and number of code word, n, is shown by,

$n=2^m-1$ (2)

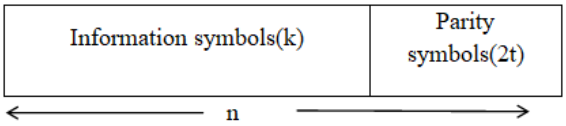

Fig.1 R-S code word's structure

III. R-S ENCODER

R-S codes are part of non-binary BCH codes where generator polynomial is situated on GF(2^m). As for a t-error-correcting Reed-Solomon code having the value $n=2^m-1$, one can see generator polynomial written below[3]:

$g(x) = (x-\alpha^i)(x-\alpha^{i+1})\ldots(x-\alpha^{i+2t})$ (3)

$g(x) = g_0+g_1x+g_2x^2+\ldots+g_{2t-1}x^{2t}+x^{2t}$ (4)

Whenever $2t=n-k$ and α is a primitive element of GF(2^m) and m_0 is the log of initial root of the code generator polynomial. Compared to other codes, R-S codes are distinct in a way that here additions and multiplication are done over GF(2^m).

The information elements m_i belongs to GF(2^m) for $0 \leq i \leq k-1$, which is used for the constructions of code word[13]. The polynomial form such as, R-S(n,k) code words expressed as,

$C = (m(\alpha_1), m(\alpha_2),..., m(\alpha_n))$
Using of support set $(1,\alpha, \alpha^2, ... \alpha^{n-1})$ (6)
Code word R-S(n,k) code formed with the help of Eq.(6)
$C = (m(1), m(\alpha),..., m(\alpha^{n-1}))$ (7)
The code word polynomial,
$c(x) = m(\alpha^{n-1})x^{n-1} + m(\alpha^{n-2})x^{n-2} +....+ m(\alpha)x + m(1)$ (8)

The encoding steps are[15]:
Step 1: Multiplication of m(x), which symbolizes information polynomial, by x^{n-k}.
Step 2:- In order to get the remainder r(x), let's divide $m(x)x^{n-k}$ by g(x)
$r(x) = m(x).x^{n-k} \bmod g(x)$
Step 3:- Then code-word polynomial c(x) is derived through m(x) and r(x)
$c(x) = m(x)x^{n-k} + r(x)$
A circuit of mod g(x) division is its most significant framework. The feedback linear shift registers, adders, multipliers, and switches is shown in the circuit[14].Below is its circuit diagram of Encoder (see Fig. 2.)

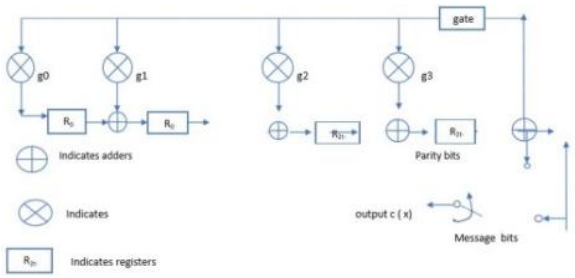

Fig.2 Circuit Diagram of Encoder

Here's a step-by-step description of the workings of the circuit:
Step 1: All the registers R_0 to R_{2t-1} are set at 0. When the switch is on the message m(x) is sent to two points: one to the output c(x) and the other, after multiplying with parity bits, to the circuit.
Step 2: After k^{th} clock cycle the m(x) is sent to the circuit and division is done. We get r(x) as co-efficient of remainder. All the values are stored in the registers.
Step 3: After k+1th clock cycle the switch is moved to position 2. Then the data in the registers start shifting one by one. After 2t=n-k clock cycle, all the data is sent to the output end. And code word c(x) is formed along with k information symbols to m(x). This is the process of encoding one block of code.
Step 4: Now we reset all the registers at 0. And again undertake encoding in the same done above for the next block of codes.

IV. R-S DECODER

The code word c(x) that we get following the process of encoding is expected to be interrupted while being communicated through channel[7]. Say e(x) is noise or error pattern. Now it's going to be attached to c(x) and the sum would be communicated to the receiver. Say for example the received-vector's polynomial is r(x), thus r(x) = c(x) + e(x). Below are expressions of c(x), e(x), as well as r(x).
$c(x) = c_0 + c_1x + c_2x^2 +....+ c_{n-1}x^{n-1}$ (8)
$e(x) = e + e_1x + e_2x^2 +....+ e_{n-1}x^{n-1}$ (9)
$r(x) = r_0 + r_1x + r_2x^2 +....+ r_{n-1}x^{n-1}$ (10)

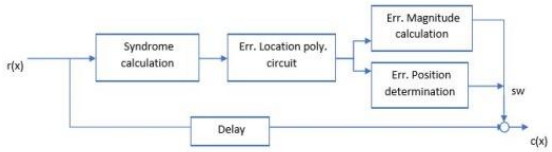

Fig.3 R-S Decoding Steps

We can perform RS decoding in the following ways:
Step 1. Calculating Syndromes
By the definition of Syndrome, $S = RH^T$
Syndrome polynomial, $s(x) = s_1 + s_2x + s_3x^2 +....$
$+ s_{2t}x^{2t-1}$ (8)
Calculating the coefficients $s_i (i = 1,2,...,2t)$ of s(x) based on r(x)
$s_i = r(\alpha^i) = \sum_{j=0}^{n-1} rj\,(\alpha i)^j$
$= r_0 + r_1\alpha^i + r_2(\alpha^i)^2 +....+r_{n-1}(\alpha^i)^{n-1}$
$= r_0 + \alpha^i(r_1+...+ \alpha^i(r_{n-3} + \alpha^i(r_{n-2} + \alpha^i r_{n-1}))...)$ (9)
Step 2. Calculating polynomial that locates errors
Once syndrome polynomial is calculated, error values as well as their locations are computed. Here, 2t values of syndrome polynomials are calculated from step one. The polynomials contain aggregate v unknown terms; here v is unknown error number before decoding[8].
Step 3. (Chien Search) addressing the issue of error position
Decoding process also involves an important stage, which is to pinpoint the locus of errors in the code word received [5]. Chien Search is used to achieve this end. Using input values, the Search finds out if output is 0.
Step 4. Deriving error values
As soon as location of error is found out, syndrome values as well as error polynomial sources are used with a view to calculating error values. Forney's Algorithm [6], which is employed to achieve this goal, can efficiently undertake a matrix inversion.
Step 5. Correcting errors
Error symbol having set bit shows received code's corresponding bit has error, and thus needs to be fixed[9]. To this end, all symbols are read again. At each location of error, the symbols received are XOR having error symbols. This way decoder fixes an error.

V. RESULTS

This paper seeks to design R-S(7,3) & R-S(15,9) encoders and decoders, which are simulated in VHDL. These encoders and decoders are synthesized on Spartan 6 FPGA board using Xilinx software. The

paper deals with the way to detect and correct errors. Identifying errors and fixing them is an essential prerequisite for seamless communication. We reduce the impact of an error during communication by way of introducing redundancy to data before communication. The redundancy enables decoder to detect and correct errors. Table 1 illustrates design of R-S(7,3) and R-S(15,9) encoder.

	R-S (7,3)	R-S (15,9)
No.of code symbols	n=7	n=15
No. of data symbols	k=3	k=9
No. of parity symbols	n-k=4	n-k=6
No. of error correcting capability	t=n-k/2=2	t=n-k/2=3
Galois Field	GF(2^3)	GF(2^4)
PrimitivePolynomial	1+x+x^3(11 decimal)	1+x+x^4(19 decimal)

Table1 Design of R-S Encoder

The RTL schematic for R-S(7,3) and R-S(15,9) encoder is shown in Fig.4 and Fig. 5

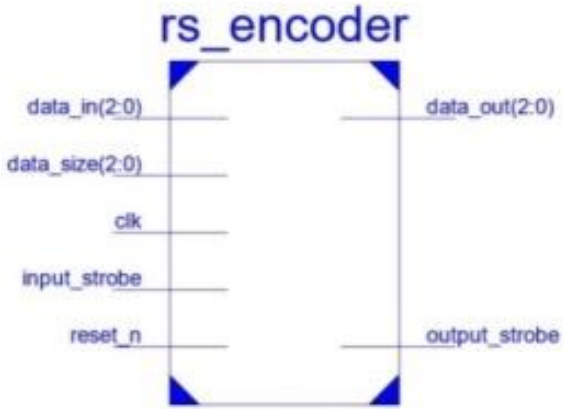

Fig. 4 R-S(7,3)Encoder

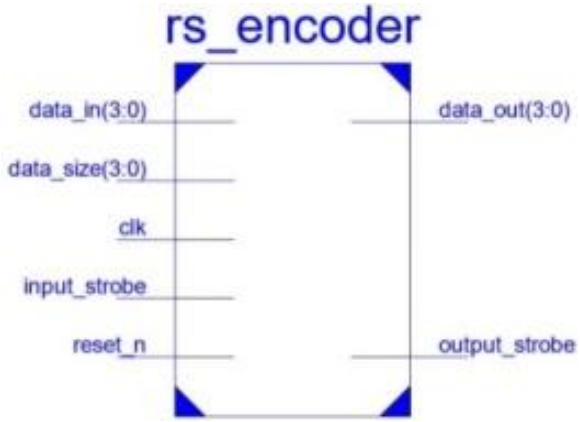

Fig. 5 R- S(15,9)Encoder

The simulation output testifies our design for R-S (7,3) & R-S(15,9) encoder, which is shown in Fig. 6 and Fig. 7. The encoder contains an error indicator, which shows errors in data.

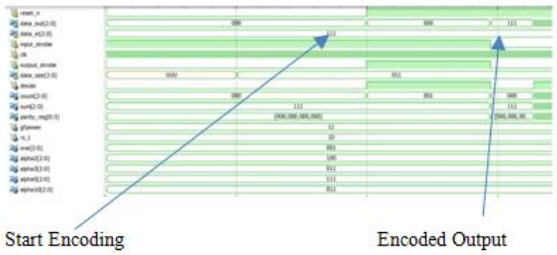

Fig. 6 Waveform of R-S(7,3) Encoder

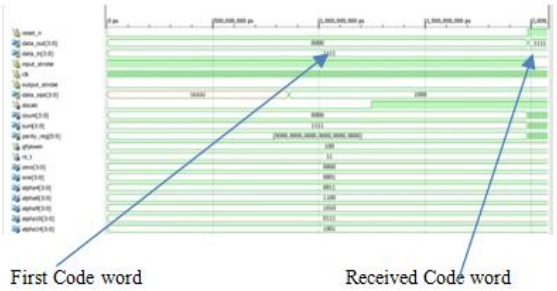

Fig. 7 Waveform of R-S(15,9) Encoder

RTL Schematic for R-S(7,3) & R-S(15,9) decoder is shown in Fig 8 and Fig 9.

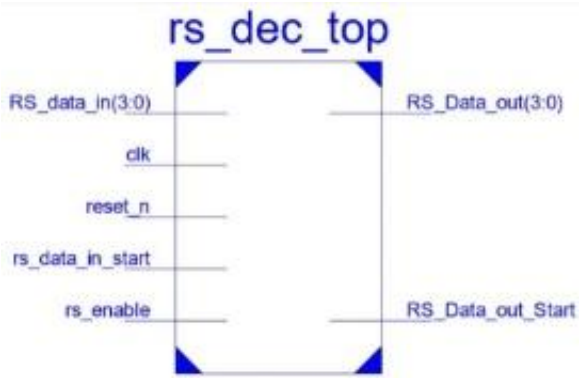

Fig. 8 R-S(7,3) Decoder

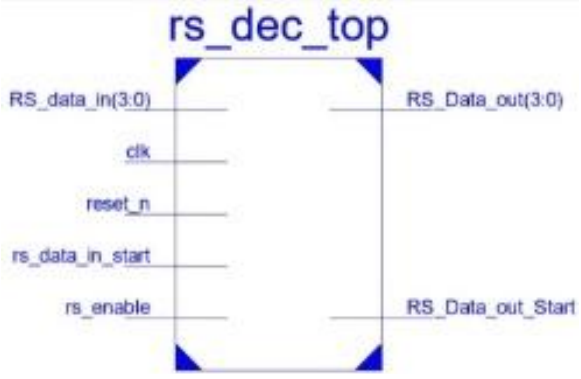

Fig. 9 R-S(15,9) Decoder

For R-S(7,3) and R-S(15,9) decoder, simulation waveform has various word vectors. If errors are greater than the capacity for correction, decoder resumes decoding failure towards the end of the received message. The waveform is shown in Fig. 10 and Fig. 11.

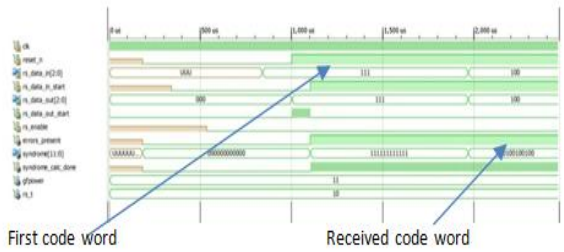

Fig.10 Waveform of R-S(7,3)Decoder

Fig.11 Waveform of R-S(15,9)Decoder

Table 2 and Table 3 show the device utilization for R-S(7,3) and R-S(15,9) encoder and decoder on target device XC6slx4-2tqg144.

Logic Utilization	R-S(7,3)		R-S(15,9)	
Number of Slices Registers	19 out of 4800	0%	32 out of 4800	0%
Number of Slices LUTs	24 out of 2400	1%	43 out of 2400	1%
Number of fully used LUT+FF pairs	19 out of 24	79%	33 out of 43	76%
Number of bonded IOBs	13 out of 102	12%	16 out of 102	15%
Number of BUFG	1 out of 16	6%	1 out of 16	6%

Table 2 Synthesis Report for device utilization of R-S(7,3) & R-S(15,9) encoder

Logic Utilization	R-S(7,3)		R-S(15,9)	
Number of Slices Registers	32 out of 4800	0%	130 out of 4800	2%
Number of Slices LUTs	18 out of 2400	0%	136 out of 2400	5%
Number of fully used LUT+FF pairs	17 out of 33	51%	88 out of 178	49%
Number of bonded IOBs	10 out of 102	9%	13 out of 102	12%
Number of BUFG	1 out of 16	6%	1 out of 16	6%

Table 3 Synthesis Report for device utilization of R-S(7,3) & R-S(15,9) decoder

For R-S(7,3) and R-S(15,9) encoder and decoder show the timing delay where the speed grade is -2. The delay report is shown in Table 4.

	Encoder		Decoder	
	R-S(7,3)	R-S(15,9)	R-S(7,3)	R-S(15,9)
Speed Grade	-2	-2	-2	-2
Delay Time (ns)	2.724	3.035	2.009	3.374
Maximum Frequency (MHz)	367.107	329.489	497.760	296.384

Table 4 Comparison Table of R-S (7,3) R-S(15,9) Encoder and Decoder

VI. CONCLUSION

Finally, simulation of the R-S encoder and decoder with constraint R-S(7,3) & R-S(15,9) has been used to correct 2 burst errors and 3 burst errors. The synthesis results show that the delay time of the R-S(7,3) encoder is 2.724ns and decoder is 2.009ns. Similarly, the delay time of R-S(15,9) encoder is 3.035ns and decoder is 3.374ns. The comparison is shown in table 4. The R-S(7,3) uses less number of logic element compared to R-S(15,9). This work shows that there is reduction in cost, less fabrication area and low complexity with high speed.

FUTURE SCOPE

The practitioners of tamper detection and copyright protection may find it to be a good reference. Reed-Solomon codes can be developed to correct multiple errors and create various types of constructions. It is interesting to study bar codes and QR codes applications, because they have the capacity to visually display the error-correction codes, making the codes easily accessible to interested users.
We can see that satellite transponder bandwidth has regularly been very popular. In fact, its popularity is growing.

REFERENCES

[1] Aqib. Al Azad, Minhazul. Huq, Iqbalur,and Rahman Rokon, " Efficient Hardware Implementation of Reed-Solomon Encoder and Decoder in FPGA using Verilog", International Conference on Advances in Electronics and Power Engineering (ICAEPE, 2011) Bangkok Dec, (117-119) 2011.
[2] Amandeep Singh, and Mandeep Kaur, "Design and Implementation of Reed-Solomon Encoder on FPGA," World Academy of Science, Engineering and Technology International Journal of Information and Communication Engineering,Vol.7 No.9 (1248-1250) 2013.
[3] British Broadcasting Corporation, "Reed-Solomon Error Correction," R & D White paper, 2002.
[4] Harshadal. Borkar, and Prof.V.N. Bhonge, "Design and Implementation of Reed-Solomon Encoder and Decoder," SSRG International Journal of Electronics and Communication Engineering, (2445-2451)Jan 2015.
[5] Manju Mangtani, " Implement Reed-Solomon Encoder/Decoder using Spartan FPGA," Journal of Research

in Engineering and Applied Sciences,Vol. 1(199-205) 2016.

[6] National Aeronautics and Space Administration (NASA) technical reports, "Tutorial on Reed-Solomon error correction coding," Lyndon B. Johnson Space Center Houston, 1990.

[7] P. Parvathi and P.Rajendra Prasad, "FPGA based design and implementation of Reed-Solomon encoder & decoder for Error Detection and Correction," Conference on Power, Control, Communication and Computational Technologies for Sustainable Growth,(261-266) 2015.

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

[9] Rajeev Kumar Patial and Priyanka Dayal, "FPGA Implementation of Reed-Solomon Encoder and Decoder for Wireless Network 802.16, International Journal of Computer Applications,Vol.68-No.16,(0975-8887) April 2013.

[10] Diplaxmi Chaudari, Mayura Bhujade, and Pranali Dhumal, "VHDL Design and FPGA Implementation of Reed-Solomon Encoder and Decoder for R-S(7,3)," SSRG International Journal of Electronics and Communication Engineering, March 2014.

[11] Chandale, Gavali, Giri, Shrivastava: FPGA Based Error Detection and Correction System using Reed Solomon Code, International Research Journal of Engineering and Technology, Vol.03(1446-1450), 2016.

[12] Wai and Yang: Field Programmable Gate Array Implementation of Reed Solomon code, RS(255,239).

[13] Szymon Czynszak: Decoding algorithms of Reed Solomon code, School of Computing Blekinge Institute, October 2013.

[14] Leonvan Pavert,"Reed-Solomon Encoding and decoding" Turku University of Applied Science, 2011.

[15] http://ptgmedia.pearsoncmg.com/images/art_sklar7_reed-solomon/elementLinks/art_sklar7_reed-solomon.pdf

★★★