Design of Reed Solomon LCC Soft Decision Decoder Based on HDD

Jeseena S
PG Student/Department of ECE
TKM Institute of Technology, Kollam,
Kerala, India

Suby Varghese
Assistant Professor/Department of ECE
TKM Institute of Technology, Kollam
Kerala, India

***Abstract*—Channel coding is an important signal processing operation for the efficient transmission of digital information over the channel. In this, the number of symbols in the source encoded message is increased in a controlled manner, in order to facilitate two basic objectives at the receiver, one is error detection and other is error correction. In high speed communication system, Reed-Solomon (RS) codes are widely used to provide error protection, especially against the burst errors. Reed Solomon codes are non-binary, linear block based forward error correction code. In this paper a Reed-Solomon low complexity Chase (LCC) decoder has been designed based on hard decision decoding algorithm. With this, the error correcting capability can be improved by incorporating the reliability information from the channel into algebraic soft decision LCC decoding algorithm. Also by introducing hard decision decoding, the sophisticated interpolation algorithm of LCC decoder can be avoided.The coding is done in VHDL language, synthesized using Xilinx ISE 13.2 and simulated using ModelSim.**

***Keywords*—Reed-Solomon (RS) codes, Hard decision decoder (HDD), Algebraic soft decision decoder (ASD), Low complexity Chase (LCC), Syndromes**

I. INTRODUCTION

One of the most important challenges in digital communication system is to provide efficient and reliable data transmission and storage systems. Channel coding is an important signal processing operation for the efficient and reliable transmission of digital information over the channel. The power of channel code or forward error correction (FEC) code is that the system can, without retransmissions, find and correct limited errors caused by a transmission or storage system. Reed-Solomon (RS) code is a well-known non-binary linear block based FEC code. It is popularly used for error correction in many applications like storage devices (CD, DVD), wireless communications, high speed modems and satellite communications.

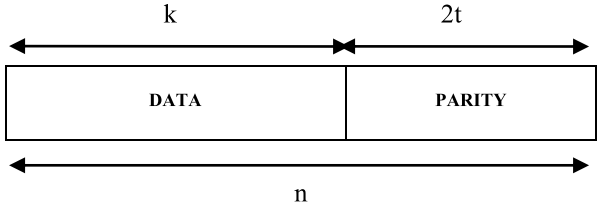

Fig. 1 Reed Solomon Codeword

RS code can be specified as RS (n, k) with m-bit symbols, where $n=2^m-1$ as shown in figure 1.The variable n is the size of the codeword symbols, k is the number of data symbols and 2t is the number of parity symbols. A predetermined sized block of data (k bytes) is encoded so that the result is a data block of size n, where n>k. This n block contains the k original data bytes, along with n-k parity bytes, representing the redundancy in the signal, for transmission over the noisy channel. Within the block, the RS algorithm works on multiple bits of data at a time, typically a byte. Each byte is a symbol, and the nature of the RS algorithm allows for the correction of whole symbols, as opposed to correcting individual bits. This means that the RS decoder can correct up to't' symbols that contain errors in a code word, where 2t=n-k. This is the particular characteristic which allows RS codes to be effective at correcting burst errors in addition to random errors.

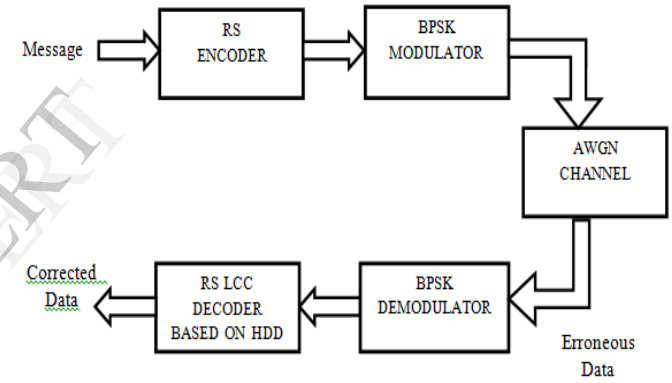

Fig. 2 System model

The proposed system consists of four main blocks: RS encoder, BPSK modulator, AWGN channel, BPSK demodulator and RS decoder as shown in figure 2. The RS encoder unit takes the incoming data and adds redundancy to obtain the codeword. The encoded message codeword are fed into a modulator, which maps each of the information sequence into signal waveforms. The channel over which the waveforms are transmitted will corrupt the waveforms by adding symmetric additive white Gaussian noise (AWGN). At the receiver, the digital demodulator processes the channel corrupted signal waveforms and reduces it into a sequence of data symbols. Among the two Reed-Solomon decoding algorithms, algebraic soft-decision decoding (ASD) has significant coding gain over hard decision decoding (HDD). Comparing the different ASD algorithms, the low complexity Chase (LCC) decoding is found to have less computation complexity with similar or higher coding gain. The LCC algorithm consists of three main steps: multiplicity assignment, interpolation and factorization, where interpolation stage involves major part of the computation. So instead of employing the sophisticated interpolation algorithm, the LCC decoding is done based on the HDD, to get the error locator and error evaluator polynomial.

This paper is organized as follows. Section 2 gives a brief overview of Reed Solomon LCC decoder based on HDD. In section 3, the simulation results of the decoder sections are discussed. Conclusions and ongoing works are discussed in section 4.

II. REED SOLOMON DECODER

The RS LCC decoder based on HDD method combines the hard-decision decoding techniques with the LCC decoding method to decode the RS codes using channel information. The block diagram of Reed-Solomon LCC decoder based on HDD is shown in figure 3. The steps of the decoder are: multiplicity assignment, unified syndrome computation, Key Equation Solver (KES), Chien Search for error location calculation, decoding failure detection, Forney's algorithm for error value calculation and error correction. The LCC decoding process creates 2^{η} different test vectors using the reliability information from the received points (channel). A unified syndrome computation algorithm is used in which the syndromes of one test vector can be obtained from syndromes of previous test vector. If an error is detected, the process of correction begins by locating the errors first. Generally Berlekamp-Massey Algorithm is used to calculate the error locator polynomial. The precise location of the errors is calculated by using Chien search algorithm. Then magnitude of the errors is calculated using Forney algorithm. In this HDD-based LCC decoding algorithm, the test vectors are selected for correction during the decoding on occurrence of the HDD failure. The magnitude of the error is added to the received codeword to obtain a correct codeword.

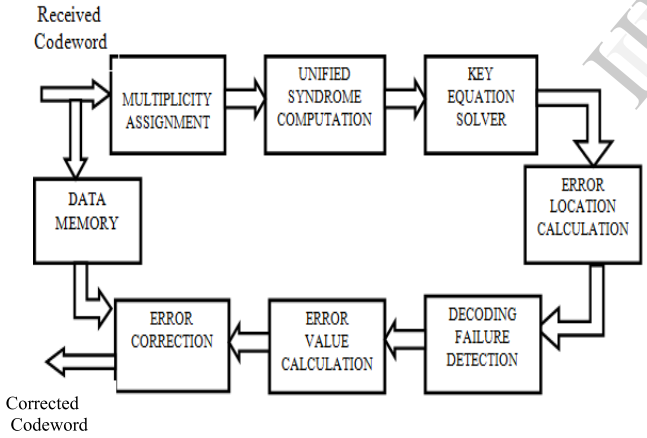

Fig. 3. RS LCC decoder based on HDD

A. Multiplicity Assignment

The RS LCC decoding process creates 2^{η} different test vectors using the reliability information from the channel, where $\eta<2t$. To create the test vectors a ratio between the probability of the hard-decision symbol and the second-best decision is obtained. This ratio indicates how good the hard decision is. The desired probability ratio for the received message polynomial r(x) for i = 0 to n-1 is defined, as in (1) Where $ri_{\_HD}$ is the hard decision of the symbol ri, and $ri_{\_2HD}$ its second best decision symbol.

$$\pi_{1Xn} = \left[\frac{P\left(r_0 \middle| P\left(r_{0\_HD}\right)\right)}{P\left(r_0 \middle| P\left(r_{0\_2HD}\right)\right)} \cdots \frac{P\left(r_{n-1} \middle| P\left(r_{(n-1)\_HD}\right)\right)}{P\left(r_{n-1} \middle| P\left(r_{(n-1)\_2HD}\right)\right)}\right] \quad (1)$$

Corresponding to the η points with the worst probability ratio, a set of 2^{η} combinations called test vectors are created by selecting the hard-decision symbol (ri) or the second-best decision ($ri_{\_2HD}$) in the η less reliable points. Once the test vectors are created, it is required to select one of them, and correct the errors of the message, if it is possible.

To create the 2^{η} test vectors, η symbols have to be marked with zero multiplicity, as the rest of symbols are marked with one multiplicity. To classify the multiplicity of the points, the probability ratio for the received message polynomial r(x) are ordered, and η points with the least values will be marked as the points with different values for the test vectors.

B. Unified Syndrome Computation (USC) Architecture

The USC architecture consists of the syndrome calculation and the syndrome update architecture. They are separated into different pipelining stages.

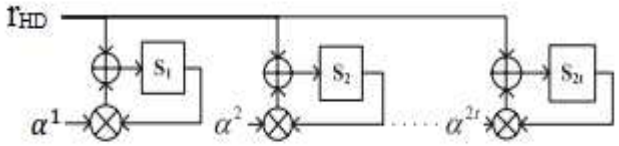

Fig. 4. Syndrome Calculation

The syndrome calculation architecture for the decoder is shown in figure 4. The syndrome consists of n–k symbols and the values are computed from the received test vector $r_{HD}(x)$. The syndrome depends only on the error vector, and is independent of the transmitted code word. The syndrome values can be obtained as (2) by substituting $x = \alpha^i$ in the received polynomial for i=1,2,…..n-k.

$$S_i = r_{HD}(\alpha^i) \quad (2)$$

With syndrome calculation architecture, the syndromes of the first test vector are calculated and stored in registers from S_1 to S_{2t}, and later pass the syndrome update module in order. At the same time, the positions α^i of η unreliable points with their $r_{i\text{-}HD}$ and $r_{i\text{-}2HD}$, i ε Z , are also stored.

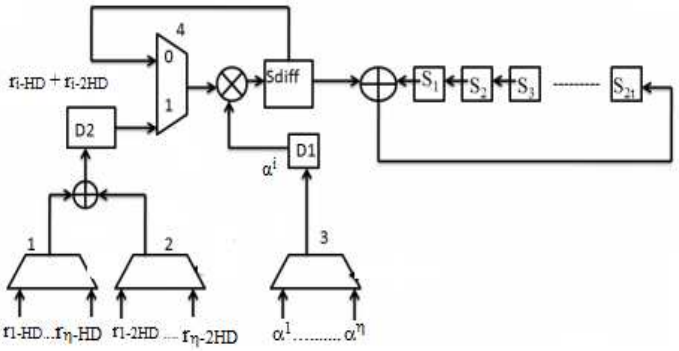

Fig. 5. Syndrome Update Architecture

In syndrome update architecture, the multiplexer 1, 2 and 3 choose $r_{\varepsilon\text{-}HD}$, $r_{\varepsilon\text{-}2HD}$ and α^{ε} of one unreliable point in Z. This selected point is the only different point between the current test vector and the next one and its position is denoted as ε. The difference of $r_{\varepsilon\text{-}HD}$ and $r_{\varepsilon\text{-}2HD}$ is calculated by sending them into

an GF (2^3) adder and is multiplied by α^{ε} to get $(r_{\varepsilon\text{-HD}} + r_{\varepsilon\text{-2HD}})\ \alpha^{\varepsilon}$. The value in S_{diff}, which is $(r_{\varepsilon\text{-HD}} + r_{\varepsilon\text{-2HD}})\ \alpha^{\varepsilon}$ is added to the value in S1, and then the sum is fed back to the register queue to update the syndromes. Also $(r_{\varepsilon\text{-HD}} + r_{\varepsilon\text{-2HD}})\ \alpha^{2\varepsilon}$ is calculated and added to S2. In this way, the stored syndromes of the first test vector can be updated by adding the corresponding syndrome difference to each item in the syndromes. Finally all the syndromes shift to the adder to finish the update.

C. Key Equation Solver

When all the syndrome values are zero then the received test vector is a correct codeword. If any of the syndrome value is nonzero then there is error in the received test vector. These syndrome values are then given as the input of the reduced inversionless Berlekamp-Massey (RiBM) algorithm. This KES block gives two polynomials-error locator polynomial and error evaluator polynomial by solving the key equation given as

$$\lambda(x)\ S(x) = \omega(x) \bmod x^{2t} \tag{3}$$

$$\text{where } \lambda(x) = 1+\lambda_1 x + \ldots\ldots . + \lambda_e x^e \tag{4}$$

$$\omega(x) = \omega_0 + \omega_1 x + \ldots.. + \omega_{e-1}\, x^{e-1} \tag{5}$$

With error locator polynomial coefficients $\lambda(x)$ and error evaluator polynomial coefficients $\omega(x)$, the errors can be corrected as the received word is being read out of the decoder.

D. Error Location Calculation

The locations of the errors are determined based on the error locator polynomial, $\lambda(x)$. Chien search is used for finding roots of the error locator polynomial. It is a brute force algorithm that evaluates the polynomial for all possible input values, and then checks to see which outputs are equal to zero. If $\lambda\ (\alpha^{-i})$ is equal to 0, then α^{-i} is an error location where i = 0,1,....,n-1.

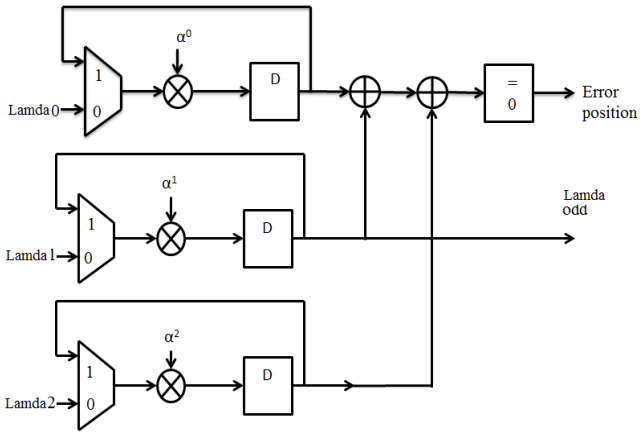

Fig. 6. Architecture for Chien Search Algorithm

E. Decoding Failure Detector

After error location calculation, the number of zeros/roots of this step is compared with degree of $\lambda(x)$ obtained from KES. If both are equal then corresponding test vector is selected for Forney's algorithm. It means that the first vector which is free from decoding failure will be used to make a decoding of the received message. Therefore, if one of the 2^{η} test vectors has less than t+1errors, then the decoder will be able to correct them in all cases. Most of the time, it will detect when more than t errors have occurred.

F. Error Value Calculation

Forney Algorithm is used for evaluating the error values. The error position and the coefficients of $\omega(x)$ is taken here as input. It is also using the Galois field multiplier as the Chien search algorithm. If the error number is equal to or less than t, $\lambda(x)$ and $\omega(x)$ will be obtained by the above equations 4 and 5 and the error value may be expressed using (6) Forney's formula, as

$$Y_i = \left. \frac{x\ \omega(x)}{\lambda_{odd}(x)} \right|_{x=\alpha^{-j}} \tag{6}$$

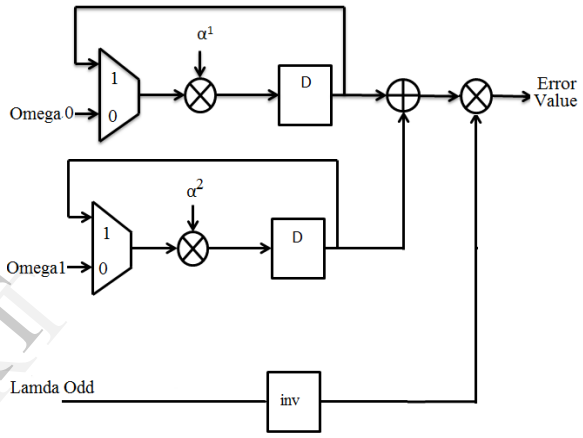

Fig. 7. Architecture for Forney algorithm

G. Error Correction

The output of the chien/forney block is the error vector. This vector is the same size as the codeword. The vector contains non zero values in locations that correspond to errors. Because the error vector is generated in the reverse order of the received codeword, a FIFO 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 the adder is the decoder's estimate of the original codeword.

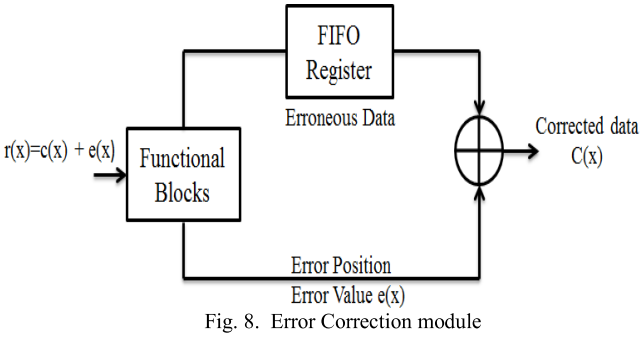

Fig. 8. Error Correction module

III. Simulation Results

The design entry is modelled using VHDL in Xilinx ISE Design Suite 13.2 and the simulation of the design is performed using Modelsim to validate the functionality of the design. MATLAB is used to obtain the AWGN channel output.

Here a (7,3) Reed-Solomon code is designed using MATLAB, for which the number of message symbols is 3 and total number of symbols in the codeword is 7. Each symbol contains 3 bits (m=3) and error correcting capability of the code is 2 symbols (t=2). So we have to add 2t=4 parity symbols. Also the multiplicity assignment for the decoder is done with hard decision and second best decision symbols. With η unreliable points, 2^{η} different test vectors are created.

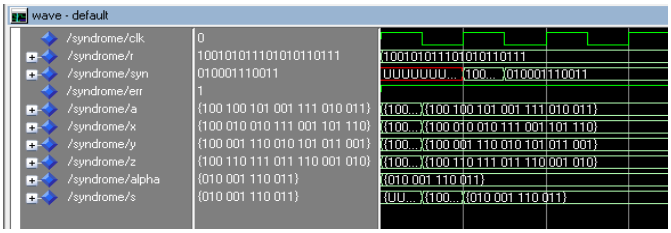

Fig. 9. Syndrome Computation output

Figure 9 shows the output of syndrome computation block. If there is no error in the received codeword 'r' then syndrome value is zero, 'err' signal will be low. Then all other blocks in RS decoder will be idle and the corrected data will be equal to received codeword. If erroneous data is received, then 'err' signal is high and it indicates presence of errors and rest of the blocks will be active.

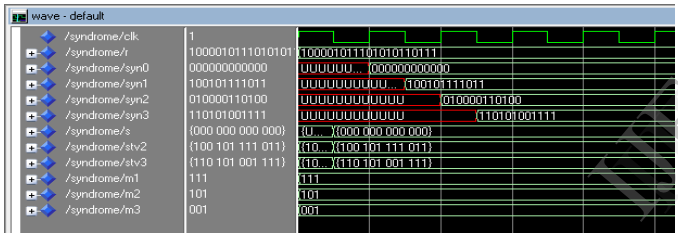

Fig. 10. Syndrome Update output

Figure 10 shows the syndrome update output of all the test vectors. By knowing the multiplicities of the received test vector and their positions, the syndrome computation of η unreliable points thereby all test vectors can be obtained.

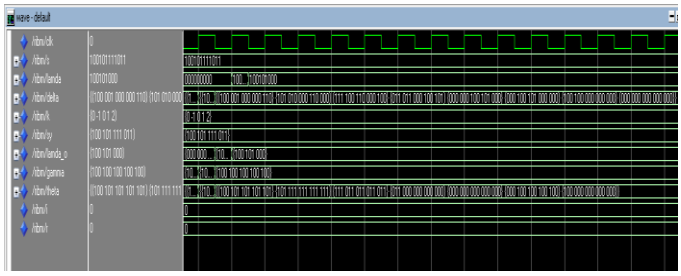

Fig. 11 RiBM output

Figure 11 shows the simulation result of RiBM block. 's' is the syndrome input. 'lamda' gives the coefficients of error locator polynomial obtained from the 'delta' array. RiBM algorithm computes the error locator polynomial in 2t=4 clock cycles.

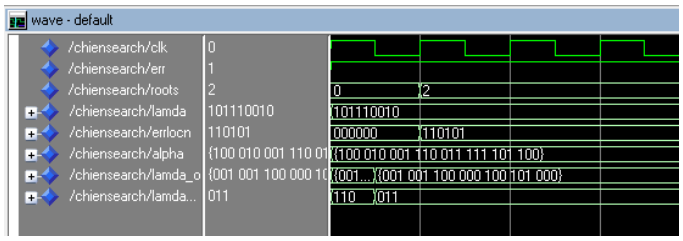

Fig. 12 Chien Search output

Figure 12 shows the simulation result of Chien Search for error location calculation. From 'lamda', the error location of the points is obtained. Also the number of roots is obtained for checking the decoder failure detection.

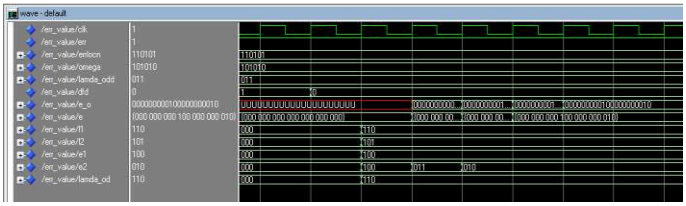

Fig. 13 Forney output

Figure 13 shows the simulation result of Forney Algorithm for error value calculation. From 'lamda_odd' and 'omega', the error values are obtained from the points of chien search.

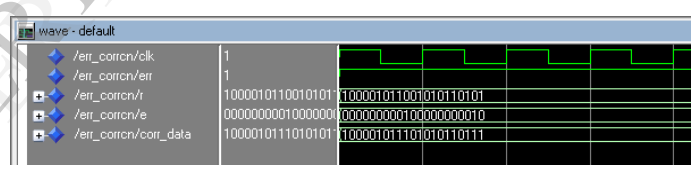

Fig. 14 Error Correction output

Figure 14 shows the simulation result of error correction that can be done by XORing the received codeword with error polynomial obtained from Chien and Forney blocks.

IV. Conclusions

The increasing demand for the efficient and reliable digital data transmission and storage systems has led to the widespread use of Reed Solomon Forward Error Correction code. The RS encoded data after passing through the AWGN channel is reached at the decoder with different multiplicities. With unified syndrome computation architecture, syndrome computation of all the test vectors is obtained in order to select the correct test vector in case of a decoding failure. Also the error polynomial is generated by solving KES to locate the error and its magnitude for error correction.

References

[1] Wei Zhang, Hao Wang, Boyang Pan,"Reduced Complexity LCC Reed Solomon Decoder based on Unified Syndrome Computation", *Proceedings of the IEEE transactions on Very Large Scale Integration (VLSI) Systems*, Vol. 21, pp.974-978, 2013.

[2] Jaydeb Bhaumik, Sundar Das, Satyajit "Design of RS (255,251) Encoder and Decoder on FPGA",*International Journal of Soft Computing and Engineering*, volume 2, pp.2231-2307, 2013

[3] Xinmiao Zhang and Jiangli Zhu "Hardware complexities of algebraic soft decision reed solomon decoders and comparisons ," *in circuits and systems (APCCAS) 2012 IEEE Asia pacific conference* pp. 216-219, 2012

[4] F. Garcia-Herrero, J. Valls, and P. K. Meher, "High speed RS(255, 239) decoder based on LCC decoding,"*Circuits Syst. Signal Process.*,vol. 30, no. 6, pp. 1643–1669, 2011.

[5] Chang Limin, Song Lu, Duan Fengyang, "Research and Realization of RS Codec based on FPGA Technique*", Proceedings of the IEEE International Conference on Mechatronics and Automation*, pp.481-486, 2009

[6] Xinmiao Zhang, "High Speed VLSI architecture for low complexity Chase Soft- decision Reed Solomon Decoding", *Information Theory and application workshop*, 2009

[7] Shu Lin, Daniel J. Costello, "Error Control Coding: Fundamentals and Applications", Second Edition, Prentice-Hall, 2005

[8] Arshad Ahmed, Ralf Koetter, Naresh R. Shanbhag, "VLSI architectures for soft decision decoding of Reed Solomon codes ", *Proceedings of IEEE international Conference on Communications,* Vol. 5,pp.2584-2590, 2004

[9] Bernard Sklar, "*Digital Communications: Fundamentals and Applications*", Second Edition, Prentice-Hall, 2001