Analysis of Reed-Solomon Codes: Application to Digital Video Broadcasting Systems

Teodor B. Iliev[1] and Georgi V. Hristov[2]

***Abstract* – Accuracy of information in any communication system is very critical. Use of Forward Error Correction (FEC) to lower the probability of error and increase transmission distance has become widespread. Reed-Solomon is a block FEC, capable of correcting multiple errors, specifically focusing on burst errors, making it popular for storage devices, wireless and mobile communication units. In this paper we have presented, discussed and analyzed the Reed Solomon encoder and decoder structures implemented in the Digital Video Broadcasting systems.**

***Keywords* – DVB, Reed – Solomon codes, FEC**

I. INTRODUCTION

Error correcting codes (channel coding) are one of the solutions available to improve the digital communication quality. The purpose of channel coding is to introduce, in a controlled manner, some redundancy in the binary information sequence to overcome the effects of noise and interference encountered during the transmission through the channel. Reed-Solomon codes are block error correction codes with burst error-correcting capabilities that have found widespread use in storage devices and digital communication systems [1, 8]. In particular, concatenated coding employing an inner convolutional code combined with a Reed-Solomon outer code constitutes an attractive scheme that is commonly encountered in many applications, and in particular in the digital video broadcasting systems (DVB) [9, 10].

RS codes are Maximum Distance Separable (MDS), known to be the most powerful linear codes for their class, and have the ability to correct both errors and erasures, defined as errors with identified error locations.

For a typical channel, the addition of RS coding allows the system to operate within approximately 4 dB of the Shannon capacity. The resulting benefit translates into higher data rates, lower bit-error rates (BER), greater transmission distance, and greater immunity to interference effects [2].

II. PROPERTIES OF REED – SOLOMON ENCODER

Reed Solomon encoding is a block encoding scheme and partially is specified as an RS (n, k), where n refers to the output codeword length and k refers to the input word length, as shown in Fig. 1. The RS code is based on the Galois field GF (2^8), and therefore has a symbol size of 8 bits. The difference $n–k=2t$ is the number of parity symbols have been appended to make the encoded block. An RS decoder can correct up to $(n–k)/2$ or t symbols, i.e. any t symbols can be corrupted in any way and the original symbols can be recovered. The RS (255, 239) was chosen which processes a data block of 239 symbols and can correct up to 8 symbol errors by calculating 16 redundant correction symbols. As an MPEG-2 packet is 188 bytes long, the code was shortened, i.e. the first 51 information bytes were set to zero and not transmitted at all. In this way the RS (204, 188) is generated [4]. Thus the DVB code splits the message into blocks 188 symbols long. The parity symbols ($2t$=204 – 188=16) are then appended to produce the full 204 symbol long code. Up to t=16/2=8 symbol errors can be then corrected.

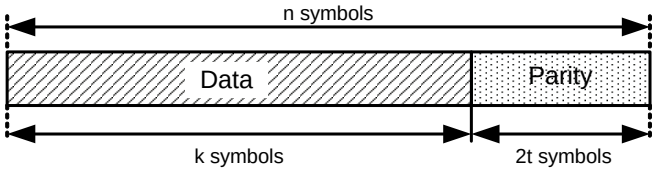

Fig.1 Code word of RS (n,k) code

The encoder forms a code word $x^{n-k}m(x) + r(x)$ by means of the following equation:

$$\frac{x^{n-k}m(x)}{g(x)} = q(x) + \frac{r(x)}{g(x)}, \qquad (1)$$

where the divisor, $g(x)$ is known as the generator polynomial. It is a polynomial of degree (n-k) and which is a factor of (x^n+1). To maximize the minimum distance between codes, the roots of this polynomial should all be consecutive. This is a direct consequence of the BCH bound, which states that the minimum distance is always larger than the number of consecutive factors of $g(x)$. The system used adapted a generator polynomial with roots from α^1 to α^{32}.

The term x^{n-k} is a constant power of x, which is simply a shift upwards n-k places of all the polynomial coefficients in $m(x)$. It happens as part of the shifting process in the architecture below. The remainder after the division $r(x)$ becomes the parity. By concatenating the parity symbols on to the end of the k message symbols, an n coefficient polynomial is created which is exactly divisible by $g(x)$.

Eq. (1) also implies that the generator polynomial is a factor of all possible code words.

The symbols in Reed Solomon coding are elements of a Galois Field (finite field). Encoding is achieved by appending the remainder of a Galois field polynomial division into the

[1]Teodor B. Iliev is with the Department of Communication Systems and Technologies, 8 Studentska Str., 7017 Ruse, Bulgaria, E-mail: tiliev@ecs.uni-ruse.bg

[2]Georgi V. Hristov is with the Department of Communication Systems and Technologies, 8 Studentska Str., 7017 Ruse, Bulgaria, E-mail: ghristov@uni-ruse.bg

message. This division is done by a Linear Feedback Shift Register (LFSR) implementation. The RS encoder with internal feedback connection corresponding to $g(x)$ is shown on Fig. 2.

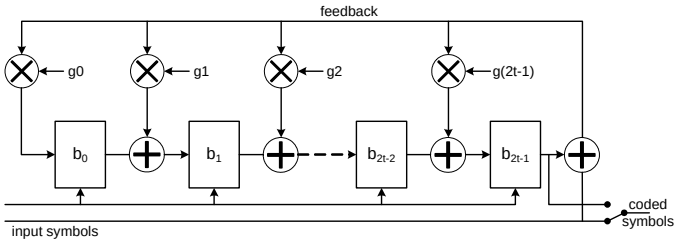

Fig.2 RS encoder with LFSR

The encoder shown on Fig. 2 is a 2t tap shift register, where each register is m bits wide. The multiplier coefficients $g0$ to $g(2t\text{-}1)$ are coefficients of the RS generator polynomial. The coefficients are fixed, which can be used to simplify the multipliers if required. The only hard bit is working out the coefficients, and for hardware implementations the values can often be hard coded.

At the beginning of a block all the registers are set to zero. From then on, at each clock cycle the symbol in each register is added to the product of the feedback symbol and the fixed coefficient for that tap, and passed on to the next register. The symbol in the last register becomes the feedback value on the next cycle. When all n input symbols have been read in, the parity symbols are sitting in the register, and it just remains to shift them out one by one.

III. Reed – Solomon Decoder

Decoding RS codes involves the extraction of two information entities from the received word. These are the set of Partial Syndromes, and the Error Locator Polynomial. From these two parameters, Error Locations and Error magnitudes are extracted, and are directly applied upon the received word to extract the original, corrected message [6].

A Generator Polynomial defines each RS code configuration. The specific configurations of the encoder and decoder are determined by this polynomial. In the decoder, the partial syndromes are computed by treating the received information as a polynomial and evaluating it at each of the roots of the Generator Polynomial. This operation is realized in hardware as a set of shift-register assemblies connected in parallel. The Partial Syndromes are computed simultaneously and are passed on afterwards to the Error Locator Polynomial unit. The Error Locator Polynomial is computed directly from the Partial Syndromes and a popular algorithm used to extract this is the Berlekamp-Massey Algorithm. The magnitudes of the corresponding errors are solved using the Forney Algorithm. An error-correcting block uses the computed error magnitudes to adjust the received codeword [5].

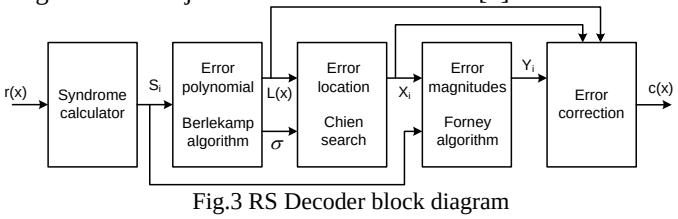

Fig.3 RS Decoder block diagram

The RS decoder consists of five major blocks as shown in Fig. 3.

The first step in decoding the received symbol is to determine the data syndrome. A codeword's syndrome $s(x)$ is the remainder of the division of the received word $r(x)$ by the generator polynomial, as implied by the following equation:

$$\frac{r(x)}{g(x)} = q(x) + \frac{s(x)}{g(x)}. \tag{2}$$

The received word, as seen in Eq. 3, can be decomposed into the code word $c(x)$ and the error component $e(x)$. Further manipulation of Eq. 2 leads to Eq. 5.

$$r(x) = c(x) + e(x) \tag{3}$$

$$\frac{c(x)+e(x)}{g(x)} = q(x) + \frac{s(x)}{g(x)} \tag{4}$$

Since all code words are divisible by the generator polynomial, only the error component will yield a remainder. Eq. 5 and 6 illustrates this:

$$\frac{c(x)}{g(x)} + \frac{e(x)}{g(x)} = q_c(x) + q_e(x) + \frac{s(x)}{g(x)} \tag{5}$$

$$\frac{e(x)}{g(x)} = q_e(x) + \frac{s(x)}{g(x)} \tag{6}$$

It can now be seen in Eq. 6 that the syndrome is independent of the message information and depends only on the error component.

In most systems, partial syndromes are computed instead of the syndrome, for reasons of simpler hardware implementation. In the computation of a partial syndrome, the divisor is no longer the entire generator polynomial, but only one of its factors, as seen in Eq. 7. There will be n-k partial syndromes for every received word, since the generator polynomial has n-k factors.

$$\frac{r(x)}{(x-a_k)} = q(x) + \frac{s_k}{(x-a_k)} \tag{7}$$

Note that the remainder s_k in Eq. 7 can be obtained by evaluating $r(x)$ at a_k, as:

$$s_k = r(a_k) \tag{8}$$

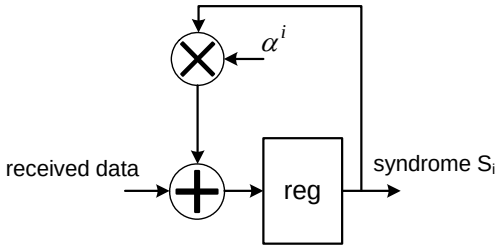

Fig.4 Partial Syndrome Calculator

Syndrome calculation can be done by an iterative process, such that the answer (2t syndrome symbols) is available as

soon as the last parity symbol has been read in. The circuit shown on Fig. 4 will generate the i^{th} syndrome, $2t$ of these will be needed for the full syndrome decoder. The syndromes depend only on the errors, not on the underlying encoded data.

A. Error Polynomial lambda – Berlecamp – Massey and Euclids algorithm

In Reed-Solomon decoding, two items of information are required to extract the original message: the error locations and the error magnitudes.

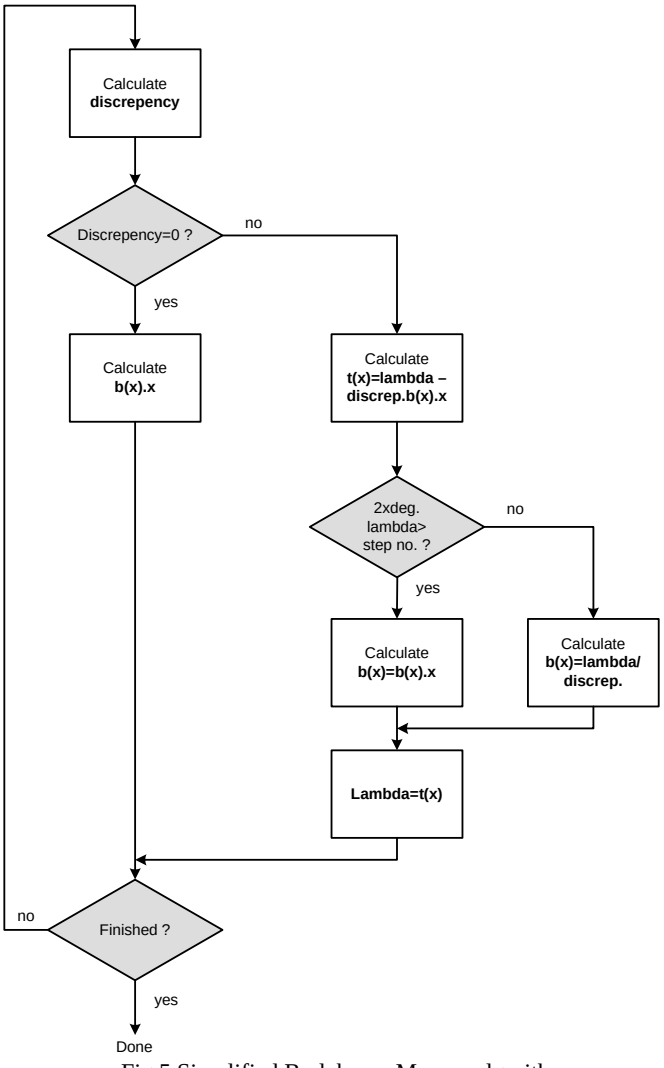

Fig.5 Simplified Berlekamp Massey algorithm

The Error Locator Polynomial is a polynomial whose roots directly define the error locations in the received word. This polynomial could be computed directly from the partial syndromes [7]. A number of methods exist to perform this computation, and the method that is most popular and most suited to this system is the Berlekamp-Massey Algorithm.

The Berlekamp-Massey algorithm is a shift-register synthesis algorithm [8] and the simplified block scheme of the algorithm is show on Fig.5. It takes as input the n-k partial syndromes and outputs the error locator polynomial $\sigma(x)$.

The algorithm aims to find an LFSR of minimal length such that the first (n-k) elements in the LFSR output sequence are the (n-k) syndromes. The taps of this shift register are the coefficients of the desired error locator polynomial, $\sigma(x)$ [8].

The algorithm iteratively solves the error locator polynomial by solving one equation after another and updating the error locator polynomial. If it turns out that it cannot solve the equation at some step, then it computes the error and weights it, increases the size of the error polynomial, and does another iteration. A maximum of $2t$ iterations are required. For n symbol errors, the algorithm gives a polynomial with n coefficients. At this point the decoder fails if there are more than t errors, and no corrections can be made. Doing so might actually introduce more errors than there were originally.

B. Error Polynomial Roots

After the Error Locator Polynomial has been computed, the roots of this polynomial have to be calculated in order to know the error locations. This is done by performing the Chien Search, which evaluates the Error Locator Polynomial at all elements of the GF(256) field. The algorithm checks if $\sigma(a^p)$ equals zero, p = 0, 1, 2 …., n, then a^p is a root of the polynomial, and α^{-p} is an error location, X_p.

Solving the key equation (Eq. 7) [7] determines the error evaluator or error magnitude polynomial, $\Omega(x)$:

$$S(x)\sigma(x) = \Omega(x) \bmod x^{n-k} \tag{9}$$

C. Error Magnitudes - Forney Algorithm

An efficient way of computing $\Omega(x)$ is to perform parallel computation of $\sigma(x)$ [9]. Solution to the key equation may be derived as follows:

$$\begin{aligned}\Omega(x) &= S(x)\sigma(x) \bmod x^{2t} = \\ &= \left(S_1 + S_2 x + \ldots + S_{2t} x^{2t-1}\right) \bullet \left(\sigma_1 + \sigma_2 x + \ldots + \sigma_t x^t\right) \bmod x^{2t} = \\ &= \Omega^{(0)} + \Omega^{(1)} x + \ldots + \Omega^{(t-1)} x^{t-1}\end{aligned} \tag{10}$$

$$\Omega^{(i)} = S_{i+1}\sigma_0 + S_i\sigma_1 + \ldots + S_1\sigma_i, \quad for\ i = 0,1,\ldots,t-1 \tag{11}$$

This leads to a direct computation of $\Omega(x)$, which requires fewer multiplications than the iterative algorithm [3].

The Forney Algorithm is used to compute for the error magnitudes, Y_i, corresponding to the respective error locations. Using the following equation:

$$Y_i = \frac{\Omega\left(X_i^{-1}\right)}{\sigma'\left(X_i^{-1}\right)}, \tag{12}$$

where X_i^{-1} indicates the root as computed from the Chien Search, and $\sigma'(x)$ the derivative of the error locator polynomial.

D. Error Correct

The error corrector block takes the received code and performs XOR-operation with the corresponding error

magnitudes computed at the respective error locations to attain the original message stream (Eq. 13).

$$c(X_i) = r(X_i) \oplus Y_i \tag{13}$$

IV. Simulation

We have presented the results of conducted simulation of Reed Solomon codes with following n and m parameters in according to DVB standard – RS (255, 239), RS (255, 235), RS (255, 223), RS (255, 205). The parameters setting of convolution code, rate $R=m/n$ selection in according to DVB standard – 1/2, 2/3, 3/4, 5/6, 7/8 and interleaver depth I settings – up to 20 [4].

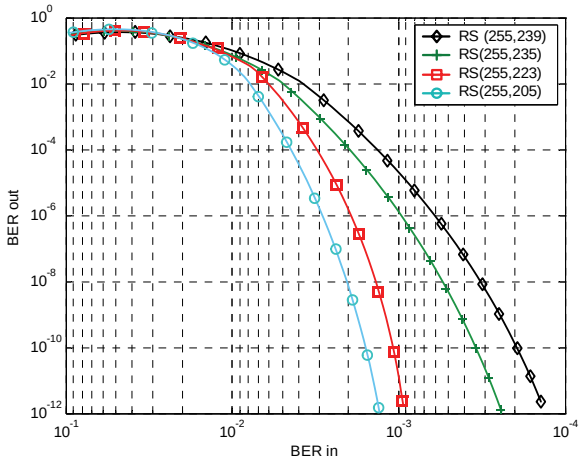

Fig. 6. Residual Bit – Error Rate value of various RS codes

Independently of whether the encoding and decoding is in frequency or time domain, the efficiency of the RS code is the same. The residual bit-error rate (BER) of various RS codes based on GF (2^8) is shown in Figure 6. Although the RS codes are symbol-oriented codes, the analysis of the efficiency takes bit errors into account. The efficiency of the code increases with an increase in the number of test symbols. At an input bit-error rate of 2.10^{-3} the residual bit error rate of the RS (255, 205) code is approx 1.10^{-10} – the coding gain is thus more than 10 to the power of 7 – whereas in the case of the RS (255, 239) code at the same input bit-error rate the output bit-error rate is 9.10^{-4} – the coding gain is only slightly greater than 0.5. For all DVB transmission standards a modified (shortened) RS (255, 239) code is used which makes it possible residual bit-error rate of approx 1.10^{-11} at an input bit--error rate of 2.10^{-4} while correcting up to 8 symbol errors per block.

The residual bit-error rate of convolutional codes of rate R is a function of E_b/N_0 (energy transmitted per bit divided by the noise-power density of the white Gaussian noise) and the parameter K describes the length of the code. The performance of the error correction increases with increased K. For the DVB standard a convolutional code of rate R=1/2 with a constraint length K=7 is used. With E_b/N_0 = 3.2 dB it is possible to achieve a bit-error rate of less than 2.10^{-4} at the output of the decoder, this ratio corresponds to the maximum of the bit-error rate at the input of the RS decoder, so finally a bit-error rate at the output of the RS decoder of less than 1.10^{-11} is obtained.

V. Conclusion

This paper has provided an overall view of the Reed Solomon codes implemented in digital television broadcast standards and we have conducted simulation with various RS codes. From the simulation is evidence that the efficiency of the codes increases with an increase in the number of test symbols.

Reliability of information is critical in any communication systems. Use of error-control codes reduce interference effects, and FECs in general, eliminate the need for retransmission of data streams. The theoretical performance of Reed-Solomon codes in bursty noise channels makes it a very good choice for FEC.

Acknowledgement

This work is a part of the project DMU-02/13-2009 "Design and performance study of an energy-aware multipath routing algorithm for wireless sensor networks" of the Bulgarian Science Fund at the Ministry of Education, Youth and Science.

References

[1] T. Iliev, I. Lokshina, D. Radev, and G. Hristov, "Analysis and Evaluation of Reed–Solomon Codes in Digital Video Broadcasting Systems", – In: Proceedings of Seventh Annual Wireless Telecommunications Symposium WTS 2008, Pomona, CA, USA, (on CD & IEEE Xplore publishing), 2008.

[2] J.-H. Jeng and T.-K. Truong, "On Decoding of Both Errors and Erasures of a Reed–Solomon Code Using an Inverse-Free Berlekamp–Massey Algorithm", IEEE Transaction on Communication, vol. 47, 1488–1494, 1999.

[3] T.C. Lin, T. K. Truong and P. D. Chen, "A Fast Algorithm for the Syndrome Calculation in Algebraic Decoding of Reed–Solomon Codes," IEEE Transaction on Communication, vol. 55, 2240–2244, 2007.

[4] U. Riemers "*Digital Video Broadcasting. The International Standard for Digital Television*". Springer-Verlag Berlin Heildeberger New York, 2001.

[5] R. Roth and G. Ruckenstein, "Efficient decoding of reed-solomon codes beyond half the minimum distance," IEEE Transaction on Information Theory, vol. 46, 246–257, 2000.

[6] W. E. Ryan and P. Conoval, "A Method of Analysis for Interleaved Reed-Solomon Coding with Erasure Decoding on Burst Error Channels", IEEE Transaction on Communication, vol. 41, 430–434, 1993.

[7] T. K. Truong, J. H. Jeng, and I. S. Reed, "Fast algorithm for computing the roots of error locator polynomials up to degree 11 in Reed-Solomon decoders," IEEE Transaction on Communication, vol. 49, no. 5, 779–783, 2001.

[8] S. B. Wicker, "*Error Control Systems for Digital Communication and Storage*". Englewood Cliffs, NJ, 1995.

[9] S. B. Wicker and V. Bhargava, "*Reed Solomon Codes and their Applications"*, NY, IEEE Press (1994).

[10] EN 301192 v1.4.1: Digital Video Broadcasting (DVB); DVB Specification for Data Broadcasting, ETSI, Nov., 2004.