

**APPLICATIONS OF REED-SOLOMON CODES  
ON OPTICAL MEDIA STORAGE**

---

A Thesis  
Presented to the  
Faculty of  
San Diego State University

---

In Partial Fulfillment  
of the Requirements for the Degree  
Master of Arts  
in  
Mathematics

---

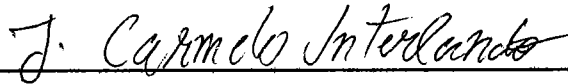
by  
Johnny Phuong Nguyen  
Fall 2011

**SAN DIEGO STATE UNIVERSITY**

The Undersigned Faculty Committee Approves the

Thesis of Johnny Phuong Nguyen:

Applications of Reed-Solomon Codes  
on Optical Media Storage



---

J. Carmelo Interlando, Chair  
Department of Mathematics and Statistics



---

Roxana Smarandache  
Department of Mathematics and Statistics



---

Carl Eckberg  
Department of Computer Science

10/26/11

---

Approval Date

Copyright © 2011  
by  
Johnny Phuong Nguyen

## **DEDICATION**

To my parents and  
my brothers Peter and Joe

## **ABSTRACT OF THE THESIS**

Applications of Reed-Solomon Codes  
on Optical Media Storage

by

Johnny Phuong Nguyen  
Master of Arts in Mathematics  
San Diego State University, 2011

This work describes in some depth the applications of the theory of error-correcting codes, in particular Reed-Solomon codes, for optical media storage, namely, CDs, DVDs and Blu-ray discs. The objective was to collect in a single place the most relevant work in the area, providing the reader with easy access to state-of-the-art technology. We cover the CIRC code of the compact disc followed by the RSPC code of the DVD. Afterwards we discuss the picket code that makes the Blu-ray error correcting system the most powerful error correcting code to date. Lastly we touch upon how to make the Blu-ray picket code even better than the original code.

## TABLE OF CONTENTS

	PAGE
ABSTRACT .....	v
LIST OF TABLES.....	viii
LIST OF FIGURES .....	ix
ACKNOWLEDGEMENTS .....	x
CHAPTER	
1 INTRODUCTION .....	1
1.1 Basics of Coding Theory .....	1
1.2 Reed-Solomon Codes and Their Uses .....	2
1.3 Contribution of This Thesis.....	3
1.4 Overview of Thesis.....	3
2 BACKGROUND .....	4
2.1 Cyclic Codes .....	4
2.2 Errors, Erasures and Burst Errors .....	4
2.3 BCH Codes .....	4
2.4 Reed-Solomon Codes .....	5
2.5 Interleaving .....	6
2.6 Decoding RS Codes with an Inversionless Berlekamp Massey Algorithm....	8
3 AUDIO CD ERROR CORRECTING CODE .....	13
3.1 Development of CDs.....	13
3.2 Cross-Interleaved Reed-Solomon Code .....	13
3.3 The Decoding and Error Correcting Process.....	14
3.4 Interpolation .....	15
4 DVD ERROR CORRECTING CODE .....	16
4.1 The Reed-Solomon Product Code.....	16
4.2 Decoding with the RSPC .....	16
4.3 Comparing Correcting Abilities of the DVD to the CD .....	18
5 BLU-RAY ERROR CORRECTING CODE.....	19
5.1 Design for Blu-Ray Error Correcting Code .....	19

	vii
5.2 The Picket Code .....	19
5.3 Decoding with the Picket Code.....	20
6 CONCLUSION AND FUTURE WORK .....	22
6.1 Comparing the Error Correcting Codes .....	22
6.2 Improving on the Picket Code in the Future .....	22
BIBLIOGRAPHY .....	25

**LIST OF TABLES**

	PAGE
Table 2.1. Interleaving to Depth $j$ .....	7
Table 2.2. $j$ -Frame Delayed Interleaving .....	8



## LIST OF FIGURES

	PAGE
Figure 1.1. Uranus from 600,000 miles away. ....	2
Figure 2.1. An example of the use of the inversionless BM algorithm. ....	12
Figure 4.1. The RSPC code used in DVDs. ....	17
Figure 5.1. Blu-ray picket code. ....	20
Figure 6.1. Illustration of the main drawbacks of the conventional picket code. ....	23

## ACKNOWLEDGEMENTS

First I thank my thesis committee, Professor J. Carmelo Interlando, Professor Roxana Smarandache, and Professor Carl Eckberg for their time.

I thank my family for their love and support throughout the last 2 years. I could not have finished this thesis without their encouragement.

A very special thanks goes to Jennifer Cheung for helping me format this thesis into  $\LaTeX$ . I could not have completed this thesis without her.

I like to thank Sony for being the only corporation to email me back in response to my inquiries.

And lastly a thank you goes to my girlfriend Nancy for her reassuring me that I can complete this thesis.

# CHAPTER 1

## INTRODUCTION

In Chapter 1, I will go over some basic ideas and processes of coding theory. Some uses of Reed-Solomon codes will also be briefly covered as well as the contribution and overview of this thesis.

### 1.1 BASICS OF CODING THEORY

Informally speaking, coding theory is about devising efficient and reliable ways of transferring information from one place to another, in the presence of noise. A source sends information to the encoder which in turn sends information through a channel. A channel is the medium over which the information is transmitted or recorded, such as telephone lines and compact discs. The receiver then gets the information from the channel and decodes it for the user. Noise in the channel may cause information to be received incorrectly. Noise is unpredictable, however it may be modeled according to some probability distribution. As examples of noise, we can cite: Scratches on a disc, lightning, solar radiation, competing transmitter, and so forth. Observe that without noise, there would be no need for the theory of error-correcting codes.

Modern communication and storage systems that work with digitally represented data, such as CD players, TV, fax machines, and mobile phones require error correcting codes because in practice nearly all channels are noisy.

More specifically, all information (be it voice, video, pictures, data) is represented by a long string of zeroes and ones, or bits. The encoder breaks down that string into information blocks of fixed length, say,  $k$  bits. Each block is then *encoded* according to a certain pre-specified rule (mapping), resulting in a codeword having  $n > k$  bits. Mathematically speaking, encoding is an injective mapping from  $\{0, 1\}^k$  to  $\{0, 1\}^n$ .

Each codeword in turn is transmitted over the noisy channel. Upon receiving a noisy version of the sent codeword, the decoder must convert it to the most likely sent codeword. Mathematically speaking, decoding is a mapping from  $\{0, 1\}^n$  to  $\{0, 1\}^k$ .

A binary code  $C$  of length  $n$  is a subset of  $0, 1^n$ . Its elements are called codewords. The information rate of  $C$  is defined as  $\frac{\log_2 |C|}{n}$  bits per block. The distance between any two codewords is the number of coordinates in which they disagree. The minimum distance of  $C$  is the minimum of the distances between any two distinct codewords.

One of the main challenges in coding theory is to construct codes with high rates and minimum distances. The rate of a code is related to its efficiency for transmitting information whereas its minimum distance is related to its capability of correcting errors (i.e., its capacity of providing reliability). Equally important challenges are to find codes whose encoding and decoding processes are of low complexity. Informally speaking, a *powerful code* is one that possesses a high rate and minimum distance, and can be easily implemented (i.e., the encoding and decoding procedures are of low complexity).

Among the large class of the so-called *linear codes*, one can find codes that present a good compromise when it comes to satisfying the above criteria. Reed-Solomon codes are nonbinary linear codes (over the field  $GF(2^n)$ ) and they are the main object of this work.

## 1.2 REED-SOLOMON CODES AND THEIR USES

Reed-Solomon codes are very powerful error correcting codes. They were developed by Irving Reed and Gus Solomon in their paper about these codes was published in June 1960. Their paper was entitled “Polynomial Codes over Certain Finite Fields”.

Reed-Solomon codes have applications from spacecraft in our solar system to compact discs, DVDs and Blu-ray discs [16].

The first ever use is from the Voyager 2 spacecraft that was launched in 1977. On April 5, 1985, the Voyager 2 captured the first image of Uranus from 200 million miles away. This is the first ever RS encoded image ever transmitted from deep space [19]. On January 24, 1986, Voyager 2 transmitted more detailed photos and other data about Uranus as it came about 50,000 miles away from the planet [18]. Figure 1.1 [18] is a picture of Uranus from about 600,000 miles.



**Figure 1.1. Uranus from 600,000 miles away.**

Reed-Solomon codes are used when the receiver needs to send information back to its transmitter. This sort of application includes but not limited to mobile data transmission systems, i.e. our cell phones, and military communication systems. RS codes also play a role in developing high-speed super computers [19].

Many of the useful of RS codes is used in channels that have a significant amount of noise. The use of fiber optics is increasing and RS codes could be used there. As stated in [19] Reed-Solomon codes are best used in mobile environments as there is no other error control system that can match the performance.

Perhaps the most well known application of Reed-Solomon codes is in optical disc storage. Not many people can say that they've received pictures from outer space but most people can say that they've listened to a compact disc (CD) or watched a movie on a digital versatile disc (DVD) or Blu-ray disc (BD). Blu-ray discs are named after the blue-violet laser that is used to read them as opposed to the red laser that reads CDs and DVDs. In 2010, DVD and Blu-ray sales totaled \$6.77 billion with an estimated 226 million units sold in the US alone [7]. There were about 326.2 million CDs sold in the US in the year 2010 [3]. Combined that's more than half a billion CDs, DVDs and Blu-ray discs sold in the last year in the US alone!

### **1.3 CONTRIBUTION OF THIS THESIS**

The main purpose was to collect, in a single work, the main applications of Reed-Solomon codes that are usually not covered in depth in the coding theory classes Math-525 and Math-625. The idea is that the reader or student can have prompt access to the three main applications of Reed-Solomon codes. One other topic that is usually not covered is the inversionless Berlekamp-Massey algorithm, which is widely used to decode RS codes, especially in the above-mentioned applications. The algorithm is described and an example is worked out.

### **1.4 OVERVIEW OF THESIS**

Chapter 2 will introduce some background information about Reed-Solomon codes and how they are formed. Interleaving and the inversionless Berlekamp-Massey algorithm will be covered as well. Chapter 3 will focus on the cross interleaved Reed-Solomon code that the CD employs to correct errors. A discussion of how the CD uses another method to further its error correcting capabilities is also discussed. Chapter 4 will cover the Reed-Solomon product code for usage in DVDs. In Chapter 5 the innovative picket code will be discussed. The picket code is the powerful error correcting system used in Blu-ray discs and is considered to be the best error correcting code used in optical disc media storage today. Chapter 6 will be the conclusion. It will also briefly cover some future applications of Reed-Solomon codes, mainly two methods of improving the Blu-ray picket code.

## CHAPTER 2

### BACKGROUND

In this chapter I will cover the necessary background from cyclic codes and Reed-Solomon codes to the ever important inversionless Berlekamp-Massey algorithm that DVDs and Blu-ray discs use to decode their respective Reed-Solomon codes.

#### 2.1 CYCLIC CODES

Let  $C$  be a subset of  $GF(q)^n = GF(q) \times GF(q) \times \cdots \times GF(q)$ ,  $n$  times where  $q$  is a prime power and  $n$  is a positive integer. Let  $S$  be the set of all polynomials of degree less than  $n$  with coefficients in  $GF(q)$ .  $C$  is a cyclic code of length  $n$  if  $C$  is a subset of  $S$  and  $v(x) \in C \implies x \times v(x) \bmod x^n - 1 \in C$ . A codeword in  $c \in C$  will be represented as either  $(c_0, c_1, \dots, c_{n-1})$  or  $c(x) = c_0 + c_1x + \cdots + c_{n-1}x^{n-1}$  which is its polynomial form.

#### 2.2 ERRORS, ERASURES AND BURST ERRORS

When the magnitude and location is known for corrupted bits, it is just called an error. An erasure is when the error location number is known but the magnitude of the error is unknown. A burst error is an error that affects many digits that are close together as opposed to randomly distributed errors that affect only one digit and not those around it [8].

#### 2.3 BCH CODES

BCH codes, invented by Bose, Chaudhuri, and Hocquenghem, are cyclic  $q$ -ary codes of length  $q^m - 1$  where  $q$  is a prime power and  $m$  is an integer  $> 0$ . A  $q$ -ary  $(n, k)$  cyclic code, where  $n$  is the length and  $k$  is the dimension, has generator polynomial of degree  $n - k$  over a Galois field,  $GF(q)$ . If a code  $C$  has length  $n = q^m - 1$  then  $C$  is a primitive code. The generator polynomial is defined as:

$$g(x) = g_0 + g_1x + g_2x^2 + \cdots + g_{n-k-1}x^{n-k-1} + x^{n-k}$$

where  $g_0 \neq 0$ ,  $g_{n-k} = 1$  and  $g_i \in GF(q)$ ,  $0 \leq i \leq n - k$ . The generator polynomial is a factor of  $x^n - 1$ .

For any positive integer  $m$ , a Galois field  $GF(q^m)$  with  $q^m$  elements can be constructed from  $GF(q)$ . The construction of  $GF(q^m)$  is based on a monic primitive polynomial  $p(x)$  of degree  $m$  over  $GF(q)$ . A polynomial  $p(x)$  is monic if the coefficient of the highest power of  $x$  is 1. Let  $\alpha$  be a root of  $p(x)$  then,

$$0, 1, \alpha, \alpha^2, \dots, \alpha^{q^m-2}$$

are all the elements of  $GF(q^m)$  and  $\alpha^{q^m-1} = 1$  [6]. An element  $\alpha \in GF(q^m)$  is *primitive* if every non-zero word in  $GF(q^m)$  can be expressed as a power of  $\alpha$ . In other words,  $\alpha^j \neq 1$  for  $1 \leq j < q^m - 1$  [8]. Every element  $\beta$  can be expressed as a polynomial in  $\alpha$ ,

$$\beta = \alpha_0 + \alpha_1\alpha + \alpha_2\alpha^2 + \dots + \alpha_{m-1}\alpha^{m-1}$$

where  $\alpha_j \in GF(q)$  for  $0 \leq j < m$ . In this way, every element in  $GF(q^m)$  has 3 forms of representation: power ( $\beta^0 = 1, \beta^1, \beta^2, \dots$ ), polynomial as seen from above, and vector form. Take for example  $\beta = \alpha_0 + \alpha_1\alpha + \alpha_2\alpha^2 + \dots + \alpha_{m-1}\alpha^{m-1}$  as previously mentioned can be represented as  $(\alpha_0, \alpha_1, \dots, \alpha_m)$  [6].

Let  $\alpha$  be a primitive element in  $GF(q^m)$ . The generator polynomial  $g(x)$  of some  $t$ -error correcting primitive  $q$ -ary BCH code is the polynomial of smallest degree over  $GF(q)$  that has  $\alpha, \alpha^2, \dots, \alpha^{2t}$  for roots. Let  $m_i(x)$  be the minimal polynomial of  $\alpha^i$  where  $1 \leq i \leq 2t$ . The generator polynomial is now defined as:

$$g(x) = \text{lcm}(m_1(x), m_2(x), \dots, m_{2t}(x)).$$

This  $q$ -ary BCH code with generator polynomial  $g(x)$  as described above has the following properties:

1. The block length is  $n = q^m - 1$ .
2. The number of parity check symbols is  $n - k \leq 2mt$ .
3. The minimum distance of the code  $d \geq 2t + 1$ .

## 2.4 REED-SOLOMON CODES

Reed-Solomon (RS) codes form a special subclass of the  $q$ -ary BCH codes. RS codes are BCH codes with  $m = 1$ . In the scope of this paper,  $q$  will always equal  $2^r$  for some integer  $r$ . Take process just described above and apply it when  $m = 1$ . Since  $\alpha^i$  is an element in  $GF(q)$ , its minimal polynomial  $m_1(x)$  is just  $x - \alpha$  where  $i \in \{1 \dots 2t\}$ . Therefore  $\text{lcm}(m_1(x), m_2(x), \dots, m_{2t}(x))$  is:

$$g(x) = (x + \alpha)(x + \alpha^2) \dots (x + \alpha^{2t}).$$

Thus,  $g(x)$  is the generator polynomial of some RS code.

A  $t$ -error correcting RS code has the following properties:

1. The block length is  $n = q - 1$ .
2. The number of parity check symbols is  $n - k = 2t$ .
3. The minimum distance of the code  $d = 2t + 1$ .
4. The dimension  $k = q - 1 - 2t$ .

5. The number of codewords  $|C| = q^k$ .

Example: Take the 2-error correcting RS code with symbols from  $GF(2^3)$  constructed with  $x^3 + x + 1$ . Here  $q = 2^3$ . The generating polynomial is:

$$g(x) = (x + \alpha)(x + \alpha^2)(x + \alpha^3) = \alpha^3 + \alpha x + \alpha^5 x^2 + \alpha^3 x^3 + x^4.$$

This is the (7, 3, 5) RS code, with  $n = 7, k = 3, d = 5$ . The number of parity check symbols is 4 and the number of codewords is  $q^k = (2^3)^3 = 512$ . Here is the construction of  $GF(2^3)$  using the primitive polynomial  $f(x) = x^3 + x + 1$ . We compute  $x^i \bmod f(x)$  to get:

<i>word</i>	$\leftrightarrow x^i \bmod f(x)$
100	1
010	$x$
001	$x^2$
110	$x^3 \equiv 1 + x$
011	$x^4 \equiv x + x^2$
111	$x^5 \equiv 1 + x + x^2$
101	$x^6 \equiv 1 + x^2$

In addition, a Reed-Solomon code can correct  $t$  errors and  $v$  erasures if:

$$2t + v < q - k + 1$$

Also, let  $C$  be a code of distance  $d$ . Then  $C$  can simultaneously correct  $t$  errors and  $e$  erasures provided that  $d - 1 \geq 2t + e$ , see[6].

## 2.5 INTERLEAVING

Burst errors are errors that occur very close to each other i.e. a scratch on a disc will produce burst errors. Interleaving is one technique to improve the burst error correcting capability of a code. It involves rearranging the order in which the digits of a code are transmitted. Messages are encoded into their corresponding codewords and then those codewords are transmitted. Suppose now that the first  $j$  codewords are selected and the first digits of those codewords are sent, and then the second digits are sent, then the third, etc. Once all the  $n_j$  digits from the first  $j$  codewords have been sent, then the proceeding set of  $j$  codewords is sent, followed by the third set, and so on. This process is called interleaving to depth  $j$  [8].

In order to interleave the codewords  $c_1, c_2, \dots$  to depth  $j$ , it is necessary that for  $i = 0, 1, 2, \dots$  the digits in each codeword are sent in a certain order. In Table 2.1 [8], each codeword is listed row by row but the digits are transmitted column by column.



**Table 2.1. Interleaving to Depth  $j$** 


---

$c_{ij+1,1}$	$c_{ij+1,2}$	$c_{ij+1,3}$	$\dots$	$c_{ij+1,n}$
$c_{ij+2,1}$	$c_{ij+2,2}$	$c_{ij+2,3}$	$\dots$	$c_{ij+2,n}$
$\vdots$	$\vdots$	$\vdots$		$\vdots$
$c_{ij+j,1}$	$c_{ij+j,2}$	$c_{ij+j,3}$	$\dots$	$c_{ij+j,n}$

---

Here is a small example. Suppose that codewords

$$\begin{aligned} c_1 &= 100110 & c_4 &= 110011 \\ c_2 &= 010101 & c_5 &= 101001 \\ c_3 &= 001011 & c_6 &= 111000 \end{aligned}$$

are sent. If they are not interleaved, then they would be sent in order, i.e.  $c_1, c_2, \dots, c_6$ . But let's say that these codewords are interleaved to depth 3. Then the first digits of the first three codewords,  $c_1, c_2$ , and  $c_3$ , are transmitted first, followed by their second digits and so on. The digits from  $c_1, c_2$ , and  $c_3$  would be sent in this order:

$$100\ 010\ 001\ 110\ 101\ 011.$$

And the same would go for the remaining three codewords  $c_4, c_5$ , and  $c_6$ . So what kind of effect does interleaving to depth  $j$  have on the burst error correcting ability of a code? If the first digit of a codeword is the  $i^{\text{th}}$  digit sent, then its remaining digits will be in positions  $i + j, i + 2j, \dots, i + (n - 1)j$  where  $n$  is the length of each codeword. Suppose that  $C$  is a  $b$  burst error correcting code. If  $C$  is interleaved to depth  $j$  then any burst of errors of length at most  $bj$  will only make a burst error length of  $b$  in a codeword  $c$ , therefore  $c$  will be decoded correctly. However, this is only true if only one burst error pattern affects  $c$  [8].

In practice, interleaving to depth  $j$  is not used because codewords need to be encoded first before any of them are sent. This problem can be solved by doing  $j$ -frame delayed interleaving. Table 2.2 [8] below illustrates how  $j$ -frame delayed interleaving works. Each codeword  $c_i$  has only one digit  $c_{i,k}$  in row  $k$  for  $0 \leq i \leq n$  and  $c_{i,k+1}$  is one row below and  $j$  columns across from  $c_{i,k}$  for  $1 \leq j \leq n - 1$ .

Of course there must be  $n$  digits in each column so zeros are placed in any position that has no digits. Here is an example of using 1-frame delayed interleaving. Let us take six

**Table 2.2.  $j$ -Frame Delayed Interleaving**


---

$c_{1,1}$	$c_{2,1}$	$\dots$	$c_{j+1,1}$	$c_{j+2,1}$	$\dots$	$c_{2j+1,1}$	$\dots$	$c_{(n-1)j+1,1}$	$\dots$	
	$c_{1,2}$	$c_{2,2}$	$\dots$	$c_{j+1,2}$	$c_{j+2,2}$	$\dots$	$c_{2j+1,2}$	$\dots$	$c_{(n-2)j+1,2}$	$\dots$
				$c_{1,3}$	$c_{2,3}$	$c_{3,3}$	$\dots$	$c_{(n-3)j+1,3}$	$\dots$	
								$\vdots$		
								$c_{1,n}$	$\dots$	

---

codewords and substitute them into Table 2.2 which will look like this:

1	0	0	1	1	1	...					
<b>0</b>	0	1	0	1	0	1	...				
<b>0</b>	<b>0</b>	0	0	1	0	1	1	...			
<b>0</b>	<b>0</b>	<b>0</b>	1	1	0	0	0	0	...		
<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	1	0	1	1	0	0	...	
<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	0	1	1	1	1	0	...

The **bolded** zeros are the zeros that are added in to take place of any position that does not have any codeword digits. So here the string of digits sent is:

1000000000001010001001....

The advantage of using  $j$ -frame delayed interleaving is that a  $b$  burst error correcting code  $C$  will correct all burst errors of length  $b(jn + 1)$ . That is of course if at most one burst of errors affects each codeword [8].

## 2.6 DECODING RS CODES WITH AN INVERSIONLESS BERLEKAMP MASSEY ALGORITHM

Here the inversionless Berlekamp-Massey algorithm will be detailed. There is another algorithm that can be used to decode RS codes called the Euclidean algorithm. However it is more efficient for a computer to use the inversionless Berlekamp-Massey algorithm. The two algorithms are described in [10].

Let  $n$  be the block length of an  $(n, k)$  Reed-Solomon code over  $GF(2^m)$  where  $k$  is the number of  $m$ -bit message symbols. Also, let  $d$  be the minimum distance of the code, where  $d - 1$  is the number of parity symbols. Then,  $k = n - (d - 1)$ . We will define the errata vectors as  $\hat{u} = e + \mu = (\hat{u}_{n-1}, \hat{u}_{n-2}, \dots, \hat{u}_0)$ , where  $e = (e_{n-1}, e_{n-2}, \dots, e_0)$  and  $\mu = (\mu_{n-1}, \mu_{n-2}, \dots, \mu_0)$  are the error and erasure vectors, respectively. Next, we represent the received vector  $r = (r_{n-1}, r_{n-2}, \dots, r_0) = c + \hat{u}$ , where  $c = (c_{n-1}, c_{n-2}, \dots, c_0)$  is the codeword vector.

Suppose that  $s$  erasures and  $v$  errors occur in the received vector  $r$ , and assume that  $s + 2v \leq d - 1$ . Then, the syndromes are:

$$S_k = \sum_{i=0}^{n-1} r_i \alpha^{ik} = \sum_{i=1}^s W_i Z_i^k + \sum_{i=1}^v Y_i X_i^k \quad \text{for } 1 \leq k \leq d - 1, \quad (2.1)$$

where  $\alpha$  is a primitive element in  $GF(2^m)$ .  $W_i$  is the  $i^{\text{th}}$  erasure value,  $Z_i$  is the  $i^{\text{th}}$  known erasure location,  $Y_i$  is the  $i^{\text{th}}$  error value, and  $X_i$  is the  $i^{\text{th}}$  error location. From (2.1), let the syndrome polynomial be defined as:

$$S(x) = \sum_{k=1}^{d-1} S_k x^k. \quad (2.2)$$

The erasure polynomial is defined as:

$$\Lambda(x) = \prod_{j=1}^s (1 + Z_j x) = \sum_{j=0}^s \Lambda_j x^j, \quad (2.3)$$

where  $\Lambda_0 = 1$ .  $\Lambda(x)$  is the polynomial with zeros at the inverse erasure locators.

The inversionless Berlekamp-Massey algorithm is outlined as follows:

1. Initially define  $\mu^{(0)}(x) = \Lambda(x)$ ,  $\lambda^{(0)}(x) = \Lambda(x)$ ,  $\lambda^{(0)} = 0$ ,  $k = 0$ ,  $\gamma^{(0)} = 1$  if  $k \leq 0$ .
2. Set  $k = k + 1$ . If  $k \geq d - 1 - s$ , stop. Otherwise, define:

$$\delta^{(k)} = \sum_{j=0}^{d-1} \mu_j^{(k-1)} S_{k+s-j}. \quad (2.4)$$

3. Use this algorithm:

$$\mu^{(k)}(x) = \gamma^{(k-1)} \mu^{(k-1)}(x) - \delta^{(k)} \lambda^{(k-1)}(x) x,$$

$$\lambda^{(k)}(x) = \begin{cases} x \lambda^{(k-1)}(x), & \text{if } \delta^{(k)} = 0 \\ & \text{or if } 2\lambda^{(k-1)} > k - 1 \\ \mu^{(k-1)}(x), & \text{if } \delta^{(k)} \neq 0 \text{ and } \lambda^{(k-1)} \leq k - 1, \end{cases}$$

$$\lambda^{(k)} = \begin{cases} \lambda^{(k-1)}, & \text{if } \delta^{(k)} = 0 \text{ or } 2\lambda^{(k-1)} > k - 1 \\ k - \lambda^{(k-1)}, & \text{if } \delta^{(k)} \neq 0 \text{ and } 2\lambda^{(k-1)} \leq k - 1, \end{cases}$$

$$\gamma^{(k)} = \begin{cases} \gamma^{(k-1)}, & \text{if } \delta^{(k)} = 0 \text{ or } 2\lambda^{(k-1)} > k - 1 \\ \delta^{(k)}, & \text{if } \delta^{(k)} \neq 0 \text{ and } 2\lambda^{(k-1)} \leq k - 1. \end{cases}$$

4. Return to step 2.

The errata locator polynomial is now defined as:

$$\tau(x) = \sigma(x)\Lambda(x) = \frac{\mu^{(d-1-s)}(x)}{\beta} = \sum_{j=0}^{s+t} \tau_j x^j, \quad (2.5)$$

where  $\beta = \mu^{(d-1-s)}(0)$  is a field element in  $GF(2^m)$ . Now that  $\tau(x)$  is known we can compute the errata locator polynomial and the errata values. The errata locator polynomial is:

$$\beta A(x) \equiv S(x)(\beta\tau(x)) \pmod{x^d}. \quad (2.6)$$

The errata values are given as:

$$\tilde{W}_j = \frac{\beta A(\tilde{Z}_j^{-1})}{\tilde{Z}_j^{-1} \beta \tau'(\tilde{Z}_j^{-1})} = \frac{\beta A(\tilde{Z}_j^{-1})}{\tilde{Z}_j^{-1} \mu^{(d-1-s)' }(\tilde{Z}_j^{-1})}, \quad (2.7)$$

where  $\tau'(\tilde{Z}_j^{-1})$  is the derivative of  $\mu^{(d-1-s)}(x)$  with respect to  $x$  evaluated at  $x = \tilde{Z}_j^{-1}$ .

Now I will outline the five step decoding process of RS codes for both errors and erasures using the inversionless Berlekamp-Massey algorithm initialized with the erasure locator polynomial and the syndromes.

1. Compute the syndrome values  $S_1, S_2, \dots, S_{d-1}$  from (2.1). If  $S_i = 0$  for  $1 \leq i \leq d-1$ , the received word is a codeword, thus as usual, we assume that no errors occurred.
2. Compute the erasure locator polynomial given in (2.3). If  $s = d-1$  or  $d-2$  then set the error locator polynomial  $\sigma(x) = 1$  and go to step 5. Else, go to step 3.
3. If  $0 \leq s < d-2$  use the inversionless BM algorithm (described above) initialized with the erasure locator polynomial to determine the modified errata locator polynomial  $\Theta(x)$  from the known syndrome values for  $1 \leq i \leq d-1$ .
4. Compute the roots of  $\Theta(x)$  using Chien search. The roots of  $\Theta(x)$  make up the set  $\{\tilde{Z}_1^{-1}, \tilde{Z}_2^{-1}, \dots, \tilde{Z}_{s+v}^{-1}\}$ , which are the inverse locations of the  $s$  erasures and  $v$  errors.
5. Compute the errata evaluator polynomial and the errata values from (2.6) and (2.7).

Once all error values are calculated, they are used to make the error polynomial:

$$\hat{e}(x) = \sum_{i=1}^{s+v} \tilde{W}_i \tilde{Z}_i^k \quad \text{for } 1 \leq k \leq d-1.$$

And the decoded polynomial  $v(x) = \hat{e}(x) + r(x)$ .

Example: Here the elements of  $GF(2^4)$  are generated by the primitive polynomial  $p(x) = 1 + x + x^4$ . Let us take  $C = (15, 9)$  RS code over  $GF(2^4)$  with a minimum distance of  $d = 7$ . So  $t = 3$ . A combination of  $s$  erasures and  $v$  errors must satisfy  $2v + s \leq d - 1 = 6$  in order for the RS code to correct all errors. The generator polynomial  $g(x)$  of  $C$  is:

$$\begin{aligned} g(x) &= (x + \alpha)(x + \alpha^2) \dots (x + \alpha^{2t}) = (x + \alpha)(x + \alpha^2) \dots (x + \alpha^6) \\ &= x^6 + \alpha^{10}x^5 + \alpha^{14}x^4 + \alpha^4x^3 + \alpha^6x^2 + \alpha^9x + \alpha^6. \end{aligned}$$

Assume the message vector to be encoded is:

$$I = (\alpha^{10}, \alpha^{12}, \alpha^8, \alpha^5, \alpha^6, \alpha^{14}, \alpha^{13}, \alpha^{11}, \alpha^9)$$

and the codeword vector  $c$ , which is a multiple of  $g(x)$ , is:

$$c = (\alpha^{10}, \alpha^{12}, \alpha^8, \alpha^5, \alpha^6, \alpha^{14}, \alpha^{13}, \alpha^{11}, \alpha^9, \alpha^0, \alpha^1, \alpha^2, \alpha^6, \alpha^{12}, \alpha^8).$$

Also assume that the erasure vector is:

$$u = (0, 0, 0, 0, 0, 0, 0, ?, 0, 0, 0, 0, 0, 0, 0)$$

and the error vector is:

$$e = (0, 0, 0, 0, \alpha^{11}, 0, 0, 0, 0, 0, 0, \alpha^7, 0, 0, 0).$$

The errata vector is:

$$\hat{u} = u + e = (0, 0, 0, 0, \alpha^{11}, 0, 0, ?, 0, 0, 0, \alpha^7, 0, 0, 0).$$

The received vector is:

$$r = c + \hat{u} = (\alpha^{10}, \alpha^{12}, \alpha^8, \alpha^5, \alpha^6, \alpha^{14}, \alpha^{13}, ?, \alpha^9, \alpha^9, \alpha^0, \alpha^1, \alpha^{12}, \alpha^6, \alpha^{12}, \alpha^8).$$

We can replace the ? with  $\alpha^9$  and then proceed to decode. In theory the ? can be replaced by any value and the algorithm will still work. By (2.1) the syndromes are:

$$\begin{aligned} S_1 &= 1 & S_4 &= \alpha^{11} \\ S_2 &= \alpha^{13} & S_5 &= \alpha \\ S_3 &= \alpha^{14} & S_6 &= 0. \end{aligned}$$

So then the syndrome polynomial is:

$$S(x) = x + \alpha^{13}x^2 + \alpha^{14}x^3 + \alpha^{11}x^4 + \alpha x^5.$$

The erasure locator polynomial is:

$$\Lambda(x) = 1 + \alpha^7x.$$

The inversionless Berlekamp-Massey algorithm is now applied to the syndromes for  $1 \leq k \leq 6$ . This is accomplished using step 3 in the previous five step process. This is illustrated in Figure 2.1 [10].

We see from Figure 2.1 that the computation ends at  $k = d - 1 - s = 5$ . And thus:

$$\mu^{(5)}(x) = \alpha^5 + \alpha^7x + \alpha x^2 + \alpha^{10}x^3.$$

$k$	$\delta^{(k)}$	$\mu^{(k)}(x)$	$\lambda^{(k)}(x)$	$\gamma^{(k)}$	$\ell^{(k)}$
0	1	$(1 + \alpha^7 x)$	$(1 + \alpha^7 x)$	1	0
1	$\delta^{(1)} = \sum_{j=0}^6 \mu_j^{(1)} S_{2-j}$ $= \mu_0^{(1)} S_2 + \mu_1^{(1)} S_1$ $= 1 \cdot \alpha^4 + \alpha^7 \cdot 1 = \alpha^7$	$\mu^{(1)}(x) = \gamma^{(1)} \mu^{(0)}(x) + \delta^{(1)} \lambda^{(0)}(x) x$ $= 1 \cdot (1 + \alpha^7 x) + \alpha^7 (1 + \alpha^7 x) x$ $= 1 + \alpha^7 x + \alpha^7 x + \alpha^{14} x^2$ $= 1 + \alpha^7 x + \alpha^{14} x^2$	$\lambda^{(1)}(x) = \mu^{(0)}(x)$ $= (1 + \alpha^7 x)$	$\gamma^{(1)} = \delta^{(1)} = \alpha^7$	$\ell^{(1)} = 1 - \ell^{(0)} = 1$
2	$\delta^{(2)} = \sum_{j=0}^6 \mu_j^{(2)} S_{2-j}$ $= \mu_0^{(2)} S_2 + \mu_1^{(2)} S_1 + \mu_2^{(2)} S_0$ $= 1 \cdot \alpha^{14} + \alpha^{17} \cdot \alpha^{14} + \alpha^{17} \cdot 1 = \alpha^7$	$\mu^{(2)}(x) = \gamma^{(2)} \mu^{(1)}(x) + \delta^{(2)} \lambda^{(1)}(x) x$ $= \alpha^7 \cdot (1 + \alpha^7 x + \alpha^{14} x^2)$ $+ \alpha^7 (1 + \alpha^7 x) x$ $= \alpha^7 + \alpha^{14} x + \alpha^{14} x^2 + \alpha^7 x + \alpha^{14} x^2$ $= \alpha^7 + \alpha^{14} x^2$	$\lambda^{(2)}(x) = x \lambda^{(1)}(x)$ $= x(1 + \alpha^7 x)$ $= x + \alpha^7 x^2$	$\gamma^{(2)} = \gamma^{(1)} = \alpha^7$	$\ell^{(2)} = \ell^{(1)} = 1$
3	$\delta^{(3)} = \sum_{j=0}^6 \mu_j^{(3)} S_{2-j}$ $= \mu_0^{(3)} S_2 + \mu_1^{(3)} S_1 + \mu_2^{(3)} S_0$ $= \alpha^7 \cdot \alpha^{14} + 0 + \alpha^7 \cdot \alpha^{14} = \alpha^7$	$\mu^{(3)}(x) = \gamma^{(3)} \mu^{(2)}(x) + \delta^{(3)} \lambda^{(2)}(x) x$ $= \alpha^7 \cdot (\alpha^7 + \alpha^{14} x^2) + \alpha^7 (x + \alpha^7 x^2) x$ $= \alpha^{14} + \alpha^{21} x^2 + \alpha^7 x^2 + \alpha^{14} x^3$ $= \alpha^{14} + \alpha^7 x^2 + \alpha^{14} x^3$	$\lambda^{(3)}(x) = \mu^{(2)}(x)$ $= \alpha^7 + \alpha^{14} x^2$	$\gamma^{(3)} = \delta^{(3)} = \alpha^7$	$\ell^{(3)} = 3 - \ell^{(2)} = 2$
4	$\delta^{(4)} = \sum_{j=0}^6 \mu_j^{(4)} S_{2-j}$ $= \mu_0^{(4)} S_2 + \mu_1^{(4)} S_1 + \mu_2^{(4)} S_0 + \mu_3^{(4)} S_{-1}$ $= \alpha^{21} \cdot \alpha^{14} + 0 + \alpha^{21} \cdot \alpha^{14} + \alpha^{17} \cdot \alpha^{14} = \alpha^{17}$	$\mu^{(4)}(x) = \gamma^{(4)} \mu^{(3)}(x) + \delta^{(4)} \lambda^{(3)}(x) x$ $= \alpha^7 \cdot (\alpha^{14} + \alpha^{21} x^2 + \alpha^{14} x^3)$ $+ \alpha^{17} (\alpha^7 + \alpha^{14} x^2) x$ $= \alpha^{21} + \alpha^{28} x^2 + \alpha^{21} x^3 + \alpha^{14} x + \alpha^{28} x^3$ $= 1 + \alpha^7 x + \alpha^{14} x^2 + \alpha^{21} x^3$	$\lambda^{(4)}(x) = x \lambda^{(3)}(x)$ $= x(\alpha^7 + \alpha^{14} x^2)$ $= \alpha^7 x + \alpha^{14} x^3$	$\gamma^{(4)} = \gamma^{(3)} = \alpha^7$	$\ell^{(4)} = \ell^{(3)} = 2$
5	$\delta^{(5)} = \sum_{j=0}^6 \mu_j^{(5)} S_{2-j}$ $= \mu_0^{(5)} S_2 + \mu_1^{(5)} S_1 + \mu_2^{(5)} S_0 + \mu_3^{(5)} S_{-1}$ $= 0 + \alpha^7 \cdot \alpha^{14} + \alpha^{17} \cdot \alpha^{14} + \alpha^7 \cdot \alpha^{14} = 0$	$\mu^{(5)}(x) = \gamma^{(5)} \mu^{(4)}(x) + \delta^{(5)} \lambda^{(4)}(x) x$ $= \alpha^7 \cdot (1 + \alpha^7 x + \alpha^{14} x^2 + \alpha^{21} x^3)$ $= \alpha^7 + \alpha^{14} x + \alpha^{14} x^2 + \alpha^{28} x^3$	$\lambda^{(5)}(x) = x \lambda^{(4)}(x)$ $= x(\alpha^7 x + \alpha^{14} x^3)$ $= \alpha^7 x^2 + \alpha^{14} x^4$	$\gamma^{(5)} = \gamma^{(4)} = \alpha^7$	$\ell^{(5)} = \ell^{(4)} = 2$

**Figure 2.1.** An example of the use of the inversionless BM algorithm.

From (2.5) we see that  $\mu^{(5)}(x) = \beta\tau(x)$  and  $\beta = \mu^{(5)}(0) = \alpha^5$ . Let us set  $\mu^{(5)}(x) = \Theta(x)$ . We need to find the roots of  $\Theta(x)$ . This is done through an exhaustive search called Chien search which will look for all the elements of  $GF(16)$  that are roots of  $\Theta(x)$ . Doing so will yield  $\{\alpha^3, \alpha^7, \alpha^{10}\}$  as the roots of  $\Theta(x)$ .

Next we need to find the derivative of  $\Theta(x)$  which is  $\Theta'(x) = \alpha^7 + \alpha^{10}x^2$ . Therefore the errata evaluator polynomial is:

$$\begin{aligned} \beta A(x) &\equiv S(x)(\beta\tau(x)) \pmod{x^d} \\ &\equiv S(x)\Theta(x) \pmod{x^7} = \alpha^5 x + \alpha^4 x^2 + \alpha^{10} x^3 \end{aligned}$$

And the errata value for  $\tilde{Z}_3 = \alpha^3$ :

$$\begin{aligned} \tilde{W}_3 &= \frac{\beta A(\tilde{Z}_3^{-1})}{\tilde{Z}_3^{-1} \beta \tau'(\tilde{Z}_3^{-1})} = \frac{\beta A(\tilde{Z}_3^{-1})}{\tilde{Z}_3^{-1} \Theta'(x)(\tilde{Z}_3^{-1})} \\ &= \frac{\alpha^5(\alpha^{-3}) + \alpha^4(\alpha^{-3})^2 + \alpha^{10}(\alpha^{-3})^3}{\alpha^{-3}(\alpha^7 + \alpha^{10}(\alpha^{-3})^2)} = \alpha^7. \end{aligned}$$

Similarly,  $\tilde{W}_7 = \alpha^2$  and  $\tilde{W}_{11} = \alpha^{11}$ .

## CHAPTER 3

### AUDIO CD ERROR CORRECTING CODE

In this chapter we will go over the application of Reed-Solomon codes in an audio CD. Chapter 3 will also cover some background on the development of the audio CD and how a CD can further improve on its error correcting capabilities through the use of a technique called interpolation.

#### 3.1 DEVELOPMENT OF CDS

The Compact Disc digital audio system has the standard components of a communication system - a transmitting and a receiving end connected by a channel, and as is usual in a communication system, many of the operations at the receiving end are the inverses of those at the transmitting end. The two audio signals that originate from the stereo input are converted from analog to digital and are recorded digitally on a magnetic tape. The channel encoder adds parity bytes, each byte consisting of eight bits, derived from two Reed Solomon Codes [17]. After this, digital control and display information containing music related data and a table of contents on the disc are added to the encoded data. The output from the channel encoder goes through a modulator before it is conveyed to the master disc. Modulation caters for requirements like reducing inter symbol interference, and so on. Next, the CD standardized format is optically recorded on the surface of a glass disc which is coated with photoresist, called the master disc. The information is transferred in the form of *pits* to a transparent plastic disc via a nickel shell called a stamper. After receiving a reflective aluminum coating protected by lacquer, the CD is ready for playing [17].

Channel errors can come from several areas. They can come from foreign particles, air bubbles in the plastic material, inaccuracies in the pits due to stamper errors [17] or in the coating and cutting of the disc [8]. These will cause errors when the information is optically read out. These commonly occur as random errors and smaller length error patterns. However, burst errors caused by fingerprints and scratches on the surface of the disc result in errors affecting several consecutive bits [17].

#### 3.2 CROSS-INTERLEVED REED-SOLOMON CODE

There are many error correcting codes used in practice today but the most commonly used method is cross-interleaved Reed-Solomon code or CIRC for short [11]. Recording in stereo requires 2 amplitude measurements taken 44,100 times per second. One sample is taken from the left and the other from the right [8]. In the encoding process, each binary word

of length 16 corresponds to an amplitude measurement which is represented by 2 field elements in  $GF(256)$  [8]. Each element is referred to as a byte. When recording in stereo, 4 bytes are produced at each “tick”. A tick is  $\frac{1}{44,100}$  of a second. The amplitude measurements from 6 consecutive ticks are formed together to make a message of length 24. After 4 parity bytes are added, this message is encoded to a codeword using  $C_1$ , a (28, 24, 5) Reed-Solomon code over  $GF(256)$  [8]. The codewords in  $C_1$  are then 4-framed delay interleaved.

An additional 4 parity bytes are added and the codewords in  $C_1$  are then encoded using  $C_2$ , a (32, 28, 5) RS code over  $GF(256)$ . Afterwards, an additional byte is added to each codeword in  $C_2$  for control and display purposes and the length of each codeword is now 33 [8]. The code  $C_2$  is then 1-framed delay interleaved [17].

### 3.3 THE DECODING AND ERROR CORRECTING PROCESS

The decoding process as well as the error correcting process will now be described. Since encoding and decoding are inverse operations, received codewords are first passed through  $C_2$  and then to  $C_1$ .  $C_2$  is used to correct single errors and  $C_1$  can be used to correct up to 4 erasures. Each code has four redundancy symbols with a minimum distance of 5, and is hence capable of correcting either 2 symbol errors or 4 symbol erasures. This code can also be used to simultaneously correct one error and detect three errors, or correct a single error and two erasures [17].

First, the effect of interleaving to depth 1 is removed and all parity check symbols are also removed from received codewords. If  $C_2$  detects no errors, it will pass on the 28 information symbols to  $C_1$  with all the erasure indications off [17]. If there is a single error,  $C_2$  will correct this. If  $C_2$  detects more than one error, it will set erasure flags on all 28 bits indicating that they are all unreliable [17]. The effect of the 4-frame delay applied earlier in the encoding process is now removed and the received word is now sent to  $C_1$ . A received word will be decoded correctly by  $C_1$  if there are no more than 4 flagged digits, that is, assuming that  $C_2$  detected all errors in the previous step.  $C_1$  then corrects up to 4 erasures, treating all flagged bytes as erasures and the unflagged ones as correct [8].

Notice that the only way  $C_2$  can fail at decoding a received word is if that word is within distance 1 of a codeword in  $C_2$  that is not the correct codeword. How many are there exactly? There are  $(2^8)^{28}$  or  $2^{224}$  codewords in  $C_2$ , only one of which is the correct word. The remaining  $2^{224} - 1$  codewords is within distance 1 of  $1 + 32(2^8 - 1)$  words of length 32. So of all the  $(2^8)^{32}$  binary error patterns that could affect a codeword in  $C_2$  only  $(2^{224} - 1)(1 + 32)(2^8 - 1)$  of them result in a codeword within distance 1 of a different codeword in  $C_2$ . This turns out to be about 1 in  $2^{19}$  [8]. The code  $C_2$  is good enough to correct bursts of length up to



4000 bits [13]. That corresponds to 16 consecutive uncorrectable words in  $C_1$  and a track length of about  $2.5mm$  [17].

### 3.4 INTERPOLATION

There is another operation in the encoding process than can be used to further the chances of correcting errors within the  $C_1$  code. The bytes within codewords in  $C_1$  are reorganized. Every codeword contains information from the left and the right,  $L_1$  to  $L_6$  and  $R_1$  to  $R_6$  respectively. There are two parity symbols added on as well in the encoding process,  $P_1$  and  $P_2$ . These bytes are arranged in this order:  $L_1L_3L_5R_1R_3R_5P_1P_2L_2L_4L_6R_2R_4R_6$ . The bytes are arranged as such so that if several consecutive bytes are still flagged after decoding, they can be treated as unreliable information. If this is the case, any unreliable value,  $L_i$ , can be replaced by averaging the amplitudes of the adjacent values  $L_{i-1}$  and  $L_{i+1}$  if those samples are reliable. This technique is called interpolating values [8]. If this interpolating technique is used, then correcting up to 4 erasures is not used. This strategy allows up to two-erasure correction, and up to single error correction [17]. The maximum interpolatable burst length is around 12,300 bits which corresponds to a track length of about  $7.7mm$ . A deliberately scratched CD or even a CD with holes drilled into it can be played without loss of music. An audible click will be heard for an undetected erroneous sample [17] or if even interpolation fails to correct an error than the clicking sound will be muted so the listener cannot hear it [9].

## CHAPTER 4

### DVD ERROR CORRECTING CODE

Like a CD, DVDs suffer from a combination of random and burst errors. However, DVDs need a more powerful error correction system because the higher physical density means that any physical imperfection will affect proportionally more bits [9]. In other words, physically a DVD and CD are the same size, but data wise a DVD contains more data space. A DVD has a data capacity of 4.7 GB whereas a CD only has 700 MB. Since the system margins are much tighter, the random error rate of a DVD is larger than that of a CD. Also, decoded data must be more reliable therefore we cannot rely on the concealment techniques used in an audio CD [9].

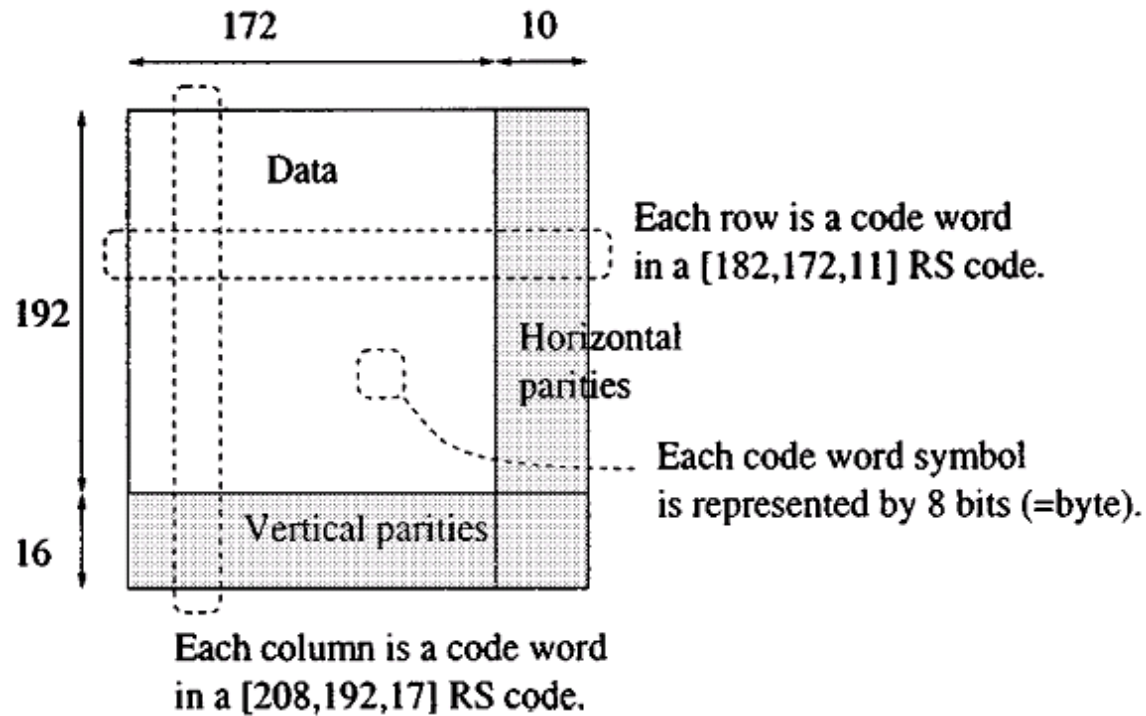
#### 4.1 THE REED-SOLOMON PRODUCT CODE

CIRC, which Compact Disc has adopted, regards data as a successive data stream, and this is a kind of linear error-correction system. However, it is different from RSPC (Reed Solomon Product Code) which is adopted by DVD. RSPC treats data as block or array. Though it needs twice as much buffer storage as a CD, the efficiency is ten times better than a CD. CIRC can correct about 500 bytes successive error, whereas RSPC can correct about 2200 bytes [9].

The RSPC code used in a DVD is graphically illustrated in Figure 4.1 [4]. Thirty-two KB of data are stored in a 192 x 172 matrix. Each row of data in this matrix is encoded into a (182, 172, 11) Reed-Solomon (RS) code word. All columns are encoded into a (208, 192, 17) RS codewords [4]. Since all codewords are stored on the disc row by row, all bursts of errors are oriented in the direction of the rows [4].

#### 4.2 DECODING WITH THE RSPC

Decoding consists of two steps. First, row correction is performed. Up to 5 byte errors can be corrected in each row. Recall that the error correcting capability of a code is  $\lfloor (d - 1)/2 \rfloor$ , where  $d$  is the distance of a code. In other words, almost all random errors are corrected. However, rows affected by bursts of errors normally have too many errors to correct [4]. In this case, row correction results either in an erroneous correction or a detection of 6 or more errors. The latter is called a decoding failure. A decoding failure most likely indicates the location of a burst of errors affecting the corresponding row [4]. Thus erasures are assigned to the bytes in such rows. With this knowledge, column correction is performed in the second step of decoding. As the column code has minimum distance of 17, if a column



**Figure 4.1.** The RSPC code used in DVDs.

has  $e$  erasures (because of decoding failures) and  $t$  errors (because of erroneous corrections), then this column is correctable if and only if  $2t + e \leq d - 1$  [4].

The decoding process will now be described in further detail. Since error correcting is done at the same time as decoding, that process will also be detailed here. The most popular RS decoder architecture today can be summarized into four steps:

1. Calculating the syndromes from the received codeword.
2. Computing the error locator polynomial  $\sigma(x)$  and the error evaluator polynomial  $\Omega(x)$ .
3. Finding the error locations.
4. Computing error values.

Once a codeword is received, its syndromes are calculated as stated above. Next, an inversionless Berlekamp-Massey (BM) algorithm is used to find the error locator polynomial  $\sigma(x)$  and the error evaluator polynomial  $\Omega(x)$  from the syndrome information obtained from step 1. To find the error locations, Chien search is used. Finally, Forney's algorithm is used to compute the error values [2].

Recall that in the decoding process, row correction is performed first. Thus, the previously described algorithm applies to row decoding only. Also recall that row correction only corrects random errors and not erasures. When decoding in the column direction, the algorithm is slightly modified. Instead of just finding the error locator polynomial  $\Omega(x)$ , an

erasure locator polynomial  $\Gamma(x)$  is also found. With both of these locator polynomials, they are combined to create an errata locator polynomial  $\Psi(x)$ , where  $\Psi(x)$  is defined as:

$$\Psi(x) = \sigma(x)\Gamma(x)$$

where  $\sigma(x)$  is the error locator polynomial.

Consequently, instead of computing the error evaluator polynomial  $\Omega(x)$ , the errata evaluator polynomial  $\Lambda(x)$  is computed. The inversionless BM algorithm is still used to find the errata locator and errata evaluator polynomial. Chien search is still used to find the errata locations and the modified Forney algorithm is used to compute the error and erasure values [15].

### **4.3 COMPARING CORRECTING ABILITIES OF THE DVD TO THE CD**

As stated earlier the maximum correctable burst length is approximately 500 bytes which is about *2.4mm* for CIRC, while it is 2200 bytes, approximately *4.6mm* for RSPC. RSPC is capable of reducing a random input error rate of  $2 \times 10^{-2}$  to a data error rate of  $10^{-15}$ , which is a factor of 10 better than in a CD [9].

## **CHAPTER 5**

### **BLU-RAY ERROR CORRECTING CODE**

Recall that in optical recording roughly two types of errors can be distinguished: single or random errors and burst errors. Single errors are caused by noise in combination with other sources of signal deterioration such as tilt of the disc or defocus of the laser spot on the disc [1]. They are called single errors because they only affect one or two bytes. Burst errors are caused by defects on the disc surface like scratches, dust, fingerprints etc[1].

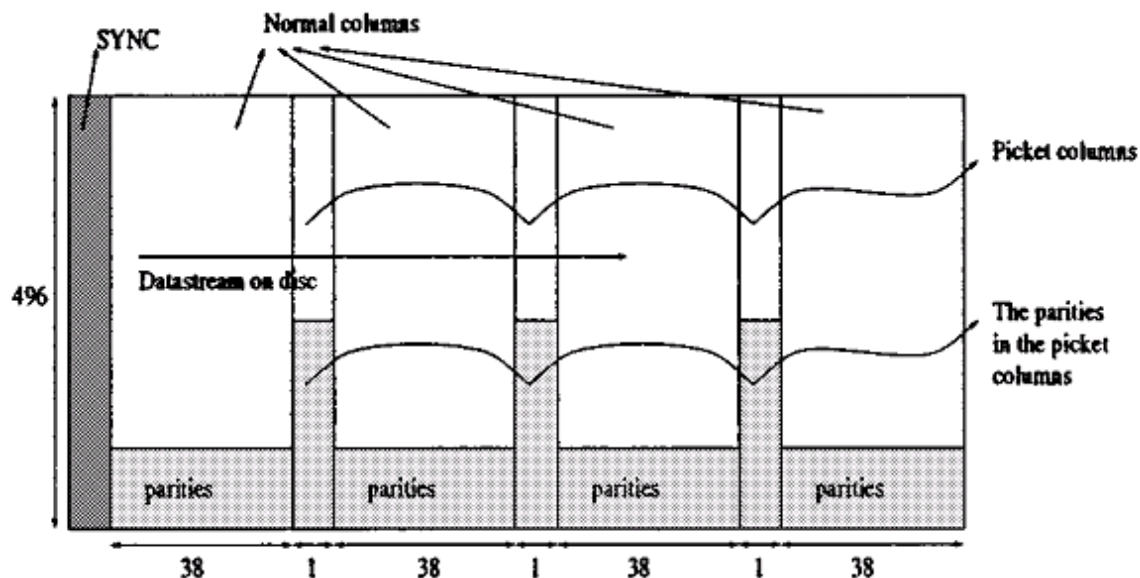
#### **5.1 DESIGN FOR BLU-RAY ERROR CORRECTING CODE**

An error correction system should be adapted to the physical properties of the medium on which the data is stored. Blu-ray Discs are more sensitive to burst errors because of its small laser spot, the thin cover layer and the high numerical aperture than the CD and DVD system. Errors on Blu-ray Discs will corrupt more data bits than on a CD or DVD disc. Therefore, the error correction system of Blu-ray Discs should be able to deal with long burst errors with little trouble [1].

Compared to DVD, the laser spot size on the entrance surface of the disc is reduced from approximately  $0.5mm$  in diameter to about  $0.14mm$ . This results in increased sensitivity to dust and scratches on the disc surface which may cause burst errors, on top of the usual random errors during readout of the recording layer. The so-called picket code is a new error detection and correction method that uses two correction mechanisms to handle these errors effectively: a long distance code (LDC) combined with a burst indicator subcode (BIS) [14].

#### **5.2 THE PICKET CODE**

In a Reed Solomon Product Code (RSPC), the row code is used for correcting random errors and detecting bursts of errors. Since Blu-ray Discs are more sensitive to burst errors, much of the horizontal parity is squandered in a sense that the horizontal parity does not have any correctional capabilities since it is only used for detection of burst errors. In the picket code, no row code exists [4]. All redundancy is put into column Reed-Solomon codes (See Figure 5.1). There are two distinguished columns: picket columns and normal columns. The collection of normal columns makes up the LDC and the collection of picket columns makes up the BIS. The picket columns have stronger error correcting capabilities when compared to the normal columns due to the fact that the picket columns contain more redundancy.



**Figure 5.1. Blu-ray picket code.**

The LDC has 304 (248, 216, 32) Reed-Solomon (RS for short) codewords. This LDC contains 64 kilobytes (KB) of user data [4]. One 64 KB block of user data is called an error correction block (ECC block). “These codewords are interleaved two by two in the vertical direction such that a block of 152 bytes by 496 bytes is formed” [1] as shown in the Figure 5.1. The LDC has sufficient parity symbols and interleaving length for correcting random errors, multiple long bursts and short bursts of errors. The burst error correction capability is strongly enhanced by using erasure correction on the erroneous symbols flagged by the BIS code.

The BIS has 24 (62, 30, 32) RS codewords. The BIS codewords are interleaved into three columns of 496 bytes each [1]. The address and control information in one ECC block is strongly protected by these BIS RS codewords. The BIS code “can be properly decoded with extremely high probability i.e. all of its errors can be corrected” [4]. The location of its corrected bytes and erroneous synchronization patterns serve as “pickets” indicating the likely position of a long burst errors in the LDC data between these pickets; when subsequent pickets have “fallen”, it is highly likely that all the data located physically in between these pickets was corrupted by a burst of errors.

### 5.3 DECODING WITH THE PICKET CODE

Picket columns are corrected first in the decoding process. Since both the LDC and the BIS codes have the same number of parity symbols per code word, there is only one RS decoder needed to decode them both [1]. This correction information is then used to locate any occurring burst errors in the normal columns. Afterwards, the symbols with errors at these locations are flagged as erasure that is these erroneous bytes are erased [1]. An erasure

and error decoding algorithm of RS codewords in the normal columns makes use of this erasure information to correct these bursts together with random errors.

The picket columns use the same algorithm as described in the DVD section for the decoding of the rows [12] [15]. Although Blu-ray decoding does not decode row wise, it still uses the same algorithm. Also the picket code performs error only decoding not erasure [12]. The LDC performs error and erasure decoding like the column decoding process of the DVD [12] [15]. Note that the Blu-ray picket code has twice as many parity bytes than the RSPC of the DVD thus the picket code is much better at correcting errors.

The maximum number of errors that can be corrected depends on the number of parity symbols added. For each two parity symbols added, one error can be corrected. This is assuming that nothing is known beforehand about the error. If the location of an error within the code word is known beforehand, only the erased value of the error has to be calculated [1]. For each parity symbol added, one erased value can be calculated, i.e. one erasure can be corrected. So it is advantageous for the error corrector to use prior knowledge of the error locations in the decoding process. Due to the nature of the errors, this is not possible for random errors, but it is very well possible for burst errors. It requires a burst indicator mechanism that can detect bursts of errors before the correction starts which is what the BIS code does [1].

## CHAPTER 6

### CONCLUSION AND FUTURE WORK

Here I will summarize the differences among the three media types. Mainly, I will compare the different error correcting codes: CIRC, RSPC, and the conventional picket code.

As explained in Chapter 3, the audio CD uses the crossed interleaved RS code, CIRC, which regards data as a successive data stream and this is a kind of linear error-correction system. However, it is different from RSPC (Reed Solomon Product Code) which is adopted by DVD. RSPC treats data as a block or an array [13]. The disadvantage of a product code structure relative to a CIRC structure is that RSPC requires twice as much memory capacity [9]. But nowadays this is not really a problem as technology has advanced enough to handle this memory obstacle.

#### 6.1 COMPARING THE ERROR CORRECTING CODES

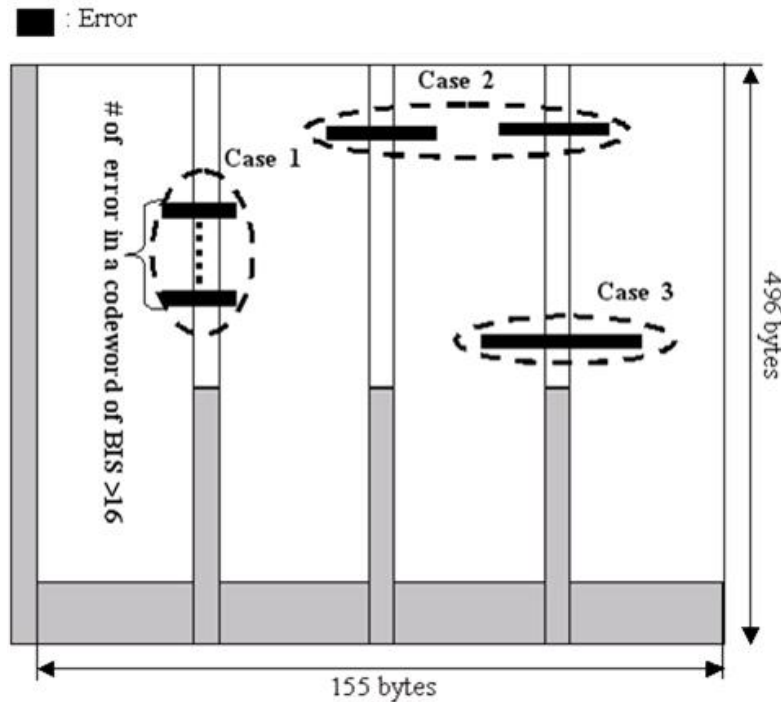
The maximum correctable burst length is approximately 500 bytes which is about  $2.4mm$  for CIRC, while it is 2200 bytes, approximately  $4.6mm$  for RSPC. RSPC is capable of reducing a random input error rate of  $2 \times 10^{-2}$  to a data error rate of  $10^{-15}$ , which is a factor of 10 better than in a CD [9]. When comparing the error rates after correction of the RSPC and the picket code, consider this experiment from [14]. A dust sprayed disc was exposed to an office environment. That disc had a byte error rate of  $4 \times 10^{-3}$ . The errors included many long and short burst errors. After error correction, the BIS code error rate was found to be below  $10^{-25}$ . The LDC was able to reduce the error rate to  $1.5 \times 10^{-18}$ . Compared to the RSPC of a DVD, the error rate was only reduced to  $5.7 \times 10^{-8}$ . Conclusively, the picket code is much more powerful than the RSPC.

#### 6.2 IMPROVING ON THE PICKET CODE IN THE FUTURE

It turns out that a product code is stronger for correcting random errors only. However, in practice, burst errors turn out to limit the actual performance of an ECC system in optical recording. Compared to Blu-ray, DVD achieves a factor 2 higher random byte error probability if no bursts are present, and compared to DVD, Blu-ray achieves a factor 10 higher burst error rate if no random byte errors are present [4].

As strong as the picket code is for the Blu-ray disc, it is not perfect. Here are 3 main drawbacks as described in [12]. Figure 6.1 from [12] illustrates the problems as well.





**Figure 6.1. Illustration of the main drawbacks of the conventional picket code.**

Case 1: BIS is composed of important information such as addressing and control data, which are required for accessing data on disk, and just performs the error-only decoding because erasure information is not supplied from the outside. Thus, if the number of byte errors in a BIS codeword exceeds 16 errors, information included in the picket columns will be compromised. If BIS can perform error and erasure decoding, more errors can be corrected.

Case 2: Under the conventional picket code, if there are errors in consecutive BIS columns in the same row, then the BIS assigns all bytes located between picket columns as erasure. In this case, it is possible that a few bytes of normal columns are erroneous. Accordingly, erasure information generated by the BIS is inaccurate and the LDC performs the error and erasure decoding based on imprecise information. Consequently, the error correction may result in reducing the error-correction efficiency of picket code. The improvement of the drawback can enhance error correction power of picket code.

Case 3: One byte of a picket column and a few bytes of the adjacent normal columns have errors. Under the conventional picket code, these errors are not considered as erasure because only one picket column is in error. This will result in a reduction in efficiency of the picket code. If LDC can perform the error and erasure decoding without the help of BIS, the problem may be solved. If the above cases mentioned frequently happen, subsequently the error-correction efficiency of the picket code will be reduced.

A new error correcting system proposed by [12] performs error and erasure decoding using the only erasure information supplied from a modulation code decoder that fixes all 3 of the above problems that the current picket code has.

Another solution to making the picket code more powerful is suggested in [5]. This patent suggests that the BIS bytes are spread out among each 64 Kb block. Each 64 Kb block can be considered as a 496 x 155 matrix. Each of the 496 rows is called a recording frame. One recording frame has 155 bytes. Normally each frame has 152 bytes that comprise the long distance code (LDC), and 3 bytes for the burst indicator code (BIS). Typically, the BIS bits are grouped into 3 bytes. But according to [5] the 24 BIS bits can be spread out into each frame. The BIS bits still do the same job as they did before. If two consecutive bits are found to be in error in the same frame, then the bytes in between them are marked as erasure. Since the BIS bits are much more spread out, more errors can be corrected.

## BIBLIOGRAPHY

- [1] BLU-RAY DISC ASSOCIATION, *White Paper Blu-ray Disc™ Format-General*, October 2010.
- [2] H. CHANG, C. B. SHUNG, AND C. YEE, *A Reed-Solomon product-code (RS-PC) decoder chip for DVD applications*, IEEE J. of SS Cir., 36 (2001), pp. 229–238.
- [3] E. CHRISTMAN, *U.S. album sales fall 12.8% in 2010, digital tracks eke out 1% gain*. Billboard, <http://www.billboard.com/news/u-s-album-sales-fall-12-8-in-2010-digital-1004137859.story#/news/u-s-album-sales-fall-12-8-in-2010-digital-1004137859.story>, accessed July 2011, 2011.
- [4] W. COENE, H. POZIDIS, M. VAN DIJK, J. KAHLMAN, R. VAN WOUDEBERG, AND B. STEK, *Channel coding and signal processing for optical recording system beyond DVD*, IEEE Trans. Mag., 37 (2001), pp. 682–688.
- [5] W. COENE, M. VAN DIJK, AND C. BAGGEN, *Method and device for encoding information words, method and device for decoding information words, storage medium and signal*. U.S. Patent 20020157055, Oct 2002.
- [6] D. COSTELLO AND S. LIN, *Error Control Coding: Fundamentals and Applications*, Prentice Hall, New Jersey, second ed., 2004.
- [7] B. FRITZ, *DVD revenue plummets 44% in 2010, SNL kagan study says*. LA Times, <http://latimesblogs.latimes.com/entertainmentnewsbuzz/2011/05/dvd-revenue-plummets-44-in-2010-study-says.html?dlvrit=71043>, accessed July 2011, 2011.
- [8] D. R. HANKERSON, D. G. HOFFMAN, D. A. LEONARD, C. C. LINDNER, K. T. PHELPS, C. A. RODGER, AND J. R. WALL, *Coding Theory and Cryptography The Essentials*, Marcel Dekker, Inc., New York, 2000.
- [9] K. IMMINK, *The digital versatile disc (DVD): System requirements and channel coding*, SMPTE J., 105 (1996), pp. 483–489.
- [10] J. JENG AND T. TRUONG, *On decoding of both error and erasures of a Reed-Solomon code using an inverse-free berlekamp-massey algorithm*, IEEE Trans. on Comm., 47 (1999).
- [11] A. KRAVTCHENKO, J. VOGEL, AND F. ROMINGER, *Error correction with a cross-interleaved reed-solomon code, particularly for CD-ROM*. European Patent 1,111,800, June 2001.
- [12] J. LEE AND S. LEE, *A new error correction technique for BD system*, IEEE Trans. on Con. Elec., 56 (2010), pp. 663–668.
- [13] X. LIN, H. JIA, AND C. MA, *Error-correction codes for optical disc storage*, Proc. of SPIE, 5643 (2005), pp. 342–347.

- [14] T. NARAHARA, S. KOBAYASHI, M. HATTORI, Y. SHIMPUKU, G. VAN DEN ENDEN, J. KAHLMAN, M. VAN DIJK, AND R. VAN WOUDEBERG, *Optical disc system for digital video recording*, Jpn. J. Appl. Phys., 39 (2000), pp. 912–919.
- [15] T. PARK, *Design of the (248, 216) Reed-Solomon decoder with erasure correction for blu-ray disc*, IEEE Trans. on Con. Elec., 15 (2005), pp. 872–878.
- [16] I. REED AND G. SOLOMON, *Polynomial codes over certain finite fields*, J. of the Soc. for Ind. and App. Math., 8 (1960), pp. 300–304.
- [17] P. SHANKAR, *Error correcting codes Reed-Solomon codes*, Resonance, 2 (1997), pp. 33–47.
- [18] VOYAGER THE INTERSTELLAR MISSION. NASA, 2010.  
<http://voyager.jpl.nasa.gov/gallery/uranus.html>, accessed July 2011.
- [19] S. B. WICKER AND V. K. BHARGAVA, eds., *Reed-Solomon Codes and Their Applications*, IEEE Press, New York, 1994.