Bachelor's Thesis

Degree Programme in Information Technology

2011

León van de Pavert

REED-SOLOMON ENCODING AND DECODING

A Visual Representation

TURUN AMMATTIKORKEAKOULU
TURKU UNIVERSITY OF APPLIED SCIENCES

Bachelor's Thesis | Abstract

Turku University of Applied Sciences

Degree Programme in Information Technology

Spring 2011 | 37 pages

Instructor: Hazem Al-Bermanei

León van de Pavert

REED-SOLOMON ENCODING AND DECODING

The capacity of a binary channel is increased by adding extra bits to this data. This improves the quality of digital data. The process of adding redundant bits is known as channel encoding.

In many situations, errors are not distributed at random but occur in bursts. For example, scratches, dust or fingerprints on a compact disc (CD) introduce errors on neighbouring data bits. Cross-interleaved Reed-Solomon codes (CIRC) are particularly well-suited for detection and correction of burst errors and erasures. Interleaving redistributes the data over many blocks of code. The double encoding has the first code declaring erasures. The second code corrects them.

The purpose of this thesis is to present Reed-Solomon error correction codes in relation to burst errors. In particular, this thesis visualises the mechanism of cross-interleaving and its ability to allow for detection and correction of burst errors.

KEYWORDS:

Coding theory, Reed-Solomon code, burst errors, cross-interleaving, compact disc

ACKNOWLEDGEMENTS

It is a pleasure to thank those who supported me making this thesis possible.

I am thankful to my supervisor, Hazem Al-Bermanei, whose intricate knowledge of coding theory inspired me, and whose lectures, encouragement, and support enabled me to develop an understanding of this subject.

This thesis would not have been possible without the support of the teachers at the University of Applied Sciences. I would not have been able to even start these studies without the support and understanding of my wife, Maija and motivation from my children Kira, Caspar and Julius.

Last but certainly not least, I would like to express my gratitude to Aschwin van der Woude, for listening to my issues, for his advice and coaching.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOTATIONS

n	length of codeword	
k	number of data symbols	
d	distance	
d_{min}	minimum distance	
t	number of correctable errors	
l	number of detectable errors	
$g(x)$	generator polynomial	degree: $n-k$
$p(x)$	error check polynomial	degree: $n-k-1$
$h(x)$	parity check polynomial	degree: k
$i(x)$	information polynomial	degree: $k-1$
$c(x)$	codeword polynomial	degree: $n-1$
$c'(x)$	received codeword polynomial	degree: $n-1$
$c'_r(x)$	corrected codeword polynomial	degree: $n-1$
$s(x)$	syndrome polynomial	degree: $n-k-1$
$e(x)$	error polynomial	degree: $n-1$
$GF(q)$	Galois field or finite field where q in $\mathbb{N}$	
$\mathbb{N}$	set of natural numbers or integers, $\{0,1,2,...\}$	

ABBREVIATIONS

ADC	Analog-to-digital converter
BCH	A class of codes named after Bose, Ray-Chaudhuri and Hocquenghem
BEC	Binary erasure channel
BSC	Binary symmetric channel
ECC	Error-correcting code
FEC	Forward error correction
MDS	Maximum distance separable
ML, MLD	Maximum likelihood, maximum likelihood decoding
RS	A class of codes named after Reed and Solomon

1 INTRODUCTION

1.1 Error detection and correction

When data is stored or transmitted, we cannot ignore encoding. The field of mathematics that deals with sending data, a digital bit stream, over a noisy channel is called coding theory. The Oxford English Dictionary says the following about code:

> Any system of symbols and rules for expressing information or instructions in a form usable by a computer or other machine for processing or transmitting information.

During World War II, and even before, as far back as classic times, messages had to be sent to allies but it was crucial they were unintelligible to the enemy. This field of cryptology was born out of necessity, a sense of survival. After the war, before governments could render the research obsolete, the people behind cryptology research showed that cryptology and eventually the theory of error detecting and correcting could be put into practical use. We can see that the field of cryptology is adjacent to and often-times overlapping with the field of coding theory (Trappe & Washington, 2006).

Firstly, some pioneers and their achievements are addressed. The mathematics behind coding follows in chapter 2. While chapter 3 goes into the theory of linear block codes, it will be the visualisation in chapter 4 that explains how burst errors can be detected and corrected, on e.g., a compact disc. Physical damage like dust or scratches or material impurities can cause erasures or burst errors in the data stream. With forward error correction techniques, like Reed-Solomon codes, these interrupts in the data stream can be detected and corrected.

1.2 History of error control coding

Pioneers of coding theory are Shannon and Hamming who were colleagues at Bell Labs. Hocquenghem in 1959 and independently Bose and Ray-Chaudhuri in 1960 were responsible for a class of codes known as BCH codes. Reed and Solomon followed with a set of cyclic codes, which are BCH codes, but are well-suited for detecting and correcting burst errors and erasures. It was not until almost a decade later when Berlekamp invented a decoding algorithm which was simplified by Massey in 1969. In 1982 the compact disc (CD) was the first mass-produced device that used these error correcting codes.

1.2.1 Shannon

Claude Shannon (1916–2001) was an electronic engineer and mathematician and during the Second World War he joined Bell Labs to work on cryptography. His work was closely related to coding theory and eventually led to publication of the article named *A Mathematical Theory of Communication* in 1948, which is now regarded as one of the founding works of communication theory. Presently, not only do many regard him as the father of information theory, but he is also credited with establishing digital computer and digital circuit design theory while he was a Master's student at MIT (Bose, 2008).

Shannon's channel coding theorem proves that if the code transmission rate is less than the maximum channel capacity, it is possible to design an error-control code with almost error-free information transmission (Bossert, 1999).

1.2.2 Hamming

Richard Hamming (1915–1998) was a contemporary and colleague of Shannon at Bell Labs. While doing research on cryptology, Hamming became inter-

ested in the idea of error correcting codes while working on a relay computer out of normal office hours. Unfortunately there were no computer operators available to react to an alarm in case an error was detected. Hamming had to devise a code that would not only detect an error, but would also be able to correct it automatically, instead of just ringing the alarm. These codes are used to add redundancy to data which aid the detection and correction of errors. Chapter 3 explains the Hamming code, which was a first in the field we now know as coding theory.

Although the Hamming code was referred to by Shannon in 1948, patent considerations prevented its independent publication until 1950.

1.2.3 Hocquenghem, Bose and Ray-Chaudhuri

Alexis Hocquenghem (1908?–1990) was a French mathematician, whose article “Codes correcteurs d'erreurs” from 1959 mentioned codes that he described as a “generalization of Hamming's work” (Hocquenghem, 1959).

Independently from Hocquenghem, Ph.D. adviser Raj Bose (1901–1987) and his student Dwijendra Ray-Chaudhuri (1933–) published “On a class of error correcting binary group codes” in 1960. This class of linear block codes is named after Bose, Ray-Chaudhuri and Hocquenghem and became known as BCH codes (Wicker & Bhargava, 1999).

1.2.4 Reed and Solomon

Irving Reed (1923–) is an American mathematician and engineer who is best known for co-inventing a class of algebraic codes known as Reed-Solomon codes (RS codes) in collaboration with Gustave Solomon (1930–1996). RS codes are seen as a special case of the larger class of BCH codes but it was not until almost a decade later, by regarding them as cyclic BCH codes, that an efficient decoding algorithm gave them the potential to their widespread

application.

1.2.5 Berlekamp and Massey

Elwyn Berlekamp (1940–) is a professor emeritus of mathematics, electrical engineering and computer science at the University of California, Berkely. While he was studying electrical engineering at MIT one of his Ph.D. advisers was Claude Shannon. Berlekamp invented an algorithm for decoding BCH codes in 1968, but it was James Massey (1934–), an information theorist and cryptographer, who simplified this algorithm in 1968 which we know as the Berlekamp-Massey algorithm (Massey, 1969).

This algorithm made it possible to develop a fast and efficient decoder with a linear feedback shift register (LSFR), but it was not until 1982 with the advent of the mass production of the CD that the digital information age as we know it was started. Immink states that “without error-correcting codes, digital audio would not be technical feasible” (Wicker & Bhargava, 1999). Today RS codes are widely in use in many applications that involve data transmission, like wireless computer networks; telephony: GSM, GPRS, UMTS; digital video broadcasting: DVB-T, DVC-C; and data storage, like hard disk drives (HDD) in computers. Memory cards in cameras and telephones, and optical storage like Compact Discs (CD), Digital Versatile Discs (DVD) and Blu-ray Discs (BD) also use Reed-Solomon codes.

1.3 Basics of Data Communication

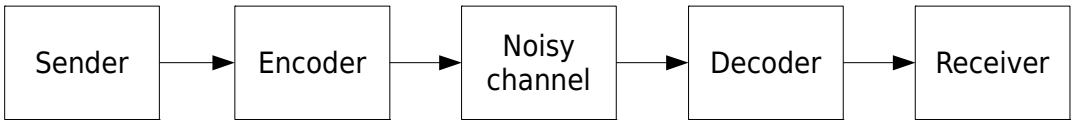

Figure 1.1: Simplified diagram of a digital transmission system

A sender transmits a message through a channel to a receiver. The channel could be air when using a wireless network or the channel could be a data cable. Noise may appear on these types of channels, so in order to receive the message with as few errors as possible, ideally the sender should use

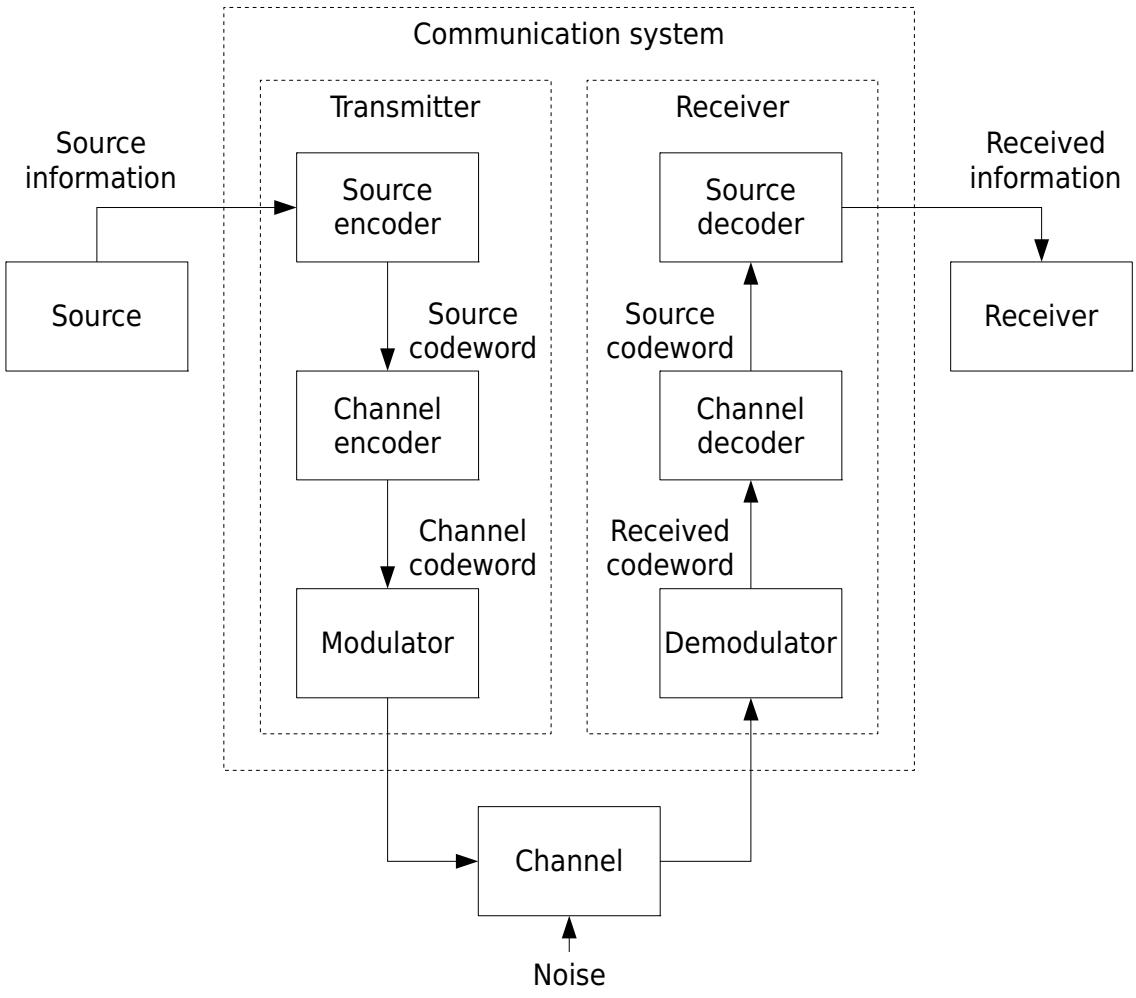

Figure 1.2: Diagram of a digital transmission system. (Blahut, 2003)

high power signal amplification and the channel should be as short as possible. However, in normal situations these are not viable solutions. GSM telephones have, in fact, very small batteries and are rather energy efficient and an Ethernet cable in a building can be up to 100 meters before an active repeater or switch has to amplify the signal. In order to use as little energy as possible and transmit over a long distance, codewords have to be encoded, as shown in Figure 1.1 and 1.2. The message is then transmitted over a channel where errors may be introduced. The received codeword needs to be decoded into the received message.

The probability that codeword r is received if codeword c is transmitted can be expressed as $P(r|c)$. In Figure 1.3, a model of a Binary Symmetric Channel (BSC) shows the event that a 1 is transmitted. There is a probability that 0 is received. The transmission is unaltered with a probability of $1-p$ (Bossert, 1999). Maximum-Likelihood Decoding in Section 3.3 gives a more in-depth view of this topic. This channel is characterised by the following conditional probabilities:

$$\begin{aligned} P(r=0|c=0) &= 1-p \\ P(r=1|c=0) &= p \\ P(r=0|c=1) &= p \\ P(r=1|c=1) &= 1-p \end{aligned}$$

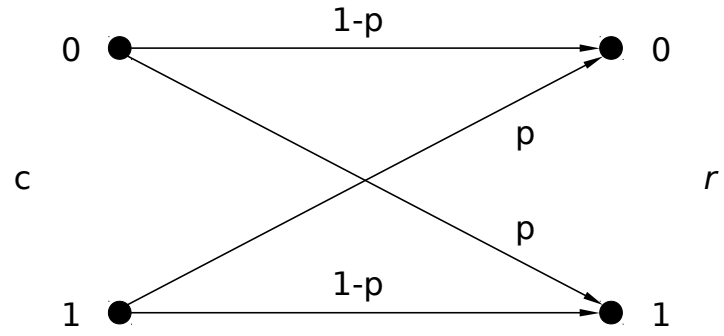

Figure 1.3: Model of the binary symmetric channel (BSC) (MacKay, 2003)

2 CODING THEORY BASICS

2.1 Linear Algebra

Mathematicians developed their coding theory using linear algebra, which works with sets of numbers or fields. These numbers can be added, subtracted, multiplied or divided. Fields like integers, the set of natural numbers $\mathbb{N}=\{0,1,2,\ldots\}$, are infinite fields; we could always imagine its largest element and add 1 to it.

Information and code can be seen as elements in a finite field which comes with some advantages when using the binary number system.

2.2 Galois Fields

A finite field F_q is a field F which has a finite number of elements and q is the order of the field. This finite field is often called a Galois field, after the French mathematician Évariste Galois (1811 – 1832) and is denoted $GF(q)$. For the purpose of this thesis we consider only binary field $GF(2)$ and its extension fields $GF(2^m)$ where $m \in \{2,3,4,\ldots\}$.

The following is always valid for all numbers in a binary Galois field (Blahut, 1983):

- fields contain 0 or 1.
- adding two numbers gives one number in the set.
- subtracting two numbers gives one number in the set.
- multiplying one number gives one number in the set.
- dividing one number by 1, as division by 0 is not allowed, gives one number in the set.
- The distributive law, $(a+b)c=ac+bc$, holds for all elements in the field.

A finite field, by definition, has to contain at least two numbers and, therefore, the smallest Galois field contains the elements or numbers 0 and 1, and is defined as $GF(2)=\{0,1\}$. Since we have a finite field with only aforementioned binary numbers, the addition of 1 and 1 in Table 2 cannot be equal to 2, but instead has to be defined as 1+1=0 where 2 is congruent to 0 modulo 2, or $2\equiv 0 \pmod 2$ (Hill, 1986). For subtraction we take $-a$ as the additive inverse of a. This inverse can be found by $a+b=c$ and we write it $b=c-a$ which is equal to $b=c+(-a)$. Substituting a and b with 0 and 1, we can see that the additive inverse of 0 is 0 and the additive inverse of 1 is 1.

Table 1: Addition for GF(2)

+	**0**	**1**
0	0	1
1	1	0

Table 2: Multiplication for GF(2)

*	**0**	**1**
0	0	0
1	0	1

Division is a multiplication with its multiplicative inverse of which we can write as:

$$\frac{a}{b}=c\,.$$

Therefore $a\cdot b^{-1}=c$ which results in $a=c\cdot b$. Because $a\cdot a^{-1}=1$, the multiplicative inverse of 1 is 1. Division is always possible for all except 0. Because division by zero is not defined and $0\cdot a^{-1}\neq 1$, zero has no multiplicative inverse.

2.3 Extension Fields

Finite fields exist for all prime numbers q and for all p^m where p is prime and m is a positive integer. $GF(q)$ is a sub-field of $GF(p^m)$ and as such the elements of $GF(q)$ are a sub-set of the elements of $GF(p^m)$, therefore $GF(p^m)$ is an extension field of $GF(q)$.

Table 3: Addition for GF(4)={0,1,2,3}

+	**0**	**1**	**2**	**3**
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

Table 4: Multiplication for GF(4)={0,1,2,3}

*	**0**	**1**	**2**	**3**
0	0	0	0	0
1	0	1	2	3
2	0	2	0	2
3	0	3	2	1

Consider $GF(4)=\{0,1,2,3\}$ in Table 3 and 4, which is not a Galois field because it is of order 4, which is not a prime. The element 2 has no multiplicative inverse and therefore we cannot divide by 2. Instead, we could define $GF(4)=\{0,1,a,b\}$ with addition and multiplication as shown in Table 5 and 6. Now all elements do have additive and multiplicative inverses.

Table 5: Addition for GF(4)={0,1,a,b}

+	**0**	**1**	**a**	**b**
0	0	1	a	b
1	1	0	b	a
a	a	b	0	1
b	b	a	1	0

Table 6: Multiplication for GF(4)={0,1,a,b}

*	**0**	**1**	**a**	**b**
0	0	0	0	0
1	0	1	a	b
a	0	a	b	1
b	0	b	1	a

These extension fields are used to handle non-binary codes where code symbols are expressed as m-bit binary code symbols, For example, $GF(4)$ consists of four different two-bit symbols and $GF(16)$ of 16 hexadecimal symbols. To obtain multiplication for binary, numbers are expressed as polynomials, they are multiplied and divided by the prime polynomial while the remainder is taken as result.

2.4 Polynomials

Let us we write $GF(4)$ as $GF(2^2)$ and take prime polynomial

$$p(x)=x^2+x+1$$

which is an irreducible polynomial of degree 2, which can be checked by multiplying $p(x)$ with polynomials of a lesser degree, like 1, x and $x+1$ (Blahut, 1983).

Table 7: Addition for GF(2²) in binary representation

+	**00**	**01**	**10**	**11**
00	00	01	10	11
01	01	00	11	01
10	10	11	00	01
11	11	10	01	00

Table 8: Multiplication for GF(2²) in binary representation

*	**00**	**01**	**10**	**11**
00	00	00	00	00
01	00	01	10	11
10	00	10	11	01
11	00	11	01	10

This gives us the structure of $GF(2^2)$ in Table 7 and 8. Note that addition in a finite field is equivalent to the logic exclusive OR (XOR) operation and multiplication is equivalent to the logic AND. In Table 9 and 10, $GF(2^2)$ is represented in polynomial form.

Table 9: Addition for GF(2²) in polynomial representation

+	**0**	**1**	**x**	**x+1**
0	0	1	x	x+1
1	1	0	x+1	x
x	x	x+1	0	1
x+1	x+1	x	1	0

Table 10: Multiplication for GF(2²) in polynomial representation

*	**0**	**1**	**x**	**x+1**
0	0	0	0	0
1	0	1	x	x+1
x	0	x	x+1	1
x+1	0	x+1	1	x

In order to describe an extension field $GF(p^m)$ it is useful to know its primitive polynomial $p(x)$, where the degree of $p(x)$ is equal to m. For example, $GF(16)=GF(2^4)=\{0000,0001,0010,\ldots,1111\}$ is a finite field that contains 16 4-bit code symbols. Addition is analogue to the example above. Multiplication can be obtained firstly by writing the symbols as polynomials to express which positions in these 4-bit codes are non-zero and, secondly, by using modulo 2 addition of coefficients in addition and multiplication.

Let α be defined as a root of polynomial $p(x)$, such that we can write:

$$p(\alpha)=0$$

Thus for $GF(16)$ with its irreducible polynomial $p(x)=x^4+x+1$ we can write:

$$\alpha^4+\alpha+1=0$$
$$\alpha^4=0-\alpha-1$$

We have already noted that subtraction is the same as addition in a binary finite field, so:

$$\alpha^4=\alpha+1$$

Therefore the polynomial of exponential α^4 is $\alpha+1$. From there we can calculate the polynomial for α^5 by:

$$\begin{aligned}\alpha^5&=\alpha\cdot\alpha^4\\&=\alpha\cdot(\alpha+1)\\&=\alpha^2+\alpha\end{aligned}$$

Now we can take $\alpha^k = \alpha \cdot \alpha^{k-1}$ for every $k < 2^m - 1$, where $m = 4$ in our example. Calculations for α^5 and α^6 in Table 11 are straight forward. However, polynomials of degree 4 may be reduced to ones of less than a degree of 4:

$$\begin{aligned}\alpha^7 &= \alpha \cdot \alpha^6 \\ &= \alpha \cdot (\alpha^3 + \alpha^2) \\ &= \alpha^4 + \alpha^3\end{aligned}$$

Substituting α^4 with $\alpha + 1$ gives

$$\begin{aligned}\alpha^7 &= \alpha + 1 + \alpha^3 \\ &= \alpha^3 + \alpha + 1\end{aligned}$$

so the polynomial of α^7 is $x^3 + x + 1$. By convention the degree of the zero polynomial is $-\infty$ (Hill, 1986). The remaining exponentials can be obtained in the same manner while keeping each polynomial of degree 3 or less because we can substitute α^4, a polynomial of degree 4, with $\alpha + 1$, which is of degree 1. Note that $\alpha^{15} = 1$. Fermat's Little Theorem says that $p^{m-1} \equiv 1 \pmod m$ where p is prime and m is a positive integer (Blahut, 1983) (Bossert, 1999).

Table 11: Elements of Galois Field GF(2^4) in different notations

Exponential	Degree	Algebraic	Polynomial	Binary	Decimal	Hexadecimal
0	-∞	0	0	0000	0	0
α^0	0	1	1	0001	1	1
α^1	1	αα	x	0010	2	2
α^2	2	α^2	x^2	0100	4	4
α^3	3	α^3	x^3	1000	8	8
α^4	4	α+1	x+1	0011	3	3
α^5	5	α(α+1)	x^2+x	0110	6	6

Exponential	Degree	Algebraic	Polynomial	Binary	Decimal	Hexadecimal
α^6	6	$\alpha(\alpha^2+\alpha)$	x^3+x^2	1100	12	C
α^7	7	$\alpha(\alpha^3+\alpha^2)$	x^3+x+1	1011	11	B
α^8	8	$\alpha(\alpha^3+\alpha+1)$	x^2+1	0101	5	5
α^9	9	$\alpha(\alpha^2+1)$	x^3+x	1010	10	A
α^{10}	10	$\alpha(\alpha^3+\alpha)$	x^2+x+1	0111	7	7
α^{11}	11	$\alpha(\alpha^2+\alpha+1)$	x^3+x^2+x	1110	14	E
α^{12}	12	$\alpha(\alpha^3+\alpha^2+\alpha)$	x^3+x^2+x+1	1111	15	F
α^{13}	13	$\alpha(\alpha^3+\alpha^2+\alpha+1)$	x^3+x^2+1	1101	13	D
α^{14}	14	$\alpha(\alpha^3+\alpha^2+1)$	x^3+1	1001	9	9

2.5 Vector Space

Linear codes can be represented as sets of vectors. Let us define a vector space $GF(q^m)$. This is a vector space of a finite dimension m. The codewords are q-ary sets of m-elements or m-tuples which form the coordinates of the endpoints of the vectors. Figure 2.1 presents two of such m-dimensional vector spaces. In such a vector space, every codeword can be presented as a the sum of two vectors give another vector in the same vector space (Bose, 2008). For example, $GF(2^2)$ is a two-dimensional vector space. It has four binary vectors.

Take vectors $v_1=[0,1]$, $v_2=[1,0]$ and $v_3=[1,1]$, then $v_1+v_2=[0,1]+[1,0]=[1,1]$, which is a vector in the same space. Vectors $v_{1,}v_{2,\ldots,}v_k$ are linear independent if there is not a single set of scalars $a_i\neq0$, such that $a_1v_1+a_2v_2+\ldots+a_kv_k=0$. For example, vectors $[0,1]$ and $[1,0]$ are linearly independent, but $[0,1]$ and $[1,1]$ are linear dependent vectors.

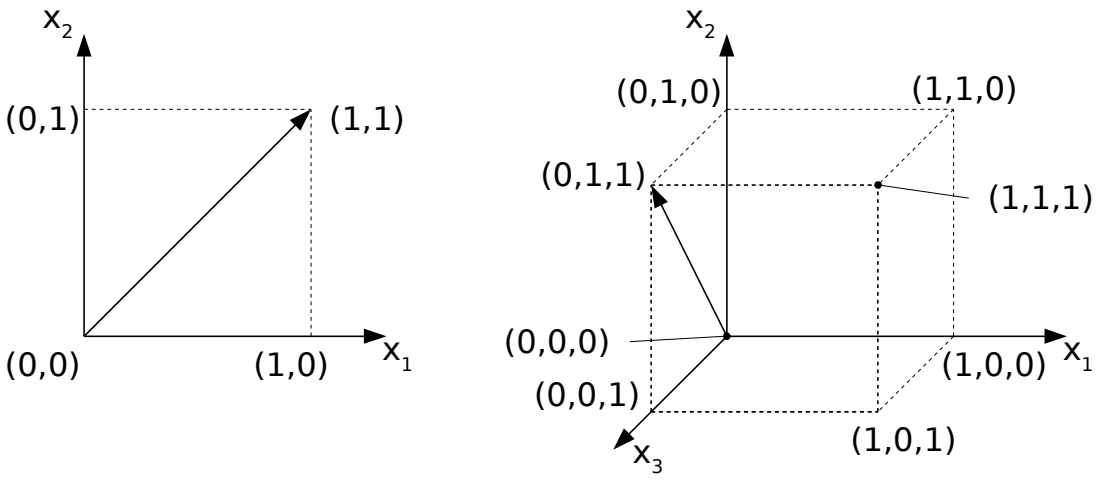

Figure 2.1: Codewords [1,1] and [0,1,1] as vectors over GF(2)

3 LINEAR BLOCK CODES

3.1 Hamming weight, minimum distance and code rate

The Hamming weight $w_H(x)$ of a codeword or vector x is defined as the amount of non-zero elements or vector coordinates, which ranges from zero to length n of said codeword.

$$w_H(x)=\sum_{j=0}^{n-1} w_H(x_j), \quad where \quad w_H(x_j)=\begin{cases}0, & x_j=0\\ 1, & x_j\neq 0\end{cases}$$

The Hamming distance $d_H(x,y)$ between two codewords or vectors x and y is defined as amount of elements or coordinates where x and y differ.

$$d_H(x,y)=\sum_{j=0}^{n-1} w_H(x_j+y_j), \quad where \quad w_H(x_j+y_j)=\begin{cases}0, & x_j=y_j\\ 1, & x_j\neq y_j\end{cases}$$

$$d_H(x,y)=w_H(x,y)$$

The minimum distance d_{min} of code C is the minimum distance between two different codewords. The minimum distance for linear codes is equal to the minimum weight (Bossert, 1999). However, a codeword containing only zeros and, therefore, having a distance of zero is disregarded as the minimum distance cannot be zero.

Let x, y be codewords in code C. A received vector, which is the sent vector x in C, plus error vector e can only be corrected if the distance between any other codeword y in C fulfil

$d_{min}(x,x+e)<d_{min}(y,x+e)$ or $w_{min}(e)<w_{min}(x+y+e)$.

Therefore $w_{min}(e)\leq\frac{d-1}{2}$, where d is the distance.

This is written as $t \leq \frac{d-1}{2}$ or $d \geq 2t+1$, where t is the amount of errors that can be corrected.

In general, a code C of length n, with M codewords, and a minimum distance $d = d(C)$, is called an (n, M, d) code. Then $M \leq q^{n-d+1}$ and the code rate of a q-ary (n, M, d) code is at most $1 - \frac{d-1}{n}$.

A linear q-ary code of length n, with k codewords or message symbols, and distance d, is called a (n, k, d) code or (n, k) code. The code rate is defined as

$$R = \frac{\log_q k}{n}$$

If, according to Shannon's channel coding theorem, rate R is less than capacity C, then the code exists but if rate R is larger than capacity C, the error probability is 1 and the length of the codeword becomes infinite.

3.2 Singleton bound

It is preferable to have a large minimum distance d so that many errors can be corrected. Also, a large amount of codewords M would allow for efficient use of bandwidth when transmitting over a noisy channel. Unfortunately, increasing d tends to increase n or decrease M. The Singleton bound is an upper bound for M in terms of n and d. A code that satisfies the Singleton bound is called a MDS code (maximum distance separable). The Singleton bound can be written as

$$q^d \leq \frac{q^{n+1}}{M}$$

for the MDS code to obtain the largest possible value of d for a given n and

M. Reed-Solomon codes are an important class of MDS codes (Trappe & Washington, 2006).

3.3 Maximum-Likelihood Decoding

There are two principles of decoding. In hard-decision decoding the received bits are believed to be either 1 or 0 in binary transmission. The decoding is done bit by bit. In soft-decision decoding, the received codewords may contain samples of bits with many values, not just 1 or 0. Calculating the closest error-free codeword is more complicated but soft-decision decoding has better performance than the hard-decision decoding.

Assuming hard-decision decoding is used, the received codeword is decoded into its closest codeword measured by its smallest Hamming distance. This minimum probability of error principle is called Maximum-Likelihood or Minimum Distance Decoding (Geisel, 1990).

The Model of the binary symmetric channel (BSC) (MacKay, 2003) in Figure 1.3 shows that the channel has binary input and output with an error probability, the channel is characterised by the following conditions if c is the transmitted code and r the received code:

$$\begin{array}{lcl} P(r=0|c=0) & = & 1-p \\ P(r=0|c=1) & = & p \\ P(r=1|c=0) & = & p \\ P(r=1|c=1) & = & 1-p \end{array} \quad \text{and} \quad 0 \leq p \leq \frac{1}{2}$$

Comparing all received codewords r to all transmitted codewords c as a direct way of correcting errors would not be inefficient. This means storing all 2^k code vectors and performing equally as many comparisons for each received codeword, resulting in error vectors of which the vector with the smallest distance is probably the transmitted codeword. A more practical decoding method would be Syndrome decoding which will be described in Section 3.5.

3.4 Hamming codes

Hamming constructed a code where he added three additional bits to four information bits. These additional or parity bits are chosen based on the information bits in the following manner:

$$p_1 = i_1 + i_2 + i_4$$
$$p_2 = i_1 + i_3 + i_4$$
$$p_3 = i_2 + i_3 + i_4$$

and form the codeword $c = (i_1, i_2, i_3, i_4, p_1, p_2, p_3)$. Hamming codes are block codes. This means that a fixed block of input data is processed into a fixed block of output data. A code is called a systematic code if the codeword starts with the information bits, followed by the parity bits, as shown in Figure 3.2. A non-systematic code has the information bits in a different order. The parity bits are the result of a modulo 2 addition, so if there is an even amount of bits, it gives 0 and 1 when there is an odd amount. If a single error occurs, i.e., a bit is flipped or reversed, the codeword no longer satisfies the equations.

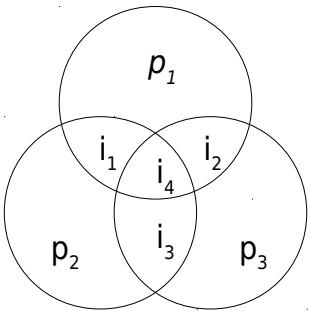

Figure 3.1: Relation between information and parity bits

Figure 3.2: An example of a systematic codeword of length n

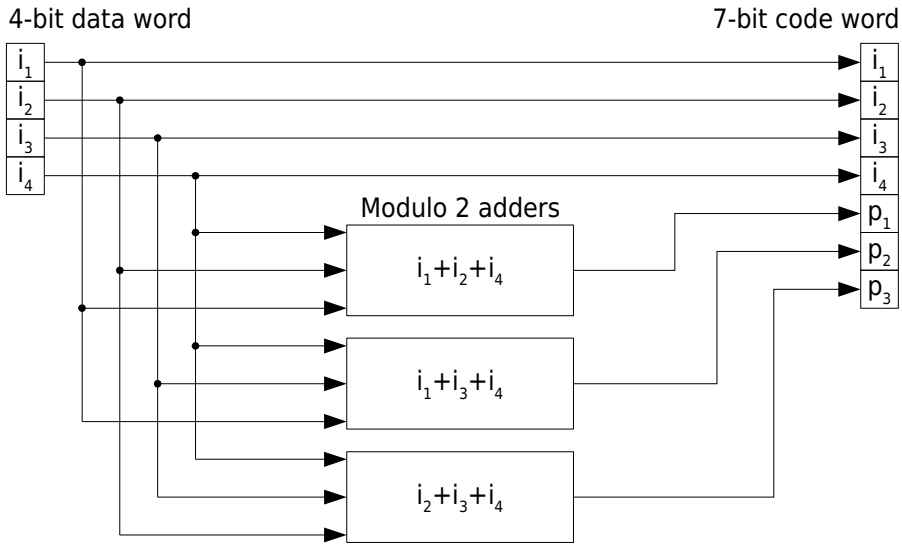

Figure 3.3: Hamming (7,4) encoder

The decoder receives a seven-bit codeword $r=(i'_1, i'_2, i'_3, i'_4, p'_1, p'_2, p'_3)$. With an algebraic method known as syndrome decoding it is possible to determine the position of the error:

$$s_1 = p'_1 + i'_1 + i'_2 + i'_4$$
$$s_2 = p'_2 + i'_1 + i'_3 + i'_4$$
$$s_3 = p'_3 + i'_2 + i'_3 + i'_4$$

The three-bit syndrome (s_1, s_2, s_3) returns $(0,0,0)$ when a received codeword contains no errors. There are seven more possible syndromes, each corresponding to the position of the error in the received codeword. The decoder then inverts the detected bit to counter the error.

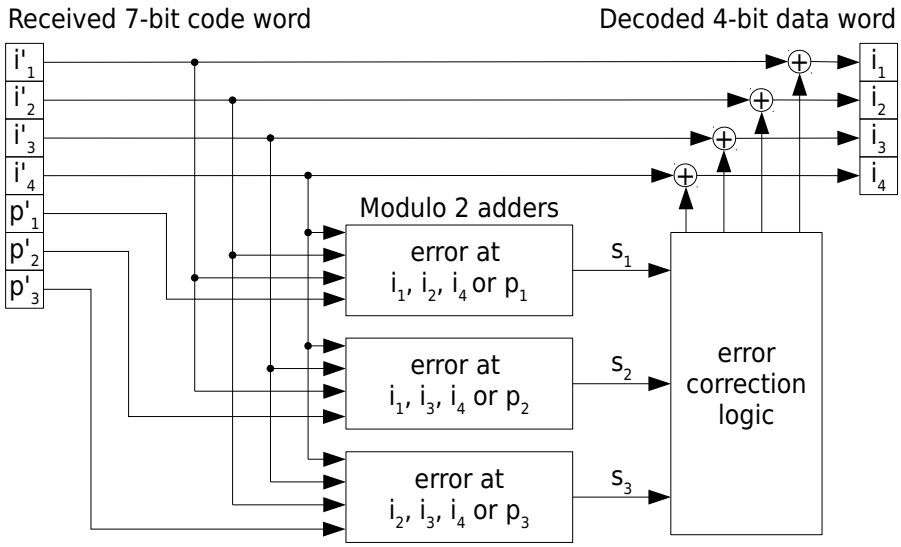

Figure 3.4: Hamming (7,4) decoder

Bose (2008) considered the space of q-ary m-tuples, where every q-ary vector of length m can be represented by its endpoint in this space. Hence, we can represent every codeword as a point in this space, and all codewords at a Hamming distance of t or less would lie within the sphere centred at the codeword and with a radius of t.

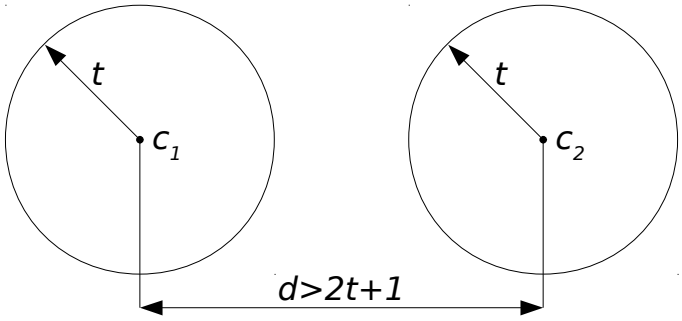

Figure 3.5: Decoding sphere

3.5 Syndrome decoding

Trappe and Washington (2006) define linear (n,k) code of dimension k and length n over a field F as a k-dimensional subspace of F^n. For example, a

linear binary code, of length n and dimension k is a set of 2^k binary codewords or n-tuples, such that the sum of any two codewords is always a codeword. To construct such a linear (n,k) code, we choose a $k \times n$ matrix known as generator matrix. The rows have to be linearly independent to produce unique codewords.

Generator matrix G is taken so that $G=[I_k, P]$, where I_k is the $k \times k$ identity matrix which determine the codewords and P is a $k \times (n-k)$ matrix that provides redundancy, the parity matrix. Now every codeword c of code C can be expressed as a linear combination of rows of G by $c = i \cdot G$. We can now calculate the generator matrix for a systematic representation. For example, a systematic Hamming (7,4) code has the following generator matrix:

$$G = [I_4|P] = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

The parity check matrix is then calculated as

$$H = [-P^T|I_3] = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

For example, encoding information bits $[1100]$ gives

$$[1 \quad 1 \quad 0 \quad 0] \times \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \equiv [1 \quad 1 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1]$$

Decoding received codeword $c'=[1100011]$ with syndrome decoding results in $[000]$ when no errors are detected. However, in our example an error was introduced in the fifth position of codeword $c'=[1100111]$, so we can expect a syndrome with non-zero elements.

$$c' \times H^T = [1 \quad 1 \quad 0 \quad 0 \quad \underline{1} \quad 1 \quad 1] \times \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ \underline{1} & \underline{0} & \underline{0} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \equiv [1 \quad 0 \quad 0]$$

The value $[100]$ can be looked up in parity check matrix H and tells that the error occurred in the fifth position from the left. Correction based on syndrome requires more steps and asks for a matrix of all single error vectors. Codeword $c=[1100011]$ and received codeword $c'=[1100111]$ give an error vector of $e=[0000100]$ or $c=c'+e$. Since we already know that $s=c'\cdot H^T$ and an error-free $c\cdot H^T$ has a syndrome with all zero elements, we now substitute c' with $c+e$ because $c=c'+e$ is equivalent to $c'=c+e$ in binary.

$$\begin{aligned} s &= c'\cdot H^T \\ &= (c+e)\cdot H^T \\ &= c\cdot H^T + e\cdot H^T \\ &= 0 + e\cdot H^T \\ &= e\cdot H^T \end{aligned}$$

We can conclude that the syndrome solely depends on the error pattern and not on the transmitted codeword.

$$e = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad s = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

3.6 Cyclic codes

Cyclic codes are widely used in data communication because their structure makes encoder and decoder circuitry simple. Hill (1986) defines code C as cyclic (n,k) code if C is a linear code of length n over a finite field and if any cyclic shift of a codeword is also a codeword. Thus,

$$(c_0, c_1, c_2, ..., c_{n-1}) \in C \text{ and } (c_{n-1}, c_0, c_1, ..., c_{n-2}) \in C.$$

Let $g(x)$ be the polynomial with the smallest degree. By dividing its highest coefficient, we may assume that the highest non-zero coefficient of $g(x)$ is 1. The polynomial $g(x)$ is called the generator polynomial for C, which must be a divisor of x^n-1 (in a binary field this is equal to x^n+1) with a degree of $n-k$. Subsequently, every cyclic code is a polynomial (Trappe & Washington, 2006).

The encoder for cyclic codes is then

$$c(x)=i(x)\cdot g(x)$$

where $c(x)$ is the polynomial with degree $n-1$ of codeword $(c_0, c_1, c_2, ..., c_{n-1})$ which is calculated as

$$c(x) = \sum_{i=0}^{n-1} c_i \cdot x^i = c_0+c_1 x+c_2 x^2+...+c_{n-1} x^{n-1}$$

and $i(x)$ is the information polynomial of degree $k-1$. Generator polynomial $g(x)$ must be of degree $n-k$.

3.7 BCH codes

A BCH code is a cyclic polynomial code over a finite field with a particularly chosen generator polynomial. Hamming codes are the subset of BCH codes with $k=2^m-1-m$ and have an error correction of 1. Generally, a family of t-

error correcting codes defined over finite fields $GF(q)$, where $2t+1<q$, are BCH codes or RS codes (Hill, 1986). The main advantage of BCH codes is the ease with which they can be decoded using syndrome and many good decoding algorithms exist. A well-known decoding algorithm is the Berlekamp-Massey algorithm. This allows very simple electronic hardware to perform the task, making the need for a computer unnecessary. This implies that a decoding device may be small and consume little power. BCH codes allow control over block length and acceptable error thresholds, which makes them very flexible. This indicates that code can be designed to meet custom requirements. Another reason they are important is that there exist good decoding algorithms that correct multiple errors. Hocquenghem, as well as Bose and Ray-Chaudhuri, discovered the class of BCH codes, but not the decoding. Peterson developed the first decoding algorithm in 1960 followed by refinement from Berlekamp, Massey and many others (Trappe & Washington, 2006).

3.7.1 Generating BCH code

It is easy to generalise the construction of a t-error-correcting code of length $n=2^m-1$ over $GF(q)=\{0,1,...,q-1\}$ provided $2t+1\leq n\leq q-1$. According to Hill (1986) it is not difficult to construct a binary BCH code over an extension field $GF(q^m)$. In order to obtain a cyclic code only the generator polynomial $g(x)$ is needed. For any integer $m\geq 3$ and $t<2^{m-1}$, there exists a primitive BCH code with parameters:

$$\begin{aligned} n &= 2^m-1 \\ n-k &\leq m\cdot t \\ d_{min} &\leq 2t+1 \end{aligned}$$

Let α be a primitive n-th root of unity of $GF(2^m)$. For $1\leq i\leq t$, let $m_{2i-1}(x)$ be the minimum polynomial of α_{2i-1}. The degree of $m_{2i-1}(x)$ is m or a factor of m. The generator polynomial $g(x)$ of a t-error-correcting primitive BCH

codes of length 2^m-1 is given by

$$g(x)=Least\,Common\,Multiple\{m_1(x),m_2(x),m_3(x),...,m_{2t-1}(x),m_{2t}(x)\}$$

and because every even power of a primitive element has the same minimal polynomial as some odd power of the element, then $g(x)$ can be reduced to

$$g(x)=LCM\{m_1(x),m_3(x),...,m_{2t-1}(x)\}$$

The degree of $g(x)$ is $m\cdot t$ or less and so is the number of parity check bits, therefore $n-k\leq m\cdot t$ (van Lint, 1999).

Generally a code is a BCH code over $GF(q)$ with $m,n,d,c \in \mathbb{N}$ chosen such that q is a prime power and $2\leq d\leq n$. Also, m is the multiplicative order of q modulo n and n is not divisible by q, so the greatest common divisor of n and q is 1 (Lidl & Pilz, 1998). In special circumstances it is that,

- A BCH code with $c=1$ is called a narrow-sense BCH code;
- A BCH code with $n=q^m-1$ is called primitive;
- A narrow-sense BCH code with $n=q^m-1$ is called a Reed-Solomon code.

The consecutive roots of the generator polynomial may run from $\alpha^c,...,\alpha^{c+d-2}$ instead of $\alpha,...,\alpha^{d-1}$. As before, let α be a primitive n-th root of unity in $GF(q^m)$, and let $m_i(x)$ be the minimal polynomial over $GF(q)$ of α_i for all i. The generator polynomial of the BCH code is defined as the least common multiple $g(x)=LCM\{m_c(x),...,m_{c+d-2}(x)\}$ (Trappe & Washington, 2006).

3.7.2 Decoding a BCH code

BCH codes can be decoded in many way and it is most common that

- Syndromes values for are calculated for the received codeword;
- Error polynomials are calculated;
- Roots of these polynomials are calculated to obtain the location of errors;
- Error values are calculated at these locations.

Let code C be a binary BCH code with distance $d \geq 3$. C is a cyclic code of length n, with generating polynomial $g(x)$. There is a n-th root of unity α such that

$$g(\alpha^{k+1}) = g(\alpha^{k+2}) = 0$$

for some integer k.

Let $$H = \begin{bmatrix} 1 & \alpha^{(k+1)} & \alpha^{2(k+1)} & \dots & \alpha^{(n-1)(k+1)} \\ 1 & \alpha^{(k+2)} & \alpha^{2(k+2)} & \dots & \alpha^{(n-1)(k+2)} \end{bmatrix}.$$

If $c = (c_0, \dots, c_{n-1})$ is a codeword, then polynomial $m(x) = c_0 + c_1 x + \dots + c_{n-1} x^{n-1}$ is a multiple of $g(x)$, so

$$m(\alpha^{k+1}) = m(\alpha^{k+2}) = 0$$

This may be rewritten in terms of H:

$$cH^T = [c_0, \dots, c_{n-1}] \begin{bmatrix} 1 & 1 \\ \alpha^{(k+1)} & \alpha^{(k+2)} \\ \alpha^{2(k+1)} & \alpha^{2(k+2)} \\ \vdots & \vdots \\ \alpha^{(n-1)(k+1)} & \alpha^{(n-1)(k+2)} \end{bmatrix} = 0 .$$

H is not necessarily a parity matrix for C, however, it can correct an error.

Suppose codeword $c' = c + e$ is received with error vector $e = (e_0, \dots, e_{n-1})$. Assuming that there is one error, the algorithm for correcting one error is to write $c' \cdot H^T = (s_1, s_2)$.

- If $s_1 = 0$ then there is either no error or more than one error and we stop here.
- If $s_1 \neq 0$, take $\frac{s_2}{s_1}$ which results in a power α^{j-1} of α.

The error is in position j and $e_j = 1$. Subtracting the error vector e from the received codeword c' gives the corrected codeword c'_r. For binary BCH

codes it is only necessary to calculate the position, because the error value is always equal to 1. In non-binary BCH codes an additional error value polynomial is needed (Trappe & Washington, 2006).

3.8 Reed-Solomon codes

RS codes, which are BCH codes, are used in applications such as spacecraft communications, compact disc players, disk drives, and two-dimensional bar codes. According to Bossert (1999) the relationship between BCH and RS codes is such that RS codes comprise a subset of BCH codes and occasionally BCH codes comprise a subset of RS codes. Van Lint (1999) defines an RS code as a primitive BCH code of length $n=q-1$ over $GF(q)$.

3.8.1 Generating a Reed-Solomon code

Let $GF(q)$ be a finite field with q elements and it generate a rather specific BCH code C over $GF(q)$ of length n, called a Reed-Solomon code. Let α be a primitive n-th root of unity of $GF(q)$ and let code C have a length of $n=q-1$. Now take d so that $1 \leq d \leq n$ and the generator polynomial $g(x)$ is given by

$$\begin{aligned} g(x) &= \prod_{i=1}^{d-1} (x-\alpha^i) \\ &= (x-\alpha)(x-\alpha^2)\dots(x-a^{d-1}) \end{aligned} .$$

Trappe and Washington (2006) state that the minimum distance for C is at least d. Since $g(x)$ is a polynomial of degree $d-1$, it has at most d non-zero coefficients. Therefore, the codeword corresponding to the coefficients of $g(x)$ has a weight of at most d. It follows that C has a weight of exactly d and the dimension of C is n minus the degree of $g(x)$

$$n-deg(g) = n-(d-1) = n+1-d .$$

Therefore, a Reed-Solomon code is a cyclic $(n, n+1-d, d)$ code with codewords corresponding to polynomials, where each $f(x)$ is a polynomial with coefficients in $GF(q)$ that cannot be factored into lower degree polynomials while assuming that the highest non-zero coefficient is 1:

$g(x)f(x)$ with $deg(f) \leq n-d$.

It follows that there are q choices for each $n-d+1$ coefficients of $f(x)$, and thus there are q^{n-d+1} codewords in code C. Therefore, an RS code is a MDS code since it makes the Singleton bound an equality.

4 VISUALISATION

We consider two kinds of errors: random errors, which are distributed randomly among individual bits; and burst errors, which occur in consecutive groups of hundreds of bits. Burst errors are usually the result of, for example, fingerprints, dust and scratches on the disc surface (Wicker & Bhargava, 1999).

Additionally to the BSC as described in Section 3.3 we should mention the Binary Erasure Channel (BEC), in case a codeword is transmitted, but nothing is received. Let c be the transmitted code with alphabet $\{0,1\}$, let r be the received code with alphabet $\{0,1,e\}$ where e denotes the erasure. This channel is characterised by the following conditional probabilities:

$$\begin{aligned} P(r=0|c=0) &= 1-p \\ P(r=e|c=0) &= p \\ P(r=1|c=0) &= 0 \\ P(r=0|c=1) &= 0 \\ P(r=e|c=1) &= p \\ P(r=1|c=1) &= 1-p \end{aligned}$$

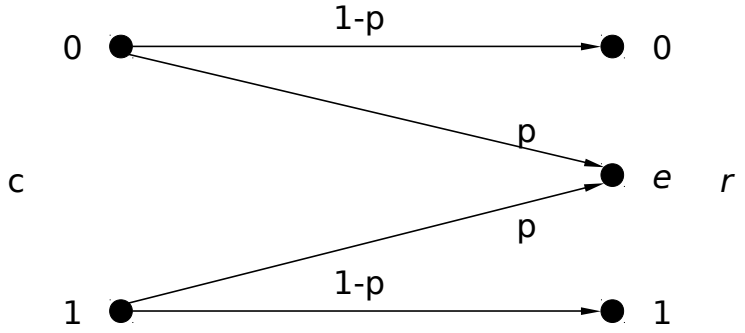

Figure 4.1: Model of the binary erasure channel (BEC) (MacKay, 2003)

Cross-interleaved Reed-Solomon code (CIRC) is well suited to deal with combinations of random as well as burst errors. For example, CIRC is used in compact discs with requirements such as:

- low redundancy
- the ability to correct random errors and burst errors
- good possibility of error concealment in case the correction capacity is surpassed.

4.1 Bit stream encoding

An analog-to-digital converter (ADC) converts sound into a digital data stream at a sample rate of 44.1 kHz, which, according to Nyquist's sampling theorem, is sufficient to reproduce a maximum audio frequency of 20 kHz. The 32-bit stereo sample has 16 bits for the left and 16 bits for the right channel. Six of these samples are then grouped in a frame of 32 audio bits, 16-bit per audio channel. The net audio bit stream is therefore $44\,100 \times 32 = 1.41$ Mbits/s. These

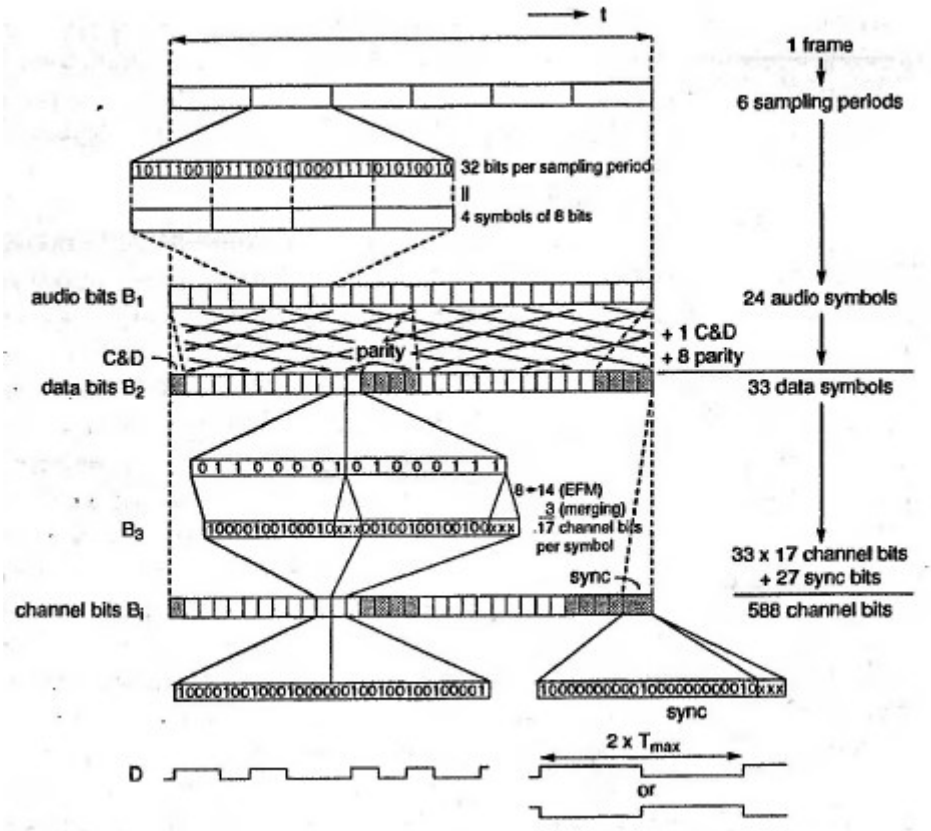

Figure 4.2: Bit streams in the encoding process (Wicker & Bhargava, 1999)

samples are then each divided to form 24 8-bit symbols per frame. Figure 4.2 shows this as bit stream B_1. In B_2, 8 parity symbols and a control and display symbol (C & D) are added such that each frame now contains 33 data symbols. The C & D symbol contains information for the listener which can be shown if the player has a display. Subsequently an eight-to-fourteen (EFM) code is used to translate these into 14-bit symbols plus three merging bits in B_3. This brings the net data bit stream rate to 1.94 Mbits/s. Then a synchronisation pattern of 27 bits is added to the frame to obtain bit stream B_i of $33\times17+27=588$ channel bits per frame in such a way that each 1 indicates a pit edge; it therefore makes no difference if pit and land were interchanged on a disc. The total bit rate after all these data manipulations is approximately 4.32 Mbits/s (Wicker & Bhargava, 1999).

4.2 Cross-interleaved Reed-Solomon Code (CIRC)

Cross-interleaving separates the symbols in a codeword, as codewords undergo a second encoding on a symbol basis. It becomes less likely that a burst from the outer decoder disturbs more than one Reed-Solomon symbol in any one codeword in the inner code.

Since the information in CIRC is interleaved in time, errors that occur at the input of the error correction system are spread over a large number of frames during decoding. The error correction system can correct a burst of thousands of data bits because the errors are spread out by interleaving. If more than the permitted amount of errors occur, they can only be detected.

The audio signal degrades gracefully by applying interpolation or muting the output signal. Key parameters to the success of concealment of errors are thoughtful positioning of the left and right audio channels as well as placing audio samples on even- and odd-numbered instants within the interleaving scheme. There are several interleaved structures used in the CD which allow

for error detection and correction with a minimum of redundancy.

A simple interleaving method is block interleaving. In block interleaving, a block of data symbols is written row by row as a $n \times m$ matrix and read column by column. It is obvious that the interleaver requires a memory capacity of $n \times m$ symbols. The CD uses a more effective interleaver, a periodic or convolutional interleaver, known as a cross-interleaver. Before transmission the symbols of the codewords are multiplexed over delay lines with differing delays, combined (demultiplexed) and send to the channel. At the receiver this process is reversed. In Figure 4.3 two Reed-Solomon codes, C_1 and C_2, are interleaved cross-wise. Outer code C_1 is RS(32,28) and inner code C_2 is RS(28,24). The symbols are 8 bits long and are elements of $GF(2^8)$. The code rate is $\frac{k_1}{n_1} \times \frac{k_2}{n_2} = \frac{3}{4}$ and for both codes the minimum distance is 5 which allows for a correction of maximum two errors in one code or four erasures. Each information frame contains 6 right and 6 left channel audio samples, denoted by R and L. Each 16-bit sample is divided into two 8-bit symbols or bytes (W) and the even- and odd-numbered audio samples are subjected to a delay of two bytes (2D). The 24 bytes are regrouped and the even- and odd-numbered samples are further separated by the parity bytes of code C_2 (Q). These 28 bytes are multiplexed and subjected to 28 different delays (1D to 27D) analogous to the convolutional interleaver as mentioned above. As a result of the convolutional interleave, one C_2 code is stored in 28 different blocks spread over 109 blocks. The required memory for a delay operator $D = 4$ is computed as $4 \times 27 \times 28/2 = 1502$ bytes. Encoder C_1 forms four parity bytes (P) after which a delay of 1 byte is inserted every other line in order to separate two adjacent symbol errors which are a result of small burst errors. Parity bytes P are inverted to prevent all zero codewords. This is important for detection of bit insertions or deletions (Wicker & Bhargava, 1999).

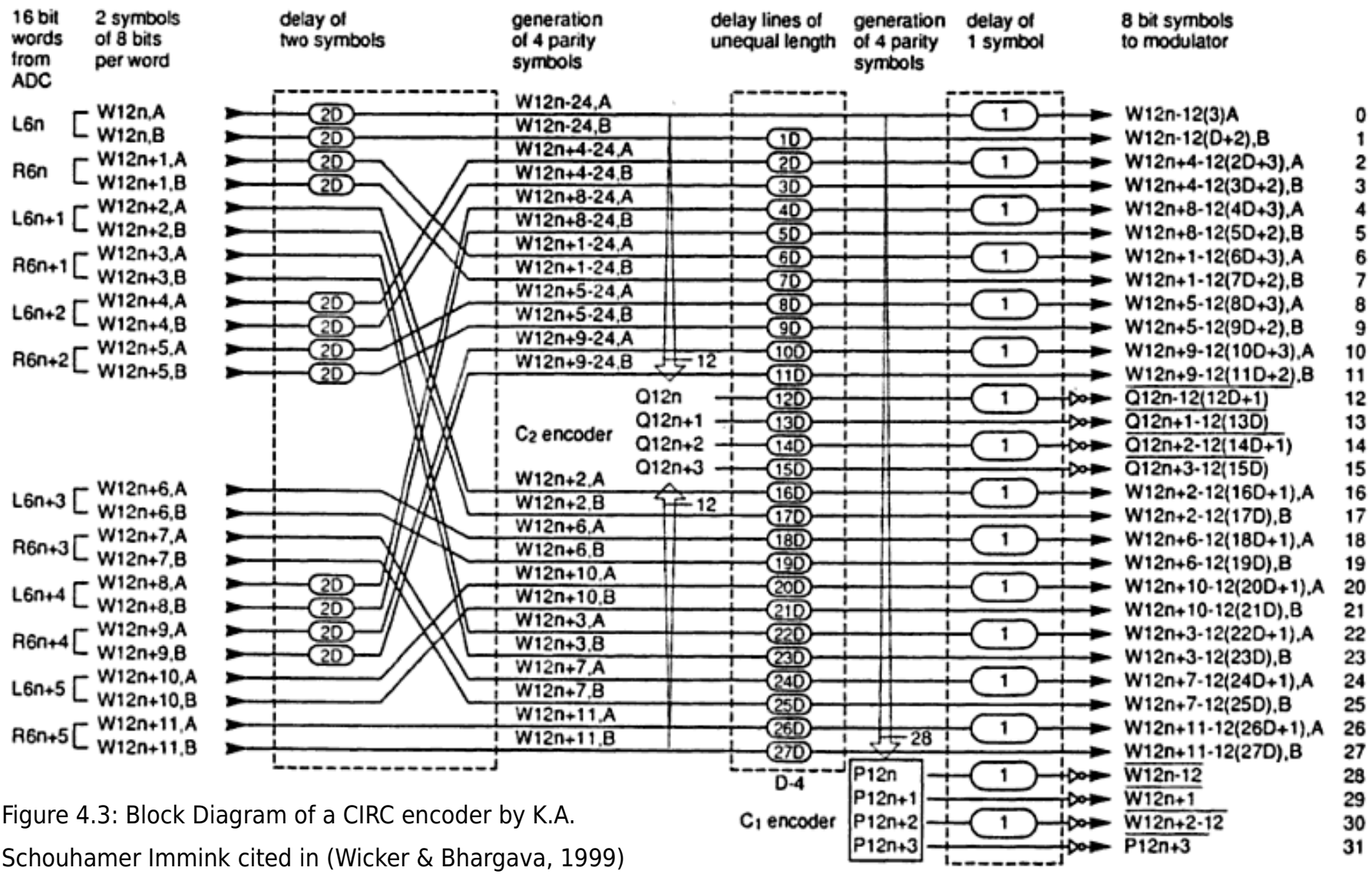

Figure 4.3: Block Diagram of a CIRC encoder by K.A. Schouhamer Immink cited in (Wicker & Bhargava, 1999)

4.3 Decoding

While the encoder has been standardised in the *Red Book audio specifications*, the decoder strategy has not been standardised. Each manufacturer is, therefore, free to choose their own decoding strategy. Analogous to the encoding process, the error correction system consists of two decoders D_1 and D_2. In most strategies, decoder D_1 corrects one error. If more than one error may occur in the 32 bits from the demodulator, D_1 attaches an erasure flag to the 28 outgoing symbols. Erasures will be spread over a number of codewords at the input of D_2. Decoder D_2 can at most correct four erasures. If more than four erasures may occur D_2 attaches an erasure flag to the 24 outgoing symbols. These flags allow the concealment system to react to the unreliable signal. The maximum fully correctable burst length and the maximum interpolation length are determined by the CIRC format. Four blocks are correctable, since code C_2 is quadruple- erasure-correcting and the maximum fully correctable burst error is about 4000 data bits. This corresponds to a track length of 2.5 mm on a CD, where an effective length on track of data bits of about 0.6 µm. About 50 blocks, roughly 12000 bits, can be concealed by interpolation. This corresponds to close to 7.5 mm of track length. Given the standardised format of the CD, a designer of a decoding integrated circuit (IC) can choose a certain decoding strategy (Wicker & Bhargava, 1999).

One way of decoding these codes depend on simultaneously solving a linear system of equations (LSE). The Berlekamp-Massey algorithm is a way of solving an LSE, but the inner working of this mechanism is not in the scope of this thesis.

5 SUMMARY AND CONCLUSION

A forward error correction code can be used effectively to detect and correct burst errors. Cross-interleave schemes with two error-correcting codes allow the outer code to declare burst errors so the inner code can correct these, because they are dispersed over codewords. The design of the interleaver and regrouping data symbols accommodate detection and correction of both random and burst errors while concealment of errors is possible if the correction capacity is exceeded.

The visualisation of the mechanism of cross-interleaving by K.A. Schouhamer Immink, one of the original designers of the compact disc standard, has explained many aspects of data encoding. However, proper visualisation and mathematical understanding of a decoder may require future studies in discrete mathematics and coding theory.

6 REFERENCES

Blahut, R.E. (1983) *Theory and Practice of Error Control Codes*. Reading: Addison-Wesley Pub. Co.

Blahut, R.E. (2003) *Algebraic codes for data transmission*. Cambridge: University Press.

Bose, R. (2008) *Information theory, coding and cryptography*. 2nd ed. New Delhi: Tata McGraw-Hill Publishing Company Ltd.

Bossert, M. (1999) *Channel Coding For Telecommunications*. New York: John Wiley & Sons.

Geisel, W.A. (1990) *Tutorial on Reed-Solomon error correction coding*. Houston: NASA, Lyndon B. Johnson Space Center.

Hill, R. (1986) *A First Course in Coding Theory*. Oxford: Clarendon Press.

Hocquenghem, A. (1959) "Codes correcteurs d'erreurs", *Chiffres*, 2, pp.147-156.

Lidl, R. & Pilz, G. (1998) *Applied Abstract Algebra*. 2nd ed. Berlin: Springer Verlag.

Lint, J.H. van (1999) *Introduction to coding theory*. 3rd ed. Berlin: Springer Verlag.

MacKay, D.J.C. (2003) *Information theory, inference, and learning algorithms*. Cambridge: University Press.

Massey, J.L. (1969) "Shift-register synthesis and BCH decoding", *IEEE Trans. Information Theory*, T-15 (1), January, pp.122-127.

Trappe, W. & Washington, L.C. (2006) *Introduction to cryptography: with coding theory*. 2nd ed. New Jersey: Pearson Prentice Hall.

Wicker, S.B. & Bhargava, V.K. (1999) *Reed-Solomon Codes and Their Applications*. New York: John Wiley & Sons.