A DESIGN OF NONPRIME MODIFIED ARRAY LDPC CODES FOR MAGNETIC RECORDING APPLICATIONS

CHALIT CHUSIN

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENT FOR THE DEGREE OF
MASTER OF ENGINEERING IN DATA STORAGE TECHNOLOGY
COLLEGE OF DATA STORAGE INNOVATION
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
2013

KMITL-2012-DS-M-001-04

หัวข้อวิทยานิพนธ์	**การออกแบบรหัสแอลดีพีซีแบบมอดดิฟายอาร์เรย์ที่ไม่เป็นจำนวนเฉพาะเพื่อการประยุกต์ใช้กับการบันทึกเชิงแม่เหล็ก**
นักศึกษา	**นายชลิต ชูสิน**
รหัสประจำตัว	51068910
ปริญญา	**วิศวกรรมศาสตรมหาบัณฑิต**
สาขาวิชา	**เทคโนโลยีการบันทึกข้อมูล**
พ.ศ.	2556
อาจารย์ที่ปรึกษาวิทยานิพนธ์	**รศ.ดร.สมศักดิ์ ชุมช่วย**

บทคัดย่อ

รหัสแอลดีพีซีแบบมอดดิฟายอาร์เรย์เป็นรหัสแอลดีพีซีชนิดหนึ่งที่มีอัตรารหัสที่สูงและสามารถทำงานได้อย่างมีประสิทธิภาพบนช่องสัญญาณแบบอุดมคติที่มีเพียงสัญญาณรบกวนเกาส์แบบขาว (Additive White Gaussian Noise: AWGN) อย่างไรก็ตามรหัสแอลดีพีซีชนิดนี้ไม่สามารถสนับสนุนการออกแบบสำหรับทุกความยาวรหัสได้ เพราะว่าประสิทธิภาพของรหัสชนิดนี้จะถูกจำกัดโดยการใช้เมทริกซ์ย่อยแบบเมทริกซ์จัตุรัสที่มีจำนวนหลักและจำนวนแถวเป็นจำนวนเฉพาะเท่านั้น ถ้าเราออกแบบเมทริกซ์พาริตีเช็กโดยใช้เมทริกซ์ย่อยที่ไม่เป็นจำนวนเฉพาะจะเกิดไซเคิลขนาดเท่ากับ 4 ในเมทริกซ์พาริตีเช็ก

วิทยานิพนธ์นี้จึงนำเสนอวิธีการออกแบบเมทริกซ์พาริตีสำหรับรหัสแอลดีพีซีแบบมอดดิฟายอาร์เรย์ที่สามารถใช้ได้กับหลายความยาวรหัส โดยการออกแบบวิธีนี้สามารถลดจำนวนไซเคิลขนาดเท่ากับ 4 ในเมทริกซ์พาริตีเช็ก เมื่อใช้เมทริกซ์ย่อยแบบเมทริกซ์จัตุรัสที่มีจำนวนหลัก และจำนวนแถวไม่เป็นจำนวนเฉพาะได้ เราได้ทำการแบ่งการออกแบบเป็น 3 ขั้นตอนหลัก เพื่อที่จะทำให้เมทริกซ์พาริตีเช็ก มีจำนวนไซเคิลขนาดเท่ากับ 4 น้อยที่สุด ขั้นตอนที่ 1 จะทำการแทนที่เมทริกซ์สลับตำแหน่งแบบดั้งเดิมของรหัสแอลดีพีซีแบบมอดดิฟายอาร์เรย์ ด้วยเมทริกซ์สลับตำแหน่ง P_{SC} โดยเมทริกซ์ชนิดนี้ ได้ออกแบบเพื่อการป้องกันการเกิดซ้ำกันของเมทริกซ์เดียวกันบนแถวเดียวกัน ขั้นตอนที่ 2 จะทำการ cyclic shift บนแถวของเมทริกซ์ที่ได้จากขั้นตอนที่ 1 ขั้นตอนสุดท้ายจะทำการจัดรูปแบบของพาริตีเช็กเมทริกซ์ ให้อยู่ในรูปของสามเหลี่ยมล่าง โดยเมทริกซ์ลักษณะนี้จะมีคุณสมบัติในการเข้ารหัสอย่างง่ายได้ เราได้ทดสอบผลลัพธ์ของการออกแบบโดยเปรียบเทียบประสิทธิภาพของรหัสที่ออกแบบกับรหัสแอลดีพีซีแบบมอดดิฟายอาร์เรย์ ที่ออกแบบโดยใช้เมทริกซ์ย่อยแบบเมทริกซ์จัตุรัสที่มีจำนวนหลักและจำนวนแถวเป็นจำนวนเฉพาะ โดยการทดลองนี้จะใช้อัตราการเข้ารหัสที่ 0.9 และความยาวรหัสเท่ากับ 512 2,000 และ 4,000 บิต ผลการจำลองสมรรถนะการทำงานพบว่า แม้จะใช้เมทริกซ์จัตุรัสที่มีจำนวนหลักและจำนวนแถวไม่เป็นจำนวนเฉพาะในการออกแบบพาริตีเช็กเมทริกซ์ ค่าอัตราความผิดพลาดบิตข้อมูลของรหัสแบบมอดดิฟายอาร์เรย์ ยังคงมีประสิทธิภาพบนช่องสัญญาณรบกวนเกาส์แบบขาวเหมือนกับรหัสมอดดิฟายอาร์เรย์ที่ใช้เมทริกซ์ย่อยที่เป็นจำนวนเฉพาะ ในวิทยานิพนธ์ฉบับนี้ยังได้นำเสนอตัวอย่างของการนำพาริตีเช็กเมทริกซ์ที่ออกแบบไปประยุกต์ใช้กับรหัสแอลดีพีซีชนิดอื่นด้วย เช่น การนำไปใช้กับรหัสแอลดีพีซีแบบอินเตอร์ลีฟมอดดิฟายอาร์เรย์และรหัสแอลดีพีซีที่ใช้วิธีการรวมพาริตีเช็กเมทริกซ์เข้าด้วยกันโดยใช้ทฤษฎีบทเศษเหลือจีน

Thesis	A Design of Nonprime Modified Array LDPC Codes for Magnetic Recording Applications
Student	Mr.Chalit Chusin
Student ID.	51068910
Degree	Master of Engineering
Program	Data Storage Technology
Year	2013
Thesis Advisor	Assoc.Prof.Dr.Somsak Choomchuay

ABSTRACT

Modified Array LDPC code is one among several type of LDPC codes. These codes are high rate codes that can provide good error rate performance in Additive White Gaussian Noise (AWGN) channels. However, these codes do not support arbitrary code lengths, because the code length of a Modified Array LDPC code with good error rate performance is limited to a submatrix size that to be a prime number. If we design the parity check matrix with nonprime submatirx to support arbitrary code length, many cycles 4 will be generated in its parity check matrix.

In this thesis, we propose a design of the Modified Array LDPC code that can support arbitrary code length. This method will reduce the number cycles 4 when designed with nonprime submatrix size. We separate the design into 3 steps to ensure that the designed matrix contains few or no cycle 4. First, we replace the original circulant permutation matrices in MAC with P_{sc} (j, k). This parameter was designed to avoid the same degree of shifting used by the permutation matrix in the same row. Secondly, we define a new matrix as a result of cyclically shifting of the rows of the parity check matrix in a blockwise manner. The identity matrices I are in the diagonal form after doing the second step. Finally, a parity check matrix is managed to be a triangular form in order to achieve the efficient encoding. We compare the designed code's performance with the original MAC. Code parameters are: code rate 0.9 with the code length of 512, 2K and 4K bits. The simulation results of our design show that even if the submatrix size is a nonprime number, it delivers the same error rate performance as the prime submatrix size in the AWGN channel. We also have applied our design to another Array LDPC code such as IMAC and Large Girth Quasi-Cyclic LDPC Codes by making use of the Chinese Remainder Theorem (CRT). For IMAC with non prime submatix size, we propose the quasi cyclic matrix that uses to interleave non prime submatix size. We also present the method to combine 2 quasi cyclic LDPC codes via CRT when GCD $\neq$ 1.

Acknowledgements

First, I would like to express my sincere thanks to my thesis advisor, Assoc. Prof.Dr.Somsak Choomchuay, who has given me much in the way of suggestion, support, and advice from the very early stages of this research as well as giving me extraordinary experiences throughout the work. I would not have achieved this much and this thesis would not have been completed without all of the support that I received from him. I really enjoyed doing the LDPC codes design in my graduate study at the College of Data Storage Innovation, KMITL.

I would also like to thank all of the professors, lecturers and supporting staff in the department of electronics engineering for their continuous encouragement and guidance during the whole period of my study at KMITL.

I would like to thank to K. Chutima Prasartkaew for supporting my research publication.

I would like to express my appreciation to Western Digital (Thailand) Company for their financial support throughout my master's program fellowship at DSTAR KMITL.

I would also like to thank my supervisor, K. Vichai, and WD's HR manager, K. Pricha Leelanukrom, for their help during my research publication.

Finally, I most gratefully acknowledge my parents and friends for all their support throughout the period of this research.

Chalit Chusin

Contents

Pages

Abstract (Thai) I
Abstract (English) II
Acknowledgement III
Contents IV
List of Tables VII
List of Figures VIII
List of symbols and abbreviations X
Chapter 1 Introduction 1
1.1 Overview of Hard Disk Drive System 1
1.1.1 Disk 2
1.1.2 Head/Suspension Assembly 2
1.1.3 Actuator Assembly 3
1.1.4 Spindle and Motor Assembly 3
1.1.5 Electronic Card 3
1.2 Read and Write Process for Hard Disk Drive 4
1.3 Perpendicular Recording Channels 7
1.3.1 System diagram of the perpendicular recording system 8
1.3.2 Noise and Distortions in Perpendicular Recording Systems 13
1.4 Error Correcting Code in Hard Disk Drive 14
1.4.1 Background on Reed-Solomon Codes 15
1.4.2 Background on LDPC Codes 16
1.5 Hard Disk Sector Structures 16
1.6 Chinese Remainder Theorem 17
1.7 Problem Identification 19
1.8 Objective and Expectation 20
1.9 Research Hypothesis 20
1.10 Conceptual Framework 20
1.11 Scope & Constrain 21

Chapter 2 Linear Block Code and LDPC Codes 22
2.1 Linear block code 22
2.1.1 Binary Linear Block Codes 23
2.1.2 Hamming distance 24
2.1.3 Generator Matrix 24
2.1.4 Parity Check Matrix and Syndrome 25

Contents (Cont)

Pages

2.2 Introduction to LDPC codes 27
2.3 Types of LDPC codes 27
2.3.1 Regular LDPC Codes 27
2.3.2 Irregular LDPC Codes 28
2.4 Representations of LDPC codes 29
2.4.1 Cycle length of LDPC codes 30
2.5 Constructing LDPC Codes 30
2.5.1 Random based Construction 31
2.5.2 Structured Constructions 32
2.6 Encoding of LDPC codes 33
2.6.1 General case 33
2.6.2 Linear time encoding for LDPC codes 38
2.7 Decoding 43
2.7.1 Bit-Flip Algorithm (Hard Decision Decoding) 43
2.7.2 Message Passing Algorithm (Soft decision decoding) 44
2.7.3 Log- domain Sum-Product Algorithm (Soft decision decoding) 47
2.8 Array LDPC Codes 54
2.9 Modified Array Codes: MAC 56
2.10 Interleaved modified array LDPC codes: IMAC 59

Chapter 3 Non Prime Modified Array LDPC Code and Results 64
3.1 Issue of Modified Array LDPC Code 64
3.2 Non prime Modified Array LDPC codes 65
3.3 LDPC simulation system overview 69
3.4 Simulation performance result 71
3.4.1 BER performance of MAC and Non prime MAC with similar parameter 72
3.4.2 BER performance of Non prime MAC and MAC with different iteration 75
3.4.3 To find the suitable iteration on decoding process 76
3.4.4 BER performance of Non prime MAC with 500, 2K and 4K bits code length 78

Contents (Cont)

Pages

Chapter 4 Application of the Non Prime MAC LDPC Codes 80

4.1 Application of Non prime MAC LDPC codes with Interleave Modified Array Codes 80

4.2 To construct Non prime MAC LDPC codes with Chinese Remainder Theorem (CRT) 83

4.2.1 The Combination of QC-LDPC Codes via CRT 84

4.2.2 Matrix Design via CRT 86

4.3 Non prime MAC LDPC Codes in Magnetic Recording Channel 92

Chapter 5 Conclusions and Perspectives 95

References 97

Appendix 101

Author Biography 119

List of tables

Tables **Pages**

Table 2.1 Linear block code with k = 3 and n = 6 23

Table 3.1 Parameters for simulate the performance of MAC for 4K block length with high code rate 72

Table 3.2 Parameters for simulate the performance of MAC and Non prime MAC ... 75

Table 3.3 Parameters for simulate the performance of Non prime MAC 77

Table 3.4 Parameters for simulate the performance of Non prime MAC 528, 2016 and 4080 block length 78

Table 4.1 Parameter for IMAC, MAC and Interleaved Non prime MAC at 4K block length 82

Table 4. 2 Parameters for 4Kbits (R=0.9) 91

Table 4. 3 Parameters for 4Kbits sector 93

List of figures

Figures		Pages
Figure 1.1	Overview of hard disk drive system	2
Figure 1.2	Writing process in perpendicular recording systems	5
Figure 1.3	Ideal transition response in perpendicular recording systems	6
Figure 1.4	Dipulse response of T50 = 20 nm and B = 15 nm	8
Figure 1.5	Longitudinal recording diagram (left) and perpendicular recording diagram (right).	8
Figure 1.6	System diagram of the perpendicular recording system	9
Figure 1.7	Write precompensation applied to the write current	11
Figure 1.8	Timing jitter in communication systems	12
Figure 1.9	Sector structure in hard disk drive	17
Figure 2.1	Binary codewords structure	22
Figure 2.2	Parity check matrix H and Tanner Graph with cycle 4	29
Figure 2.3	Length 4 and 6 cyclesLength 4 and 6 cycles	30
Figure 2.4	Example of (a) Gallager's construction, (b) MacKay's construction	31
Figure 2.5	Random construction based on [Luby et al. 2001]	32
Figure 2.6	Parity check matrix H in approximate lower triangular form	38
Figure 2.7	Message passed between bit node and check node	47
Figure 2.8	Message send from variable node i to check node j	48
Figure 2.9	Message send from check node j to variable node i	49
Figure 2.10	Modify message for next iteration	49
Figure 2.11	Compute soft output	50
Figure 2.12	Tanner graph for example 2.1	51
Figure 2.13	Comparison of an array code with regular LDPC[8].	56
Figure 2.14	Performance of LDPC codes over an AWGN channel with Array LDPC code[9].	57
Figure 2.15	Performance of LDPC code with 4 Kbit over the AWGN channel	59
Figure 2.16	Structure of parity check matrix H for Array LDPC	61
Figure 2.17	Performance of MAC LDPC code and IMAC LDPC codes at 4K block length	62
Figure 3.1	Pattern diagram of Modified Array LDPC code with prime submatrix (L = 11).	65
Figure 3.2	Pattern diagram of Modified Array LDPC code with prime submatrix (L = 12).	65
Figure 3.3	Pattern diagram of array LDPC code with non-prime submatrix size (L = 12).	67

List of figures (Cont)

Figures Pages

Figure 3.4 Pattern diagram of array LDPC code with non-prime submatrix after cyclic shift right 67

Figure 3.5 Parity check matrix for the Non prime Modified Array LDPC codes (j = 3,k = 12 and L= 12). 68

Figure 3.6 Parity check matrix for the Modified Array LDPC codes (j = 3, k = 12 and L= 12). 69

Figure 3.7 LDPC simulation system overview 70

Figure 3.8 Simulation performance of MAC for 4 Kbits block length with prime and non prime submatrix 73

Figure 3.9 Simulation performance of MAC and non prime MAC for 4 Kbits block length. 74

Figure 3.10 Simulation performance of MAC and non prime MAC LDPC codes with 5, 10, 20, and 30 iterations 76

Figure 3.11 Simulation performance of non prime MAC LDPC codes on different iterations 77

Figure 3.12 Simulation performance of non prime MAC LDPC codes on 528, 2016 and 4080 block length 79

Figure 4.1 Simulation result of non prime IMAC and existing LDPC's performance 83

Figure 4.2 Codes performance of ours and MAC's at R = 0.90 92

Figure 4.3 Channel model for Magnetic recording systems[42]. 93

Figure 4.4 Codes performance of Non prime MAC and MAC at 4096 bits 94

List of symbols and abbreviations

z(t)	read back signal
d_i	sequence of the transitions (can be 0, +1 or -1)
B	channel bit spacing
$s(t)$	transition response
$h(t)$	dipulse response
V_{max}	zero to peak amplitude
T_{50}	width when transition response changes from *-A/2* to *A/2*
a_k	input data symbols
b_k	precoded symbols
τ_k	timing offset
r(t)	received waveform
a_i	transition jitter
(n, k, d)	binary block code, which n code bits, k information bit where d is the Hamming distance.
C	Codeword, where $C = [c_1, c_2, \ldots, c_n]$
m	Message bit, where $m = [m_1, m_2, \ldots, m_k]$
R_c	code rate for codeword
G	generator matrix
I_k	a k x k identity matrix
H	Parity check matrix
E	error vector, where $e = [e_1, e_2, \ldots, e_n]$
w_r	row weight
w_c	column weight
p	parity bit
d_{min}	minimum distance
$E(H_k)$	Exponent matrices
$M(H_k)$	Mother matrices
σ^2	noise variance
q_{ij}	message sent from the i^{th} variable node to j^{th} check node
r_{ji}	message sent from j^{th} check node to the i^{th} variable node
α	permutation matrix with dimention L X L
ω	quasi cyclic matrix matrix with dimention L X L
NRZ	Non-return-to-zero
ECC	Error correction code
MTR	Maximum transition run

List of symbols and abbreviations (Cont)

NLTS	Nonlinear transition shift
MMSE	Minimize the mean squared error
PRML	Partial response Maximum Likelihood
ISI	Inter symbol interference
AWGN	Additive white Gaussian noise
PAM	Pulse amplitude modulation
ML	Maximum-likelihood
ITI	Inter track interference
RS codes	Reed-Solomon code
SER	Sector error rate
LDPC codes	Low density parity check codes
BP	Belief propagation
MDS	Maximum distance separable
SNR	Signal to noise ratio
CRT	Chinese Remainder Theorem
LLR	Log-likelihood ratio
QC-LDPC codes	Quasi cyclic Low density parity check codes
Gcd	Greatest common divisor
SPA	Sum product algorithm
MAC	Modified Array Low density parity check code
IMAC	Interleaved Modified Array Low density parity check code

Chapter 1
Introduction

Computer memories are divided into two types. The first type is a group of working memories such as RAM, ROM and other semiconductor devices. These devices are directed addressable by the CPU. The second type is a group of mass storage memories such as hard disk drive, CD and DVD. The mass storage devices are not directed addressable by the CPU. Mass storage devices can be classified into three technologies: magnetic storage, optical storage, and semiconductor memory. From these technologies, the magnetic storage technology is more interested than others. Since, it can provide the high storage capacity at low costs with high flexibility.

The hard disk drive is a type of magnetic storage technology. It is firstly invented by IBM in 1956. The substantial progress has been achieved in recording areal density and data rate. The present trend in hard disk design is a moving toward a small feature size with high capacities. The design can be implied by reducing the size of track width or bit size. The small track width can lead to high capacities but generates more error than the large track width. So, the error correcting codes were used to eliminate the errors on hard disk drive.

This chapter will begin with the overview of the Hard Disk Drive System. Read and Write channels are introduced in Section 1.2. Perpendicular recording channels are presented in Section 1.3. The error correcting code used to protect and correct the error in hard disk drive is discussed in Section 1.4. Hard Disk Drive structure is discussed in Section 1.5. The Chinese Remainder Theorem or CRT is explained in Section 1.6. The problem identification of this thesis, Objective and Expectation, Research Hypothesis, Conceptual Framework, Scope & Constraint are given in Section 1.7, 1.8, 1.9, 1.10 and 1.11, respectively.

1.1 Overview of Hard Disk Drive System

Hard Disk Drive (HDD) is a mechatronic system that includes several components. These components can be classified into magnetic components, mechanical components and electronic components. The HDD was invented firstly by IBM in 1956. IBM 350 RAMAC (Random Access Method of Accounting and Control) was introduced in 1957 with a capacity of 5 MB (MB: Megabits). It had an areal density of 2 Kbits/inch^2 (kilobits per Square inch). Today, HDD's capacity is larger than 1 TB (TB: terabyte) with an areal density more than 420 Gigabits/inch^2 (gigabit per square inch). In the future, HDD's areal density will be continuously increased.

Now, the new generation HDD has capacity 200000 times when compare to RAMAC. Over the past 20 years, we have seen the physical size of HDD changed from a 5.25 inch form factor to a 2.5 inch form factor on today. The capacity of HDD also are doubling every 12 to 18 months. The current physical components used in HDD can be presented in Figure 1.1. The media and the head are magnetic components of the hard disk drive system. They are assigned to be used in storage binary information. In HDD, the information is recorded and retrieved by the read/write head. The information bits will be stored in concentric data tracks on a rotating disk. The spindle motor is used to move the head on the track. The main functions and special features of HDD components are briefly explained in this section.

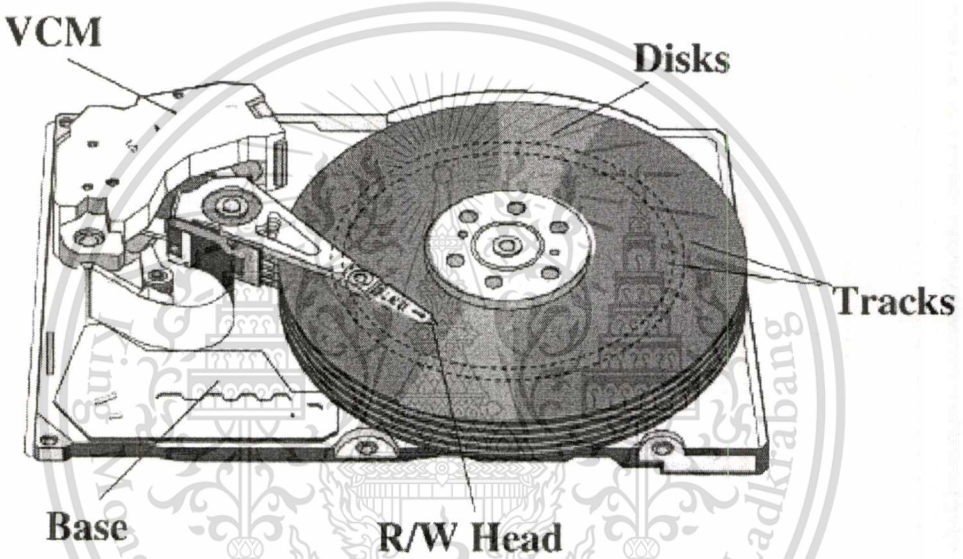


Figure 1.1 Overview of hard disk drive system

1.1.1 Disk

A typical HDD includes one or more flat rotating disk with two magnetic surfaces that are called platters. The data are recorded on continuously spinning disks which are fabricated by sequentially depositing of an alloy film, a carbon overcoat layer and a lubricating layer. Each layer has a film thickness ranging from several to several tens of nanometers. The alloy film contains a magnetic recording layer for retaining the information. To produce the uniform of good read back signal from the R/W, the media's surface of the disk must be smooth.

1.1.2 Head/Suspension Assembly

In HDD, the R/W head is processed by thin film and metalized structure that exhibits magneto resistive effect. Both read and write operations in disk drives were performed by a single head. These heads are referred to a slider those are

positioned only microinches above the recording medium on an air bearing surface. To read or write process, the current will flow through the MR head to generate magnetic signals for record on the disk surface. A gimbal is used to connect the slider to a suspension for pitch and roll rotations. The suspension is attached to the arm of the actuator by a ball swaging.

1.1.3 Actuator Assembly

In a hard disk drive system, a set of actuator assembly is employed to position the read/write head on the certain track on the disk surface. This actuator assembly consists of a VCM, data flex cable or printed circuit cable, actuator arms and crash stops at both ends of travel. The data are read/written from/to the magnetic disk surfaces of the R/W heads. These heads were mounted on the slider which is attached to the actuator arm. The function of the actuator in HDD is called VCM actuator. The coil of VCM actuator extends between yoke and magnets. The VCM driver or amplifiers are the part of the actuator assembly which is mainly used to drive or give the power to the VCM actuator.

1.1.4 Spindle and Motor Assembly

Spindle motor is a DC motor that is used to spin the stack of disks in HDD. It must stable, reliable and consistent turning power for thousands of hours. All hard disks use a servo to control a DC spindle motor. The spindle motor is configured to connect the hard disk platter without the use of belts or gears. The fluid dynamic bearing or aerodynamic bearing spindles are normally used in high performance of hard disk drives. The speed of spindle motor is around 10,000 RPM or more. Many models of hard disk drives, for the desktop and mobile environment, are still using the spindle speed of 5,400 RPM or 7,200 RPM. The hard disk drives with high performance are using higher spinning rates such as 10,000 RPM or even 15,000 RPM. The most critical component in hard disk's spindle motor is the set of spindle motor bearings at each end of the spindle shaft. With the demand for higher areal density and faster spindle speed, the fluid dynamic bearing (FDB) spindle motor is adopted in HDDs.

1.1.5 Electronic Card

The electronic card is an interface to the host personal computer. The widely used integrated electronic circuits in hard disk drives are the Integrated Drive Electronics (IDE), the Advanced Technology Attachment (ATA), and the Small Computer System Interface (SCSI). These integrated circuits have a powerful driver of the spindle motor, VCM, R/W electronics, servo demodulator, microcontroller or

digital signal processor (DSP) for servo control, ROM and RAM for microcode and data transfer.

1.2 Read and Write Process for Hard Disk Drive

In hard disk drives, data are recorded on the disk with binary bits. The bits are written on the tracks, which are circular bands around the disk center. In perpendicular recording systems, the polarity of the bits is represented by the upward or downward orientation of the magnetic field in the medium. Writing bits are the process to magnetize the magnetic field of the medium into different directions. Figure 1.2 gives an illustration of the writing process. The recording flux is controlled by the current in the write head. The amplitude of the current should be large enough to magnetize the medium to saturation and the direction of the current determines the magnetization direction. By spinning the disk, the latter bit will be written overlapping the previous one. The rotational speed is controlled carefully to achieve the desired channel bit spacing.

The datas are written on the disk sector by sector. A sector has user data bits plus other overhead bits as describe in 1.4. Each track includes multiple sectors. Now, a 4 Kbits and 4 Kbytes sector size is proposed in hard disk drive. When the magnetization pattern on the disk is read, the disk spins and the read head moves over the track to get the magnetic field above the medium. The read back signal can be regarded as a superposition of the isolated transition responses at the transition positions of the magnetic fields. We can write the of read back signal as follows:

$$z(t) = \sum_i d_i s(t - iB) \tag{1.1}$$

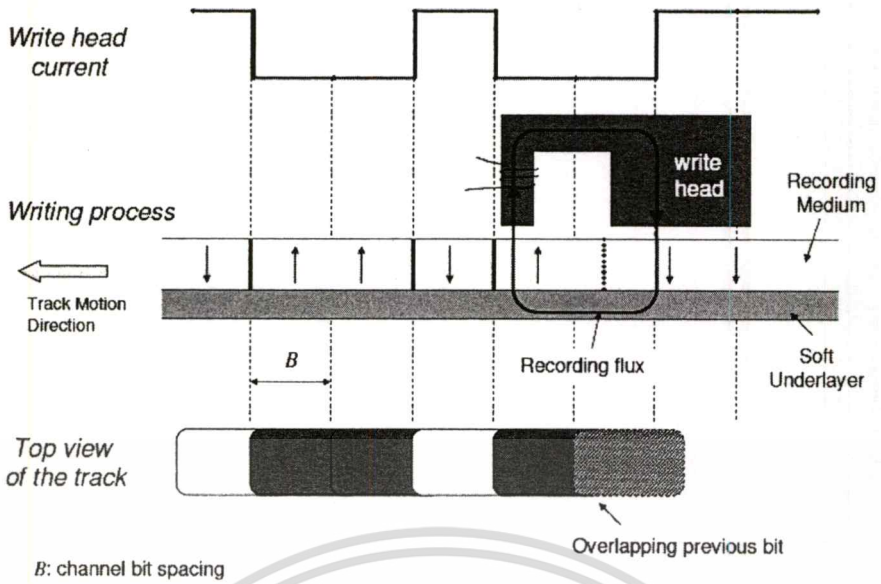


Figure 1.2 Writing process in perpendicular recording systems.

where $\{ d_i \}$ is the sequence of the transitions, B is the channel bit spacing and $s(t)$ is the transition response. The transition "d_i" can be 0, +1 or -1, where 0 represents no transition at position i, and ±1 represent transitions of different directions. The isolated transition response can be analyzed and measure from the disk. There are two approximations of the ideal transition response that are used in simulations. One of them uses the tanh function, such as in [11, 12]. The transition response can be expressed as:

$$s(t) = V_{max}\, erf\left(\frac{0.954t}{T_{50}}\right) \tag{1.2}$$

where V_{max} is the zero to peak amplitude and T_{50} is the width when $s(t)$ changes from $-A/2$ to $A/2$, as shown in Figure 1.3. Since the tanh function is an odd function, T_{50} also is the value that satisfies

$$s\left(T_{50}/2\right) = A/2 \tag{1.3}$$

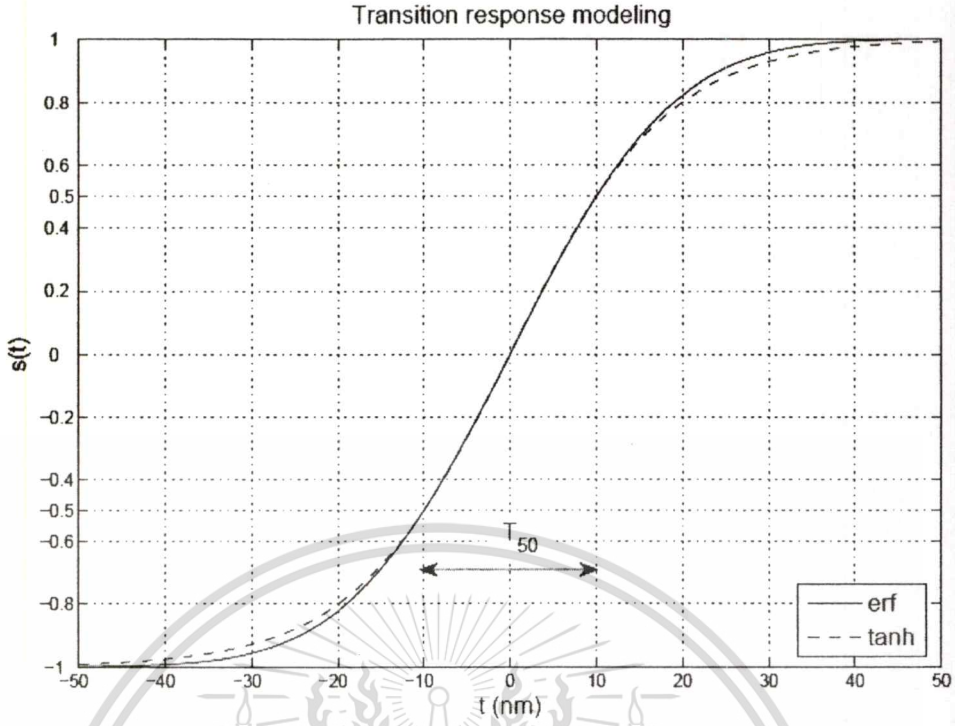


Figure 1.3 Ideal transition response in perpendicular recording systems.

where *erf* ( ) has the form

$$erf(t) = \frac{2}{\sqrt{\pi}} \int_0^t e^{-u^2} du \tag{1.4}$$

The parameter T_{50} in this approximation has the same meaning as in the tanh approximation. It satisfies to the equation (1.3) as well. A comparison of the two approximations is shown in Figure 1.4. They are using the parameters T_{50} = 20 nm and V_{max} = 1.

The noiseless read back signal can be expressed as the superposition of the dipulse response, where the input data are considered to be the NRZ pattern. Let the binary data sequence is $\{x_i\}$, where $x_i \in \{+1,-1\}$. The transition d_i can be calculated as

$$d_i = \frac{x_i - x_{i-1}}{2}. \tag{1.5}$$

Denoting the dipulse response by $h(t)$, the read-back signal can be represented by

$$z(t) = \sum_i x_i h(t - iB) \tag{1.6}$$

where $h(t) = \big(s(t) - s(t - B)\big) / 2$. An example of the dipulse response is shown in Figure 1.4, which uses the error function approximation of the transition response.

1.3 Perpendicular Recording Channels

A hard disk drive system comprises with many individual components that are designed from various fields of physics and engineering. For the signal processing, we will focus on the recording channel. Past over 40 years, the bits on the hard disk drives were recorded as horizontal magnetic fields on the surface of the medium, which is called longitudinal recording. When the recording density increases, the superparamagnetic effect will occur. The microscopic magnetic grains on the disk becomes so tiny and their magnetic orientations are not stable at room temperature. Though increasing the coercivity, the "field" required to write a bit, of the media can avoid the superparamagnetic effect, the write field cannot be arbitrarily large because of the materials of the head. With perpendicular recording technology, the orientation of the magnetic field of each bit is perpendicular to the surface of the medium. Because of, the geometry was changed in the writing process. The maximum fringing writes field was generated by the head with the same material. The write field is twice as big as the longitudinal recording. Perpendicular is unlike the longitudinal recording because the demagnetizing field in perpendicular recording decreases with higher density. Therefore, the higher recording density can be achieved on perpendicular recording. Figure 1.5 shows the Longitudinal recording diagram and perpendicular recording diagram.

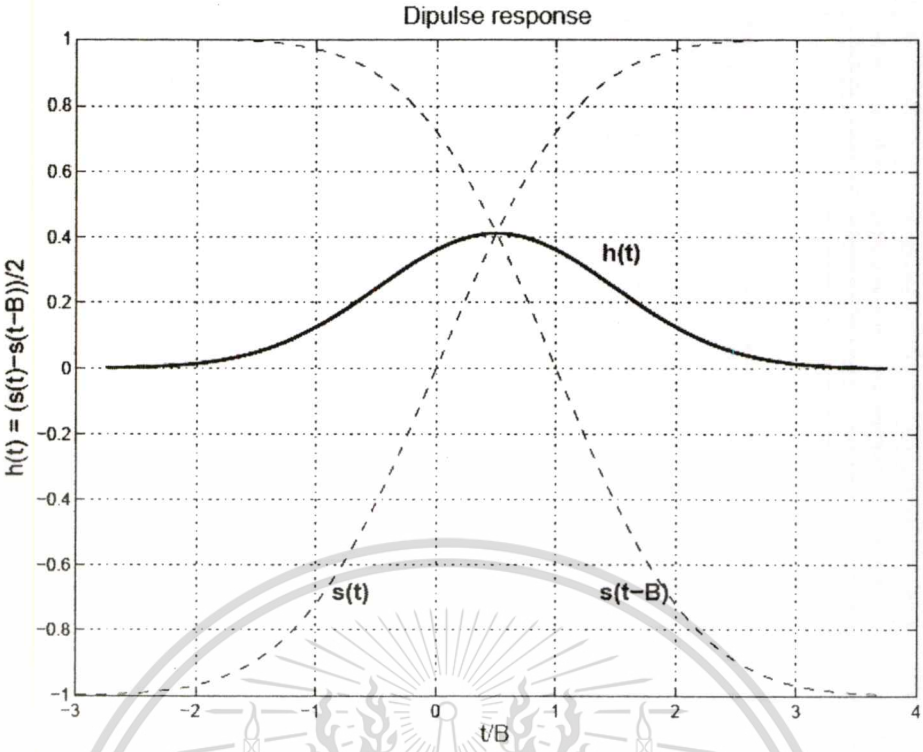


Figure 1.4 Dipulse response of T_{50} = 20 nm and B = 15 nm.

1.3.1 System diagram of the perpendicular recording system

From the signal processing and coding techniques, the data storage system can be modeled as a communication system. A system diagram of the magnetic recording system is shown in Figure 1.6. The writing process can be regarded as the transmitter. The binary user data are usually encoded with an error correction code (ECC) and modulation code. A precoder is sometimes used. The transition positions in the write current are adjusted by the write precompensation process before going through the coil of the write head. The write precompensation process can reduce the effect of the nonlinear transition shift.

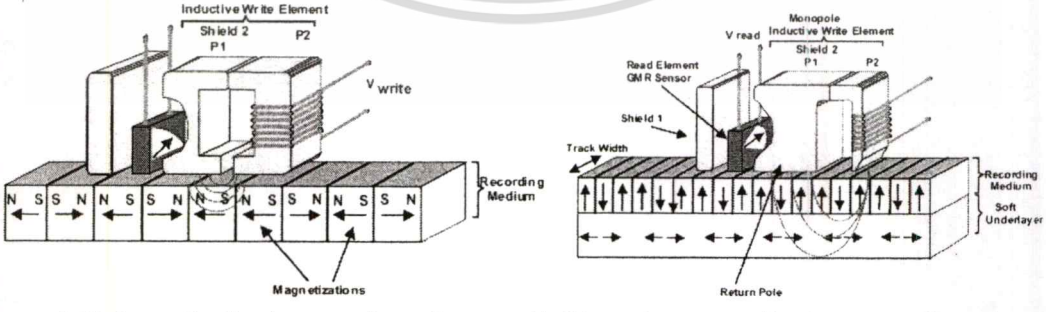


Figure 1.5 Longitudinal recording diagram (left) and perpendicular recording diagram (right).

The read back and data recovery process can be regarded as a receiver in the communication system. The read back signal from the read head is an analog waveform. It is processed by the front end circuit, which usually contained a preamplifier and an analog low pass filter. The timing recovery loop provides the clock for sampling the digital signal. The sample signal is usually equalized before the detector. The modulation code and the error correcting code are decoded afterwards to finally recover the user data. The signal processing and coding techniques used in the system can introduce as Figure 1.6.

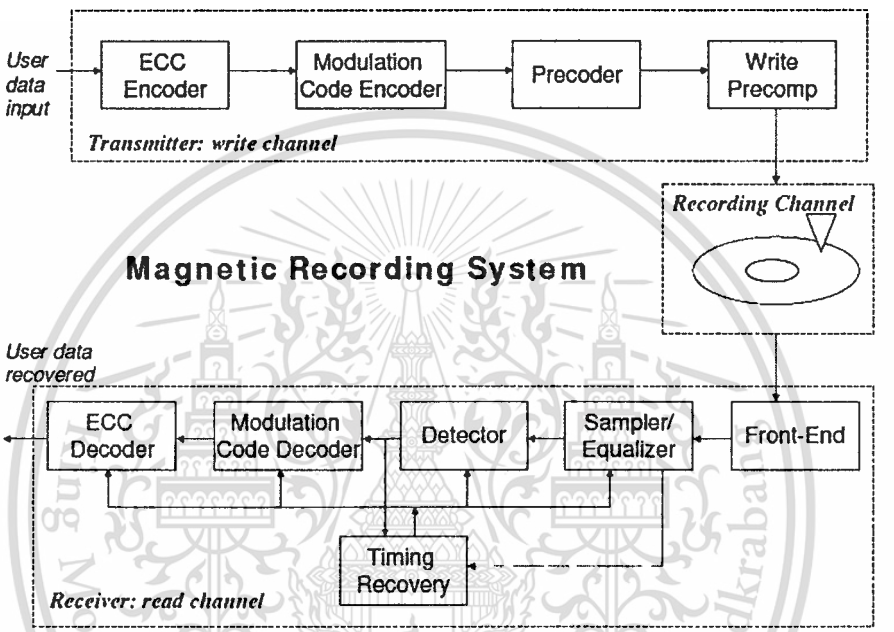


Figure 1.6 System diagram of the perpendicular recording system.

Error Correction Codes - It was used to eliminate the effects of noise that was introduced by the channel. For hard disk drives, the error rate of the recovered user data needs to be very low, for example 10^{-15}, which is much lower than the other communication system such as in wireless communication. An error correction code is used after the detection and modulation code. It uses to correct the errors which occurred during the recording process. More detail for error correction code will be explained in Section 1.4.

Modulation Code - It also calls the constrained codes. These codes are used in digital recording systems to avoid some of the data recovery failures such as timing recovery failure and detection failure. For example, a run length limited (*d, k*) codes were used in the peak detection systems. The code limits the run length of zeros in a codeword. This constraint is very suitable for the transition of the magnetic fields on the disk. If there is no transition on a long stretch of track, there is a long sequence of zeros in the transition sequence. The timing recovery loop

could lose track of the clock. Therefore, we need to make sure that the run length of zeros cannot be longer than a certain number *k*. However, the run length of zeros cannot be smaller than d bits in order to prevent interference between two neighboring transitions. If two transitions are too close, they will be affected by the ISI and the detection might fail.

In PR4 channels, a (*0, G/I*) modulation code is usually used. The parameter *G* limits the run length of zeros in the transition sequence for the sake of the timing recovery. That is a same parameter as the k constraint in the (*d, k*) codes. The constraint *I* is used to ensure the successful detection by the Viterbi detector. Since the trace back length of the Viterbi detector is limited in a real system, the maximum number of zeros in the substrings of the sequence needs to be limited by *I*. The variations of other constraints are used in the magnetic recording channels. For example, a DC free constraint is extremely useful for a perpendicular recording channel where the DC is nonzero. Some constrained codes, such as the maximum transition run (MTR) constrained codes, increase the distance between events in the PRML detector to improving the system performance. The modulation codes add overhead to the data sequence to ensure the constraint. Thus, the rate of the code is very important for the system designers. The constrained codes used in real systems usually have very high code rates.

Precoder - A portion of the equalization task can be shifted to the transmitter by precoding the data symbols at the transmitter. The data symbols are written to the channel. With the knowledge of the channel characteristics, the precoder can be chosen to partly undo the channel distortion. So, it can reduce the equalization burden at the receiver. Precoders can also be chosen to prevent catastrophic error propagation. For the PR4 system, the precoder is denoted by the polynomial fraction $1/1 \oplus D^2$. The precoded symbols $\{b_k\}$ are arrived at from the data symbols $\{a_k\}$ according to $b_k = b_{k-2} \oplus a_k$.

Write precompensation - A dominant and nonlinearity found in magnetic recording is called a nonlinear transition shift (NLTS). NLTS causes the written bit transitions to be shifted, if there exist previously written transitions within a reasonable distance. These demagnetizing fields interfere with the write bubble (used to magnetize the media) and cause the bit transitions to be written in unintended positions. In longitudinal recording, NLTS moves the written bit transitions against the recording direction. The shift bit transitions occur in the opposite direction in perpendicular recording.

The write precompensation is typically used as a means to deal with NLTS. The idea is to delay the transitions in the write current. That is used to offset the " forward " shifts of the written transitions caused by NLTS. Figure 1.7 shows the write precompensation subsystem of the recording system. When a bit transition is to be written, the write current was offset by a certain amount determined by the NLTS measurements. After the NLTS occurs, the bit transition will be evenly spaced. Today, write precompensation is found in almost every commercial hard drive.

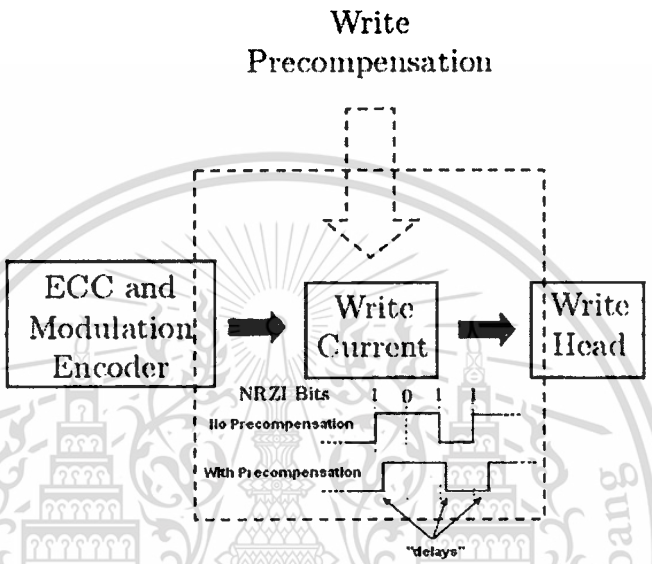


Figure 1.7 Write precompensation applied to the write current.

Partial Response Equalization - The magnetic recording channel is a channel with intersymbol interference. In early longitudinal magnetic recording systems, peak detection was used for low density recording. The pulses produced by two neighboring transitions are enough apart so that the interference between each other can be ignored. However, with the increasing of recording density, the interference between pulses becomes more severe and the peak detection gives a very poor performance. A partial response maximum likelihood detector was proposed to deal with the channel ISI. In order to reduce the number of states in the detector, an FIR equalizer is used to equalize the channel response to a desired. One class of equalization target, labeled " Class-4 " or " extended Class-4 ", is widely used in longitudinal magnetic recording channels with different channel bit densities. A general expression of these target polynomials takes the form

$$g(D) = (1-D)(1+D)^N \tag{1.7}$$

where $N \geq 1$. When $N = 1$, a simplified channel uses $g(D)$ as the channel transfer function. It is usually called a PR4 (Partial Response Class-4) channel. When $N \geq 2$, the channels are called extended Class-4 channels. It denoted by E^{N-1} *PR4*. There are many studies on the design of the equalizer such as in [23, 24, 25]. The design objective is usually to minimize the mean squared error (MMSE) of the targeted signal and the equalized channel output. Since the channel parameters might be changed slightly on one disk drive because of the temperature variation. The equalizer can be designed to be adaptive to the channel. So, a stable targeted signal is provided for the detector afterwards.

Detection - Before the use of partial responses, the detection used peak detection. A partial response target, a Viterbi, or Viterbi-like detector is used to detect the data sequence. The conventional Viterbi detector is a maximum likelihood sequence detector for an ISI channel with AWGN. For any received sequence $\overline{r}$, the maximum likelihood sequence detector gives the most probable input data sequence $\overline{x}$. So, the probability $\Pr(\overline{r}|\overline{x})$ can be maximized. The Viterbi algorithm runs on a trellis. The number of trellis states is determined by the partial response target length. The detected sequence is selected to be the one that minimizes the accumulated branch metric. For the conventional Viterbi detector, the branch metric is the Euclidean distance between the received samples and the labeled target outputs.

The noise in the perpendicular recording channel is dominated by the transition noise which cannot be modeled as AWGN. After the equalizer, the noise is usually colored. Therefore, the conventional Viterbi detector is not the optimal one. So, the modifications to the conventional Viterbi detector were proposed. A noise predictive maximum likelihood detector was proposed in [26]. It uses a noise prediction filter in the Viterbi algorithm. The pattern dependent noise is a predictive detector, which uses a different noise prediction filter for each branch to capture the data dependent nature of the jitter noise. The system performance can be improved by using these modified detectors.

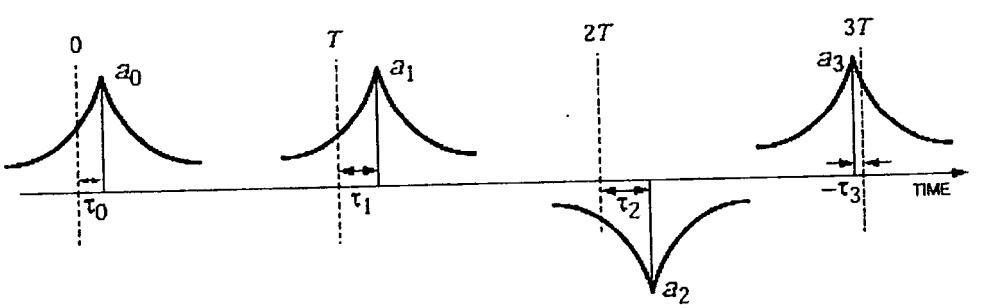


Figure 1.8 Timing jitter in communication systems.

Timing Recovery – It is an essential part of digital communication and storage systems. The underlying physical channel is an analog in nature whereas the user data is digital. At the transmitter, we obtain a continuous time signal from the discrete time of user data by modulation. From the sampling theorem, it is sufficient to collect samples of this waveform. In other words, we can undo the modulation performed and get back to the discrete time domain. Figure 1.8 shows an example of a modulation scheme used in communication systems. The modulator uses the data symbols $\{a_k\}$ to modulate a pulse shape $h(t)$. The final waveform on the channel is obtained by shifting consecutive scaled pulses by the symbol duration T. We assume that $h(t)$ is a band limited that baud rate samples are sufficient statistics at the receiver. This modulation scheme is called pulse amplitude modulation (PAM). Ideally, the all pulses are shifted with respect to each other by multiples of T. It is sufficient to sample at multiples of T once we have the correct initial sampling instant. However, the different pulses in the waveform suffer from timing jitter,with the k^{th} pulse has a timing offset of τ_k from the ideal location kT. The ideal sampling instants are $kT+\tau_k$ instead of kT. The timing recovery problem is estimating the timing offsets $\{\tau_k\}$.

The maximum-likelihood (ML) timing estimator picks the timing offset estimates τ_k to minimize the cost function $J(\hat{\tau};a)$, where

$$J(\hat{\tau};a)=\int_{-\infty}^{\infty}\left(r(t)-\sum_l a_l h\left(t-lT-\hat{\tau}_l\right)\right)^2 dt, \qquad (1.8)$$

where $r(t)$ is a received waveform. It can represent the trained ML estimator, where the data symbols $\{a_k\}$ are assumed to be known at the receiver. The ML timing estimator is prohibitively complex except in a few simple cases. In general, suboptimal timing recovery methods are used to allow the practical implementation at the cost of performance.

1.3.2 Noise and Distortions in Perpendicular Recording Systems

The read back signal will be affected by various noises and distortions, which cause errors during the recovery of the data. The fundamental noise sources are media noise, the noise from the head and the noise from the electronic circuits such as the preamplifier [13]. The noise from the head and the electronic circuits are usually modeled as the additive white Gaussian noise (AWGN).

The noise from the media is the dominant noise source in perpendicular recording systems [14]. Among the components of the media noise, the transition noise is dominant [15]. The transition noise originates from the randomness of the

magnetization of grains in the transition region. It can be modeled as a random jitter of the transition position, which can be included in the read back signal as follows:

$$z(t) = \sum_i d_i s\left(t + a_i - iB\right) \tag{1.9}$$

where a_i is the transition jitter for transition d_i. If d_i is zero, there is no transition and a_i is also zero. When d_i is non zero, the transition jitter a_i is a random variable. The transition jitter a_i is assumed to have a Gaussian distribution or a truncated Gaussian distribution. The random variables representing the transition jitters are assumed to be independent to each other.

Nonlinear distortions can degrade the system performance, such as the nonlinear transition shift and the read head nonlinearities [16]. The nonlinear transition shift (NLTS) is due to the magnetic field from the neighboring recorded transitions on the same track and from adjacent tracks. Because of the neighboring magnetization, the read head could operate in a nonlinear regime [17,18].

Inter track interference (ITI) is another problem that arises with increasing density. It can be modeled as a linear signal added to the main track signal. The signal processing techniques for ITI in perpendicular channel were studied [19]. Thermal asperities also happen in longitudinal recording. They arise from the defects on the medium surfaces. The friction between the asperity and the magnetoresistive read head causes the head to heat rapidly, which results in a change of the head resistance and therefore the read back signal. Long bursts of errors usually happen because of thermal asperities. In practice, thermal asperities can be detected and canceled by signal processing techniques as proposed in [20,21].

1.4 Error Correcting Code in Hard Disk Drive

For Hard Disk Drive, each sector of data contains 512 bytes or 4,096 bits of user data. In addition to these bits, an additional number of bits are added to each sector for the implementation of error correcting code or ECC (sometimes also called error correction code or error correcting circuits). These bits do not contain data. They contain the information that can be used to correct any problems encountered trying to access the real data bits. . We use ECC to correct and detect the error on the Hard Disk Drive. The concept of this method is encoding the information bit by adding some redundant bits into the information to generate the code word. So, the cordword will have the capability to detect and correct the error. This codeword can be recovered to be the original form by decoding with related method. The

decoder adapts the detecting and correcting capability of the coding scheme to reduce some of the error.

Reed-Solomon (RS) code, named for researchers Irving Reed and Gustave Solomon who first discovered the general technique that the code employs, is the error correcting codes used for decades the error. Reed Solomon codes are widely used for error detection and correction in various computing and communications media such as magnetic storage, optical storage, high-speed modems and data transmission channels. They have been chosen because they are easier to decode than most other similar codes. It can detect (and correct) the large numbers of missing bits but require the least number of extra bits in the encoding process.

However, many other coding schemes were proposed to improve the system sector error rate (SER) and save the overhead further. Multilevel error correcting codes were also proposed to handle both random and burst errors in the recording systems [38, 39]. The low density parity check (LDPC) codes are the most likely candidate today to substitute for the RS code in magnetic recording systems because of its channel capacity approaching property [40]. The decoding of the LDPC codes uses a message passing algorithm, which is practical for implementation [41].

1.4.1 Background on Reed-Solomon Codes

Many advances in coding theory during its first few decades of development involved algebraic codes. The structure of these codes can be exploited to yield practical encoding and decoding algorithms. The major design goal was to maximize the minimum Hamming distance. One reason for this is that bounded distance decoding is guaranteed to correct any number of errors smaller than half the minimum distance.

RS codes are the most popular algebraic codes. They were introduced in 1960 by Irving Reed and Gustave Solomon [3]. A (n, k) RS code of length n and dimension k has a maximum distance separable (MDS) linear code with minimum distance $d_{min} = n-k+1$. With respect to minimum distance, RS codes are optimal because they achieve the Singleton Bound [4]. RS codes also have efficient with a hard decision decoding algorithms such as the Berlekamp-Massey (BM) algorithm. They can guarantee to correct the error up to $\left(d_{\min} - 1\right)/2$ errors. They are very well in channels with a mixture of both burst and random errors and can achieve very low error rates. However, a major drawback of RS codes is the lack of decoding algorithms that make good performance.

1.4.2 Background on LDPC Codes

LDPC codes are linear block codes with a very small fraction of non-zero entries in the parity check matrices. They were introduced by Gallager in his doctoral dissertation [5] in the same year that RS codes were proposed. However, their value went unrecognized for decades until being rediscovered by MacKay [6]. Tanner's graph [7] is the new way of graphically depicting LDPC codes being a significant exception. Using Tanner's method, LDPC codes can be represented by bipartite graphs by using a set of variable nodes, corresponding to the codeword symbols, and a set of check nodes, corresponding to the parity-check constraints of the codes. The sparse bipartite graph structure of LDPC codes turns out to work very well with belief propagation (BP), a low-complexity message-passing decoding. It can achieve the capacity of several channels. More about LDPC codes and their decoding algorithms can be found in Chapter 2.

1.5 Hard Disk Sector Structures

The basic unit of data storage on a hard disk is a sector. Each sector is specified with three basic parts namely, the sector header, the data area and the Error Correcting Code (ECC). The sector header contains information used by the drive and controller. This information includes sync bytes, address identification, flaw flag and header parity bytes. The address identification is used to ensure that the mechanics of the drive have positioned the read/write head over the correct location. The data area contains the recorded user data. The ECC field contains codes based on the data field, which is used to check and possibly correct errors that may have been introduced into the data.

Since the introduction of PC-DOS with 512 bytes sector by IBM in 1981, we are still using a 512 bytes sector on the disk to store our data until 2010. However, the 512 byte sector size has turned into a limiting factor after the hard disk drive industry is increasing the areal densities in hard disk drive on a yearly basis. That involves a continuous loss of SNR. So, ECC algorithms used in hard drives must be improved to maintain data integrity or sector failure rate. But, this can't be accomplished by adding more ECC at the current 512 bytes format. This bottleneck has been recognized by all major hard disk drive producers.

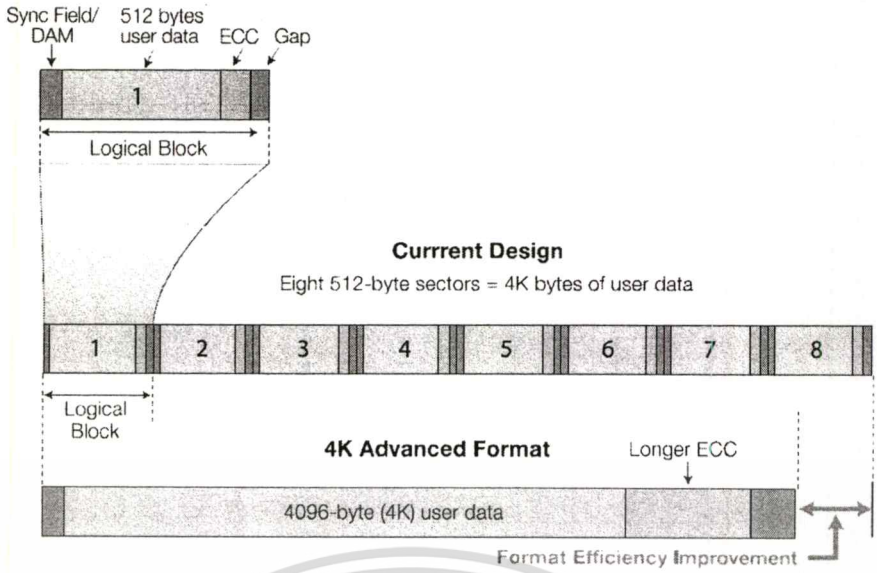


Figure 1.9 Sector structure in hard disk drive.

The new hard disk drives read/write data to 4 Kbyte (4,096 bytes) physical sectors on the media. The 4 Kbyte is the equivalent of putting eight 512 byte sectors into the new 4Kbyte sector. This approach provides two benefits illustrated in Figure 1.9. First, optimize the sector header with each smaller sector. The drive uses less space to store the same amount of sync bytes, address identification, flaw flag and header parity bytes resulting in a format efficiency improvement (see blue arrow in Figure 1.9). The second benefit is a larger and more powerful error correction code (ECC) can be utilized. So, it can be providing better integrity of user data.

1.6 Chinese Remainder Theorem

The Chinese Remainder Theorem is a result about congruences in number theory and its generalizations in abstract algebra. It was first published in the 3rd to 5th centuries by Chinese mathematician Sun Tzu. In its basic form, the Chinese remainder theorem will determine a number n that when divided by some given divisors leaves given remainders.

Theorem: Suppose that $m_1, m_2, \ldots, m_r$ are pairwise relatively prime positive integers, and let $a_1, a_2, \ldots, a_r$ be integers. Then the system of congruences,

$$x \equiv a_k \bmod(M_k) \text{ for } k = 1, 2, \ldots, r$$

has a unique solution modulo $M = M_1 \times M_2 \times \cdots \times M_r$, which is given by:

$$x \equiv a_1 M_1 b_1 + a_2 M_2 b_2 + \ldots + a_r M_r b_r \bmod(M),$$

where $M_k = M/m_k$ and $b_k \equiv (M_k)^{-1} \bmod (m_k)$ for $k = 1, 2, \ldots, r$.

Proof: Notice that gcd $(M_k, m_k) = 1$ for $k = 1, 2, \ldots, r$. Therefore, every b_k exists (and can be determined easily from the extended Euclidean Algorithm). Now, notice that since $M_k b_k \equiv 1 \bmod (m_k)$, we have $a_k M_k b_k \equiv a_k \bmod (m_k)$ for $k = 1, 2, \ldots, r$. On the other hand, $a_k M_k b_k \equiv 0 \bmod (m_j)$ if j is not k (since m_j divides M_k in this case). Thus, we see that $x \equiv a_k \bmod (m_k)$ for $k = 1, 2, \ldots, r$. If there were two solutions, say x_0, and x_1, then we would have $x_0 - x_1 \equiv 0 \bmod (m_k)$ for $k = 1, 2, \ldots, r$, so $x_0 - x_1 \equiv 0 \bmod (M)$, i.e., they are the same modulo M.

Example 1.1: Find the smallest multiple of 10 which has remainder 2 when divided by 3, and remainder 3 when divided by 7. We are looking for a number which satisfies the congruences,

$$x \equiv 2 \bmod(3),\ x \equiv 3 \bmod(7),\ x \equiv 0 \bmod(2) \text{ and } x \equiv 0 \bmod(5).$$

Since 2, 3, 5, and 7 are all relatively prime in pairs, the Chinese Remainder Theorem tells us that there is a unique solution modulo $210 = 2 \times 3 \times 5 \times 7$. We calculate the M_k and b_k as follows:

$$M_1 = 210 / 2 = 105;\ b_1 \equiv (105)^{-1} \bmod(2) = 1$$

$$M_2 = 210 / 3 = 70;\ b_2 \equiv (70)^{-1} \bmod(3) = 1$$

$$M_3 = 210 / 5 = 42;\ b_3 \equiv (42)^{-1} \bmod(5) = 3$$

$$M_4 = 210 / 7 = 30;\ b_4 \equiv (30)^{-1} \bmod(7) = 4.$$

We have:

$$\begin{aligned} x &\equiv 0(M_1 b_1) + 2(M_2 b_2) + 0(M_3 b_3) + 3(M_4 b_4) \\ &= 0 + 2(70)(1) + 0 + 3(30)(4) = 140 + 360 \\ &= 500 \bmod 210 \equiv 80. \end{aligned}$$

The Chinese remainder theorem can be used in the LDPC code, which constructing QC-LDPC codes of large length from QC-LDPC codes of small length. The method is combining the exponent matrices for given QC-LDPC codes of smaller length to obtain the exponent matrix for a QC-LDPC code of large length. For k = 1, 2, . . . , s, let C_k be a QC-LDPC code whose parity-check matrix H_k is an $mL_k \times nL_k$ binary matrix. The corresponding exponent and mother matrices are given by $E(H_k) = (a_{ij}^k)$ and $M(H_k)$. Assume that all $M(H_k)$ are the same and $\gcd(L_l, L_k) \geq 1$ for all $l,k(l \neq k)$. We will explain more detail on the combining method via CRT in Section 4.2.

1.7 Problem Identification

As the hard disk drive's capacities increase by reducing the size of track width, many errors were generated in the system. So, the error correcting code was used to protect and correct the error. Low density parity check codes (LDPC) have recently received considerable attention as the coding technique for the storage channels. They can show the performance close to the Shannon Capacity and exhibit simple encoding and decoding structures. The performance of LDPC codes depends on the parity check matrix. The parity check matrix can separate into two types of the constructions. The first one is random construction that offers flexibility in term of design and construction. The lack of connection regularity between row and column in random construction will increases decoder interconnection complexity. This presents serious disadvantages in terms of storing and accessing a large parity check matrix, encoding data, and analyzing code performance. These problems will be overcome with the structure construction. The structure parity check matrix has regular interconnection patterns, which reduces hardware complexity for encoders and decoders. From a short to moderate block lengths, the structural parity check matrix such as array LDPC codes perform significantly better performance while compared to the random regular LDPC codes.

However, array LDPC codes do not support arbitrary code lengths. Because of the code length of an array LDPC code with good error rate performance is limited to a multiple of a prime submatrix size. Therefore, when we use an array LDPC code in a magnetic recording system, we have to add dummy bits to match with the code length of the array LDPC code that we need to design. Since the dummy bits contain no information data, the code rate is degraded. That directly impacts to magnetic recording system because it needs to keep more information in the sector. So, we propose the Nonprime Modified Array LDPC codes in this thesis. It can support arbitrary code lengths because the submatrix size of the Nonprime Modified Array LDPC codes can be a nonprime number.

1.8 Objective and Expectation

Recently, LDPC codes have been widely studied due to its superior performance. It has been accepted by many industry standards such as DVB, IEEE802.16 and IEEE802.11n. One of the concerns related to the performance of LDPC is its parity check matrix. So, the goal of this thesis is designing the parity check matrix that achieves the suitable performance in any arbitrary code length. Most of the design uses prime parameters to design their parity check matrix because of they aware 4 cycle generate in their design when use the non prime parameter. This thesis will introduce the non prime parameter of the parity check matrix that fixes the problem in LDPC code. The design is based on the Array LDPC codes and Modified Array LDPC codes. The results of this design can use in arbitrary code length but still has attractive properties such as simple encoding, low error floor and capability of detecting and correcting burst error.

1.9 Research Hypothesis

The performance of LDPC codes depends on the parity check matrix. The parity check matrix was constructed with parameters such as row and column weights, rate, girth, and code length. The structure parity check matrix has regular interconnection patterns, which reduces hardware complexity for encoders and decoders. From a short to moderate block lengths, the algebraically constructed QC-LDPC codes perform significantly better as compared to the random regular LDPC codes. However, the structure parity check matrix is limitation in terms of code rate, codeword length and code girth. The code's performance is inferior to the randomly generated codes when using longer block length. Then, this thesis will study the problem of structure parity check matrix when designs with long block length. The non-prime parameter with parity check matrix will introduce to fix the problem of cycle-4 long block length LDPC.

1.10 Conceptual Framework

The originally Low Density Parity Check codes or LDPC codes were proposed as random construction. So, the implementation of these codes is more complexity and using more memory requirements. After that, the LDPC with structure parity check matrix or Array codes was proposed to reduce hardware complexity for encoders and decoders. Array LDPC codes were developed by many researchers for improving the performance of LDPC code. Modified Array LDPC codes are the type of Array LDPC codes. They were designed by applying the cyclic shift to Array code for achieving the properties of simple encoding. But, these array matrix still achieve the poor performance when design with non prime submatrix size. So, this thesis

will propose the design of the parity check matrix that achieve a good performance when design with non prime submatrix size.

1.11 Scope & Constraint

This thesis will present the construction of Array LDPC codes for support an arbitrary code length design. The propose method will eliminate the 4 cycles in the parity check matrix while design the code with non prime parameter. The performance of our design methods was tested by simulating with MATLAB program. The code performance of the propose method will compare with Modified Array LDPC codes and Interleaved Modified Array LDPC codes on white Gaussian noise channel. In this thesis, we investigate the code on the 4Kbits block length that is the block length in magnetic recording applications.

This thesis will organize as in the following.

The first chapter explains an overview of hard disk drive system, the magnetic recording channel, hard disk sector structures, error correcting code in hard disk drive, problem identification, literature review of LDPC codes, objective and expectation, research hypothesis, conceptual framework, scope & constraint.

The second chapter gives an overview of linear block codes. Perpendicular Recording Channels such as read and write process, noise and distortions and system diagram is also present. The types of LDPC code, encoding and decoding method with examples and some of the literature review discusses in this chapter.

The third chapter shows the design and performance of purpose method. The different parameter which characterizes the LDPC performance was simulated to find the suitable parameter for magnetic recording application.

The fourth chapter shows the application to apply the design method to another design.

The fifth chapter is a conclusion and some perspectives are finally given at the end of this thesis.

Chapter 2
Linear Block Code and LDPC Codes

In this chapter, an introduction of Linear Block Codes and LDPC codes will present. The chapter begins in Section 2.1 with the basic concept of Linear Block Code. Then, an introduction of LDPC code present in Section 2.2. The types of LDPC codes with regular and irregular are introduced in Section 2.3. Then, the Tanner graphs, cycle length and girth of LDPC codes present in Section 2.4. Section 2.5 discusses on the method to construct LDPC codes. The encoding and decoding process on LDPC codes were described in Section 2.6 and 2.7. The several constructions of array LDPC codes presented in Section 2.8, 2.9 and 2.10.

2.1 Linear block code

Linear block codes are the concept of the simple codes that are basically an extension of the single bit parity check codes for error detection. A single bit parity check code is one of the most common forms of detecting transmission errors. This code uses one extra bit in a block of n data bits to indicate whether the number of 1s in a block is odd or even. If a single error occurs, the parity bit will not correspond to the number of detected 1s in the information bit sequence. So, the single error is detected.

Linear block codes extend this notion by using a larger number of parity bits to either detect more than one error or correct for one or more errors. Unfortunately linear block codes trade their error detection or correction capability for either bandwidth expansion or a lower data rate. This thesis will restrict the attention to binary codes, where both the original information and the corresponding code consist of bits taking a value of either 0 or 1.

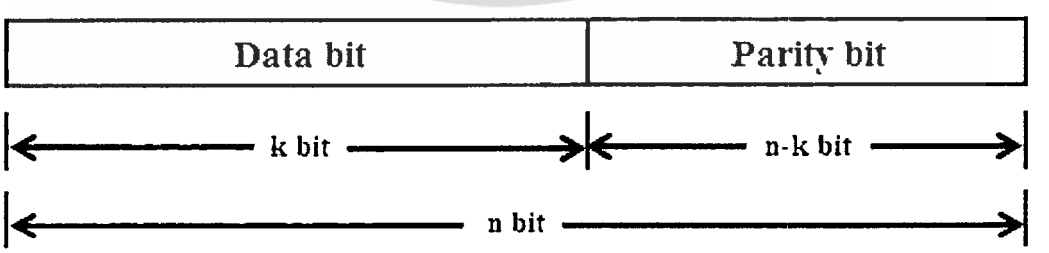


Figure 2.1 Binary codewords structure.

2.1.1 Binary Linear Block Codes

A binary block code generates a block of n coded bits from k information bits. We call an (n, k, d) binary block code, where d is the Hamming distance. The elements of the code C are called codewords. The input message, there is a total of 2^k distinct messages, will transform into a n-tuple codewords C by the encoder process, where $n > k$. The codewords C show in Figure 2.1. A codewords is divided into two parts, the message part and the redundant checking part. The message part consists of k information bits and the redundant checking part consists of $n - k$ parity check bits, which are linear sums of the information digits. A linear block code with this structure is referred to as a linear systematic block code. A codewords are transmitted across the channel with the code rate

$$R_c = k / n \tag{2.1}$$

In fact, a binary block code is linear if the modulo 2 sum of two code words is also a code word. The block code given in Table 2.1 is a (6, 3) linear code. One can easily check that the sum of any two code words in this code is also a code word.

Table 2.1 Linear block code with $k = 3$ and $n = 6$

$\mathbf{m} = [m_1\ m_2\ m3]$	$C = [c_1\ c_2\ c_3\ c_4\ c_5\ c_6]$
0 0 0	0 0 0 0 0 0
1 0 0	1 0 0 1 0 0
0 1 0	0 1 1 0 1 0
1 1 0	1 0 1 1 1 0
0 0 1	1 0 1 0 0 1
1 0 1	0 1 1 1 0 1
1 1 0	1 1 0 0 1 1
1 1 1	0 0 0 1 1 1

2.1.2 Hamming distance

Hamming distance of the code *C* is the smallest distance between any two different codewords. The Hamming distance between two vectors of the same length is defined as the number of positions in which these two vectors differ. The Hamming distance between a vector and the all zero vector is called the weight of the vector. The Hamming distance *d* is an important parameter of a code *C*. A code with Hamming distance *d* can correct *t* errors

$$t = \frac{\left| d_{\min} - 1 \right|}{2} \tag{2.2}$$

And it can detect *e* errors.

$$e = d_{\min} - 1 \tag{2.3}$$

2.1.3 Generator Matrix

The generator matrix is a compact description of how codewords are generated from message bits in a linear block code. The design goal in linear block codes is to find generator matrices such that their corresponding codes are easy to encode and decode. This matrix also has a powerful error correction/detection capabilities. Consider an (*n*, *k*) code with k message bits denoted as

$$m = [m_1, m_2, \ldots, m_k] \tag{2.4}$$

That is encoded into the codeword

$$C = [c_1, c_2, \ldots, c_n] \tag{2.5}$$

It can represent the encoding operation as a set of n equations defined by

$$C = mG \tag{2.6}$$

where the $k \times n$ generator matrix G for the code is defined as

$$G = \begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1n} \\ g_{21} & g_{22} & \cdots & g_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ g_{k1} & g_{k2} & \cdots & g_{kn} \end{bmatrix} \tag{2.7}$$

Since a linear block code is a subspace of dimension k in the larger n dimensional space. The k row vectors, g_{ij} =1, of G must be linearly independent. So, they span the k dimensional subspace associated with the 2^k codewords. Since the set of basis vectors for this subspace is not unique, the generator matrix is also not unique. A systematic linear block code is described by a generator matrix of the form

$$G = [I_k \mid P] = \left[\begin{array}{cccc|cccc} 1 & 0 & \cdots & 0 & p_{11} & p_{12} & \cdots & p_{1(n-k)} \\ 0 & 1 & \cdots & 0 & p_{21} & p_{22} & \cdots & p_{2(n-k)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & p_{k1} & p_{k2} & \cdots & p_{k(n-k)} \end{array}\right] \tag{2.8}$$

where I_k is a $k \times k$ identity matrix and P is a $k \times (n - k)$ matrix that determines the redundant, or parity, bits to be used for error correction or detection.

2.1.4 Parity Check Matrix and Syndrome

The parity check matrix is used to decode linear block codes with generator matrix G. The parity check matrix H corresponding to a generator matrix $G = [I_k|P]$ is defined as

$$H = \left[P^T \mid I_{n-k}\right] \tag{2.9}$$

It is easy to verify that $GH^T = 0$, where 0 denotes an all zero $k \times (n\text{-}k)$ matrix. Recall that a given codeword C in the code is obtained by multiplication of the message bits sequence m by the generator matrix G.

$$CH^T = mGH^T = 0 \tag{2.10}$$

For any input sequence *m*, where 0 denotes the all zero row vector of length *n-k*. Thus, multiplication of any valid codeword with the parity check matrix results in an all zero vector. This property is used to determine whether the received vector is a valid codeword or has been corrupted.

Based on the notion of syndrome, let *r* be the received codeword resulting from transmission of codeword *C*. In the absence of channel errors, $r = C$. However, if the transmission is corrupted, one or more of the codeword symbols in *r* will differ from those in *C*. We can write the received codeword as

$$r = C + e \tag{2.11}$$

Where $e = [e_1, e_2, ..., e_n]$ is the error vector indicating which codeword symbols were corrupted by the channel. We define the syndrome of *r* as

$$S = rH^T \tag{2.12}$$

If *r* is a valid codeword then $S = CH^T = 0$. Thus, the syndrome equals the all zero vector if the transmitted codeword is not corrupted. It is corrupted in a manner such that the received codeword is a valid codeword different from the transmitted codeword. If the received codeword *r* contains the detect errors, then $S \neq 0$. If the received codeword contains correctable errors, then the syndrome identifies the error pattern corrupting the transmitted codeword, and these errors can then be corrected. Note that the syndrome is a function only of the error pattern e and not the transmitted codeword *C*, since

$$S = rH^T = (C + e)H^T = CH^T + eH^T = 0 + eH^T. \tag{2.13}$$

Since $S = eH^T$ corresponds to n-k equations in n unknowns, there are 2^k possible error patterns that can produce a given syndrome *S*. However, the probability of bit error is typically small and independent for each bit. The most likely error pattern is the one with minimal weight that corresponds to the least number of errors introduced in the channel. If an error pattern $\hat{e}$ is the most likely error associated with a given syndrome *S*, the transmitted codeword is typically decoded as

$$\hat{C} = r + \hat{e} = C + e + \hat{e} \tag{2.14}$$

2.2 Introduction to LDPC codes

Low density parity check (LDPC) codes are forward error correction codes and are the class of linear block codes. They were corresponding to the sparse parity check matrix *H*. A parity check matrix with size *m* by *n* is low density because it has a small number of 1's in each row $w_r << m$ and each column $w_c << n$. LDPC codes are regular if w_c is constant for every column and $w_r = w_c (n=m)$ is also constant for every row. In LDPC encoding, the codeword $(c_0,c_1,c_2,c_3, ...,c_n)$ consists of the message bits $(m_0,m_1,m_2,...,m_k)$ and some parity check bits. The equations derived from the H matrix are used to generate parity check bits. LDPC codes were first proposed by Robert Gallager in 1962 but they were forgotten for a long time due to the requirement of high complexity computation when very long codes are considered. In 1993, Berrou et. al. proposed turbo codes. This is the new encoding/decoding technique, with the complexity is slightly larger than the convolution codes. Their performance was approaching to the Shannon capacity of the AWGN channel. The invention of turbo codes led the research community to focus on iterative decoding algorithms. In 1995, LDPC codes were rediscovered by MacKay and Neal. After the rediscovery of LDPC codes, they were widely used for the communication system. Because of, they have a lower decoding complexity and the code's performance is closer to the capacity of the Shannon limit.

2.3 Types of LDPC codes

A low density parity check (LDPC) codes are characterized by its sparse parity check matrix. The parity check matrix contain zero elements and few nonzero elements. The important parameters are codeword length, *n*, rows, *j*, columns, *k*, and number of parity bits, $m = n - k$. The number of nonzero elements in a row of the parity check matrix is called the row weight, denoted by w_r. The number of nonzero elements in a column of the parity check matrix is called the column weight, denoted by w_c. There are two general classes of LDPC code:

2.3.1 Regular LDPC Codes

A LDPC code is regular if the number of 1's in column w_c and the number of 1s in row w_r in a parity check matrix *H* are constant. Different types of regular coding can be stressed in coding theory field. Mainly, the well known ones can be listed as follows: Gallager Codes, Quasi Cyclic Codes, Array Codes and Random Codes. Moreover different code rates are possible for different techniques. The sample of regular matrix is shown as

$$H=\begin{bmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix} \quad (2.15)$$

The example matrix from equation (2.15) is regular with w_c = 2 and w_r = 4. It is also possible to see the regularity of this code while looking at the graphical representation in Figure 2.2. There is the same number of incoming edges for every *v* node and is also for all the *c* nodes. The code rate for regular LDPC is

$$R=1-(w_c / w_r) \quad (2.16)$$

2.3.2 Irregular LDPC Codes

A LDPC code is irregular if the number of 1's in columns and rows are not constant in a parity check matrix *H*. Irregular LDPC Codes have an important impact in the coding theory since it is stated in [27]. They perform better than regular ones. Different types of irregular codes have been developed. They can be listed as follows: Modified Array Codes, Poisson, Sub-Poisson, Moderately Super-Poisson, Very Super-Poisson, Fast encoding versions. Irregular LDPC codes can be parameterized by the degree polynomials $\lambda(x)$ and $\rho(x)$, which can be defined as

$$\lambda(x)=\sum_{i=2}^{d_l}\lambda_i x^{i-1} \quad (2.17)$$

and

$$\rho(x)=\sum_{i=2}^{d_r}\rho_i x^{i-1} \quad (2.18)$$

where $\lambda(x)$ and $\rho(x)$ are the fractions of edges belonging to degree-*i* variable and check nodes. d_l and d_r are the maximum variable and check node degrees respectively. The optimization of the $\lambda(x)$ and $\rho(x)$ is found by optimization algorithm. The code rate for irregular LDPC is

$$R=1-\left[\left(\sum_{k=1}^{n} w_{ck} / k\right)-\left(\sum_{j=1}^{n} w_{rj} / j\right)\right] \quad (2.19)$$

2.4 Representations of LDPC codes

Although a LDPC codes are defined by a sparse parity check matrix *H*. A bipartite graph, also known as Tanner graph [28], can be used to represent the code. A bipartite graph is a graph whose nodes can be divided into two sets that each node is connected to a node in the other set. The two sets of nodes in a Tanner graph are called check nodes and variable nodes that represent rows and columns respectively. Figure 2.2 shows a parity check matrix with a corresponding Tanner graph. The a^{th} check node is connected to b^{th} variable node if $H_{a,b}=1$. Check nodes $f_0..f_3$ represent the four rows of the matrix, whereas $v_0..v_8$ are the columns. The number of edges in each check node is equal to the row weight and the number of edges in each variable node is equal to the column weight. The row and column weights are four and two respectively in this example.

The Tanner graph is not only allows efficient decoding of the received word but also offers a base to study various aspects that influence the performances of the LDPC codes. Some of these concepts are cycle and girth.

Definition 2.1 (Cycle). *In a graph, a cycle is a path that starts from a node* n *and ends in* n.

Definition 2.2 (Girth). *The* girth *of a graph is the length of the smallest cycle in that graph.*

In Figure 2.2, a cycle 4 (i.e. a cycle of length 4) is depicted. The girth of the graph is evidently 4 since considered one of the important parameters of a LDPC code. It is commonly accepted that the presence of short cycles in the graph is one of the main parameters that affect the coding gain. So, the LDPC will design to avoid the cycle 4 in *H* matrix.

$$H=\begin{bmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix}$$

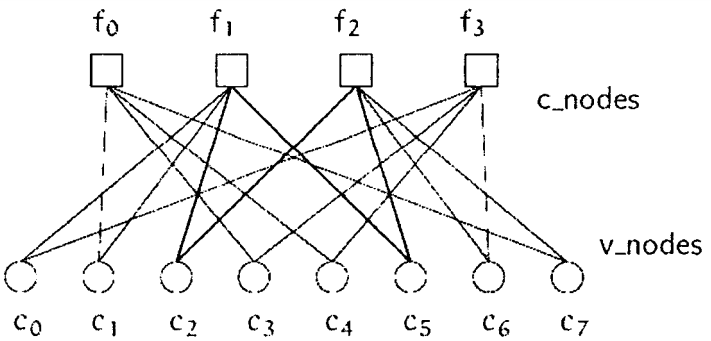


Figure 2.2 Parity check matrix H and Tanner Graph with cycle 4.

2.4.1 Cycle length of LDPC codes

A Tanner Graph is possible to view the definition of the minimum cycle length of a code. The minimum number of edges traveled from one check node to return to the same check node. Length 4 and 6 cycles with corresponding to parity check matrix configurations are shown in Figures 2.3 respectively.

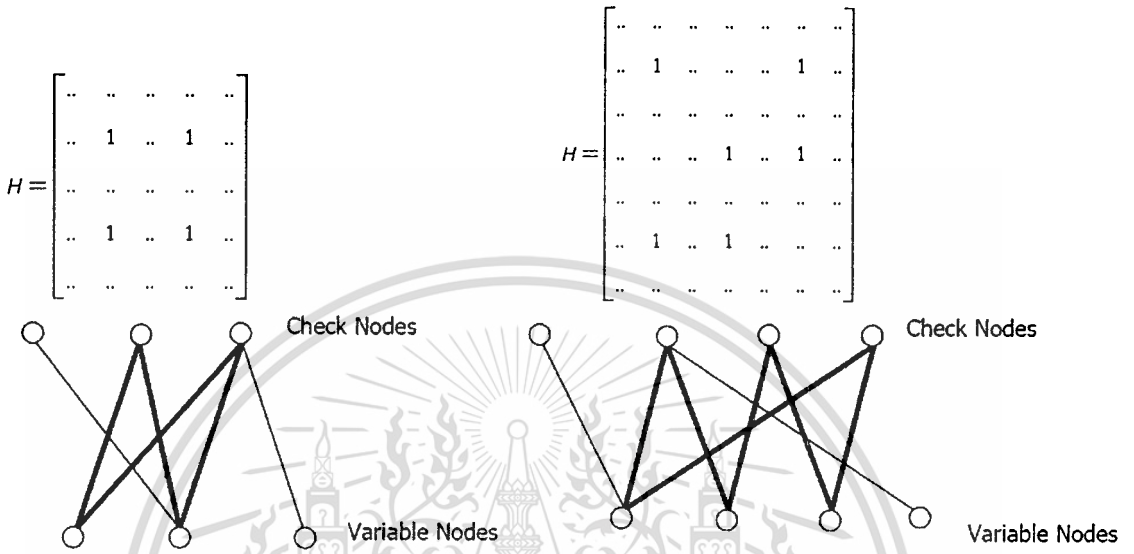


Figure 2.3 Length 4 and 6 cyclesLength 4 and 6 cycles.

Figure 2.3 has been shown that the existence of these cycles degrades the performance during the iterative decoding process. Therefore when generating the parity check matrix, the minimum cycle length permitted must be determined. It is possible to control the minimum cycle length when generating the matrix. However, the computational complexity and time increases exponentially with each increase in minimum cycle length.

2.5 Constructing LDPC Codes

The construction of LDPC codes requires the definition of the pattern of connection between rows and columns of a parity check matrix or between check nodes and variable nodes of a corresponding Tanner graph. The construction process will consider LDPC code parameters such as row and column weights, rate, girth and code length. The main objectives in code construction are good decoding performance and easier hardware implementation. A LDPC code could be constructed such that it has low hardware complexity and cost. This is mostly achieved by having row and column connections that have a regular pattern. Good error correcting performance and low complexity hardware characteristics could also be optimized at the same time. However, putting constraints on construction methods to obtain hardware aware codes may degrade or limit decoding

performance. Construction methods can be either random (unstructured row-column connections) or structured (row-column connections predefined in some way). Random constructions have flexibility in design and construction but lack row and column connections regularity., which increases decoder interconnection complexity. Structured constructions may have regular interconnection patterns but often produce a class of codes limited in rate length and girth.

2.5.1 Random based Construction

Random constructions connect rows and columns of a parity check matrix without any structure or predefined connection pattern. They are actually pseudo random connections done by computer searches. Constructions could be done in the Tanner graph by connecting check nodes to variable nodes with edges or in the parity check matrix by connecting rows to columns with 1's entries where all other entries are 0's. Randomly adding edges to a Tanner graph or 1's entries in the parity check matrix will not produce a desired rate and will probably have cycles of four. However, the resulting code could be optimized by either post processing or by putting constraints on the random choices as the code is built. Post processing exchanges or deletes some connections in order to get a desired girth and rate. Random construction with constraints adds a connection in the code if it does not violate the desired girth or row and column weights.

The first constructions of LDPC codes were random ones. The parity check matrix is the concatenation and/or superposition of sub-matrices. These submatrices are created by processing some permutations on a particular submatrix which usually has a column weight of 1. R.Gallager's construction is based on a short matrix H_0. Then j matrices $\pi_i(H_0)$ are vertically stacked on H_0, where $\pi_i(H_0)$ denotes a column permutation of H_0.

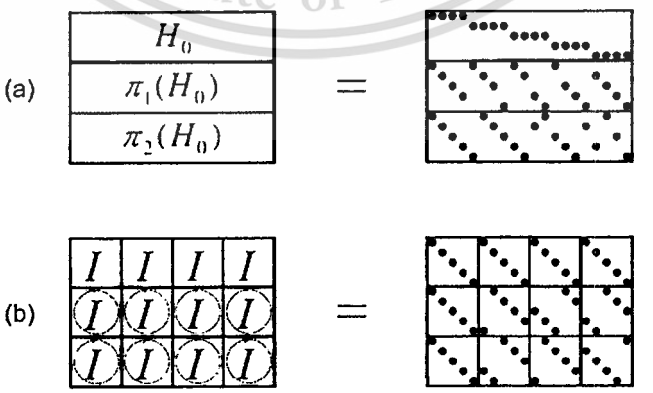


Figure 2.4 Example of (a) Gallager's construction, (b) MacKay's construction.

Mackay [6] was interested in a minimum the number of short cycles in the Tanner graph representing the parity-check matrix. He designed the parity check matrix that has short cycles. These methods were illustrated on Figure 2.4. Regular and irregular codes can be also constructed like [29], where the 2 sets of nodes are created, each node appearing as many times as its degree's value. Then, a one to one association is randomly mapped between the nodes of the 2 sets, like illustrated in Figure 2.5.

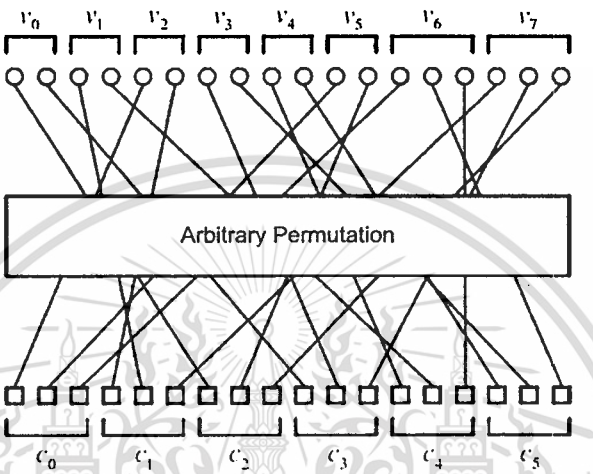


Figure 2.5 Random construction based on [Luby et al. 2001].

2.5.2 Structured Constructions

Structured construction methods put constraints on row and column connections to get a desired or predefined pattern. The main objectives of this method is achieving the good performance as the random LDPC code. At the same time, it has a connection pattern that is easier to implement in hardware. There are many structured methods already developed producing codes differing in structure (row and column interconnection pattern), performance and hardware implementation complexity. Developed methods include these requirements such as Finite Geometry [30] [31], algebraic method and etc. Some of structured methods will present in Section 3.8, 3.9 and 3.10. Structured on connections can reduce hardware complexity and the cost of encoders and decoders. For example, a code matrix divided into submatrices can be mapped onto a decoder with the number of processing nodes equal to the number of row and column submatrices. Since there are fewer submatrices compared to individual rows or columns, the number of processing nodes and interconnections is also reduced. In most structured codes, several rows or columns have a common connection rule. Hardware implementations of codes can be simplified in a group of messages between check and variable nodes that can be routed using a single connection rule. A major

drawback of structured codes is that performance is degraded compared to random codes for longer codes.

2.6 Encoding of LDPC codes

Although LDPC codes achieve better performance and have low decoding complexity compared to Turbo codes. But, they have high encoding complexity and encoding delay. So, the encoding of LDPC codes has been considered as the major problem on LDPC codes. The encoding for LDPC codes comprise of two tasks. First, start with constructing a sparse parity check matrix then generates codewords by using a parity check matrix. There are two methods employed to encode messages into codewords for LDPC codes. The first method is encoded by multiplying the vector corresponding to the information bits with the generator matrix as using in any linear code. The second method is an approximate lower triangulations that proposed by Richardson and Urbanke to reduce the complexity of the first method.

2.6.1 General case

Similar to all other linear block codes, an LDPC code can be encoded by multiplying the message m by a generator matrix *G*. This is represented as

$$C = mG \tag{2.20}$$

Where

$C = [\, c_1\ c_2\ c_3\ \ldots\ c_n \,]_{(1xN)}$ is the N bit codeword

G is the N x K generator matrix of the code in which

$$GH^T = 0 \tag{2.21}$$

The codeword C can be written in systematic form by

$$C = \left[m_1 m_2 \ldots m_k\ p_1 p_2 \ldots p_{(n-k)} \right]$$

or

$$C = \left[m \middle| p \right] \tag{2.22}$$

Where

p is the $1x(N\text{-}K)$ of parity bit

m is the $1xK$ of message bit

Since LDPC codes are usually defined through parity check matrix H, rather than their generator matrix G, the encoding process can recall as

$$CH^T = 0 \tag{2.23}$$

From equation (2.23), we can find p for generate the codewords as

$$H = \left[H_1 \middle| H_2 \right] \text{ and } H^T = \left[\frac{H_1^T}{H_2^T} \right] \tag{2.24}$$

Replace C and H in equation (2.23) with equation (2.22) and (2.24), where H_2 is KxK matrix

$$CH^T = \left[m \middle| p \right] \left[\frac{H_1^T}{H_2^T} \right] = 0$$

$$CH^T = mH_1^T + pH_2^T = 0$$

So, the matrix p present as

$$p = mH_1^T + (H_2^T)^{-1} \tag{2.25}$$

We can put the p value from equation (2.25) in equation (2.22) to generate the codewords. Equation (2.23) can use to verify the codewords. An example 2.1 supposes that we want to send the message m = [1011] over the channel can encode as follow.

Example 2.1 :

Step 1 : Considering and parity check matrix that used to encode message bit as below

$$H=\begin{bmatrix}1&0&0&0&1&1&1\\0&1&0&1&1&0&1\\0&0&1&1&0&1&1\end{bmatrix}$$

Step 2 : Find H_1^T and H_2^T by equation (2.24)

$$H=[H_1\,|\,H_2]=\left[\begin{array}{cccc|ccc}1&0&0&0&1&1&1\\0&1&0&1&1&0&1\\0&0&1&1&0&1&1\end{array}\right]$$

So,

$$H_1^T=\begin{bmatrix}1&0&0\\0&1&0\\0&0&1\\0&1&1\end{bmatrix} \text{ and } H_2^T=\begin{bmatrix}1&1&0\\1&0&1\\1&1&1\end{bmatrix}$$

Step 3 : Find matrix $\left(H_2^T\right)^{-1}\left(H_2^T\right)^{-1}$

$$\left(H_2^T\right)^{-1}=\frac{1}{\left|H_2^T\right|}.\text{Adjoint}\left(H_2^T\right)$$

where

$\left|H_2^T\right|$ is the determinant of matrix H_2^T that is not equal to 0

$\text{Adjoint}\left(H_2^T\right)$ is the transpose matrix of $\text{Cofactor}\left(H_2^T\right)$

$\text{Cofactor}\left(H_2^T\right)$ is the element product of matrix $Minor\left(H_2^T\right)$ and $(-1)^{i+j}$,

where i and j is the position value on each element

$Minor(A)$ is a determinant of submatrix

Step 3.1 : Find $\left|H_2^T\right|$

$$\left|H_2^T\right| = \begin{matrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{matrix} \begin{matrix} 1 & 1 \\ 1 & 0 \\ 1 & 1 \end{matrix} = (0+1+0)-(0+1+1) = 1-2 = -1$$

Step 3.2 : Find Adjoint$\left(H_2^T\right)$

$$H_{2_{11}}^T = +\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} = -1, H_{2_{12}}^T = -\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} = 0, H_{2_{13}}^T = +\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} = 1$$

$$H_{2_{21}}^T = -\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} = -1, H_{2_{22}}^T = +\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} = 1, H_{2_{13}}^T = -\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} = 0$$

$$H_{2_{31}}^T = +\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = 1, H_{2_{32}}^T = -\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} = -1, H_{2_{13}}^T = +\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} = -1$$

So,

$$\text{Adjoint}\left(H_2^T\right) = \left[cofactor\left(H_2^T\right)\right]^T = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 1 & 0 \\ 1 & -1 & -1 \end{bmatrix}^T = \begin{bmatrix} -1 & -1 & 1 \\ 0 & 1 & -1 \\ 1 & 0 & -1 \end{bmatrix}$$

Step 3.3 : Find matrix $\left(H_2^T\right)^{-1}$

$$\left(H_2^T\right)^{-1} = \frac{1}{\left|H_2^T\right|}.\text{Adjoint}\left(H_2^T\right) = \frac{1}{-1}\begin{bmatrix} -1 & -1 & +1 \\ 0 & +1 & -1 \\ +1 & 0 & -1 \end{bmatrix} = \begin{bmatrix} +1 & +1 & -1 \\ 0 & -1 & +1 \\ -1 & 0 & +1 \end{bmatrix}$$

Step 4 : Find parity bit p by using equation (2.25)

$$mH_1^T = \begin{bmatrix} 1 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \end{bmatrix}$$

Then,

$$p = mH_1^T + (H_2^T)^{-1} = \begin{bmatrix} 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} +1 & +1 & -1 \\ 0 & -1 & +1 \\ -1 & 0 & +1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

we can get the codewords as

$$C = \begin{bmatrix} m | p \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \end{bmatrix}$$

We can use equation (2.23) to confirm our codeworeds

$$CH^T = 0, \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$$

For this method, the encoder has complexity in the order of $O\left(n^2\right)$ operations. If n is large for LDPC codes, from thousands to hundreds of thousands of bits, the encoder can become prohibitively complex on computation. To overcome the encoding complexity, linear time encoding for LDPC codes was designed to reduced complexity encoding.

2.6.2 Linear time encoding for LDPC codes

The complexity of conventional encoding is essentially proportional to the code length and becomes a significant problem when using with long code lengths. To solve this problem, Richardson and Urbanke [27] proposed an efficient encoding algorithm for LDPC codes. The idea is to do a transformation of the parity check matrix by using only row and column permutations to keep H sparse. Any arbitrary sparse matrix can be converted into the desired parity check matrix H with an approximate lower triangular form as shown in Figure 2.6.

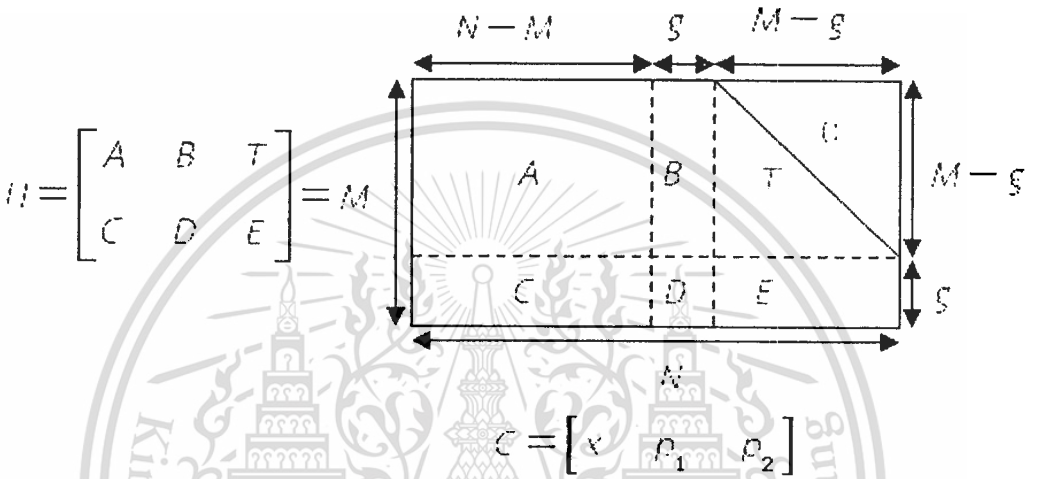


Figure 2.6 Parity check matrix H in approximate lower triangular form.

An alternative approach to encoding LDPC based on approximate lower triangulations can separate into 3 step process.

Step 1 : Perform row and column permutations to bring H into an approximate lower triangular form:

$$H_t = \begin{bmatrix} A & B & T \\ C & D & E \end{bmatrix}$$

The g rows of H in submatrices C, D, and E is called the gap of the approximate triangular representation. The submatrix T is a lower triangular matrix of size $(m - g) \times (m - g)$. If H_t is full rank, the submatrix A is size $(m - g) \times k$, B is size $(m - g) \times g$,C is $g \times k$, D is $g \times g$ and finally E is $g \times (m - g)$. Keeping the size of the gap low will reduce the encoding complexity for the LDPC code.

Step 2 : Once the lower triangular format of T is obtained, we use Gauss elimination to clear E which is equivalent to the following pre-multiplication:

$$\tilde{H}=\begin{bmatrix} I & 0 \\ -ET^{-1} & I \end{bmatrix}\begin{bmatrix} A & B & T \\ C & D & E \end{bmatrix}=\begin{bmatrix} A & B & T \\ -ET^{-1}A+C & -ET^{-1}B & 0 \end{bmatrix}=\begin{bmatrix} A & B & T \\ \hat{C} & \hat{D} & 0 \end{bmatrix} \quad (2.26)$$

where we denote

$$\hat{C}=-ET^{-1}A+C \quad (2.27)$$

$$\hat{D}=-ET^{-1}B+D \quad (2.28)$$

Step 3 : To encode the message using $\tilde{H}$ the codeword $c=\left[c_1,c_2,K,c_n\right]$ is divided into three parts by $c=\left[m,p_1,p_2\right]$, where $u=\left[m_1,m_2,K,m_k\right]$ is the message, $p_1=\left[p_{1_1},p_{1_2},K,p_{1_g}\right]$ holds the first g parity check bits and $p_2=\left[p_{2_1},p_{2_2},K,p_{2_{m-g}}\right]$ The codeword c must satisfy the syndrome test $c\tilde{H}=0$,then

$$Am+Bp_1+Tp_2=0, \quad (2.29)$$

$$\hat{C}m+\hat{D}p_1+0p_2=0 \quad (2.30)$$

Since E has been cleared, the parity bits in p_1 depend on the message bits, and can be calculated independently of the parity bits in p_2. If $\hat{D}$ is invertible, p_1 can be found by $p_1=\hat{D}^{-1}\hat{C}m$. If $\hat{D}$ is not invertible the columns of $\tilde{H}$ can be permuted until it is. By keeping g as small as possible the added complexity load of the matrix multiplication to obtain is kept to a minimum.

Once p_1 is known p_2 can be found by $p_2=-T^{-1}\left(Am+Bp_1\right)$. The sparseness of A , B and T can be employed to keep the complexity of this operation low and, as T is lower triangular, 2 p can be found using back substitution. An example of encode the message m = [1 1 0 0 1] by Linear time encoding can show as follow example.

Example 2.2: We wish to encode the message m = [1 1 0 0 1] with the LDPC code length 10 and rate -1/2 by

$$H=\begin{bmatrix}1&1&0&1&1&0&0&1&0&0\\0&1&1&0&1&1&1&0&0&0\\0&0&0&1&0&0&0&1&1&1\\1&1&0&0&0&1&1&0&1&0\\0&0&1&0&0&1&0&1&0&1\end{bmatrix}$$

Step 1 : Perform row and column permutations to bring H into an approximate lower triangular form. For this H we swap the 2nd and 3rd rows and 6th and 10th columns to obtain:

$$H_t=\left[\begin{array}{cccccc|cc|ccc}1&1&0&1&1&0&0&1&0&0\\0&1&1&0&1&1&1&0&0&0\\0&0&0&1&0&0&0&1&1&1\\\hline1&1&0&0&0&1&1&0&1&0\\0&0&1&0&0&1&0&1&0&1\end{array}\right]=\begin{bmatrix}A&B&T\\C&D&E\end{bmatrix},\ \text{where } g=2$$

Step 2 : Once the lower triangular format of T is obtained, we use Gauss elimination to clear E which is equivalent to the following premultiplication from equation (2.26):

$$\tilde{H}=\begin{bmatrix}A&B&T\\\hat{C}&\hat{D}&0\end{bmatrix}$$

Find $\hat{C}$ and $\hat{D}$ with equation (2.27) and (2.28)

$$\hat{C} = -ET^{-1}A + C$$

$$= \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \end{bmatrix} + \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\hat{D} = -ET^{-1}B + D$$

$$= \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$

So,

$$\tilde{H} = \begin{bmatrix} A & B & T \\ \hat{C} & \hat{D} & 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}$$

When applying Gauss-Jordan elimination to clear E only $\hat{C}$ and $\hat{D}$ are effected, the rest of the parity-check matrix remains sparse.

Step 3 : To encode the message using $\tilde{H}$:

$$p_1 = \hat{D}^{-1}\hat{C}m = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}\begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \end{bmatrix}\begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

$$p_2 = -T^{-1}\left(Am + Bp_1\right)$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\left(\begin{bmatrix} 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix}\right)$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\left(\begin{bmatrix} 1 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}\right)$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

So, codeword $c = \left[m, p_1, p_2\right] = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \end{bmatrix}$

This method is the most popular one for encoding LDPC codes and it has been adopted by the IEEE 802.11n and IEEE 802.16e standard. The advantage of these codes is decreases encoding complexity and lower memory requirement.

2.7 Decoding

The process of decoding tries to recover the transmitted codeword c from the received codeword v by using the parity check matrix H. Since $cH^T = 0$ defines the set of equations that must be satisfied in order to retrieve the received codeword, the relations between check nodes and variable nodes are taken from the Tanner graph where the (mod 2) sum of the values attached to variable nodes returns 0 to the check node in case the codeword is valid.

In Gallager's thesis, he introduced two algorithms: the bit flipping algorithm based on hard decision decoding and the belief propagation algorithm based on soft decision decoding. These algorithms are subclasses of message passing algorithms which are iterative decoding algorithms. The step for iterative algorithms are the passed messages that are probability estimates sent, updated and exchanged between variable and check nodes of the Tanner graph, check nodes measure the reliability of bit probability using estimations from adjacent variable nodes, the variable nodes estimate the probability that a given bit is 0 or 1 based on the estimation of the received bit from connected check nodes. Operations in the decoding algorithm are simplified by implementing log domain to replace multiplication and division operations by adding and subtracting, which reduces the implementation complexity of the decoder.

2.7.1 Bit-Flip Algorithm (Hard Decision Decoding)

The bit flipping algorithm is a hard decision algorithm deals with simple messages. The idea of this algorithm is to flip the least number of bits until all the parity check bit is satisfied. Suppose that each Bit node starts with a value of either zero or one. For any iteration, the bit node decides either to flip its value or to keep it unchanged. When a large number of the neighboring check equations are unsatisfied, the bit node decides to flip its value. This follows from the assumption that the bit node value which is in error. The decoding process will stop after a maximum number of iterations are reached or if all the parity check equations are satisfied. It can summarize the decoding algorithm in fallow:

Step 1 : Initialization. Variable nodes are assigned corresponding bit values from the received vector. Then, send these values as messages to the check nodes that connected with each variable node.

Step 2 : Check Nodes Update. Each check node will calculate a response for each variable node that connected with other bits. Then, the result with a sum of 0 will

send to satisfy the parity check equations. If all equations are satisfied the algorithm will terminate.

Step 3 : Bit Update. The variable nodes receive values from check nodes and determine the original received bit. The result in variable nodes depends on the majority voting of values received from check nodes. This process is repeated until satisfied all parity check equations or until the number of suggested iterations is reached. If the maximum number of iterations is reached, the algorithm will terminate and declares failure to converge. Note that the design of LDPC sparse H matrix ensures the existence of one or no transmission error in each parity check equation.

This algorithm is often interest for very high speed applications, such as optical networking. The hard decision has lower complexity than soft decision. We will explain the soft decision in the next section.

2.7.2 Message Passing Algorithm (Soft decision decoding)

Message Passing Algorithm or MPA is an iterative decoding algorithm. It used a factor graph to represent the code as linear block codes and LDPC codes. It is also known as a sum product algorithm (SPA) and it is normally referred as the belief propagation decoding algorithm. Contrary to the hard decision decoder, the belief propagation algorithm takes into consideration the information on the reliability of the received symbol given by the demodulator. The message passing algorithm used to estimate probabilities of bit such as intrinsic and extrinsic information. The extrinsic information is depending only on other nodes. It doesn't affect the node that it sent information to. In messages that contain intrinsic information, nodes will be dominated by its current value.

There are two types of probabilities that express the relation between a variable and an event E. The first is a priori probability of u with respect to E which is the probability that u is equal to a. It can denote by

$$P_t^{priori}(u=a)=P(u=a) \tag{2.31}$$

The second probability is a posteriori probability of u with respect to E. It is the probability of u given the outcome E and it is denoted by

$$P_t^{post}(u=a)=P(u=a|E) \tag{2.32}$$

and it can be written as

$$P(u=a|E)=\frac{1}{P(E)}P(E|u=a)P(u=a) \tag{2.33}$$

The term $P(E|u=a)$ is proportional to the extrinsic probability that describes new information for u obtained from E. and the extrinsic probability is given by

$$P_E^{ext}(u=a)=dP(E|u=a) \tag{2.34}$$

Where d is a constant that normalize the probability sum to 1. Thus the relation between these probabilities can be written as:

$$P_E^{post}(u=a)=P_E^{priori}(u=a)P_E^{ext}(u=a) \tag{2.35}$$

Since our concern is binary case, we can use log-likelihood ratio (LLR) in which probability of variable u is expressed in terms of a real number. Thus LLR of u is defined as

$$LLR(u)=\log\frac{P(u=1)}{P(u=0)}=\log\frac{p}{1-p} \tag{2.36}$$

where $p = P(u = 1)$. Now equation (2.36) can be rewritten as

$$LLR_E^{post}(u)=LLR_E^{priori}(u)+LLR_E^{ext} \tag{2.37}$$

where LLR(u) is positive if $p \geq 0.5$ and negative if $p < 0.5$. The extrinsic information reflects the incremental gain in knowledge of a posteriori information over a priori information. The message passing algorithm is based on a priori, extrinsic and a-osteriori probabilities. The a-priori information is taken from channel, where the extrinsic information comes from other nodes. The steps for MPA iterative decoding are listed below.

Step 1 : Initialization. The decoder is initialized the values by giving variable nodes from the received vector y_n of n bits. The initial probability bit that sent is 1 or 0 given by received vector. It is calculated for each variable node by

$$L(u_i) = \ln \frac{P(u_i = 1 | y_i)}{P(u_i = 0 | y_i)} \tag{2.38}$$

where $0 < i < n - 1$. In AWGN channel $L(u_i) = 2y_i / \sigma^2$, where σ^2 is the noise variance. Messages to check nodes and variable nodes, and check nodes LLRs are initialized to zero by sending values on variable nodes to connected check nodes.

Step 2 : Updating Check nodes. In this step LLR and check node to variable node messages are calculated. The message for each check nodes based on variable node messages. LLR for check nodes of number m denoted by λ_j where $0 < j < m - 1$ is given by

$$\lambda_j = \sum_{all-messages} \ln \left[\tanh \left(\frac{abs(\Omega_{i,j})}{2} \right) \right] \tag{2.39}$$

where $\Omega_{i,j}$ is the message from variable node i to check node j. check node message to variable node messages are given by

$$\Lambda_{j,i} = 2 \times a \tanh \left\{ \exp \left\{ \ln(\lambda_i) - \ln \left(\tanh \left(\frac{\Omega_{i,j}}{2} \right) \right) \right\} \right\} \tag{2.40}$$

Step 3 : Updating variable nodes. LLR and outgoing messages of variable nodes are calculated and LLR is given by

$$\lambda_i = L(u_i) + \sum_{all-messages} \Lambda_{j,i} \tag{2.41}$$

LLR is the sum of all incoming messages with addition to the initial value of variable node. Messages from variable node to check nodes are given by check nodes LLR minus messages received on that edge as given

$$\Omega_{i,j} = \lambda_i - \Lambda_{i,j} \tag{2.42}$$

Step 3 : Decision. The values of variable nodes are decided as 1 or 0 by the value λ_i. If λ_i < 0 then LLR_i = 0, and LLR_i = 1. If λ_i ≥ 0 or LLR × H^T = 0, then take the value of LLR as an estimation of codeword c_n at the decoder output. If not, go to step 2. Maximum number of iterations is decided so the algorithm stops in case the algorithm doesn't halt.

2.7.3 Log-domain Sum-Product Algorithm (Soft decision decoding)

Log-domain Sum-Product Algorithm or Log-SPA algorithm is the soft decision decoding algorithm for LDPC codes. Log-SPA is the types of sum product algorithm. In the decoding LDPC, we can separate them into 2 processes. First, construct the Tanner graph from the parity check matrix. Then, compute each message bit by using the Log-SPA algorithm. This algorithm uses the concept of message passes between check nodes and variable nodes as shown in Figure 2.7. In the graph, the check nodes and bit nodes have interchange soft information. The variable q_{ij} is the message sent from the i^{th} variable node to j^{th} check node along a connecting edge, and r_{ji} is the message sent from j^{th} check node to the i^{th} variable node along a connecting edge. The message q_{ij} is computed based on the values sent from check nodes connecting to the i^{th} variable node excluding the j^{th} check node. In summary, the Log-SPA algorithm goes through 5 steps as follows:

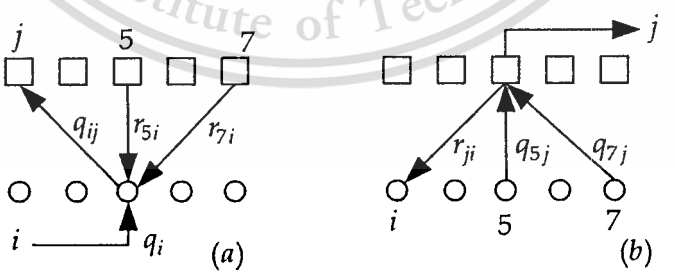


Figure 2.7 Message passed between bit node and check node (a) from i^{th} bit node to j^{th} check node (b) from j^{th} check node to i^{th} bit node.

Step 1 : Compute the initial value of $L(q_{ij})$ transmitted from the variable node i to check node j; for all i ; $1 \leq i \leq n$

$$L(q_{ij}) = L(c_i) = \frac{2y_i}{\sigma^2} = LLR_i = \log\left(\frac{P(c_i = 0 \mid y_i)}{P(c_i = 1 \mid y_i)}\right) \qquad (2.43)$$

Where

$L(c_i)$ denotes log likelihood ratio

σ^2 denotes derivation of white noise

$P(c_i = x \mid y_i)$ denotes probability for given input y_i

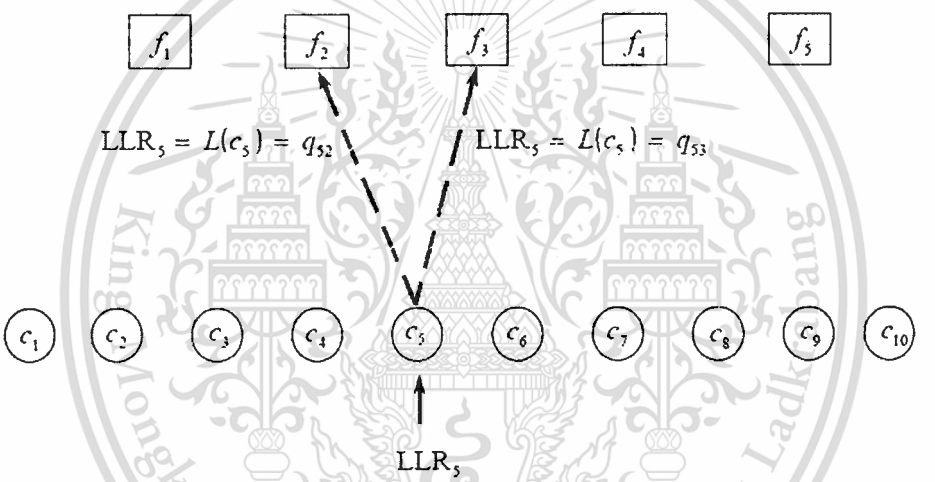


Figure 2.8 Message send from variable node i to check node j.

Step 2 : Compute $L(r_{ji})$ transmitted from the check node j to variable node i; for all i; $1 \leq i \leq n$. Let $\phi(x) = \log\left(\frac{e^x + 1}{e^x - 1}\right)$.

$$L(r_{ji}) = \prod_{i \in V_{j\setminus i}} \alpha_{ij} \phi\left(\sum_{i \in V_{j\setminus i}} \phi(\beta_{ij})\right) \qquad (2.44)$$

where

$$\alpha_{ij} = \operatorname{sgn}\{L(q_{ij})\}, \text{ and } \beta_{ij} = |L(q_{ij})| \qquad (2.45)$$

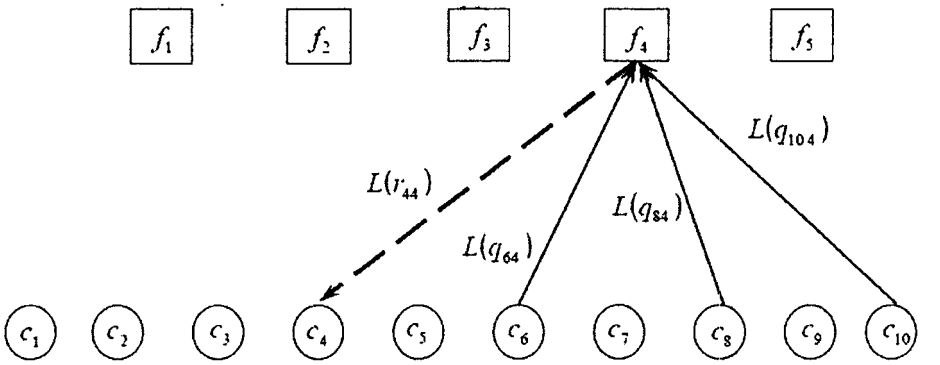


Figure 2.9 Message send from check node *j* to variable node *i*.

Step 3 : Modify $L\left(q_{ij}\right)$ and used it as the data transmitted from the variable node *i* to check node *j*; for all *i*; $1 \leq i \leq n$.

$$L(q_{ij}) = L(c_i) + \sum_{j \in C_{i \setminus j}} L\left(r_{ji}\right) \tag{2.46}$$

$C_{i \setminus j}$ represent the sum of all $L\left(r_{ji}\right)$ that connect with variable node *i* except $L\left(r_{ji}\right)$ that use the same edge with $L(q_{ij})$. Figure 2.10 show the $\text{Modify}\, L\left(q_{ij}\right)$.

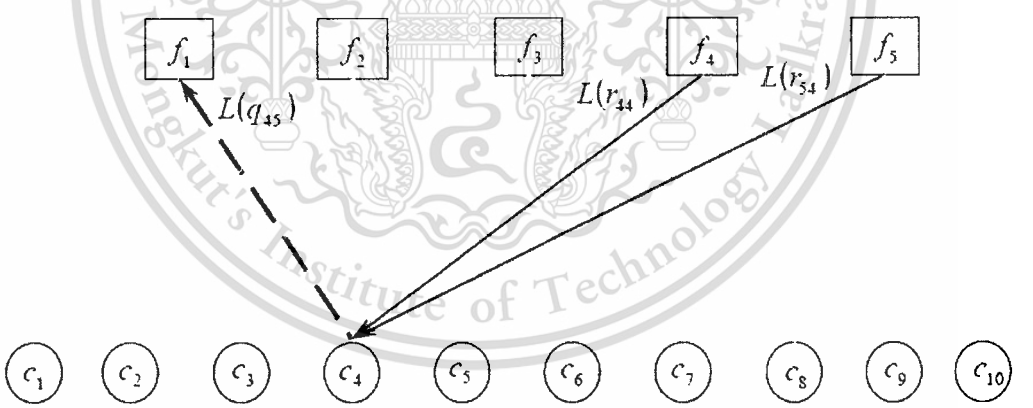


Figure 2.10 Modify message for next iteration.

Step 4 : Compute the soft output.

$$L(Q_i) = L(c_i) + \sum_{j \in C_i} L\left(r_{ji}\right) \tag{2.47}$$

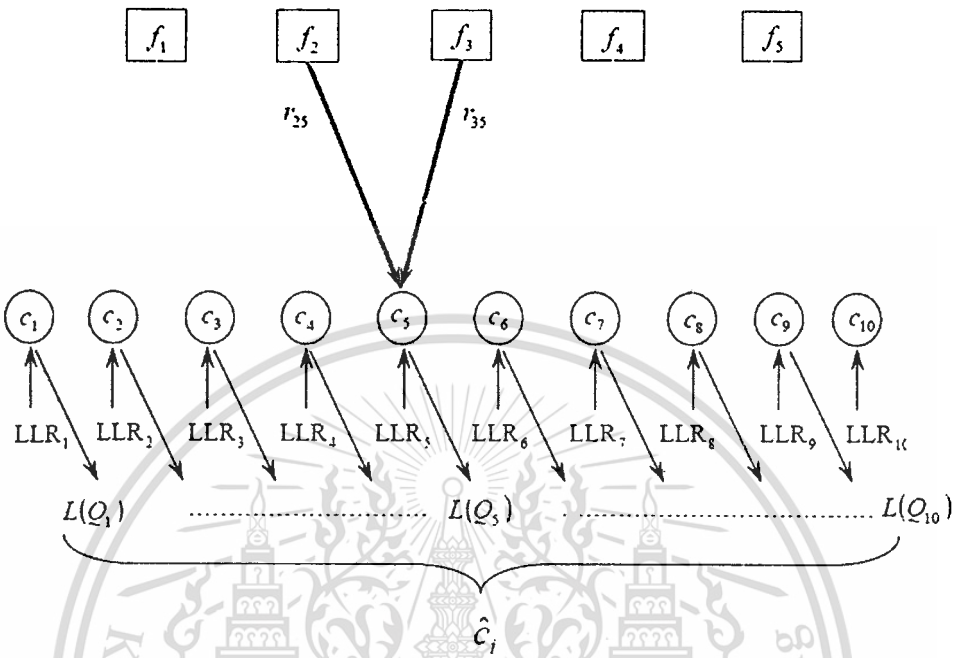


Figure 2.11 Compute soft output.

Step 5 : The soft output obtained in step 4 is then used in the hard decision as,

$$\hat{c}_i = 1 \text{ if } L(Q_i) < 0, \text{ otherwise } \hat{c}_i = 0 \tag{2.48}$$

If $\hat{c}H^T = 0$ or if maximum numbers of iteration is reached then stop, else continue iterations from step 1. We can skip step 3 to step 4 if no iteration in our decoding. The example 2.3 will show the decoding process of codeword from example1 with Log-SPA :

Example 2.3 : The message m = [1 0 1 1] was encoded to c = [1 0 1 1 1 0 0] by parity check matrix in example 3.1. After pass the channel with noise, we receive massage before decode as y_i = [2.1 -2.1 2.1 -0.1 2.1 -2.1 -2.1]. This example will use Log-SPA for decoding process.

From example 2.1 the parity check matrix H and codeword are

$$H = \begin{array}{c} \begin{matrix} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{bmatrix} \end{array} \begin{matrix} f_1 \\ f_2 \\ f_3 \end{matrix}, \qquad C = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \end{bmatrix}$$

It can write the Tanner graph as Figure 2.12.

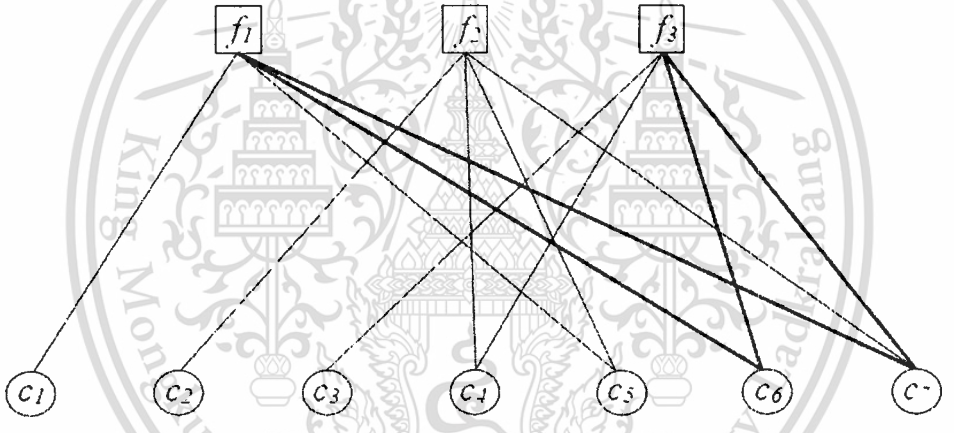


Figure 2.12 Tanner graph for example 2.1.

Step 1 : Compute the initial value of $L(q_{ij})$ transmitted from the variable node i to check node j; for all i ;$1 \le i \le n$

$$\sigma^2 = \frac{\sum (x_i - \overline{x})^2}{N} = \frac{\sum x_i^2}{N} - (\overline{x})^2$$

$$\sum x = 2.1 - 2.1 + 2.1 - 0.1 + 2.1 - 2.1 - 2.1 = -0.1$$

$$\overline{x} = \frac{-0.1}{7} = -0.0143, \overline{x}^2 = 0.0002$$

$$\sum x_i^2 = 4.41 + 4.41 + 4.41 + 0.01 + 4.41 + 4.41 + 4.41 = 26.47$$

$$\frac{\sum x_i^2}{N} = \frac{26.47}{7} = 3.78$$

$$\sigma^2 = 3.78 - 0.0002 = 3.78$$

$$L(c_i) = \frac{2y_i}{\sigma^2} = \begin{bmatrix} 1.11 & -1.11 & 1.11 & -0.05 & 1.11 & -1.11 & -1.11 \end{bmatrix}$$

Step 2 : Compute $L(r_{ji})$ transmitted from the check node j to variable node i; for all i; $1 \le i \le n$.

$$L(r_{ji}) = \prod_{i \in V_{j\setminus}} \alpha_{ij} \phi\left(\sum_{i \in V_{j\setminus}} \phi(\beta_{ij}) \right)$$

$$L(r_{11}) = [\text{sgn}]\phi\left[\phi|L(q_{51})| + \phi|L(q_{61})| + \phi|L(q_{71})| \right]$$

$$= [+--]\phi[\phi(1.11) + \phi(1.11) + \phi(1.11)] = [+]\phi[0.297 + 0.297 + 0.297] = 0.379$$

$$L(r_{15}) = [\text{sgn}]\phi\left[\phi|L(q_{11})| + \phi|L(q_{61})| + \phi|L(q_{71})| \right] = [+]\phi[0.297 + 0.297 + 0.297] = 0.379$$

$$L(r_{16}) = [\text{sgn}]\phi\left[\phi|L(q_{11})| + \phi|L(q_{51})| + \phi|L(q_{71})| \right] = [-]\phi[0.297 + 0.297 + 0.297] = -0.379$$

$$L(r_{17}) = [\text{sgn}]\phi\left[\phi|L(q_{11})| + \phi|L(q_{51})| + \phi|L(q_{61})| \right] = [-]\phi[0.297 + 0.297 + 0.297] = -0.379$$

$$L(r_{22}) = [\text{sgn}]\phi\left[\phi|L(q_{42})| + \phi|L(q_{52})| + \phi|L(q_{72})| \right] = [+]\phi[1.602 + 0.297 + 0.297] = 0.097$$

$$L(r_{24}) = [\text{sgn}]\phi\left[\phi\left|L(q_{22})\right| + \phi\left|L(q_{52})\right| + \phi\left|L(q_{72})\right|\right] = [+]\phi[0.297 + 0.297 + 0.297] = 0.379$$

$$L(r_{25}) = [\text{sgn}]\phi\left[\phi\left|L(q_{22})\right| + \phi\left|L(q_{42})\right| + \phi\left|L(q_{72})\right|\right] = [-]\phi[0.297 + 1.602 + 0.297] = -0.097$$

$$L(r_{27}) = [\text{sgn}]\phi\left[\phi\left|L(q_{22})\right| + \phi\left|L(q_{42})\right| + \phi\left|L(q_{52})\right|\right] = [+]\phi[0.297 + 0.297 + 1.602] = 0.097$$

$$L(r_{33}) = [\text{sgn}]\phi\left[\phi\left|L(q_{43})\right| + \phi\left|L(q_{63})\right| + \phi\left|L(q_{73})\right|\right] = [-]\phi[1.602 + 0.297 + 0.297] = -0.097$$

$$L(r_{34}) = [\text{sgn}]\phi\left[\phi\left|L(q_{33})\right| + \phi\left|L(q_{63})\right| + \phi\left|L(q_{73})\right|\right] = [+]\phi[0.297 + 0.297 + 0.297] = 0.379$$

$$L(r_{36}) = [\text{sgn}]\phi\left[\phi\left|L(q_{33})\right| + \phi\left|L(q_{43})\right| + \phi\left|L(q_{73})\right|\right] = [+]\phi[0.297 + 1.602 + 0.297] = 0.097$$

$$L(r_{37}) = [\text{sgn}]\phi\left[\phi\left|L(q_{33})\right| + \phi\left|L(q_{43})\right| + \phi\left|L(q_{63})\right|\right] = [+]\phi[0.297 + 0.297 + 1.602] = 0.097$$

$$L(r_{ji}) = \begin{bmatrix} \overset{11}{0.379} & \overset{15}{0.379} & \overset{16}{-0.379} & \overset{17}{-0.379} & \overset{22}{0.097} & \overset{24}{0.379} & \overset{25}{-0.097} & \overset{27}{0.097} & \overset{33}{-0.097} & \overset{34}{0.379} & \overset{36}{0.097} & \overset{37}{0.097} \end{bmatrix}$$

Step 4 : Compute the soft output.

$$L(Q_i) = L(c_i) + \sum_{j \in C_i} L\left(r_{ji}\right)$$

$$L(Q_1) = \overset{c_1}{1.11} + \overset{r_{11}}{0.379} = 1.489$$

$$L(Q_2) = \overset{c_2}{(-1.11)} + \overset{r_{22}}{0.097} = -1.013$$

$$L(Q_3) = \overset{c_3}{1.11} + \overset{r_{33}}{(-0.097)} = 1.013$$

$$L(Q_4) = \overset{c_4}{(-0.05)} + \overset{r_{24}}{0.379} + \overset{r_{34}}{0.379} = 0.708$$

$$L(Q_5) = \overset{c_5}{1.11} + \overset{r_{15}}{0.379} + \overset{r_{25}}{(-0.097)} = 1.392$$

$$L(Q_6) = \overset{c_6}{(-1.11)} + \overset{r_{16}}{(-0.379)} + \overset{r_{36}}{0.097} = -1.4$$

$$L(Q_7) = \overset{c_7}{(-1.11)} + \overset{r_{17}}{(-0.379)} + \overset{r_{27}}{0.097} + \overset{r_{37}}{0.097} = -1.303$$

$$L(Q_i) = [1.489 \ -1.013 \ 1.013 \ 0.708 \ 1.392 \ -1.4 \ -1.303]$$

Step 5 : The soft output obtained in step 4 is then used in the hard decision by

$$\hat{c}_i = 1 \text{ if } L(Q_i) < 0, \text{ otherwise } \hat{c}_i = 0$$

So, $C = [1\ 0\ 1\ 1\ 1\ 0\ 0]$; $m = [1\ 0\ 1\ 1]$; $p = [1\ 0\ 0]$

The Log-SPA algorithm is very interesting on decoding process because this algorithm has lower complexity and more numerically stable than the SPA algorithm. The SPA requires message multiplications, whereas the Log-SPA implementation uses message additions. This function is very necessary when implement to hardware because the multiplications will take up many clock cycles compared to additions.

2.8 Array LDPC Codes

Array LDPC codes were invented by Fan in 2000. Array LDPC codes are two dimensional codes that have been proposed for detecting and correcting burst errors. This type of codes reduces the complexity to construct the parity check matrix. They provide the good bit error rate performance (BER) same as the random LDPC codes. When array codes are viewed as binary codes, their parity check matrices exhibit sparseness, which can be exploited for decoding them as LDPC codes using the sum product algorithm. Therefore, array codes provide the framework for defining a family of LDPC codes that lend themselves to deterministic constructions.

The Array LDPC codes parity check matrix is specified by three parameters: a prime number p and two integers k and j such that $k, j \le p$. It has dimensions jp x kp and is given as

$$H = \begin{bmatrix} I & I & I & \cdots & I \\ I & \alpha & \alpha^2 & \cdots & \alpha^{k-1} \\ I & \alpha^2 & \alpha^4 & \cdots & \alpha^{2(k-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ I & \alpha^{j-1} & \alpha^{2(j-1)} & \cdots & \alpha^{(j-1)(k-1)} \end{bmatrix} \qquad (2.49)$$

where I is the $p \times p$ identity matrix and α is a $p \times p$ permutation matrix representing a single left or right cyclic shift. This yields the code rate is

$$R = 1 - \left(\frac{(p.j) - (j+1)}{p^2} \right).$$

The example of permutation matrix for $p = 5$ show below

$$I = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{(5\times5)} \quad \alpha = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}_{(5\times5)} \quad \alpha^2 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}_{(5\times5)}$$

$$\alpha^3 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}_{(5\times5)} \quad \alpha^4 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}_{(5\times5)} \quad \alpha^5 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{(5\times5)}$$

The parameters j and k provide the column and row weight of H. The code length is kp and parity bit is jp. By the construction, the matrix H is 4 cycle free because no two rows have overlapping 1's in more than one position.

In Figure 2.13, Fan's array LDPC code was compared with and random LDPC code with similar parameters. This code had parameters $p = 59$, $n = 30$ and $r = 3$. So, there are $N = 1740$ columns and $M = 174$ rows in the parity check matrix. This parity check matrix had 1's 3 bit per columns, where the same number of 1's in each column of randomly structure. From the graph in Figure 2.13, it can be seen that Fan's array LDPC code performs slightly better performance than randomly structure.

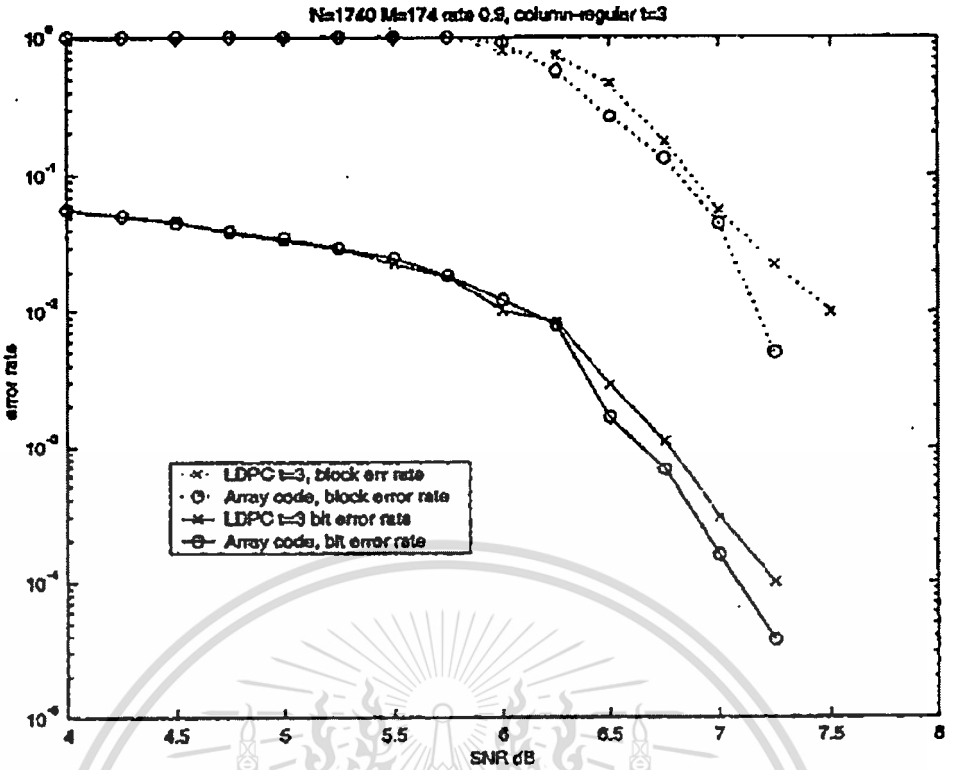


Figure 2.13 Comparison of an array code with regular LDPC[8].

2.9 Modified Array Codes: MAC

In 2001, Eleftheriou and Olcer study the family of LDPC codes that used for ADSL transmission. They found that the simulation results for binary transmission of Fan's Array LDPC code over an AWGN channel show excellent performance compared to the comparable length codes by MacKay obtained via a random construction. Iterative decoding with the sum product algorithm was employed. The results are shown in Figure 2.14 in terms of bit error rate (BER), block error rate (BLER) and corresponding channel capacities (CapB and C_Bck, respectively). The figures also indicate the BER for uncoded binary transmission.

In 2002, Eleftheriou and Olcer proposed the new method to achieve an efficient encoding in the parity check matrix of Array LDPC code. They proposed the modified array codes by applying cyclic shift to the Fan's array. They obtain the parity check matrix in the desired form of the lower triangular elements of the *jp x jp* leftmost subblock of *H* are set to zero as shown in equation (2.50). A parity check matrix *H* of a modified array code can be described in terms of the parameter (*p, j, k*). This structure yields the code rate of $R = 1 - (j / k)$.

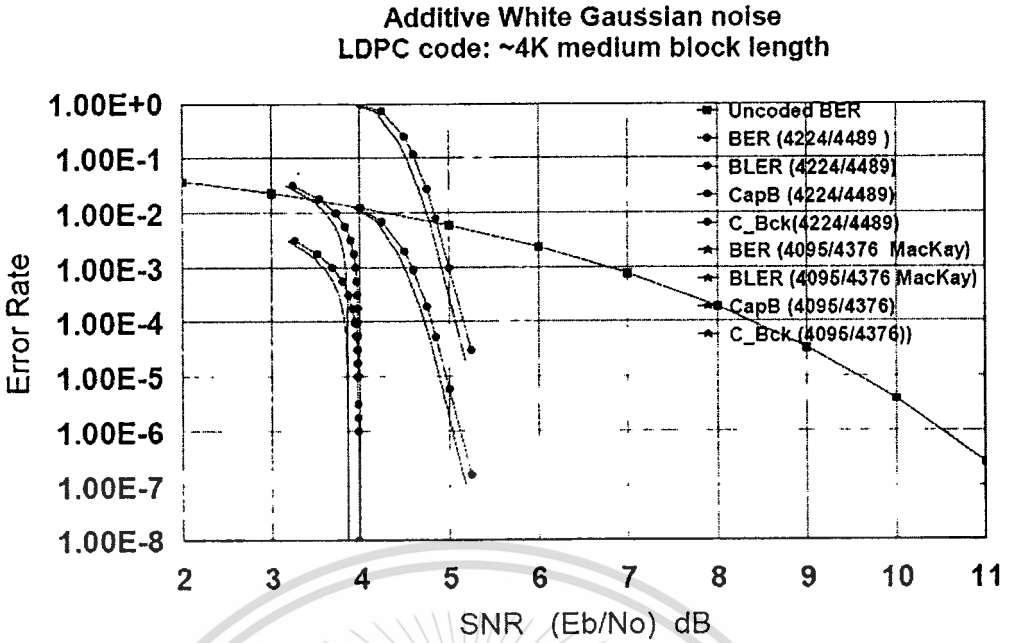


Figure 2.14 Performance of LDPC codes over an AWGN channel with Array LDPC code[9].

$$H=\begin{bmatrix} I & I & \cdots & I & I & \cdots & \cdots & I \\ 0 & I & \alpha & \cdots & \alpha^{(j-2)} & \alpha^{(j-1)} & \cdots & \alpha^{(k-2)} \\ 0 & 0 & I & \cdots & \alpha^{2(j-3)} & \alpha^{2(j-2)} & \cdots & \alpha^{2(k-3)} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \cdots & I & \alpha^{(j-1)} & \cdots & \alpha^{(j-1)(k-j)} \end{bmatrix} \quad (2.50)$$

where p is prime and $j \le k \le p$ and 0 is the $p \times p$ null matrix. I is the $p \times p$ identity matrix. The matrix α is a $p \times p$ circulant permutation matrix. This code has the code length kp, parity bit jp and it need input bit $(k\text{-}j)p$. Efficient encoding is achieved from H without the need to compute the generator matrix of the code. As the upper triangular form of H , there are no cycle 4 in the corresponding Tanner graph. This code can use the same algorithm as Fan's array on decoding process. MAC offers superior performance to Fan's array as it can reduce number of 1's in the lower triangle. This effort also led to a simpler encoder.

For MAC, it can use the property of simple encoding to encode the input bit by using equation (2.23). To generate equation (2.51), we will transpose equation (2.23) by

$$H_{(mxn)}c^{T}_{(n-1)} = 0^{T}_{(mx1)} \quad (2.51)$$

Then, c in equation. (2.51) can be written into Systematic form. We will get the formula to encoding the MAC array LDPC codes as

$$H_{(mxn)}\left[\frac{p}{m}\right]^{T} = 0^{T}_{(mx1)} \tag{2.52}$$

The result from equation (2.52) has the complexity of encoding process less than the process that we discuss in Section 2.6. Next, we will show the example to encoding the LDPC codes with parameter p = 5, j = 4 and k = 5. After we put the value of parameter p, j and k to equation (2.52), we will get the formula as

$$H_{(20x25)}\left[\frac{p_{(20x1)}}{m_{(5x1)}}\right]^{T} = 0^{T}_{(20x1)} \tag{2.53}$$

If we multiply the parity check matrix with the code word, we will get totally 20 formulas as show in equation (2.54). After that, we can find the value of the parity bit by putting the input value to the formula in equation (2.54). The formula of after multiplication between parity check matrix and codeword are shown as below.

$$\begin{aligned}
&p_{20} + m_3 = 0 \\
&p_{19} + m_2 = 0 \\
&p_{18} + m_1 = 0 \\
&\vdots \\
&\vdots \\
&p_2 + p_7 + p_{12} + p_{17} + m_2 = 0 \\
&p_1 + p_6 + p_{11} + p_{16} + m_1 = 0
\end{aligned} \tag{2.54}$$

From equation (2.54), it shows that the complexity of the formula is the linearity because MAC realigns the parity check matrix to the lower triangular form.

In the simulation result from Eleftheriou and Olcer work, the codes have lengths N = 4489, and assume j = 4, respectively. The bit error rate (BER) and block error rate (BLER) performance for binary transmission over the AWGN channel can shown as Figure 2.15. Their performance achieved is as good as or better than the performance of the randomly constructed LDPC codes of comparable lengths and

rates and it has the same performance as array LDPC codes that shown in Figure 2.14.

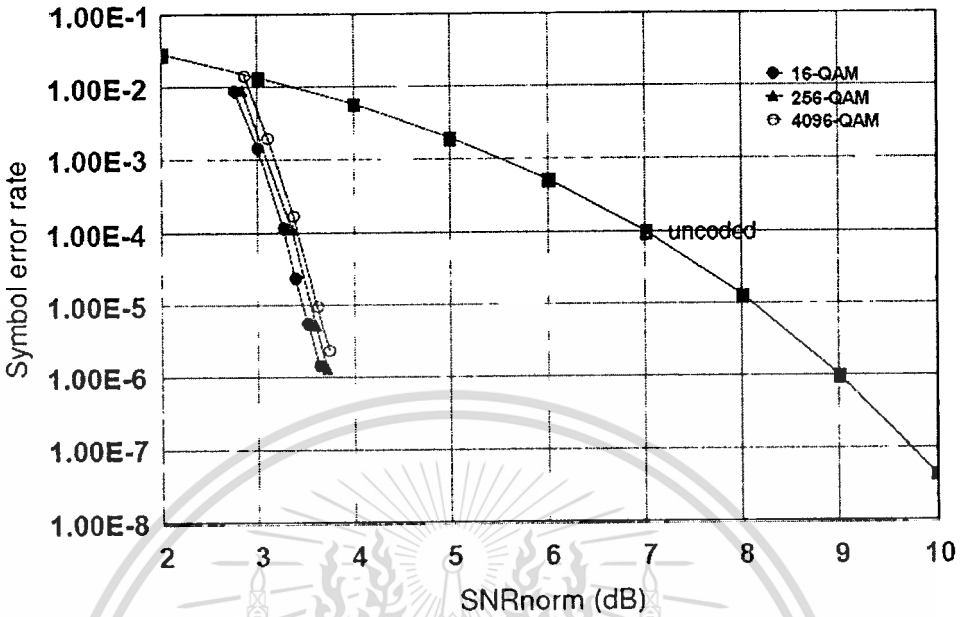


Figure 2.15 Performance of LDPC code with 4 Kbit over the AWGN channel.

2.10 Interleaved modified array LDPC codes: IMAC

Singhaudom has proposed the interleaved modified array LDPC codes or IMAC by introducing the quasi cyclic matrix into the cyclic shift of MAC codes. The parity check matrix for this designing show in equation (2.55). A parity check matrix *H* of an Interleaved modified array code still describe in terms of the parameter (*p*, *j*, *k*). This structure yields are also having the code rate of $R = 1 - (j / k)$ same as MAC.

$$H=\begin{bmatrix} I & I & I\omega & I\omega^2 & I\omega^3 & \cdots & I\omega^j \\ 0 & I & \alpha\omega & \alpha^2\omega^2 & \alpha^3\omega^3 & \cdots & \alpha^{(k-2)}\omega^j \\ 0 & 0 & I & \alpha^2\omega^2 & \alpha^4\omega^3 & \cdots & \alpha^{2(k-3)}\omega^j \\ \vdots & \vdots & \vdots & I & \alpha^3\omega^3 & \cdots & \vdots \\ 0 & 0 & \cdots & 0 & I & \cdots & \alpha^{(j-1)(k-j)}\omega^j \end{bmatrix} \quad (2.55)$$

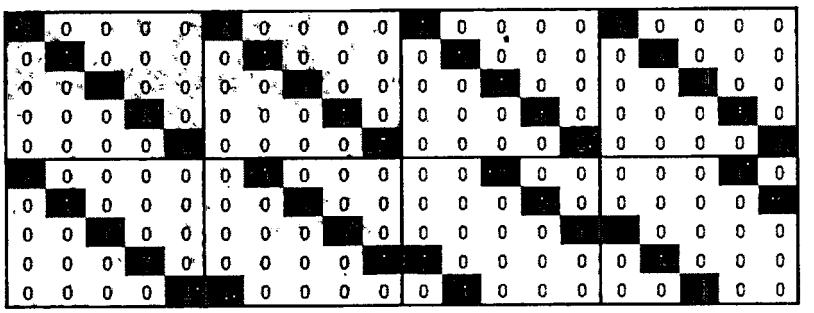
From equation (2.55), the structure of the parity check matrix H still design in the form of lower triangular as MAC array LDPC codes. Matrix 0, α and *I* have the same property as MAC array LDPC codes. This code has the code length *kp*, parity bit *jp* and it need input bit (*k-j*). ω is the quasi cyclic matrix that constructs from identity matrix by cyclic shift matrix *I* as below.

$$I = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{(5\times5)} \quad \omega = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{(5\times5)} \quad \omega^2 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{(5\times5)}$$

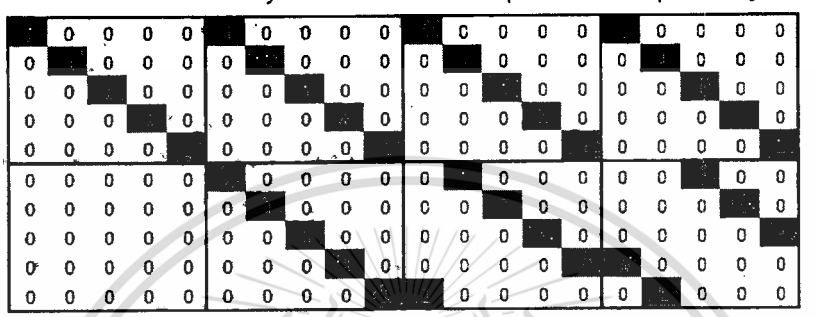
$$\omega^3 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{(5\times5)} \quad \omega^4 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{(5\times5)} \quad \omega^5 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{(5\times5)}$$

Example 2.4 The parity check matrix *H* for Array LDPC code, Modified Array LDPC code and Interleaved Modified Array LDPC code with parameter p = 5 , j = 2 and k = 4 can show as

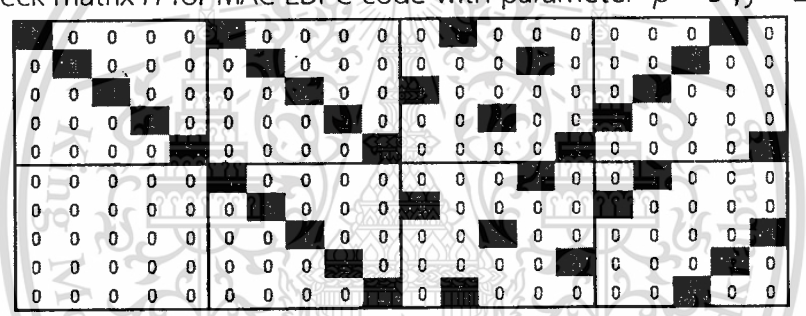
$$\begin{bmatrix} I & I & I & I \\ I & \alpha & \alpha^2 & \alpha^3 \end{bmatrix}_{(2\times4)} \rightarrow \begin{bmatrix} I & I & I & I \\ 0 & I & \alpha & \alpha^2 \end{bmatrix}_{(2\times4)} \rightarrow \begin{bmatrix} I & I & I\omega & I\omega^2 \\ 0 & I & \alpha\omega & \alpha^2\omega^2 \end{bmatrix}_{(2\times4)}$$



(a) parity check matrix H for Array LDPC code with parameter $p = 5$, $j = 2$ and $k = 4$



(b) parity check matrix H for MAC LDPC code with parameter $p = 5$, $j = 2$ and $k = 4$



(c)parity check matrix H for IMAC LDPC code with parameter $p = 5$, $j = 2$ and $k = 4$

Figure 2.16 Structure of parity check matrix H for Array LDPC.

If we consider the structure of parity check matrix H from (a) and (b), we found row 1 until row 10 have the same position of 1's. For the row 6-10, the position of 1's in (b) were shifted right p bits from (a) because of Modified Array LDPC code need the property of simple encoding in its structure. From (b) and (c), we can see column 1 until column p has only 1's in each column. We can get p formulas from the codeword by using $CH^T = 0$. But, these formulas are easier to find the unknown values than (a) because the unknown values are less than (a). For the structure of (c) when it was interleaved (b) by Quasicyclic matrix ω, we found column 1 - 10 of (b) and (c) have the same position of 1's. Because of, these columns didn't multiply by Quasicyclic matrix ω. So, we will not see the difference in columns 1-10 in Modified Array LDPC code and Interleaved Modified Array LDPC code. The columns 11-20 in (c) have different position of 1's in (b) because they

were multiplied by Quasicyclic matrix ω. From the structure both (b) and (c), it has the same number of 1's in each row and each column.

From Singudom's work, he designed the code lengths around 4 Kbits that use in a magnetic recording system. He compared his simulation performance to MAC LDPC codes with 3 code rate by changing parameter *p*. The comparison of bit error rate (BER) performance with MAC LDPC codes can show as Figure 2.17. His parity check matrix achieved better performance than MAC LDPC codes for all code rate because his design has the random structure more than MAC LDPC code.

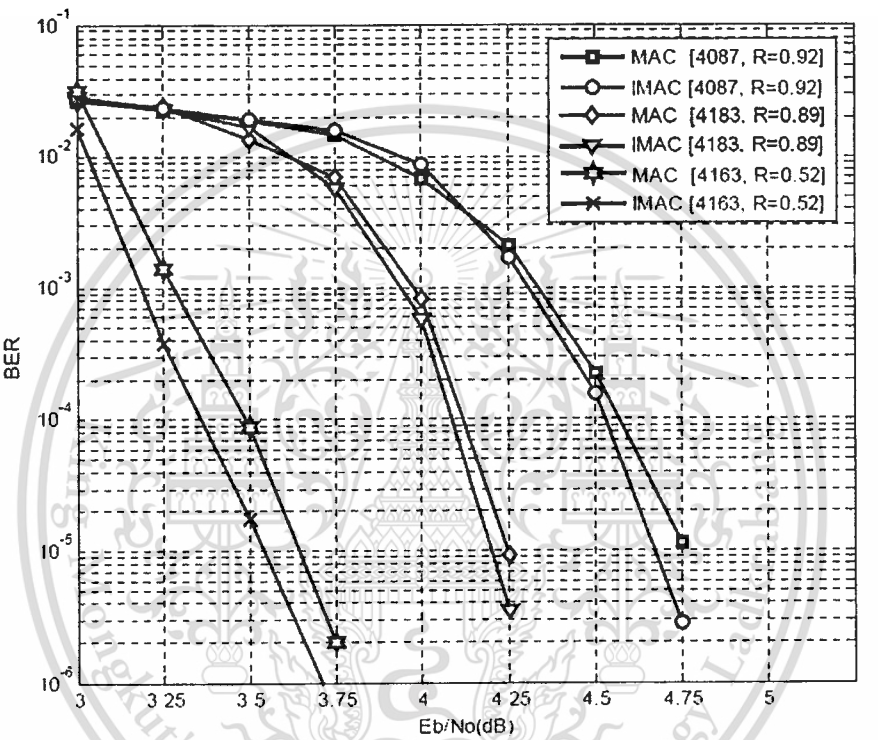


Figure 2.17 Performance of MAC LDPC code and IMAC LDPC codes at 4K block length.

In Chapter 2, we explain the linear block code and the introduction of LDPC code. The basic concept of generator matrix, the parity check matrix and syndrome are discussed in this chapter. This is the important key parameter for the linear block code design. The introduction of LDPC codes such as code structure, Tanner graphs, Cycle length of LDPC codes, encoding and decoding process are also explained. Then, we discuss on the types of array LDPC codes. To generate the parity check matrix, the random structure LDPC code has more complexity than the array LDPC code. Array LDPC codes were designed to reduce the complexity on the construction of the parity check matrix. But it has the difficult to encode the input bit. So, MAC array LDPC codes were developed to get the property of simple to encoding by realign the parity check matrix of the lower triangular form. Next,

Singudom use the idea of MAC array LDPC codes to increase the code performance by interleaved the parity check matrix of MAC array LDPC codes with quasi cyclic matrix. Chapter 3, we will discuss the problem the array LDPC code and propose the new algorithm to fix the problem.

Chapter 3
Non Prime Modified Array LDPC Code and Results

As described in Chapter 2, LDPC is a type of error correcting code that uses to correct the error in the communication system. This code is specified by sparse parity check matrix *H*. It can represent by a Tanner graph. The design of an *H* matrix is a crucial step to obtain the code performance. There are 2 types of parity check matrices: random and structural. In general, the random type offers better performance but the construction of the random matrix is more complicated when applied to the long code length. Since this thesis present about the magnetic recording system with 4 Kbits per sector. So, the parity check matrix *H* will design to support this sector size.

Modified Array LDPC code is the type of structural design. It is specified by three parameters *j*, *k*, *L* (*j*, *k* ≤ *L*). Parameters *j*, *k*, and *L* represent the row and column weights of the LDPC code, and the submatrix size. This type of LDPC code achieves the good bit error rate (BER) performances in AWGN channels with high code rate. It also gets the property of simple to encoding with $CH^T = 0$. But, Modified Array LDPC code doesn't provide the good error rate performance when the submatrix size *L* was designed with non prime number. Because of, the design with non prime number has the possibility to generate cycle 4. The code length of a Modified Array LDPC code can construct with a multiple of submatrix size. Thus, it is difficult to construct an array LDPC code with good error rate performance that supports arbitrary code lengths. Therefore, when we use a Modified Array LDPC code in a communication system, we have to add dummy bits to match the code length of the array LDPC code used. However, since the dummy bits contain no information data. So, the transmission rate is degraded.

3.1 Issue of Modified Array LDPC Code

The Modified Array LDPC code is very interesting in the communication system. Because of, it achieved the good performance in high code rate application and can encode with simple method. However, they do not support arbitrary code lengths. The code length of Modified Array LDPC code with good error rate performance is limited to a multiple of a prime number. If we construct the parity check matrix *H* with non prime submatrix size, many cycles 4 can be generated. The cycles 4 make their information loop back to themselves which degrades the performance of LDPC codes. Since, the decoding process is based on the belief propagation (BP) algorithm. Figure 3.1 and 3.2 show the examples of the pattern

diagrams for prime (submatrix size = 11) and non-prime numbers (submatrix size = 12). The vertical list on the pattern diagrams shows the row numbers and the horizontal list shows the column numbers. The entries are the permutation submatrix. The permutation submatrix is decided by its row and column numbers where it is located. These two matrixes were constructed by the Modified Array LDPC code method that described in Chapter 2. From Figure 3.2, many cycles 4 were generated when design with non prime submatrix size.

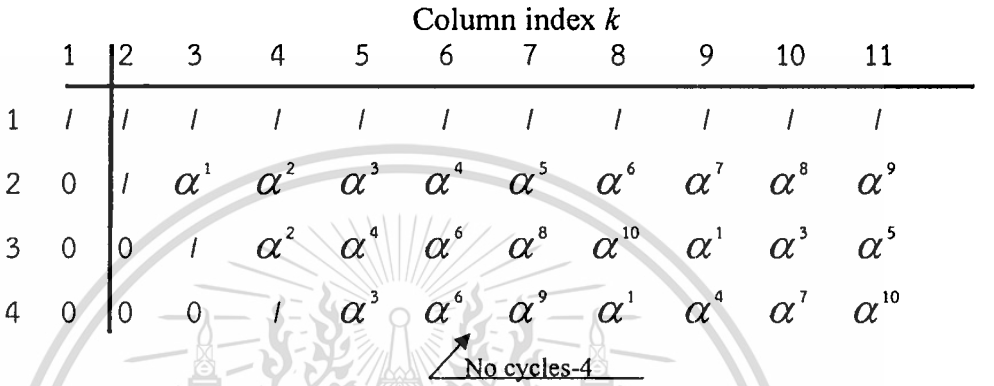


Figure 3.1 Pattern diagram of Modified Array LDPC code with prime submatrix (L = 11).

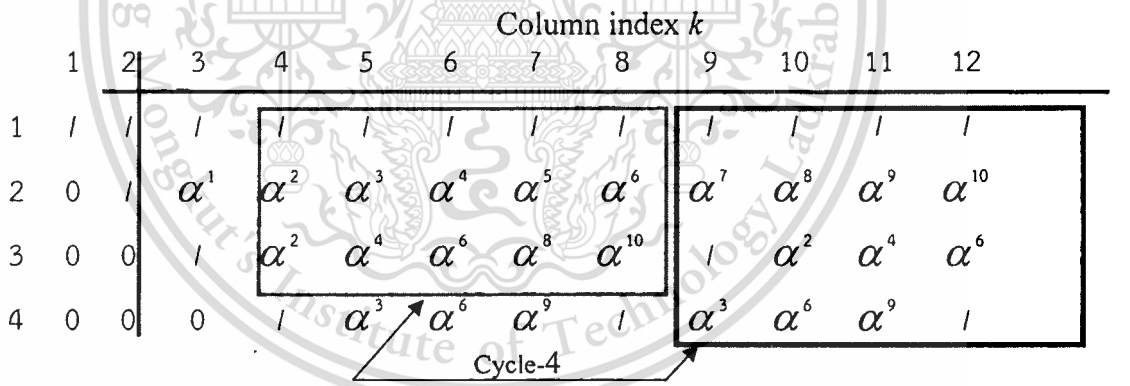


Figure 3.2 Pattern diagram of Modified Array LDPC code with prime submatrix (L = 12).

3.2 Non prime Modified Array LDPC codes

Based on the limitation of using of prime number parameters is that the designed codes can support only some particular code lengths. So, this thesis proposes the Non prime Modified Array LDPC that can use nonprime parameters to design the code with the good performance. Since the submatrix size of the Non prime Modified Array LDPC code can be nonprime. The Non prime Modified Array LDPC code will offer arbitrary code lengths. We separate the design to 3 step process to ensure that the proposed Modified Array LDPC code contains few or no

cycle 4. First, we design the Non prime array LDPC code H_N as eq (3.1). For convenience, we replace the cyclic shift of the array code with P_{SC} (j, k). In the proposed Non prime Modified Array LDPC code, the permutation submatrix is decided to eliminate all cycle 4 regardless of size. The proposed cyclic shift $P_{SC}(j,k)$ is expressed as equation (3.2)

$$H_N = \begin{bmatrix} I & I & I & \cdots & I & \cdots & I \\ I & \alpha^{P_{SC}(2,2)} & \alpha^{P_{SC}(2,3)} & \cdots & \alpha^{P_{SC}(2,j)} & \cdots & \alpha^{P_{SC}(2,k)} \\ I & \alpha^{P_{SC}(3,2)} & \alpha^{P_{SC}(3,3)} & \cdots & \alpha^{P_{SC}(3,j)} & \cdots & \alpha^{P_{SC}(3,k)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ I & \alpha^{P_{SC}(j,2)} & \alpha^{P_{SC}(j,3)} & \cdots & \alpha^{P_{SC}(j,j)} & \cdots & \alpha^{P_{SC}(j,k)} \end{bmatrix} \quad (3.1)$$

where,

$$P_{SC}(j,k) = (j-1)(k-1) + \left| \frac{(j-1)(k-1)}{L} \right| \quad (3.2)$$

In equation (3.2), the first term of the cyclic shift is according to row and column. The second term is the cyclic shift that reduces cycles 4. Figure 3.3 shows the pattern diagram of the permutation submatrix for the array LDPC code with non-prime submatrix size. The parity check matrix can generate by equation (3.1) and (3.2). Figure. 3.3 shows that all cycle 4 have been eliminated from the proposed permutation process.

To eliminate as a lot of cycle 4 as possible, we purpose the second step to permute the submatrix position according to row number. As the result of this step, we show the pattern diagram for the H_{SN} with nonprime submatrix size (L = 12) by equation (3.3).

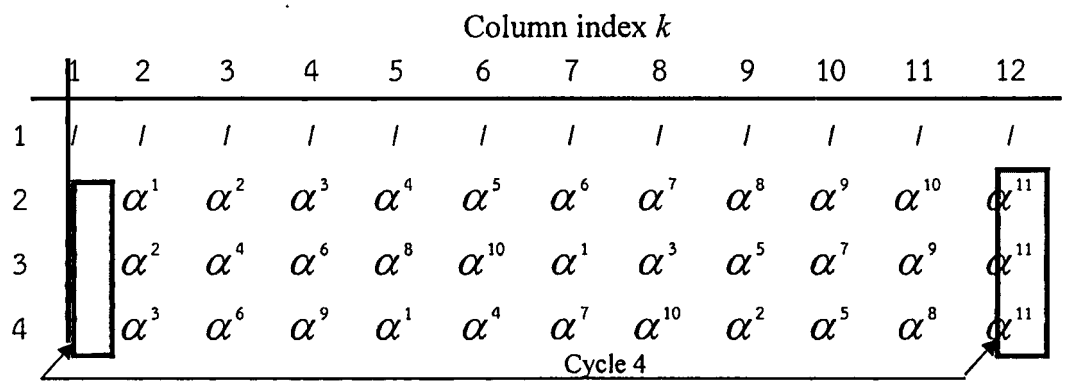


Figure 3.3 Pattern diagram of array LDPC code with non-prime submatrix size (L = 12).

$$H_N = \begin{bmatrix} / & / & / & \cdots & / & \cdots & / \\ \alpha^{P_{SC}(2,k)} & / & \alpha^{P_{SC}(2,2)} & \cdots & \alpha^{P_{SC}(2,j)} & \cdots & \alpha^{P_{SC}(2,k-1)} \\ \alpha^{P_{SC}(3,k-1)} & \alpha^{P_{SC}(3,k)} & / & \cdots & \alpha^{P_{SC}(3,j)} & \cdots & \alpha^{P_{SC}(3,k-2)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \alpha^{P_{SC}(j,k-2)} & \alpha^{P_{SC}(j,k-1)} & \alpha^{P_{SC}(j,k)} & \cdots & / & \cdots & \alpha^{P_{SC}(j,k-j+1)} \end{bmatrix} \quad (3.3)$$

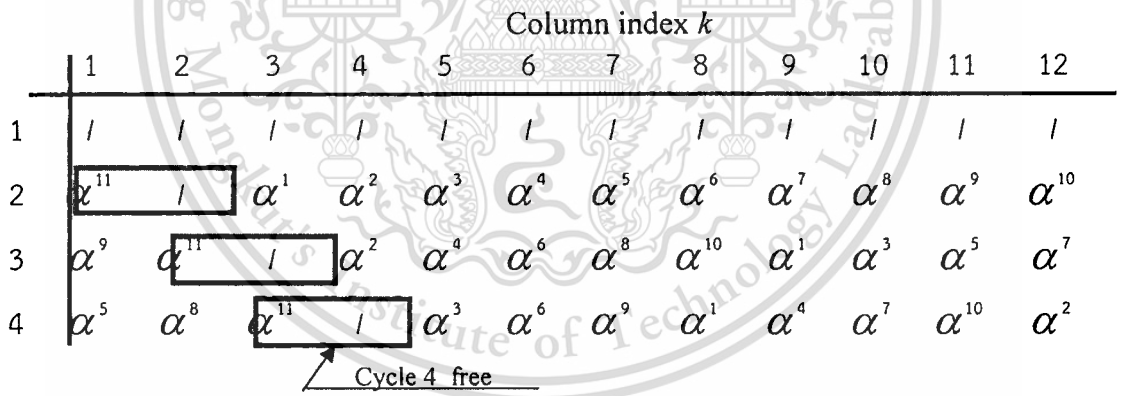


Figure 3.4 Pattern diagram of array LDPC code with non-prime submatrix after cyclic shift right.

In the second step, array LDPC code can eliminate the cycle 4 by permuting each column according to each row as per equation (3.3). But it can't get the properties of simple encoding as the MAC and IMAC as described in Chapter 2. So, we propose the third step to restructure equation (3.3) in such a way that its lower left corner holds triangular zeros as equation (3.4). We modified cyclic shift $P_{SC}(j,k)$ in equation (3.2) to $P_{NP}(j,k)$ as expressed as equation (3.5) to quite understand when replace the j, k value with design value.

$$H_N = \begin{bmatrix} I & I & I & \cdots & I & \cdots & I \\ 0 & I & \alpha^{P_{SC}(2,2)} & \cdots & \alpha^{P_{SC}(2,j)} & \cdots & \alpha^{P_{SC}(2,k-1)} \\ 0 & 0 & I & \cdots & \alpha^{P_{SC}(3,j)} & \cdots & \alpha^{P_{SC}(3,k-2)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & I & \cdots & \alpha^{P_{SC}(j,k-j+1)} \end{bmatrix} \quad (3.4)$$

where,

$$P_{NP}(j,k) = (j-1)(k-1) + \left\lfloor \frac{(j-1)(k-j)}{L} \right\rfloor \quad (3.5)$$

A parity check matrix H of a Non prime Modified Array LDPC codes can be described in terms of the parameter (L, j, k) and has the form as represented by equation (3.4). L is the submatrix size and $j \le k \le p$, 0 is the $L \times L$ null matrix and I is the $L \times L$ identity matrix. The matrix P_{NP} is a $L \times L$ cyclic shift matrix which calculates as equation (3.5). The parity check matrix of Non prime Modified Array LDPC codes can define by H_{NP}. It has codeword length $N = k \times L$, number of parity bit $M = j \times L$ and message block length $K = p \times (k - j)$ and the code rate is $R = 1 - j/k$. The Non prime Modified Array LDPC codes also led to a simple encoding as MAC because it still has lower triangle structure.

Example 3.1 Let $j = 3$, $k = 12$ and $L = 12$, we can show the H matrix for the Non prime Modified Array LDPC codes by Figure 3.5 and Modified Array LDPC codes by Figure 3.6.

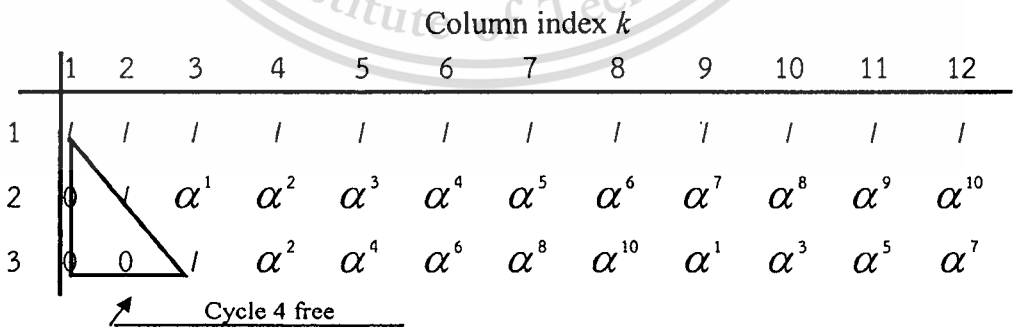


Figure 3.5 Parity check matrix for the Non prime Modified Array LDPC codes ($j = 3$, $k = 12$ and $L = 12$).

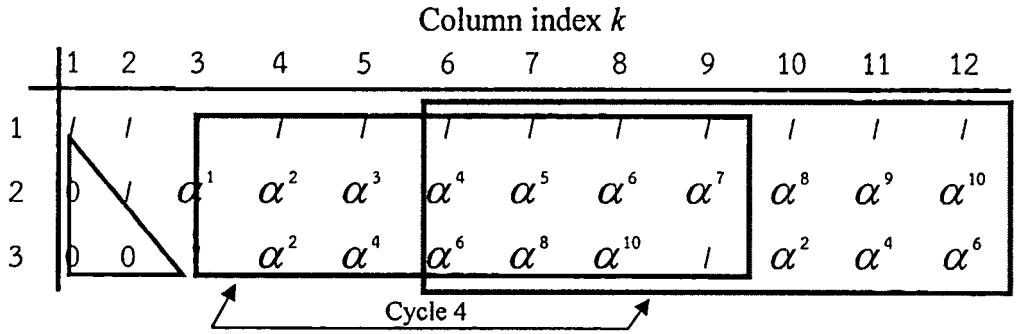


Figure 3.6 Parity check matrix for the Modified Array LDPC codes (j = 3, k = 12 and L= 12).

Figure 3.5 and 3.6 show a design of the parity check matrix of Non prime Modified Array LDPC codes and Modified Array LDPC codes while using similar parameter. It can be shown that the design by Non prime Modified Array LDPC codes method can eliminate the cycle 4 but the Modified Array LDPC codes still had the cycle 4 in their matrix. Furthermore, the design of Non prime Modified Array LDPC codes still had the triangular form. So, it can use the property of simple encoding to encode the data as Modified Array LDPC codes.

3.3 LDPC simulation system overview

The Non prime Modified Array LDPC codes and Modified Array LDPC codes (MAC) has been simulated to compare the result by using MATLAB. A simulation system is presented in Figure 3.7. First, the information vectors were generated by the MATLAB. Then, they are sent to the LDPC encoder. Next, it modulates the encoded signal by using BPSK modulation. The modulated codewords are sent across additive white Gaussian (AWGN) channel. The received bits enter the decoder to correct all the errors that have occurred through transmission and find the original data. The decoder uses Sum-Product algorithm and is quite general. It can work with any class of the parity check matrix and LDPC code. This decoder does the computations in different processing nodes in serial. Different parameters of the decoder can be changed during the process. For example, signal to noise ratio (SNR), the number of the iterations for decoding.

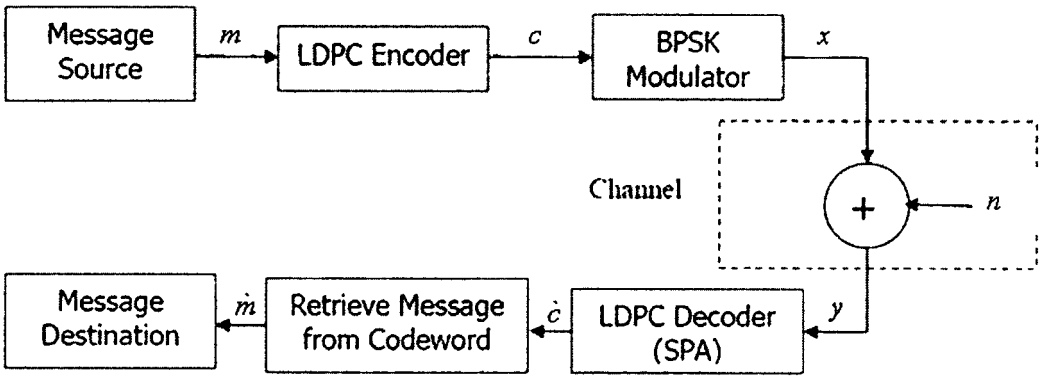


Figure 3.7 LDPC simulation system overview.

- m = *information*
- c = *Codeword*
- x = *Modulated signal*
- n = *AWGN noise*
- y = *Received signal*
- $\hat{c}$ = *Estimated codeword*
- $\hat{m}$ = *Estimated message*

Note: All above signals are vectors in terms of simulation implementation

Message Source: The message source is the user data. In terms of magnetic recording system, the message source would be the information that the user needs to store in the hard disk. The simulation utilized a random message generator.

LDPC Encoder: The LDPC encoder is adding redundant data or parity data to the user data such that it can be recovered by a receiver when a number of errors were received. In terms of simulation implementation, the encoding process is done via a parity check matrix. This process uses the design of Non prime Modified Array LDPC codes and Modified Array LDPC codes in further detail in Section 2.6 to encode the user data.

BPSK Modulator: The BPSK (Binary Phase Shift Keying) modulator maps the input binary signals, to an analog signal for recording to the media. In simulation, the BPSK signal is represented by the mapping: {0, 1}.

Channel: The channel is the medium of which information is transmitted from the transmitter to the receiver. In magnetic recording system, this is a media. The addition of noise normally occurs in the channel. In the simulations, the channel is modeled as an AWGN (Additive White Gaussian Noise) channel. The resulting noise added to the system follows the zero-mean normal distribution, with variance *NO/2*, and *NO* is the single sided noise power spectral density.

LDPC Decoder: The decoder is implemented when the user needs to get the information back. In terms of simulation implementation, decoding is a process that

loops through passing messages back and forth along the tanner graph under certain conditions are satisfied, or a maximum number of passes have occurred. This process uses the SPA to decoding the message.

Retrieve Message From Codeword: This simple process retrieves the estimated message from the estimated codeword. In the simulation this process is done via a simple function call following estimating the codeword by loop back the message after decoding 30 iterations.

Message Destination: The message destination is data that get from the user. In magnetic recording system, this would be the user watching the movie on a computer or copying file from hard disk to an external drive. In the simulations, the message destination is used to compare with the message that generate by message source to find the performance of the parity check matrix.

Eb/N_0 (the energy per bit to noise power spectral density ratio) is an important parameter in digital communication or data transmission. It is a normalized signal-to-noise ratio (SNR) measure, also known as the "SNR per bit". It is especially useful when comparing the bit error rate (BER) performance of different digital modulation schemes without taking bandwidth into account.

3.4 Simulation performance result

In this thesis, we will consider to simulate the design system into 4 topics as below

1. To compare the bit error rate (BER) performance of Modified array LDPC codes and Non prime modified array LDPC codes with similar parameter.

2. To check the bit error rate (BER) performance of Non prime modified array LDPC codes and Modified array LDPC codes when design with different iteration.

3. To find the suitable iteration on decoding process that achieves the good bit error rate (BER) performance.

4. To find the bit error rate (BER) performance of Non prime modified array LDPC codes when design with the 500, 2K and 4K bits code length.

3.4.1 BER performance of MAC and Non prime MAC with similar parameter

For Modified array codes and Non prime modified array codes, they can be described with three parameters j, k, L ($j, k \leq L$). Parameters j, k, and L represent the row and column weights of the code and L represent the submatrix size. The input message $m = L (k - j)$, codeword length $N = kL$, number of parity bit $p= jL$ and code rate is $R = 1 - (j / k)$.

Table 3.1 Parameters for simulate the performance of MAC for 4K block length with high code rate

Parameter	MAC	Non prime MAC	MAC	Non prime MAC	MAC	Non prime MAC
j	5	5	5	5	5	5
k	61	60	56	55	52	51
L	67	68	73	74	79	80
code rate (1-j/k)	0.92	0.92	0.91	0.91	0.90	0.90
input bit (L(k-j))	3,752	3,740	3,723	3,700	3,713	3,680
parity bit (jL)	335	340	365	400	395	400
code bit (kL)	4,087	4,080	4,088	4,070	4,108	4,080
iterations	30	30	30	30	30	30

To compare the performance of both methods, we will set the code rate and code length with similar parameters. For this design, we set the code length about 4000 to 4100 bits and the code rate is 0.9 that it is usually used in a magnetic recording system. The test parameters are shown in table 3.1.

From Table 3.1, we need to design the system by fixing the code rate and the codeword length. Due to the constraint of high code rate, the parameter j has to be low when comparing with k value because the relationship between j and k that shows in Table 3.1 We also see that k value was decreased when increase L value because the input bit was fixed at 4 Kbits.

Figure 3.8 shows the simulation result of the design with parameter in Table 3.1. The plots show the bit error rate performance when design MAC with prime and non prime submatrix size L for block length 4 Kbits. We simulate the code with code rate 0.92, 0.91 and 0.90 at 30 iterations.

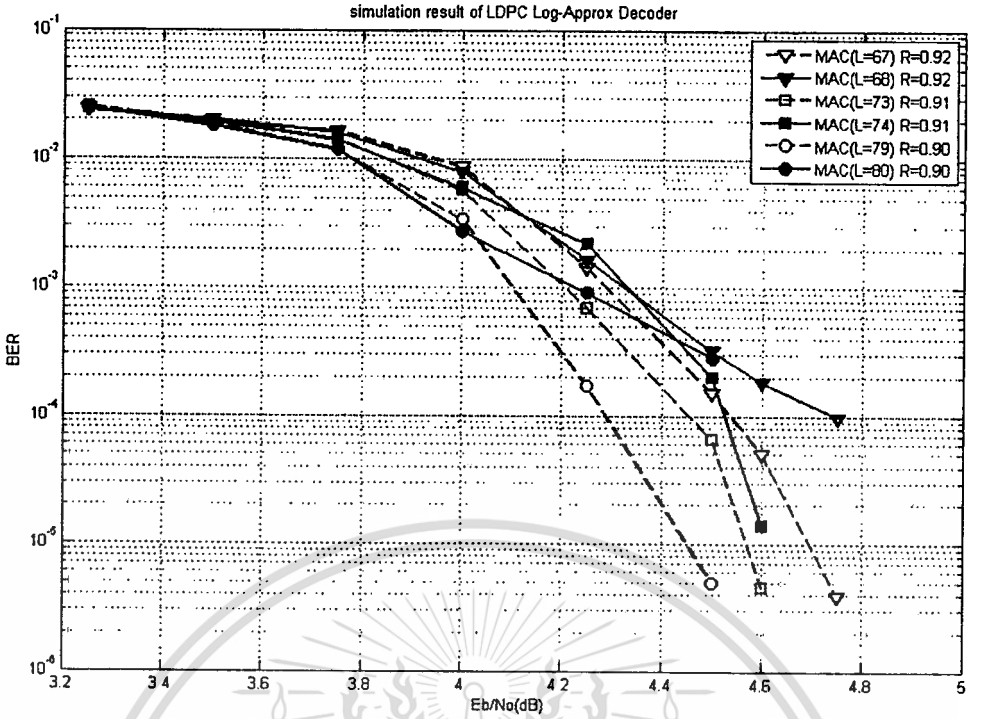


Figure 3.8 Simulation performance of MAC for 4 Kbits block length with prime and non prime submatrix.

In this design, we focus on the parameter *L*. This parameter will vary the code rate when we fix *j* and code length to 4Kbits. From the Table 3.1, it shows that the code rate will change if parameter *L* was changed. The code rate will decrease when we increase *L*. It means that if we need to design the system with high code rate, we must choose the *L* parameters with small value. The minimum distance of the code is very important to design the code. Because of, the iterative decoding performances are high when the LDPC code has the minimum distance. The minimum distance for high *L* is better than the low *L*. So, it is possible to design the code with high *L* to achieve the high performance of the code.

If we consider bit error rate (BER) performance by the code rate from Figure 3.8, it can see that the design of code rate 0.9 has better performance than code rate 0.91 and 0.92. Because of, the code rate 0.90 was designed with higher *L* than the others. It means that the code rate 0.90 has the number of redundant bits more than the rest. With the same number of sectors, code rate 0.90 can store the user data less than code rate 0.91 and 0.92. We also found that the designing code with prime submatrix size *L* has better performance than non prime submatrix size *L* for every code rate designing. The performance of MAC design with non prime submatrix size *L* very poor because it has many cycle 4 in their matrix as describe in

Section 3.1. So, we can conclude that MAC suitable for designing with prime submatrix size L only.

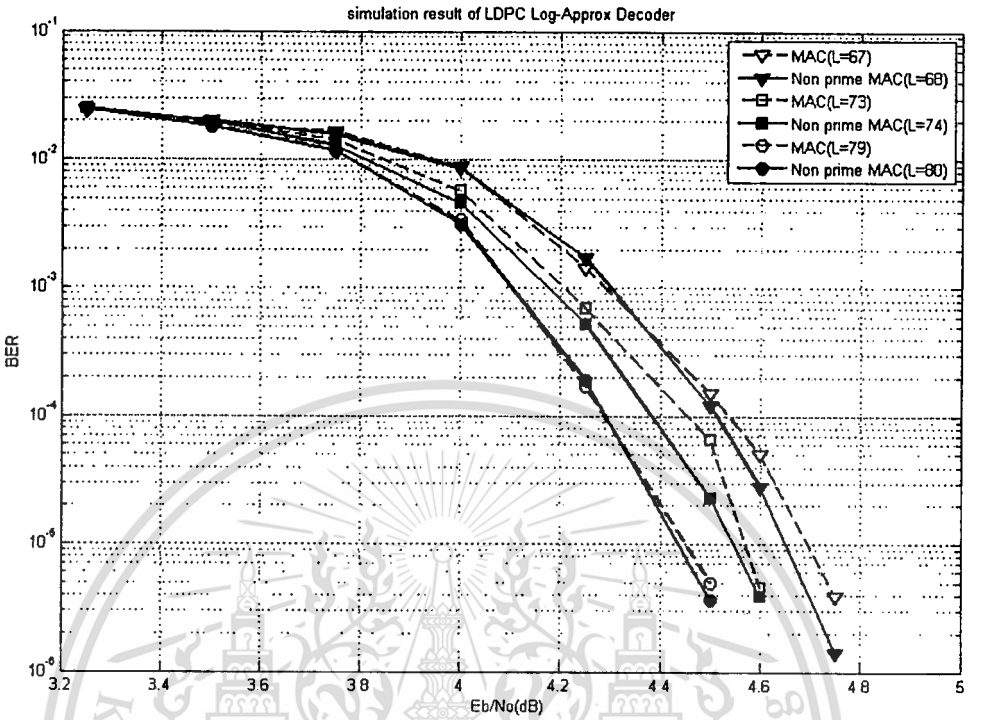


Figure 3.9 Simulation performance of MAC and non prime MAC for 4 Kbits block length.

Next, we need to compare the performance of our method with MAC. We also use the parameters in Table 3.1 to simulate the performance result. MAC with non prime submatrix size L was replaced by our design as equation (3.4) and (3.5). Figure 3.9 shows the comparable result of Non prime MAC with MAC. From these simulation results, we can see that the Non prime MAC LDPC codes with non prime submatrix size L and the MAC LDPC codes with prime submatrix size L have almost the same bit error rate performance at code rate 0.90 and 0.91. We can also see that the Non prime MAC LDPC codes have better bit error rate performance than the MAC LDPC codes with non prime submatrix size L for every code rate. Because of, the Non prime MAC LDPC codes have no cycle 4. The difference in error rate performance between the Non prime MAC LDPC codes and the MAC LDPC codes with prime submatrix size is owing to the difference in code length. Therefore, the Non prime MAC LDPC codes can realize the MAC LDPC codes that support arbitrary code lengths and has no need to use dummy bits, while achieving good error rate performance.

3.4.2 BER performance of Non prime MAC and MAC with different iteration.

This topic shows the bit error rate (BER) performance trend for both Non prime MAC LDPC codes and MAC LDPC codes. In the simulation, we change the iteration number on decoding process. For the simulation, we set parameter *j*, *k* and *L* as shown in Table 3.2. The system was designed with the code rate 0.92 and set the iteration for decoding to 5, 10, 20 and 30.

Table 3.2 Parameters for simulate the performance of MAC and Non prime MAC.

Parameter	MAC	Non prime MAC
j	5	5
k	61	60
L	67	68
code rate (1-j/k)	0.92	0.92
input bit (L(k-j))	3,752	3,740
parity bit (jL)	335	340
code bit (kL)	4,087	4,080
iterations	5/10/20/30	5/10/20/30

Figure 3.10 shows the bit error rate (BER) performance between MAC and non prime MAC. This simulation was designed with 5, 10, 20 and 30 iterations at decoding process and using the same parameter on any iteration. From the plots, it shows that the non prime MAC has the same performance as MAC for every iteration. The code performance was increased when increase the number of iterations. Because of, the decoding process for each iteration will reduce the number of errors as described in Section 2.7. For the result from Figure 3.9 and 3.10, we can use the non prime MAC instead of MAC for any code length design if we don't need to add the dummy bit to the design.

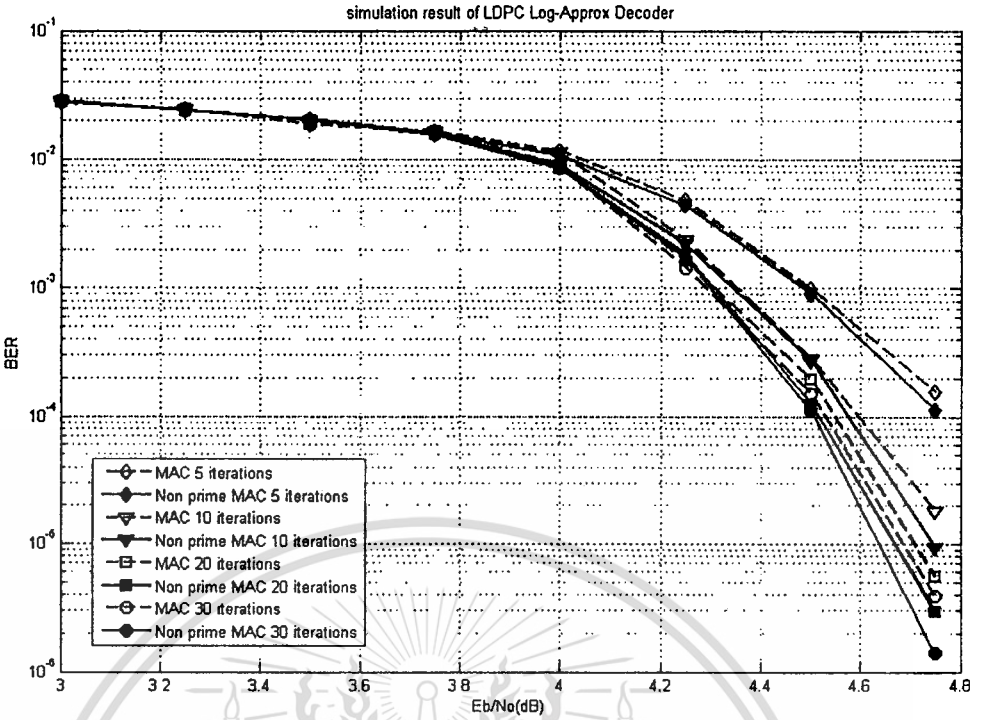


Figure 3.10 Simulation performance of MAC and non prime MAC LDPC codes with 5, 10, 20, and 30 iterations.

3.4.3 To find the suitable iteration on decoding process.

As the result in Section 3.4.1 and 3.4.2, we can use the non prime MAC LDPC codes to design the error correcting code instead of MAC LDPC codes without performance reduction. In this section, we design the non prime MAC LDPC codes by changing the iteration number of the decoding process to verify the trend of bit error rate (BER) performance. Figure 3.11 shows the performance of non prime MAC LDPC codes on 5, 10, 20, 30, 50 and 100 iterations. The parameters for the simulation were shown in Table 3.3.

Table 3.3 Parameters for simulate the performance of Non prime MAC

Parameter	Non prime MAC
j	5
k	60
L	68
code rate (1-j/k)	0.92
input bit (L(k-j))	3,740
parity bit (jL)	340
code bit (kL)	4,080
iterations	5/10/20/30/50/70/100

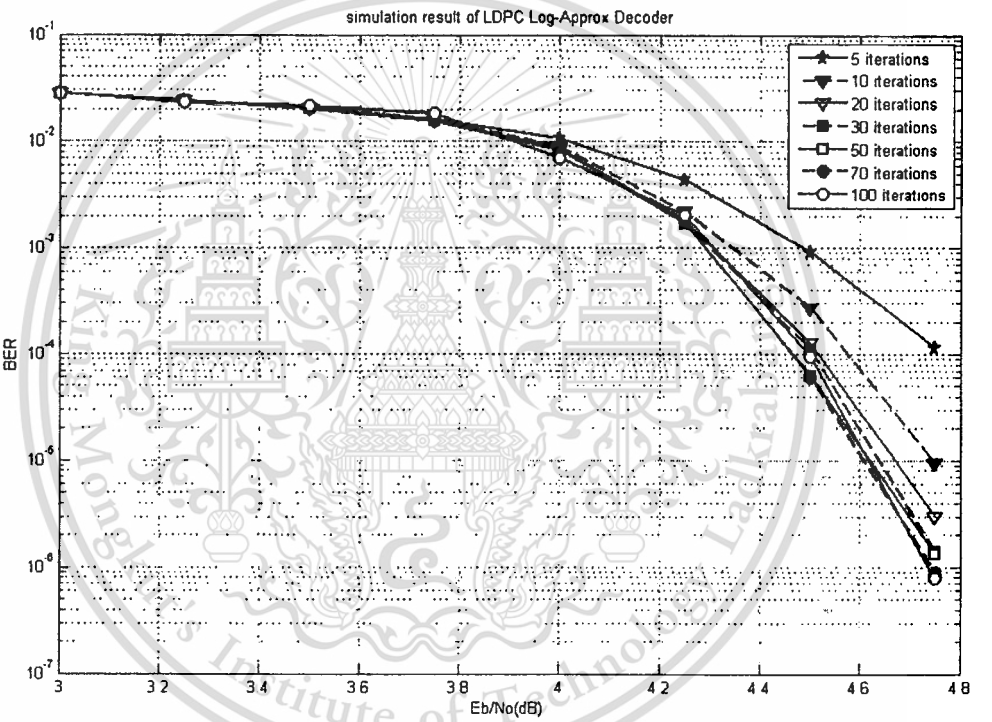


Figure 3.11 Simulation performance of non prime MAC LDPC codes on different iterations.

From the bit error rate performance of the code in Figure 3.11, we found that the number of iterations distinctly impact to the bit error rate performance. If we consider the bit error rate performance on 50 iterations, it can see that the bit error rate performance is better than the design at 5, 10, 15, 20 and 30 iterations at every range of SNR. The bit error rate performance has a little bit increasing when change the iteration from 50 to 70 and 100. Because of, the number of the error bit was reduced every decoding loop. If we use the higher number of iterations, it will impact to the hardware designing because it needs a larger space than the small

iteration. We use 30 iterations of the decoding process to simulate the performance. This iteration achieves the good performance and don't use more space in hardware designing.

3.4.4 BER performance of Non prime MAC with 500, 2K and 4K bits code length.

This section will show the bit error rate performance of the Non prime MAC LDPC codes with the different code length designing. For the simulation, we set the code length to 528, 2016 and 4080 bits with the code rate 0.8 - 0.9. The test parameters are shown in Table 3.4.

Table 3.4 Parameters for simulate the performance of Non prime MAC 528, 2016 and 4080 block length

Parameter	Non prime MAC	Non prime MAC	Non prime MAC
j	3	4	5
k	22	42	60
L	24	48	68
code rate (1-j/k)	0.87	0.90	0.92
input bit (L(k-j))	456	1,824	3,740
parity bit (jL)	72	192	340
code bit (kL)	528	2,016	4,080
iterations	30	30	30

Figure 3.12 shows the bit error rate (BER) performance of Non prime MAC LDPC codes at 528, 2016 and 4080 code length. From the plots, it shows that the bit error rate (BER) performance of non prime MAC is $3X10^{-6}$, $3X10^{-6}$ and $1.5X10^{-6}$ at 528, 2016 and 4080 code length. It can see that the non prime MAC LDPC codes can't achieve the good performance when simulate with the short block length. Because of, the decoding process for LDPC code use concept of message passes between check nodes and variable nodes. If we design the parity check matrix with short block length or small number of 1's, it has the low number of message passing between bit node and check node. These check messages are important for the code performance on decoding process. The Non prime MAC LDPC codes suitable to design with high code length more than short code length.

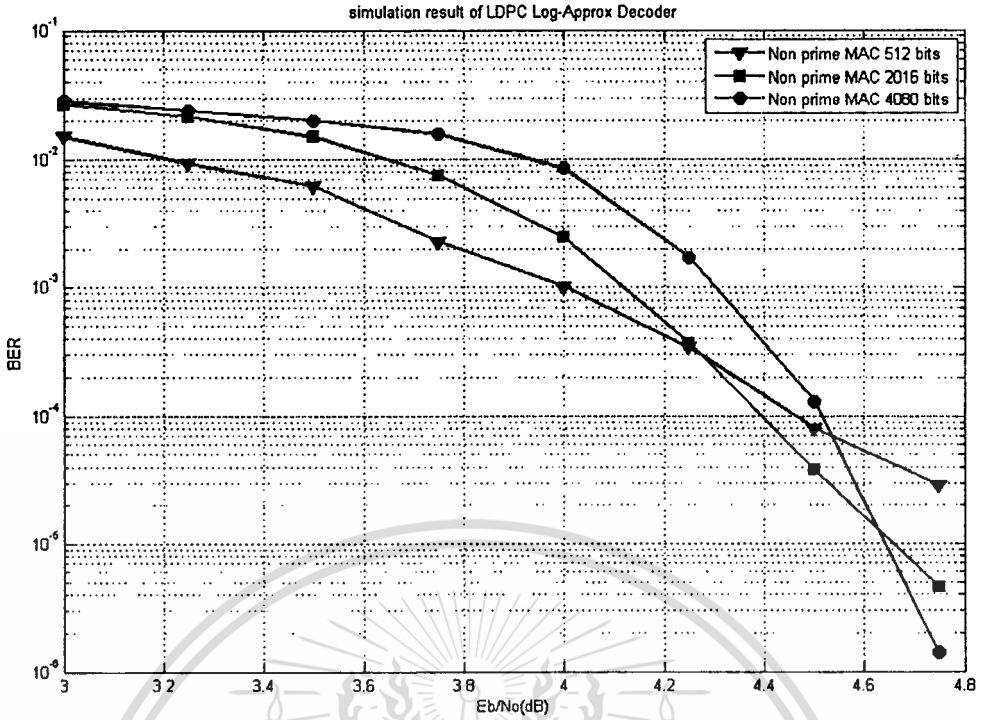


Figure 3.12 Simulation performance of non prime MAC LDPC codes on 528, 2016 and 4080 block length.

In Chapter 3, we introduced the designing method on parity check matrix of LDPC codes that fix the problem of many cycle 4 generated when design with the non prime submatrix size *L*. We show the simulation performance of the propose code at the code rate and code length about 0.9 and 4 Kbits. That is the parameter often use in magnetic recording design. The simulation result shows that the propose code with non prime submatrix size *L* have the good performance as design with MAC array LDPC codes with prime submatrix size *L*. Next Chapter, we will apply our propose code to another design.

Chapter 4
Application of the Non Prime MAC LDPC Codes

In the previous chapter, a problem of using non prime submatrix size L in MAC LDPC codes and the designing of Non prime MAC LDPC codes was described. This chapter will explain the methods to improve the bit error rate performance of Non prime MAC LDPC codes by applying our method to another LDPC code. Interleave Modified array LDPC codes and Non prime MAC LDPC codes with the Chinese Remainder Theorem (CRT) were discussed.

Section 4.1 will indicate how Interleave Modified array LDPC codes can be constructed from the Non prime MAC LDPC codes. This is exactly modified of the construction technique described in the previous section. Section 4.2 presents the method to construct the long length parity check matrix by combining 2 parity check via CRT method. Section 4.3 discuss on the Non prime MAC LDPC Codes in Magnetic Recording Channel.

4.1 Application of Non prime MAC LDPC codes with Interleave Modified Array Codes.

Interleave Modified Array LDPC Codes (IMAC), a type of Low Density Parity Check Codes (LDPC), are the high rate codes that can achieve good error rate performance in additive white Gaussian noise (AWGN) channels. They have many advantages required in magnetic recording system such as low error floor, capability of detecting and correcting burst error. However, IMAC array LDPC codes do not support arbitrary code lengths. Because of the code lengths of IMAC array LDPC codes with good error rate performance are limited by a multiple of a prime number. This section will design IMAC array LDPC codes for supporting arbitrary code lengths by applying the Non prime MAC array LDPC codes to IMAC array LDPC codes. The design is based on the construction the rows that haven't the same submatrix in the same row by include the idea of both methods together.

From Section 2.10, IMAC array LDPC codes interleaved the parity check matrix of MAC array LDPC codes by quasi cyclic matrix or ω. If we construct the quasi cyclic matrix with the non prime submatrix size L, ω will generateshownn theqsame column as show in equation (4.1).

$$\omega = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{(6\times6)} \tag{4.1}$$

From equation (4.1), we see that ω size 6x6 generate more 1's in the same column. If we use it directly to interleave the array LDPC, it will generate more cycle 4 in the parity matrix that will reduce the performance of the code. So, we do left shifting of the row 4-6 to generate a new matrix (T) as shown in equation (4.2). So, we can see that it doesn't generate cycle 4 in the same column as the original ω.

$$T = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}_{(6\times6)} \tag{4.2}$$

Subsequently, we construct T^2 by doing right shifting of the matrix. T^3 can be obtained similarly. They are shown in equation (4.3). Fairly obvious, $T^n = T$. The final H matrix then achieved as shown in equation (4.4).

$$T^2 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{(6\times6)} \rightarrow T^3 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}_{(6\times6)}, T^n = T \tag{4.3}$$

$$H=\begin{bmatrix} I & I & IT & IT^2 & \cdots & \cdots & IT^j \\ 0 & I & \alpha^{P_{NP}(2,3)}T & \cdots & \alpha^{P_{NP}(2,j)}T^{j-1} & \cdots & \alpha^{P_{NP}(2,k-1)}T^j \\ 0 & 0 & I & \alpha^{P_{NP}(3,4)}T^2 & \alpha^{P_{NP}(3,j)}T^{j-1} & \cdots & \alpha^{P_{NP}(3,k-2)}T^j \\ \vdots & \vdots & \vdots & \ddots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & \cdots & I & \cdots & \alpha^{P_{NP}(j,k-j+1)}T^j \end{bmatrix} \quad (4.4)$$

In equation (4.4), we interleaved the Non prime MAC array LDPC codes with T matrix to improve the bit error rate performance of the codes. The structure of the parity check matrix will be the random form that can achieve better performance than structural form. We show the example of H matrix for the Non prime MAC array LDPC codes with parameter j = 3, k = 12 and L= 12 by

$$H=\begin{bmatrix} I & I & IT & IT^2 & IT^3 & IT^3 & IT^3 & IT^3 & IT^3 & IT^3 & IT^3 & IT^3 \\ 0 & I & \alpha T & \alpha^2T^2 & \alpha^3T^3 & \alpha^4T^3 & \alpha^5T^3 & \alpha^6T^3 & \alpha^7T^3 & \alpha^8T^3 & \alpha^9T^3 & \alpha^{10}T^3 \\ 0 & 0 & I & \alpha^2T^2 & \alpha^4T^3 & \alpha^6T^3 & \alpha^8T^3 & \alpha^{10}T^3 & \alpha T^3 & \alpha^3T^3 & \alpha^5T^3 & \alpha^7T^3 \end{bmatrix}$$

We will investigate the bit error rate performance of Interleaved Non prime MAC array LDPC codes on the long block length. Because of, IMAC array LDPC codes have the superior performance and has the high code rate. The 4 Kbits block length is very interesting to consider because it uses in the magnetic recording system which require high code rate. The test parameters are shown in Table 4.1.

Table 4.1 Parameter for IMAC, MAC and Interleaved Non prime MAC at 4K block length

Parameter	IMAC	MAC	IMAC	MAC	Purpose
j	5	5	5	5	5
k	61	61	60	60	60
L	67	67	68	68	68
code rate (1-j/k)	0.92	0.92	0.92	0.92	0.92
input bit (L(k-j))	3,752	3,752	3,740	3,740	3,740
parity bit (jL)	335	335	340	340	340
code bit (kL)	4,087	4,087	4,080	4,080	4,080
Iterations	30	30	30	30	30

We compare the result with IMAC with similar block length and code rate. It sees that the purpose matrix that construct by equation (4.5) have the similar result with the prime IMAC and better than prime MAC. BER versus *Eb/N0* plot is shown in Figure 4.1.

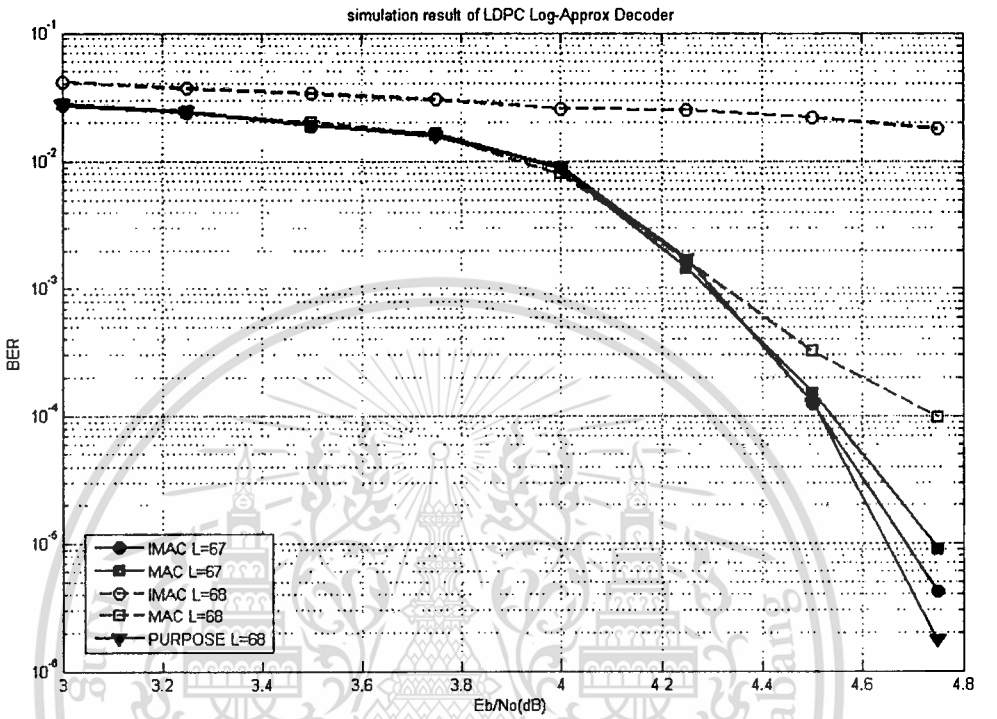


Figure 4.1 Simulation result of non prime IMAC and existing LDPC's performance.

The result of the parity check matrix, the same block length and same code rate, has a very superior performance when compared to MAC array LDPC codes and IMAC array LDPC codes that construct with non prime submatrix size *L*. MAC array LDPC codes and IMAC array LDPC codes with non prime submatrix size *L* has very poor performance because many cycle 4 were generated in the parity check matrix. But, the design matrix hasn't cycle 4. Non prime MAC array LDPC's matrix was interleaved by matrix *T*. It has superior performance when compared with Prime MAC array LDPC codes and it has better than Prime IMAC array LDPC codes.

4.2 To construct Non prime MAC LDPC codes with Chinese Remainder Theorem (CRT)

The improved QC-LDPC codes that use the Chinese Remainder Theorem (CRT) has been studied lately by [32], [33] and [34]. The CRT can be used to address the problems of the cycle 4 issue and/or girth reduction in case of long block length QC-LDPC coded. Reference S. Myung and K. Yang had proposed a method for

constructing long length QC-LDPC codes of small length QC-LDPC codes using the CRT. They were applying the CRT method to array codes and presented a family of high rate regular QC-LDPC codes with no cycle 4. Y. Liu has proposed a generalization of CRT combining method to design better QC-LDPC codes. Their results show that a BER of 10^{-6} can be obtained. Their generalized combining method outperformed 0.5 dB of SNR compared to S. Myung and K. Yang method. Recently, X. Jiang and M. H. Lee had presented the remedy to two problems associated with CRT based QC-LDPC codes: how to extend the code length code without reducing the girth and how to design codes with a prescribed girth easily. They have proposed a method of combining QC-LDPC codes via CRT. In contrast to [32], [33] and [34] has constructed H_2 with large girth submatrix while H_1 still follows. The resulted codes have flexible length, flexible rates, large girth and suite well iteration decoding. Based on the prime number parameters of the matrices have more focused to regular LDPC codes and irregular LDPC codes. A limitation of using of prime number parameters is that the designed codes can support only some particular code lengths. So, we investigate the application of CRT in designing the irregular LDPC codes with nonprime block length.

4.2.1 The Combination of QC-LDPC Codes via CRT

The parity check matrix of QC-LDPC codes shown in equation (4.6) consists of small square blocks of $L \times L$ zero matrix or circulant permutation matrices.

$$H = \begin{bmatrix} P^{\alpha_{11}} & P^{\alpha_{12}} & \cdots & P^{\alpha_{1(k-1)}} & P^{\alpha_{1k}} \\ P^{\alpha_{21}} & P^{\alpha_{22}} & \cdots & P^{\alpha_{2(k-1)}} & P^{\alpha_{2k}} \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ P^{\alpha_{j1}} & P^{\alpha_{j2}} & \cdots & P^{\alpha_{j(k-1)}} & P^{\alpha_{jk}} \end{bmatrix}_{j \times k} \tag{4.6}$$

Where $P^{\alpha_{mn}}; (1 \le m \le j, 1 \le n \le k)$ represents an $L \times L$ circulant permutation matrix obtained by cyclically right shifting the identity matrix, I to the right by α_{mn} times, and $\alpha \in \{0,1,\ldots,L-1,\infty\}$. The zero matrix of size $L \times L$ is denoted by I^{∞}. The exponent matrix, $E(H)$ of the given H in equation (4.6) is defined as

$$E(H)=\begin{bmatrix} \alpha_{11} & \alpha_{12} & \cdots & \alpha_{1(k-1)} & \alpha_{1k} \\ \alpha_{21} & \alpha_{22} & \cdots & \alpha_{2(k-1)} & \alpha_{2k} \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ \alpha_{j1} & \alpha_{j2} & \cdots & \alpha_{j(k-1)} & \alpha_{jk} \end{bmatrix} \tag{4.7}$$

The parity check matrix H can be obtained by replacing each α_{mn} entry of $E(H)$ with $P^{\alpha_{mn}}$. The mother matrix, $M(H)$ of the given H is obtained by replacing zero matrix and circulant permutation matrix in H with " 0 " and " 1 ", respectively. α_{mn}^{q} notes the exponent $(mn)^{th}$ of the matrix q.

Let $L_1, L_2, \ldots, L_t$ be pair-wise relatively prime positive integers. Let C_q be a QC-LDPC code whose binary parity check matrix, H_q with dimension of $jL_q \times kL_q$. A combining method proposed in [32] was used to construct a QC-LDPC code as the following procedures.

Step1: If $GCD(L_q, L_r)=1$ for all $q,r \le t$ and $(q \neq r)$

If $\alpha_{mn}^{q} \neq \infty$ for $1 \le q \le t$, then

$$\alpha_{mn}=\sum_{q=1}^{t}\alpha_{mn}^{q} b_q L_q \bmod L \tag{4.8}$$

where

$$L=L_1 L_2 \ldots L_t, \quad L_q=\frac{L}{L_q} \tag{4.9}$$

and $b_q L_q \equiv 1 \bmod L_q$. Otherwise, $\alpha_{mn}=\infty$.

Step 2 : The exponent matrix $E(H)$ for H is defined by

$$E(H)=(a_{mn}) \tag{4.10}$$

Step 3 : The parity check matrix H can be obtained by replacing each exponent coupling of $E(H)$ with $P^{\alpha_{mn}}$.

For example, consider two QC-LDPC codes that have parity check matrices, H_1 and H_2 that $GCD(L_1, L_2)=1$, $E(H_1) = (a_{mn}^{1})$, $E(H_2) = (a_{mn}^{2})$ and $M(H_1) = M(H_2)$. While they are combined using CRT, the exponent matrix $E(H) = (c_{mn})$ is given by

$$c_{mn}=\left\{a_{mn}^{1} b_1 L_1 + a_{mn}^{2} b_2 L_2\right\} \bmod L \tag{4.11}$$

where b_1 and b_2 are two integers such that $b_1 L_2 \equiv 1 \bmod L_1$ and $b_2 L_1 \equiv 1 \bmod L_2$. Then the parity check matrix, H can be obtained by exponent coupling of $E(H)$ and the $L_1 L_2 \times L_1 L_2$ circulant permutation matrix, P where

$$P_{mn} = \begin{cases} 1 & \text{if } n+1 \equiv m \bmod L \\ 0 & \text{otherwise} \end{cases} \tag{4.12}$$

Points to be noted here; [32],[33] and [34] have designed the codes only with prime-prime number combination. Moreover, the design of H_1, H_2 ,..., H_t is quite complicate when designing nonprime–nonprime or nonprime–prime combinations. In next section we introduce the method to construct H_1, H_2, ..., H_t that compatibles with the rules of CRT even if nonprime–nonprime or non prime–prime combination are the cases.

4.2.2 Matrix Design via CRT

In this section, we propose the construction method with nonprime number parameters for both $GCD(L_1, L_2) = 1$ and $GCD(L_1, L_2) \neq 1$. To extend an $jL_q \times kL_q$ to $jL \times kL$, parity matrix, H can be constructed by combining $E(H)$ via CRT. This method can be separated into 2 cases. First case, $GCD(L_1, .., L_q) = 1$, is the combination of either prime-prime such as L_7 and L_3, or prime-nonprime such as L_7 and L_4. In the second case, $GCD(L_1, .., L_q) \neq 1$, that can be either nonprime-prime such as L_6 and L_3 or nonprime-nonprime such as L_6 and L_4. The prime-prime choice is in fact the same as [32]. The design of each case is elaborated as the follows:

Step 1 : Construct the H_1 parity matrix as eq (2.49) given in [8]

Step 2 : Construct H_2 by check the case of GCD.

Case 1: Design to support $x = GCD(L_1, ..., L_q) = 1$

- Construct H_2 by eq (2.49) and cyclic shift down for one time and replace the 1st row with I for all a^2_{1n}

Case 2: Design to support $x = GCD(L_1, ..., L_q) \neq 1$

- Construct H_2 by
  - the 1st row construct with I for all a^2_{1n}
  - Construct a^2_{2n} for $2 \leq n \leq k$ by $a^2_{2n} = (I + y)$, $a^2_{2n} - a^1_{2n} \mid x$ and $y = 0, 1, ..., x-1$

- Construct a^2_{ij} in H_2 for $3 \le m \le j$ and $2 \le n \le k$ by $a^2_{mn} = a^1_{mn} + x$
- $a^2_{mn} = I$, and $L = (\prod_{i=1}^{t} L_i) / x$

Step 3 : Using CRT to combine H_1 and H_2 to form H

Step 4 : Check whether matrices, H satisfy the conditions that all remainders of $a_{mn} / L_q = a^q_{mn}$ for all m, n.

Step 5 : Rearrange the obtained H matrix into Modified array codes structure, since Modified array codes yields attractive properties in encoding.

In the case where $GCD(L_1,...,L_q) \neq 1$, $a^1_{mn}, a^2_{mn}, ..., a^q_{mn}$ in each matrix, H_k must be compatible with the rules of CRT that $a^2_{mn} - a^1_{mn} \mid GCD(L_1, L_2)$. Therefore we can construct the H_2 with the modified version of H_1. To comply with CRT rules, all the remainders a_{mn} of H must satisfy the condition that $a_{mn} / L_q = a^q_{mn}$ for all m, n. Then we will show the combination matrix via CRT when $GCD(L_1,...,L_q) \neq 1$ that illustrated by the follow example.

Example 4.1 : (Prime-Prime, $GCD(L_1, L_2) = 1$); To design the parity check matrix, H for 2050 codeword LDPC codes. With $j = 3, k = 10, L = 205$, we can have $L_1 = 41, L_2 = 5$.

Step 1 : Construct the H_1 parity matrix as

$$H_1 = \begin{bmatrix} I & I & I & I & I & I & I & I & I & I \\ I & \alpha^1 & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 & \alpha^7 & \alpha^8 & \alpha^9 \\ I & \alpha^2 & \alpha^4 & \alpha^6 & \alpha^8 & \alpha^{10} & \alpha^{12} & \alpha^{14} & \alpha^{16} & \alpha^{18} \end{bmatrix}$$

Step 2 : Construct H_2 by check the case of GCD.

Check whether $x = GCD(41,5) = 1$. So, we will use case 1 to construct H_2.

$$H_2 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \alpha^1 & \alpha^2 & \alpha^3 & \alpha^4 & 1 & \alpha^1 & \alpha^2 & \alpha^3 & \alpha^4 \end{bmatrix}$$

Step 3 : Using CRT to combine H_1 and H_2 to form H

For H_1 and H_2 , we can write $E(H_1)$ and $E(H_2)$ as follow.

$$E(H_1) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 0 & 2 & 4 & 6 & 8 & 10 & 12 & 14 & 16 & 18 \end{bmatrix},$$

$$E(H_2) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 & 4 & 0 & 1 & 2 & 3 & 4 \end{bmatrix}$$

After get $E(H_1)$ and $E(H_2)$, we will using equation (4.11) to calculate $E(H) = (c_{mn})$ for the design parity check matrix.

$$L = \frac{41 \times 5}{1} = 205, L_1 = 205/41 = 5, L_2 = 205/5 = 41,$$

$$b_1(5) \equiv 1 \bmod 41, b_1 = 33; b_2(41) \equiv 1 \bmod 5, b_2 = 1$$

Then, we have $c_{22} \equiv \{(1 \times 33 \times 5) + 0\} \bmod 205 = 165$. The obtained $E(H)$ and H matrix is given as

$$E(H) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 165 & 125 & 85 & 45 & 5 & 170 & 130 & 90 & 50 \\ 0 & 166 & 127 & 88 & 49 & 10 & 176 & 137 & 98 & 59 \end{bmatrix}$$

$$H=\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \alpha^{165} & \alpha^{125} & \alpha^{85} & \alpha^{45} & \alpha^{5} & \alpha^{170} & \alpha^{130} & \alpha^{90} & \alpha^{50} \\ 1 & \alpha^{166} & \alpha^{127} & \alpha^{88} & \alpha^{49} & \alpha^{10} & \alpha^{176} & \alpha^{137} & \alpha^{98} & \alpha^{59} \end{bmatrix}$$

Step 4 : Check whether matrices, H satisfy the conditions that all remainders of $a_{mn} / L_q = a_{mn}^{q}$ for all m, n. For the example $a_{24} / L_1 = 85/41 = 3$. That is the same value as a_{24}^{1} in $E(H_1)$.

Step 5 : Rearrange the obtained H matrix into Modified array codes structure, since Modified array codes yields attractive properties in encoding.

$$H=\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & \alpha^{165} & \alpha^{125} & \alpha^{85} & \alpha^{45} & \alpha^{5} & \alpha^{170} & \alpha^{130} & \alpha^{90} \\ 0 & 0 & 1 & \alpha^{166} & \alpha^{127} & \alpha^{88} & \alpha^{49} & \alpha^{10} & \alpha^{176} & \alpha^{137} \end{bmatrix}$$

Example 4.2 : (Nonprime – Nonprime, $GCD(L_1, L_2) \neq 1$); Consider codeword, C which has a 3×12 sub matrix with $L = 174$. We use two circulant permutation matrices, H_1 and H_2 with $L_1 = 87$ and $L_2 = 6$, respectively. As $GCD\ (H_1, H_2) = 3$, we are sticking to case 2.

Step 1 : Construct the H_1 parity matrix as

$$H_1=\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \alpha^{1} & \alpha^{2} & \alpha^{3} & \alpha^{4} & \alpha^{5} & \alpha^{6} & \alpha^{7} & \alpha^{8} & \alpha^{9} & \alpha^{10} & \alpha^{11} \\ 1 & \alpha^{2} & \alpha^{4} & \alpha^{6} & \alpha^{8} & \alpha^{10} & \alpha^{12} & \alpha^{14} & \alpha^{16} & \alpha^{18} & \alpha^{20} & \alpha^{22} \end{bmatrix}$$

Step 2: Construct H_2 by check the case of GCD.

Check whether $x = GCD(87,6) = 3$. So, we will use case 2 to construct H_2.

$$H_2=\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \alpha^{1} & \alpha^{2} & 1 & \alpha^{1} & \alpha^{2} & 1 & \alpha^{1} & \alpha^{2} & 1 & \alpha^{1} & \alpha^{2} \\ 1 & \alpha^{5} & \alpha^{1} & \alpha^{3} & \alpha^{5} & \alpha^{1} & \alpha^{3} & \alpha^{5} & \alpha^{1} & \alpha^{3} & \alpha^{5} & \alpha^{1} \end{bmatrix}$$

Step 3 : Using CRT to combine H_1 and H_2 to form H

For H_1 and H_2 , we can write $E(H_1)$ and $E(H_2)$ as follow.

$$E(H_1)=\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \\ 0 & 2 & 4 & 6 & 8 & 10 & 12 & 14 & 16 & 18 & 20 & 22 \end{bmatrix},$$

$$E(H_2)=\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 0 & 1 & 2 & 0 & 1 & 2 & 0 & 1 & 2 \\ 0 & 5 & 1 & 3 & 5 & 1 & 3 & 5 & 1 & 3 & 5 & 1 \end{bmatrix}$$

Using equation (4.11) to calculate $E(H) = (c_{mn})$ for the design parity check matrix.

$$L=\frac{87\times 6}{3}=174, L_1=87, L_2=6, L_{11}=29, L_{12}=3, L_{21}=3, L_{22}=2, L_{12}=L_{21}=3$$

So, we choose L_{12} only

$$\dot{L}_{11}=174/29=6, \dot{L}_{12}=174/3=58, \dot{L}_{21}=174/2=86,$$
$$b_{11}(6)\equiv 1 \bmod 29, b_{11}=5\ b_{12}(58)\equiv 1 \bmod 3, b_{12}=1$$
$$b_{22}(87)\equiv 1 \bmod 2, b_{22}=1, c_{22}\equiv\{(1\times 5\times 6)+(1\times 1\times 58)+(1\times 1\times 87)\} \bmod 174=1$$

The obtained $E(H)$ and H matrix is given as

$$E(H)=\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 90 & 91 & 92 & 6 & 7 & 8 & 96 & 97 & 98 \\ 0 & 89 & 91 & 93 & 95 & 97 & 99 & 101 & 103 & 105 & 107 & 109 \end{bmatrix},$$

$$H=\begin{bmatrix} I & I & I & I & I & I & I & I & I & I & I & I \\ I & \alpha^{1} & \alpha^{2} & \alpha^{90} & \alpha^{91} & \alpha^{92} & \alpha^{6} & \alpha^{7} & \alpha^{8} & \alpha^{96} & \alpha^{97} & \alpha^{98} \\ I & \alpha^{89} & \alpha^{91} & \alpha^{93} & \alpha^{95} & \alpha^{97} & \alpha^{99} & \alpha^{101} & \alpha^{103} & \alpha^{105} & \alpha^{107} & \alpha^{109} \end{bmatrix}$$

Step 4 : Check whether matrices, H satisfy the conditions that all remainders of $a_{mn} / L_q = a_{mn}^q$ for all m, n

Step 5 : Rearrange the obtained H matrix into Modified array codes structure, since Modified array codes yields attractive properties in encoding.

$$H = \begin{bmatrix} I & I & I & I & I & I & I & I & I & I & I & I \\ 0 & I & \alpha^{1} & \alpha^{2} & \alpha^{90} & \alpha^{91} & \alpha^{92} & \alpha^{6} & \alpha^{7} & \alpha^{8} & \alpha^{96} & \alpha^{97} \\ 0 & 0 & I & \alpha^{89} & \alpha^{91} & \alpha^{93} & \alpha^{95} & \alpha^{97} & \alpha^{99} & \alpha^{101} & \alpha^{103} & \alpha^{105} \end{bmatrix}$$

To illustrate the performance of codes designed with the technique proposed in the previous section, we design the codes of 4Kbits block length with the slightly high code rate, i.e. 0.81 and 0.9. The high rate is generally required by the data storage system. Designed parameters are shown in Table 4.2 below. The channel is assumed to be AWGN. The decoder iteration limit is set to 30.

Table 4. 2 Parameters for 4Kbits (R=0.9)

Type of parameter	Block Length (bits)	j	k	L
MAC: prime [9]	4120,3708	4	40	103
CRT: nonprix pri	4080,3672(51,2)	4	40	102
CRT: nonprix nonpri	4080,3672(51,6)	4	40	102
CRT: nonprix nonpri	4080,3672(51,34)	4	40	102

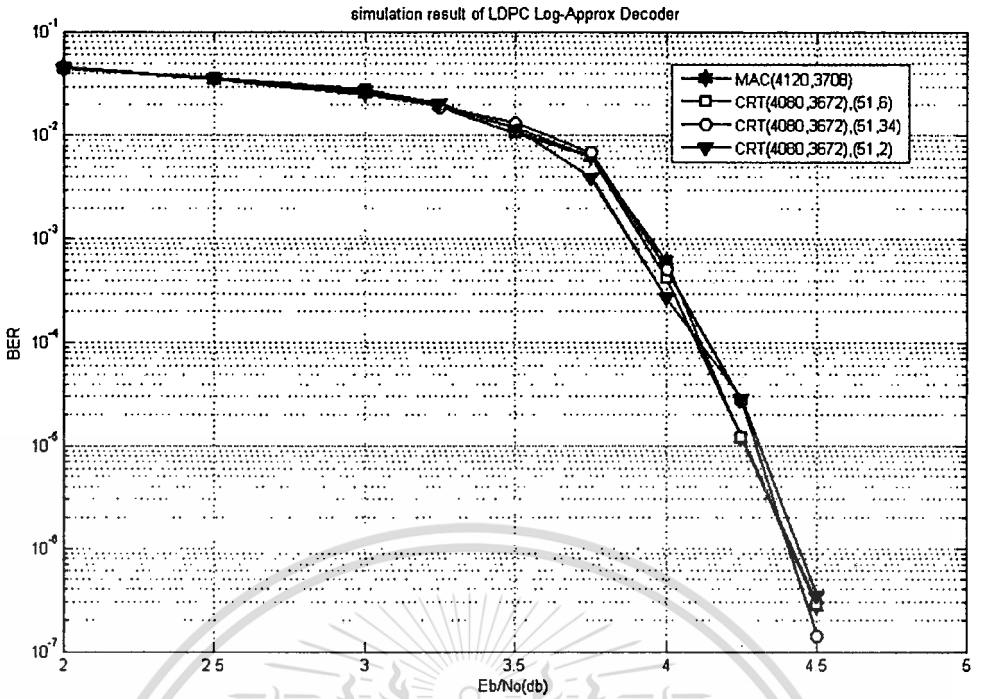


Figure 4.2 Codes performance of ours and MAC's at R = 0.90.

The available results of [9] were used to compare with our results at similar block length and code rate. Figure 4.2 shows the results of bit error rate (BER) performance, where $(n,k),(L_1,L_2)$ represent code length, information length, and sub-matrix size of H_1 and H_2, respectively. It can be seen that our work has superior performance compared to the original Modified array codes. The code's performance seems to vary upon the selection of the pairwise parameters. The pair of nonprime-prime yields slightly better performance compared to others

4.3 Non prime MAC LDPC Codes in Magnetic Recording Channel

Figure 4.3, present the read channel model that use in the current magnetic recording technology. Three levels of coding are used to decoding and encoding the information bit. This particular ordering of codes is a recent development in the drive industry and is referred to as reverse concatenation. Older hard disk drives used a forward concatenation scheme where the Modulation and Reed-Solomon code blocks are swapped. Reverse concatenation results in a more efficient coding structure and is also an enabling component for the move to iterative detection. Each of these coding levels has an important role in the function of the read channel. Iterative Detection Read Channel (IDRC) is the outermost level of coding and is very efficient, maximizing format efficiency and adding only 0.5% redundancy to the data stream. Its task is to guarantee that the read signal has continuous

information content. This allows both timing and gain information to be extracted from the read signal, making the data stream self-clocking. IDRC also prevents stressful patterns from being written to the disk. The Reed-Solomon code block are using as the error correction codes. It provide on-the-fly (OTF) correction for errors that remain after the iterative detection process. Soft information-aided Reed-Solomon decoding is also available in a recovery mode for increased error correction capability. The Reed-Solomon code block also ensures that the Modulation code operates on error-free data. The binary LDPC code is integral to the iterative detection process.

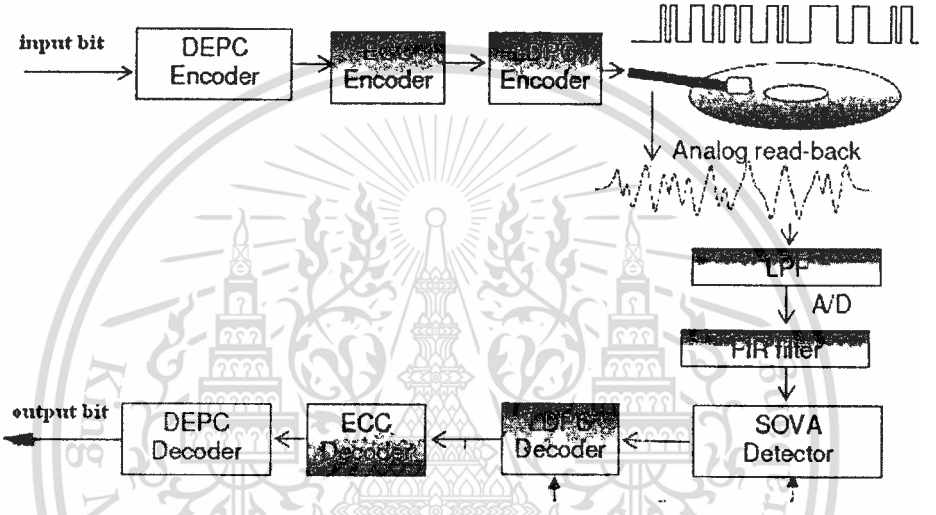


Figure 4.3 Channel model for Magnetic recording systems[42].

Various design of the parity check matrices are presented to encode and decode the information bit. In general, the parity check matrix for an LDPC code constructs with the structural method to reduce the hardware complexity. Array LDPC codes often use as the structural method for generate the parity check matrix. But, it achieve the good performance when only design with prime submatrix size. For 1 sector of hard disk drive, we need 4096 codeword bits to record on the media. Figure 4.4 show the performance of Array LDPC code for Table 4.3.

Table 4. 3 Parameters for 4Kbits sector

Parameter	j	k	L	input bit (L(k-j)	parity bit (jL)	code bit (kL)
MAC	5	61	67	3,752	335	4,087
Non prime MAC	5	64	64	3,776	320	4,096

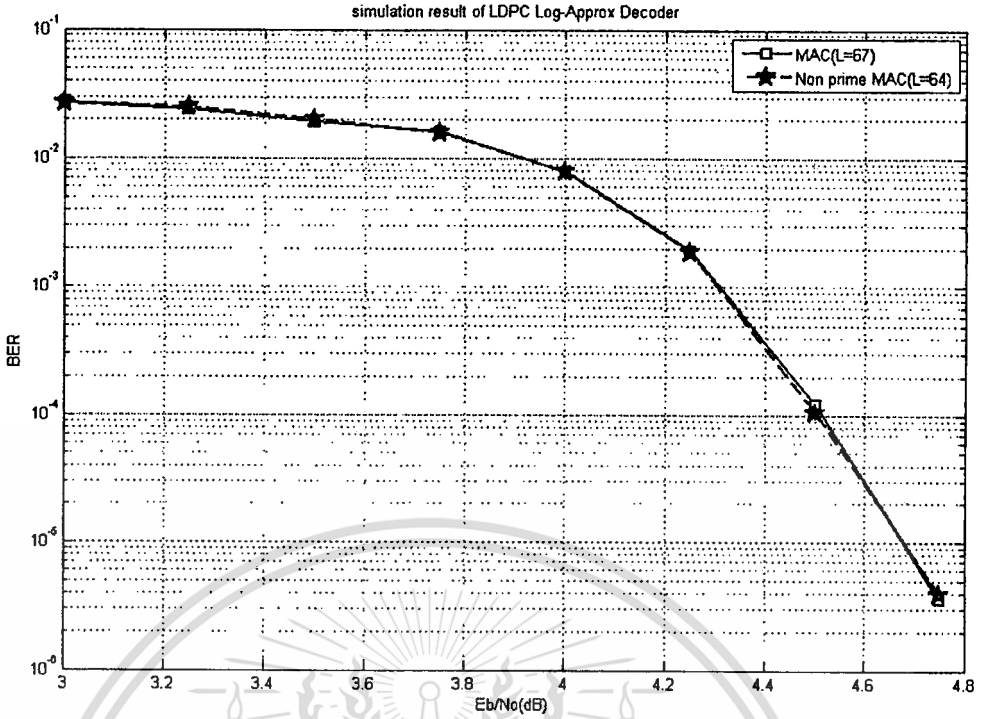


Figure 4.4 Codes performance of Non prime MAC and MAC at 4096 bits.

If we use the Array LDPC codes whose parity check matrix *H* consist of sixty-one 67 × 67 circulant submatrices, we need to add 9 bits of redundancy bit to every sector. We don't need to add the redundancy bit to the sector when we apply our design with sixty-four 64 × 64 circulant submatrices to generate codeword for LDPC encoder. From Table 4.3, we can sse that the parity check matrix *H* with 64 circulant submatrices can record input bit more than the parity check matrix *H* with 67 circulant submatrices. We can conclude that our design method can record the user data more than original MAC code around 1% with the same bit error rate performance.

Chapter 4, we introduce the method to apply Nonprime modified array LDPC code to another code such as Interleaved modified array LDPC code and construct a long length matrix via CRT. In additions the codes still have the attractive properties of LDPC codes (such as simple encoding, low error floor and capability of detecting and correcting burst error) because purpose matrix has the structure like Modified array codes. Our introduce code has superior performance compared to Modified array codes.

Chapter 5
Conclusions and Perspectives

The low density parity check codes (LDPC) were discovered in the early 1960's by R. Gallager. They have then been largely forgotten until their rediscovery in the midnineties. Nowadays, LDPC codes are currently the best performing channel coding schemes. They can achieve the performance reach to Shannon's limit. LDPC codes can be thought as a generic term for a class of error correcting codes distinguished from others by a very sparse parity check matrix, H. It was defined by a wide range of parameters. LDPC's performance improves as the block length increases and it had better performance than Turbo codes at high code lengths. The versatility of LDPC codes enables to perform optimizations to design for optical communications, magnetic storage, multi-input and multi-output channels, and of satellite transmission. The designing of the parity check matrix in LDPC codes is now a hot topic in the field of improving the code's performance.

In this thesis, a MAC array LDPC code published in 2002 by E. Eleftheriou and S. Olcer was modified for using in arbitrary code length design. For the previous array LDPC design with Array LDPC codes, MAC LDPC codes and IMAC LDPC codes, it do not support arbitrary code lengths because the code length of an array LDPC code with good error rate performance is limited to a multiple of a prime submarix size. So, this thesis introduced the design of Non prime MAC array LDPC codes. It was shown how to eliminate the cycle 4 that were generated even if design with non prime submatirx size. They can achieve good error rate performance while supporting arbitrary code lengths. We also showed that the Non prime MAC array LDPC codes achieve better bit error rate performance than the MAC array LDPC codes because it can realize arbitrary code length and has no need to use dummy bits. The addition of dummy bits to matches with the code length of the array LDPC code will degrade the transmission rate in a communication system or reduce the data bit to storing in a magnetic recording system. So, this design is very useful for the application that was limited by the space such as magnetic recording system. The purpose design still has attractive properties such as simple encoding, low error floor and capability of detecting and correcting burst error same as MAC. The Non prime IMAC array LDPC codes, the application of the proposed method, show the improvement of bit error rate performance of our design.

We can apply the introduced design method to another design LDPC code to improve the performance of the code such as extend the shorten array code length to long code length without reducing the code performance via CRT. Codes

constructed with our method differ from Myung and Yang's method that matrix is generated by combining two quasi-cyclic LDPC codes using Chinese Remainder Theorem. Moreever H_1 and H_2 size must be primes and $GCD(L_1, L_2) = 1$. In contract, in our approach there is no need that L_1 and L_2 must be prime to each other; i.e. $GCD(L_1, L_2) \neq 1$. With this property, our design code can support the arbitrary code lengths. We can design matrix H_1 and H_2 with any submatrix size. Our design has quite complicated than Myung and Yang's method but it can generate large girth like Jiang's method that extend the code length of a shortened array code without reducing the girth of a QC-LDPC code with a prescribed girth easily. In additions, the codes still have the attractive properties of LDPC codes (such as simple encoding, low error floor and capability of detecting and correcting burst error) because purpose matrix has the structure like Modified array codes. Our performance results show that even though there is some cycle 4 in H_2, they do not affect on the code's performance. Our code has superior performance compared to Modified array codes. Furthermore, we can use the interleave method to improve the code's performance.

In the future, the propose scheme can be extended for the high capacity of magnetic recording application. It can be used for practical implementation of 4 Kbytes sector in HDD. It can also apply to the other array LDPC codes for improving the bit error rate performance. To design the Non prime MAC array LDPC codes with the good bit error rate performance with the short block length also very interesting in expanding the propose method to use in another application.

References

[1] Lacan J., Delpeyroux E. "The q-ary image of some *qm*-ary cyclic codes: Permutation group and soft-decision decoding." **IEEE Trans. Inform. Theory.**, vol. 48, no. 7, July 2002, pp. 2069-2078.

[2] Lim F., Fossorier P. M., Kavcic A. "Code automorphisms and permutation decoding of certain Reed-Solomon binary images.," **IEEE Trans. Inform. Theory.**, vol. 56, no. 10, October 2010, pp. 5253 -5273.

[3] Irving S., Reed and Gustave S. "Polynomial codes over certain finite fields." **J. Soc. Indust. Appl. Math.**, vol. 8, no. 2, 1960, pp. 300-304.

[4] Singleton C. R. "Maximum distance q-nary codes". **IEEE Trans. Inf. Theory.**, vol. 10, no. 2, 1964, pp. 116–118.

[5] Gallager R. "Low-density parity-check codes." **IRE Transactions on Information Theory.**, vol. IT-8, Jan. 1962, pp. 21-28.

[6] Mackay J. C. D., Neal R. "Near Shannon Limit Performance of Low Density Parity Check Code." **Electronics Letter**, vol. 33, Mar 1997, pp. 457-458.

[7] Tanner M. R. "A Recursive Approach to Low Complexity Code." **IEEE Trans. Information Theory.**, Sep. 1981, pp. 533-547.

[8] Fan L. J. "Array Codes as low-density parity-check codes." **Proc. 2nd Int. Symp. Turbo Code.**, Sept. 2000, pp. 543-546.

[9] Eleftheriou E., Olcer S. "Low-density parity Check Codes for Digital Subscriber Lines." **Proc.2002 Int. Conf. on Comm.**, April-May 2002, pp. 1752-1757.

[10] Singhaudom W., Noppankeepong S., Suphithi P."Design of Hign-Rate Modified Array Codes for Magnetic Recording System." **ECTI International Conference.**, May 2007.

[11] Moon J., Park J. "Detection of prescribed error events: application to perpendicular recording." **Communications, 2005. ICC 2005. 2005 IEEE International Conference**, vol. 3, May 2005, pp.2057–2062.

[12] Madden M., Oberg M., Wu Z., He R. "Read channel for perpendicular magnetic recording." **IEEE Trans. Magn.**, vol. 40, no. 1, Jan. 2004, pp. 241–246.

[13] Bertram N. H. 1994. **Theory of Magnetic Recording**. Cambridge University Press.

[14] Sawaguchi H., Nishida Y., Nakagawa T., Takano H., Aoi H. "Signal processing and coding techniques for perpendicular recording." **Joint NAPMRC 2003.Digest of Technical Papers**, Jan. 2003, p. 8.

[15] Hashimoto M., Miura K., Muraoka H., Aoi H., Nakamura Y., "Influence of magnetic cluster size distribution on signal to noise ratio in perpendicular magnetic recording media." **IEEE Trans. Magn.**, vol. 40, no. 4, July 2004, pp. 2458–2460.

[16] Wilson B. "Signal processing for perpendicular recording." **Joint NAPMRC 2003. Digest of Technical Papers**, Jan. 2003, p. 9.

[17] Lee H.S., Bain J., Hong S. Y., and Lee H. "Influence of polarity of media DC background on write/read performance in perpendicular media with an antiferromagnetically stabilized soft underlayer" **IEEE Trans. Magn.**, vol. 41, no. 10, Oct. 2005, pp.3157–3159.

[18] Dhagat P., Palmer D., Xu B. "Neighborhood induced reader bias shift in perpendicular recording." **Journal of Applied Physics**, vol. 93, no. 10, May 2003, pp. 7729–7731.

[19] Tan W., Cruz J. "Signal processing for perpendicular recording channels with intertrack interference." **IEEE Trans. Magn.**, vol. 41, no. 2, Feb. 2005, pp. 730–735.

[20] Erden M., Kurtas E. "Thermal asperity detection and cancellation in perpendicular magnetic recording systems." **IEEE Trans. Magn.**, vol. 40, no. 3, May 2004, pp. 1732–1737.

[21] Mathew G., Tjhia I. "Thermal asperity suppression in perpendicular recording channels." **IEEE Trans. Magn.**, vol. 41, no. 10, Oct. 2005, pp. 2878–2880.

[22] Soriaga B. J. 2005. "On near-capacity code design for partial response channels." Ph.D. dissertation, Department of Electrical and Computer Engineering, University of California, San Diego.

[23] Kovintavewat P., Ozgunes I., Kurtas E., Barry R. J., McLaughlin W. S. "Generalized partial-response targets for perpendicular recording with jitter noise." **IEEE Trans. Magn.**, vol. 38, no. 5, 2002.

[24] Moon J., Zeng W. "Equalization for maximum likelihood detectors." ***IEEE Trans. Magn.***, vol. 31, no.2, Mar. 1995, pp. 1083–1088.

[25] Yang H., Mathew G. "Joint design of optimum partial response target and equalizer for recording channels with jitter noise." **IEEE Trans. Magn.**, no. 1, Jan. 2006, pp.70–77.

[26] Coker J., Eleftheriou E., Galbraith R., Hirt W. "Noise-predictive maximum likelihood (NPML) detection." **IEEE Trans. Magn.**, vol. 34, no. 1, Jan 1998, pp. 110–117.

[27] Richardson J. T., Shokrollahi A. M., Urbanke L. R. "Design of capacityapproaching irregular low density parity-check codes." **IEEE Trans. Inf. Theory**, vol. 47, no. 2, Feb. 2001, pp. 619-637.

[28] Tanner M. R. "A Recursive Approach to Low Complexity Codes." **IEEE Transactions on Information Theory**, vol. IT-27, no. 5, September 1981, pp. 533-547.

[29] Luby G. M., Mitzenmacher M., Shokrollahi A. M., Spielman A. D. "Improved Low-Density Parity- Check Codes Using Irregular Graphs." **IEEE Transactions on Information Theory,** vol. 47, no. 2, February 2001, pp. 585-598.

[30] Kou Y., Li S., Fossorier P. C. M. "Low-density Parity-Check Codes:Construction based on Finite Geometries." **IEEE Globecom 2000, San Fransisco, CA,** vol. 2, no. 7, November 2000, pp. 825- 829.

[31] Kou Y., Li S., Fossorier P. C. M. "Low-density Parity-Check Codes based on Finite Geometries: A Rediscovery and New Results." **IEEE Transactions Information Theory,** vol. 47, no. 7, November 2001, pp. 2711-2736.

[32] Myung S., Yang K. "A combing method of quasi-cyclic LDPC codes by the Chinese Remainder Theorem." **IEEE Comm. Lett.**, vol. 9, Sept. 2005, pp. 823-825.

[33] Liu Y., Wangm X., Chen R., He Y. "Generalized Combining Method for Design of Quasi-Cyclic LDPC Codes." **IEEE comm. Lett.**, vol. 12, no.5, May 2008, pp. 392-394.

[34] Jiang X., Lee H. M. "Large Girth Quasi-Cyclic LDPC Codes Based on the Chinese Remainder Theorem." **IEEE comm. Lett.**, vol. 13, no. 5, May 2009, pp. 342-344.

[35] Prasartkaew C., Choomchuay S. "A Parity Check Matrix Design for Irregular LDPC Codes with 2K Block Length." **ISPACS Int. Symp. on Intelligent Signal Processing and Comm. Systems,** Dec. 7-9, 2009.

[36] Chaichanavong P., Siegel H. P. "Tensor-product parity code for magnetic recording." **IEEE Trans. Magn.**, vol. 42, no. 2, Feb. 2006, pp. 350–352.

[37] Park J., Moon J., "High-rate error-correction codes targeting dominant error patterns." **IEEE Trans. Magn.**, vol. 42, no. 10, Oct. 2006, pp. 2573–2575.

[38] Abdel-Ghaffar K., Hassner M. "Multilevel error-control codes for data storage channels." **IEEE Trans. Inf. Theory,** vol. 37, no. 3, May 1991, pp. 735–741.

[39] Fahrner A. 2004. "On signal processing and coding for digital magnetic recording system." Ph.D. dissertation, Department of Telecommunications and Applied Information Theory, University of Ulm, Germany.

[40] J. B. Soriaga 2005. "On near-capacity code design for partial response channels." Ph.D. dissertation, Department of Electrical and Computer Engineering, University of California, San Diego,.

[41] Ryan E.W. "An introduction to LDPC codes," in CRC Handbook for Coding and Signal Processing for Recording Systems, B. Vasic and E. M. Kurtas, Eds. CRC Press, 2004.

[42] พรชัย ทรัพย์นิธิ, **เนื้อหาประกอบการสอนวิชาการประมวลสำหรับการบันทึกข้อมูล (Signal Processing for Disk Data Storage)** สาขาวิชาวิศวกรรมโทรคมนาคม คณะวิศวกรรมศาสตร์ และวิทยาลัยนวัตกรรมการจัดการข้อมูล (D*STAR) สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

Appendix

NON-PRIME PARAMETERS OF LDPC CODES WITH SYMMETRICAL SUB-MATRIX

Chalit Chusin[1] , Chutima Prasartkaew[2] and Somsak Choomchuay[3]

[1]College of Data Storage Technology and Applications, King Mongkut's Institute of Technology Ladkrabang, BKK, Thailand
Tel:/Fax: + 66-3-881-5966, E-mail: ithonene@hotmail.com
[2]College of Data Storage Technology and Applications, King Mongkut's Institute of Technology Ladkrabang, BKK, Thailand
Tel:/Fax: + 66-2-326-4731, E-mail: prasartkaew@yahoo.com
[3]Faculty of Engineering, King Mongkut's Institute of Technology Ladkrabang, BKK 10520, Thailand
Tel: +66-2-326-4222 Ext.114, Fax: +66-2-739-2398, E-mail: kchsomsa@kmitl.ac.th

ABSTRACT

The performance of Low-density parity-check (LDPC) codes is based on the parity check matrix design. In this paper, the irregular symmetrical parity check matrix was designed using of non-prime number. Then the arbitrary block length can be obtained. The advantages of symmetrical matrix are: 1) easily and quickly design, 2) the good performance as good as random construction was obtained and 3) higher minimum distance, the higher error detection and correction ability was obtained. Moreover, the performance of designed non-prime parity check matrix is higher when compared with MAC and SC-Array. Our designed parity check matrix still has the general properties of LDPC codes.

Index Terms— *LDPC Codes, non-prime, symmetrical, coding*

1. INTRODUCTION

A Low Density Parity Check (LDPC) codes is error correcting codes which has code performance close to Shannon limit. This code has many advantages such as low error floor, high code rate, no cycle-4 and capability of detecting and correcting burst error. In this decade, many researches are concerning the LDPC codes performance development. A design of parity check matrix is one challenge of those researches, since of the performance of LDPC codes depends on the parity check matrix design. In the parity check matrix design, the sparse one's should be decreased but must be enough for data error correction. Richardson *et al.* [1] presented the LDPC codes construction algorithm to yield the performance very close Shannon limit with a difference of only 0.0045 dB. It has very high performance when used for long block length. Eleftheriou and Olcer [2] and Singhaudom *et al.* [3] proposed a parity check matrix structure for LDPC codes, using 'Cyclic Shift' construction. Their code performances are equivalence to the random parity check matrix structure LDPC codes. The performance test results show that the longer block length, the higher performance yielded.

LDPC Codes hold two types of parity check matrix: regular and irregular. For regular parity check matrix, Gallager [4] purposed a construction of a random parity check matrix which has the same number of one element for every row and also has the same number of one element for every column, and without cycle-4. Fan [5] presented a design of an array parity check matrix structure. The complexity of parity check matrix construction is lower but still has the good performance as good as of the random parity check matrix. For irregular parity check matrix, because of its higher performance, when compared to a regular one, researchers are interesting on the later, e.g., [7], [2], [3] and [6]. In those works, the LDPC codes performances were improved by the variety of parity check matrix design. The complexity of the encoder can be made lower by using the triangular form matrices [2], [3] and [6].

In the design of parity check matrix of $jp \times kp$ sizes, the design parameters such as j, k and p should be suitably defined. Where j and k are integers, p is prime number and $j,k \leq p$. The traditional designs using of prime number parameter are presented in [4], [1], [5], [2], [3] and [6]. [4] has proposed the construction of regular LDPC codes with random parity check matrix and [1] have presented the evaluation of performance of this LDPC codes and the results shown that it performance close to the Shannon limit. [5] has proposed the array structure sub-matrix of parity check matrix with is modified from [4] and still be the regular LDPC codes for long block length. [2] have presented the modification of array structure sub-matrix from regular LDPC codes of [5] to be irregular using the concept of quasi-cyclic shift (α) into sub-matrix. [3] modified the parity check matrix from [2] by using interleave quasi-cyclic shift (ω). [6] have presented the parity check matrix for short and medium block lengths which is modified from [2] and [3].

For non-prime number sub-matrix constructions, this was firstly presented by Abematsu *et al.* [8] and followed by Chusin *et al.* [9]. Where, [8] modified the work proposed by [2] but with non-prime number construction parameters instead of prime number. Similarly [9] have modified the sub-matrix given by [3] and with non-prime number.

In this paper we propose a new design parity check matrix for arbitrary code length aims at improves the coding performance suitable for 4K bits block length using of symmetrical sub-matrix with non-prime number construction parameters. This paper is organized as follows: General perspective of LDPC codes is given in section II where

encoding and decoding are also included. Section III gives brief details of a symmetrical matrix. A design of parity check matrices for arbitrary and non-prime code lengths is outlined in section IV and the corresponding performance tests are reported in section V. Finally, the paper is concluded in section VI.

2. LDPC CODES

Low-density parity-check (LDPC) codes proposed by Gallager in 1962 [4], have attracted much attention given their good performance. The encoding and decoding of LDPC codes can be done by using of sparse parity check matrix H. The relationship between codeword bits and parity checks bit of the parity check matrix can be graphically presented by a Tanner graph [10]. A parity-check matrix H has n columns and m rows, and the codeword consists of n bits, which satisfy m checks, the number of message bits will be $k=m$, and the code rate is $R_c=k/n$. The number 1's in the parity check matrix in rows and columns represents an edge between the *i-th* bit node c_i and the *j-th* check node f_j.

For encoding of the LDPC codes, it is similar to other linear block codes, it has the relation;

$$C_{(1\times n)} = \left[m_{(1\times m)} \middle| p_{(1\times n-m)} \right] \quad (1)$$

where $p_{(1\times n-m)}$ denotes the parity portion, and $m_{(1\times m)}$ denotes the message portion respectively.

$$C_{(1\times n)} H^T_{(n\times m)} = 0 \quad (2)$$

where C is a codeword matrix, and H is a parity check matrix. In a systematic form, C can be written as:

There are several methods for decoding the LDPC codes e.g.: Believe Propagation (BP), Sum-Product (SP), and Message Passing (MP). The Log-domain Sum-Product algorithm was used in this paper; it is the message passes between check nodes and bit nodes. In each pass the log likelihood ratio (LLR) is recorded for is probability of its likely symbol. By means of this method, the variable q_{ij} be the message sent from the i^{th} bit node to j^{th} check node along a connecting edge, and r_{ji} is the message sent from j^{th} check node to the i^{th} bit node along a connection edge. The message q_{ij} is computed based on the values sent from check nodes connecting to the i^{th} bit node excluding the j^{th} check node.

3. SYMMETRICAL MATRIX

A symmetrical matrix is a square matrix that is equal to its transpose matrix. Let A is a symmetrical matrix. Then

$$A = A^T \quad (3)$$

The entries of a symmetrical matrix are symmetric with respect to the main diagonal (top left to bottom right). So if the entries are written as $A = \left(a_{ij}\right)$, then $a_{ij} = a_{ji}$. This also implies that

$$A^{-1}A^T = I, \quad (4)$$

where I is the identity matrix. The elements of a symmetric matrix A have the form:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{12} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{2n} & \cdots & a_{nn} \end{bmatrix} \quad (5)$$

The parity check matrix based on the symmetrical matrix can easily be constructed and yields a good performance as same as randomly parity check matrix. It can be seen that the '1' elements in the constructed symmetrical parity check matrix is fairly well distributed. As a result, the higher value of the minimum distance is obtained. Consequently, the better error correction can be achieved. The code performance should be as good as, or similar to the random construction presented in [7].

4. DESIGN OF PARITY CHECK MATRICES

In this section, we detail the formulation of the H matrix by using the idea of symmetrical matrix instead of permutation. The new design symmetrical sub matrix is used to form H matrix in the same manner of the forming of array code proposed by [8].

4.1 Some Related Works

Fan [5] has introduced the array structure parity matrix that can offer comparable performance when compared to a random generated parity matrix reported by Gallager [4]. Other features: low noise floor and no existence of cycle-4, are kept as the original features of LDPC codes proposed in [4]. Fan's matrix is shown below.

$$H(p,j,k) \triangleq \begin{bmatrix} I & I & I & \cdots & I \\ I & \alpha & \alpha^2 & \cdots & \alpha^{k-1} \\ I & \alpha^2 & \alpha^4 & \cdots & \alpha^{2(k-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ I & \alpha^{j-1} & \alpha^{2(j-1)} & \cdots & \alpha^{(j-1)(k-1)} \end{bmatrix} \quad (6)$$

Where I is an identity matrix $(p\times p)$,

α is a position permutation matrix $(p\times p)$.

This yields the code rate of $R = 1 - \frac{pj-j+1}{p^2}$.

Eleftheriou and Olcer [2] have proposed the modified array codes (MAC) by applying cyclic shifting to Fan's array [5] given herewith in Eq. (7). The structure noted by Eq. (7) have the code rate of $R = 1-(j/k)$. MAC offers superior performance to Fan's array as it can reduce number of "1" in the lower triangle. This leads to simpler encoder while preserving other LDPC's features.

$$H=\begin{bmatrix} I & I & \cdots & I & I & \cdots & \cdots & I \\ 0 & I & \alpha & \cdots & \alpha^{(j-2)} & \alpha^{(j-1)} & \cdots & \alpha^{(k-2)} \\ 0 & 0 & I & \cdots & \alpha^{2(j-3)} & \alpha^{2(j-2)} & \cdots & \alpha^{2(k-3)} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \cdots & I & \alpha^{(j-1)} & \cdots & \alpha^{(j-1)(k-1)} \end{bmatrix} \quad (7)$$

where I is an identity matrix $(p\times p)$.

α is a position permutation matrix $(p\times p)$.

Singhaudom *et al.* [3] have proposed the interleaved modified array LDPC developed base on the concept of [2] and named it as IMAC. By introducing the quasi cyclic matrix into the cyclic shifting the obtained matrix is given in Eq. (8). The identity matrix and the interleave matrix are shown in Eq. (9). The IMAC is superior to MAC when the block length is particularly long.

$$H=\begin{bmatrix} I & I & I\omega & I\omega^2 & I\omega^3 & \cdots & I\omega^j \\ 0 & I & \alpha\omega & \alpha^2\omega^2 & \alpha^3\omega^3 & \cdots & \alpha^{(k-2)}\omega^j \\ 0 & 0 & I & \alpha^2\omega^2 & \alpha^4\omega^3 & \cdots & \alpha^{2(k-3)}\omega^j \\ \vdots & \vdots & \vdots & I & \alpha^3\omega^3 & \cdots & \vdots \\ 0 & 0 & \cdots & 0 & I & \cdots & \alpha^{(j-1)}\omega^{(k-j)} \end{bmatrix}_{(jp\times kp)} \quad (8)$$

Where ω is the quasi cyclic matrix that constructed from identity matrix by cyclically-shifting of the matrix, I, i.e. $\omega^{n-1}=I$.

$$I=\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{(5\times 5)} \text{ and } \omega=\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{(5\times 5)} \quad (9)$$

Abematsu *et al.* [8] have proposed the Size Compatible (SC)-Array LDPC Codes of which the parity check matrix shown in Eq. (10). The design can support arbitrary code lengths while achieving good error rate performance; it contains few or no cycle-4. It should be noted that [8] has used non-prime sub-block while [5] has. For the SC-array LDPC code, the permutation sub-matrix is decided to eliminate all cycle-4. Such a proposed cyclic shift $P_{sc}(j,k)$ is expressed by Eq. (11).

$$H=\begin{bmatrix} I & I & \cdots & \cdots & I & \cdots & I \\ \alpha^{P_{sc}(2,k)} & I & \cdots & \cdots & \alpha^{P_{sc}(2,j)} & \cdots & \alpha^{P_{sc}(2,k-1)} \\ \alpha^{P_{sc}(3,k)} & \alpha^{P_{sc}(3,k-1)} & I & \cdots & \alpha^{P_{sc}(3,j)} & \cdots & \alpha^{P_{sc}(3,k-2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \cdots & \vdots \\ \alpha^{P_{sc}(j,k)} & \alpha^{P_{sc}(j,k-1)} & \cdots & \cdots & I & \cdots & \alpha^{P_{sc}(j,k-j+1)} \end{bmatrix} \quad (10)$$

where

$$P_{sc}(j,k)=(j-1)(k-1)+\left\lfloor\frac{(j-1)(k-1)}{L}\right\rfloor \quad (11)$$

The IMAC cannot support the arbitrary code lengths, since the code length is restricted to be a multiple of prime number. Chusin *et al.* [9] have presented a design of parity check matrix to address this problem. Their design is based on the construction rows that don't have the same sub-matrix in the same row. They have shown that, with non-prime number, they achieved the same error rate performance as the prime sub-matrix size of IMAC in AWGN channels. The performance for long block lengths is also better when compared with [8].

4.2 LDPC Codes with Symmetrical Sub-matrix

We designed the H matrix by using of symmetrical matrix shown in Eq. (12) instead of permutation matrix. The symmetrical matrix S is of dimension $q\times q$ where q could be either prime or non-prime number. For the designed matrix size $q\times q$, let's define the binary element of matrix be $s_{xy}=\{0,1\}$ and the index r is bound to $r=(q/3)\times 2$. x and y are row and column indices respectively.

In designing of S matrix, the row elements can be computed as given by a pseudo code below:

```
Odd number row:
    bb=0;  aa=0;  x=1;
For cc=1 to q              (row)
for dd=1 to q            (column)
    if  ((2*q)-dd)-(2*bb)=dd   then
s=1;  goto 10;
else
s=0
end;
       end;
10:    cc=cc+2;  bb=bb+1;
if   cc = ((2*q)-cc)-(2*bb)  then
   goto 20;
     end;
20: end
```

The odd number columns can be generated by the transposition of obtained odd number rows as $S_{y,x}=\left(S_{x,y}\right)^T$.

The construction or even number (2, 4, 6, ...) row and column of sub-matrix, S is to generate '1' at the $x=y$ position of each column. If there is no the '1', add this position with the '1', otherwise adding the '0'.

The resulted matrix is shown in Eq. (12);

$$S=\begin{bmatrix}
0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & s_{i,j}\\
0 & s_{2,2} & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0\\
0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & s_{(x+2),(y-1)} & 0\\
0 & 0 & 0 & s_{4,4} & \cdots & 0 & 0 & 0 & \cdots & 0 & 0\\
 & & & \ddots & & \vdots & \vdots & \vdots & & \vdots & \vdots\\
\vdots & \vdots & \vdots & & \ddots & s_{r,s} & 0 & 0 & \vdots & \vdots & \vdots\\
 & & & & & 0 & 0 & s_{(r+2),(r-1)} & \vdots & \vdots & \vdots\\
 & & & & & 0 & s_{(r+1),(r-2)} & 0 & \vdots & \vdots & \vdots\\
\vdots & \vdots & \vdots & & & & & & \vdots & \vdots & \vdots\\
0 & 0 & s_{(y-1),(x+2)} & & & & & & \vdots & \vdots & \vdots\\
s_{y,x} & 0 & 0 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & 0
\end{bmatrix} \quad (12)$$

The example designed matrix of which $q=10$ is shown in Eq. (13).

$$S=\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1\\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0\\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix} \quad (13)$$

As shown in Eq. (13), it can be seen that the matrix hold transpose property or $S^T = S$ and there is not cycles 4 in sub-matrix. We then construct one time shifting or S^1 by performing cyclically-right shifting of the original S. The obtained matrix is given in Eq. (14).

$$S^1=\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1\\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix} \quad (14)$$

Other degree shifting can be performed similarly. The block shifting scheme is arranged similar to that proposed by [8]. Finally, the parity check matrix can be generated as given in Eq. (15). We use this formulation in code design for performance evaluation. In this design, three important parameters consist of j, k and L (j, k ≤ q). Where j and k are integers and q is non-prime number of sub-matrix size.

$$H=\begin{bmatrix}
I & I & I & I & \cdots & \cdots & I\\
0 & I & \lambda^{(2,3)} & \cdots & \lambda^{(2,j)} & \cdots & \lambda^{(2,k-1)}\\
0 & 0 & I & \lambda^{(3,4)} & \lambda^{(3,t)} & \cdots & \lambda^{(3,k-2)}\\
\vdots & \vdots & \vdots & \ddots & \vdots & \cdots & \vdots\\
0 & 0 & \cdots & \cdots & I & \cdots & \lambda^{(j,k-j+1)}
\end{bmatrix} \quad (15)$$

where

$$\lambda^{(j,k)} = S^{P_T(j,k)} \text{ and,}$$

$$P_T(j,k) = (j-1)(k-1) + \left\lfloor \frac{(j-1)(k-j)}{q} \right\rfloor$$

5. PERFORMANCE EVALUATION

The performance of our constructed parity check matrix was investigated for 4K bits blocks length. This is actually useful in the application of magnetic recording system where the sector size is 512 bytes. To compare with some exist publication the sub-matrix size of 68 is used. The test parameters are shown in Table I as below with iteration number of 30. The minimum distance of available published work was determined at d_{min}=.336 [2]. Our work has d_{min}= 341. This implies that our work should have higher capability of error detection and correction.

TABLE I: PARAMETERS FOR 4KBIT BLOCK SIZE

	j	k	q	R	Block size (bit)
MAC [2]	5	61	67	0.918	4087
SC-Array[8]	5	60	68	0.917	4080
This paper	5	60	68	0.917	4080

The design code is run for 30 iterations to the random-generated input sequence as also performed by MAC and SC-array. The obtained performance is shown in Fig. 1 below. The proposed code offers the same performance as MAC and SC-Array do. It only yields slightly better performance when E_b / N_0 is greater than 4.2 dB.

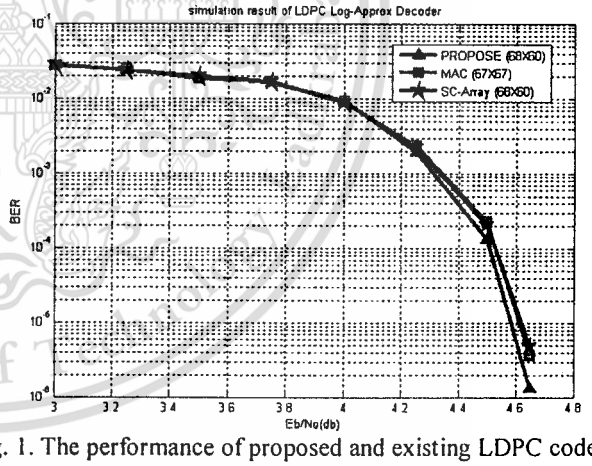


Fig. 1. The performance of proposed and existing LDPC codes (Iteration = 30)

The modified matrix proposed in this paper was also test for the block length of a particular size – say 4080, but at different sub-matrix size. This effect results in the changing of the code rate. We therefore intend for the code rate of better than 0.5. Parameters are given in Table II.

TABLE II: PERFORMANCE TEST PARAMETERS FOR EACH SUB-MATRIX SIZE

Block Size (bit)	j	k	q	R	Iterations
4080	5	60	68	0.917	30
4080	5	51	80	0.902	30
4048	5	44	92	0.886	30
4030	5	26	155	0.808	30
4063	5	17	239	0.706	30
4082	5	13	314	0.615	30
4100	5	10	410	0.5	30

Of the 30 iterations limit, the obtained result is shown in Fig. 2. It can be seen that the performance of the code can be improved when the sub-matrix size is increased as the sub-matrix size varies from 68 to 155. On the other hand, when the code rate is better than 0.8. However the decline in performance is observed when the sub-matrix size is more than 155. Therefore the good performance could be obtained only in particular range of sub-matrix size.

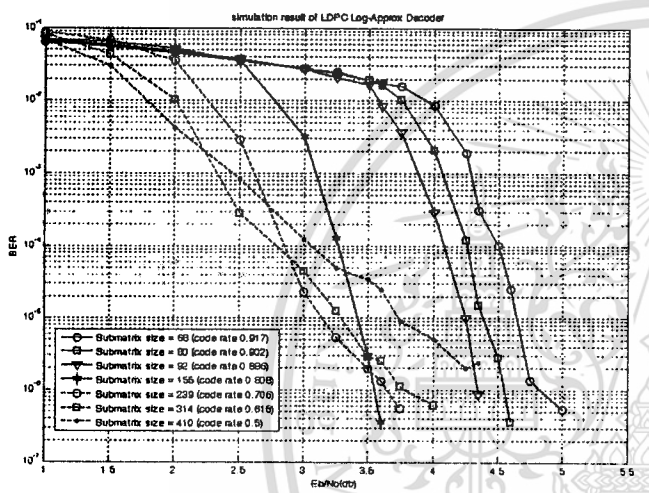


Fig. 2. The test results using different sub-matrix sizes (Iteration = 30)

6. CONCLUSIONS

The design of high-rate code with block sizes suitable for magnetic recording system is illustrated. The block size of the current application is about 4K bits. However, the actual size is determined based on particular data rate (Mbit/sec.) which practically optimized upon several parameters. To ease the design constrain we are trying to design a code that support any block size. The proposed parity check matrix is designed based on symmetrical property rather than permutation effort. Although the design code offers similar or slightly better performance compared to the published MAC [2] and SC-Array [8], the simulation shows that the size of sub-matrix affects the code performance. The larger sub-matrix leads to higher minimum distance and achieve the better performance than the smaller one. However, the sub-matrix size should less than 155, as discussed in the previous section. This proposed matrix still possesses suitable properties for magnetic recording system such as low-complexity encoding process, low error floor and capability of detecting and correcting burst errors.

In summary, the advantages of symmetrical matrix are: 1) easily and quickly design, 2) the good performance as good as random construction was obtained and 3) higher minimum distance, the higher error detection and correction ability was obtained. For future works, we plan to evaluate the performance to the 4Kbytes that using in the new generation of recording technology.

7. REFERENCES

[1] Chung, S.-Y., Forney, G. D., Jr. Richardson, T. J., and Urbanke, R. L., "On the design of low-density parity-check codes within 0.0045 dB of the shannon limit," *Electron. Lett.*, vol. 5, pp. 58-60, Feb. 2001.

[2] E. Eleftheriou and S. Olcer, "Low-density parity check codes for digital subscriber lines," *Proc. 2002 Int. Conf. on Comm.*, pp.1752-1757., April – May, 2002.

[3] W. Singhaudom, S. Noppankeepong, P. Suphithi, "Design of Hign-Rate Modified Array Codes for Magnetic Recording System," *ECTI International Conference*, May 2007.

[4] R. Gallager, "Low-density Parity-check Code," *IRE Trans. Information Theory*, Jan. 1962, pp.21-28.

[5] J. L. Fan, "Array codes as low-density parity-check codes," *Proc. 2nd Int. Symp. Turbo Codes*, France, Sep 2000, pp 543-546.

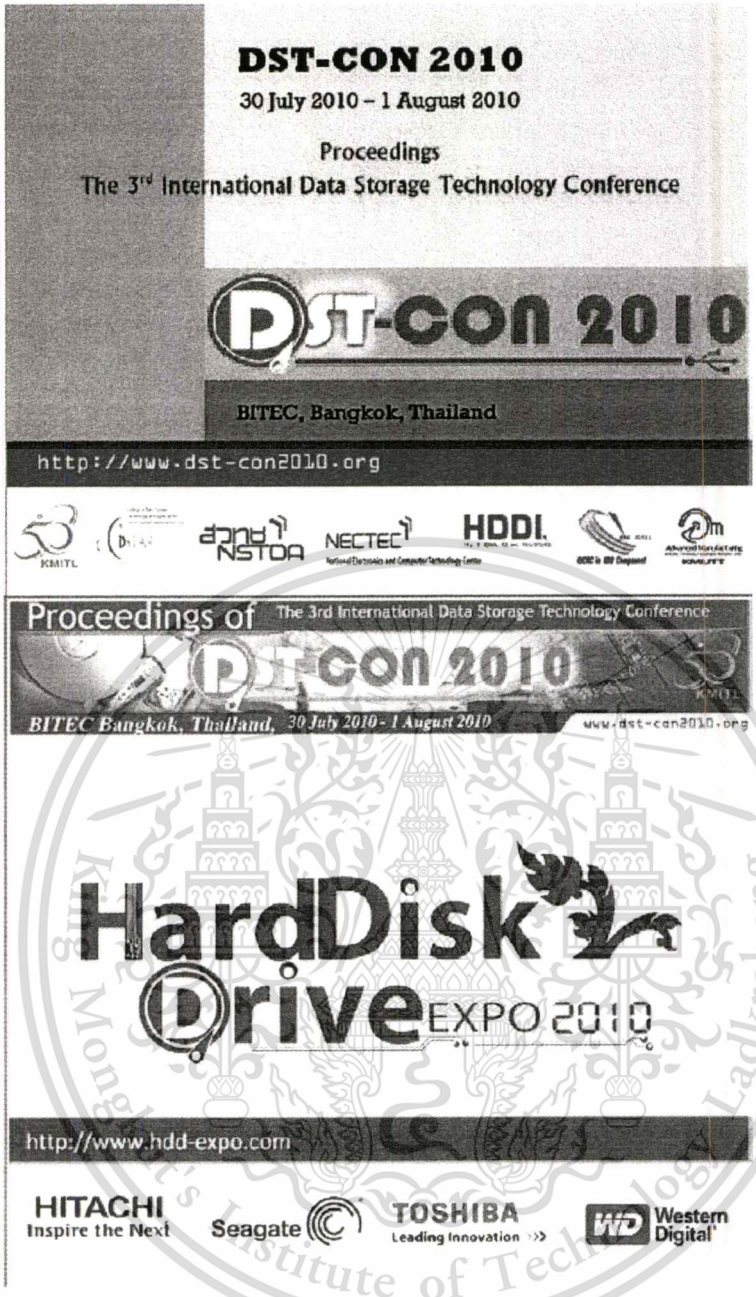
[6] C. Prasartkaew, S. Choomchuay, "A Parity Check Matrix Design for Irregular LDPC Codes with 2K Block Length," *ISPACS International Symposium on Intelligent Signal Processing and Communication Systems*, 2009.

[7] Richardson T.J., Shokrollahi MA, Urbanke R.L. "Design of capacity-approaching irregular low-density parity-check codes," *Information Theory IEEE Trans.* Volume 47, pp.619-637, Feb 2001.

[8] Abematsu, D.; Ohtsuki, T.; Jarot, S.P.W.; Kashima, T.." Size Compatible (SC)-Array LDPC Codes" *Vehicular Technology Conference*, 2007.

[9] C. Chusin, C. Prasartkaew, S. Choomchuay, , "A Design of Non-prime LDPC Based on Interleave Modified Array Codes," *Proceeding of DST-CON 2010*, Bangkok. 2010.

[10] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Information Theory*, pp.533-547, Sept.1981.



A Design of Non-prime LDPC Based on Interleave Modified Array Codes

Chalit Chusin[1], Chutima Prasartkaew[2], and Somsak Choomchuay[3]
[1,2] College of Data Storage Technology and Applications, King Mongkut's Institute of Technology Ladkrabang BKK, Thailand
[1]Tel:/Fax: + 66-3-881-5966, E-mail: ithonenc@hotmail.com
[2]Tel:/Fax: + 66-2-326-4731, E-mail: prasartkaew@yahoo.com
[3]Faculty of Engineering, King Mongkut's Institute of Technology Ladkrabang, BKK 10520, Thailand
Tel: +66-2-326-4222 Ext.114, Fax: +66-2-739-2398, E-mail: kchsomsa@kmitl.ac.th

Abstract— **Interleave Modified Array Codes (IMAC) is a type of Low Density Parity Check Codes (LDPC), which is used to achieve high rate codes and good error rate performance in additive white Gaussian noise (AWGN) channels. IMAC has many advantages in magnetic recording system such as low error floor, capability of detecting and correcting burst error etc. However, IMAC doesn't support arbitrary code lengths, as the code lengths of IMAC with good error rate performance are limited by a multiple of a prime number. This paper presents a design of parity check matrix to support arbitrary code length. The design is based on the construction rows that don't have the same sub-matrix in same row. We conduct computer simulations to evaluate the bit error rate (BER) performance of Non-prime IMAC in AWGN channels. We show that even if the sub-matrix size is a non-prime number, IMAC achieve the same error rate performance as the prime sub-matrix size in AWGN channels.**

Keywords- LDPC; Non-prime blocklenth; Array Codes

I. INTRODUCTION

Low Density Parity Check (LDPC) codes were first introduced by Gallager [1] in 1962.They have recently attracted wide attention in the coding community because their performance is near Shannon limit when use iterative message-passage decoding [2] for decode.

In magnetic recording systems, higher codes rate must be considered since the internal clock rates of read channels gets higher [3]. So, the high rate codes are obviously desirable. "Modified Array Codes" was proposed to use in the communication systems for medium code rate [4]. The codes have the performance similar to Regular / Irregular LDPC codes but they have more attractive properties such as capability of detecting and correcting burst error, simple encoding and low error floor. Interleave Modified Array Codes was designed for high rate applications which the parity matrix is superior to Modified Array Codes when the block length is particularly long [5].

Interleave Modified Array Codes (IMAC) is specified by three parameters; j,k,L where $(j,k) \le L$. Parameters j, k and L represent row weights, column weights and the sub-matrix size of the LDPC code. In general, to construct IMAC with good error rate performance, the sub-matrix size *(L)* has to be prime in order because it avoid the possibility of 4-cycle. The codes length of an IMAC can construct with the multiple of sub-matrix size. Obviously, it is difficult to construct an IMAC with good error rate performance for support arbitrary lengths. Consequently, we have to add a dummy bit to the code when IMAC isn't implemented on some code length. The transmission rate will be apparently degraded since the dummy bits contain no information. In this paper, we are proposing the new design of IMAC that can supports arbitrary code lengths. This feature can be obtained when the sub-matrix size is non-prime. The rest of this paper is organized as follows. The encoder and decoder are given in brief in the next section. Sum product algorithm (SPA) is highlighted. In section III, we detail the design of a **H** matrix that can support arbitrary block lengths. This section begins with some necessary works in **H** matrix design. In the second part we elaborate the non-prime block length design. Then in section IV, before the conclusion, we do evaluate our design compared to some published literatures.

II. LDPC CODES

The LDPC codes are generally specified by sparse parity check matrix **H** that can be represented by a Tanner graph. The graph expresses the relationship between codeword bits and parity checks bit. A parity-check matrix **H** has m rows and n columns. The codeword consists of n bits which satisfy m checks. The number of message bits will be $k = (n-m)$ and the rate of codes is $R = k/n$. The number 1's in the parity check matrix in rows and columns represents an edge between the *i-th* bit node c_i and the *j-th* check node f_j.

An example of the parity check matrix **H** of a LDPC code of dimension m, n = (3, 7) is shown as

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{bmatrix}_{(m \times n)} \quad (1)$$

The Tanner graph representing the LDPC code of the above matrix (1) can be constructed as shown in Fig. 1.

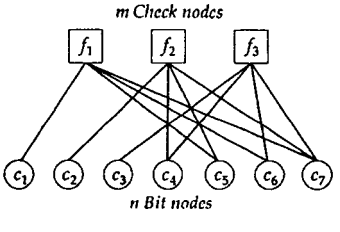


Figure 1. A Tanner graph for the matrix given in (1)

Number of 1s in the matrix reflects whether the code is regular or irregular one. In addition, the performance of the designed code is very much depended on the structure of the matrix. In the subsequent section, encoding and decoding of LDPC codes are given.

A. Encoder

Similar to other linear block codes, LDPC codes can be encoded with the relation;

$$C_{(1\times n)}H^{T}_{(n\times m)} = 0 \tag{2}$$

where C is a codeword matrix, and **H** is a parity check matrix. In a systematic form, C can be written as:

$$C_{(1\times n)} = \left[m_{(1\times m)} \mid p_{(1\times n-m)} \right] \tag{3}$$

Here $m_{(1\times m)}$ denotes the message portion, and $p_{(1\times n-m)}$ denotes the parity portion respectively.

B. Decoders

There are several methods used to decoding the LDPC codes. There are: Believe Propagation (BP), another name is Message Passing (MP) or Sum-Product (SP) [1] and Log-domain Sum-Product algorithm (log-SPA) [6]. In the Log-domain Sum-Product algorithm (SPA), the message simply passes between check nodes and bit nodes. In each pass the log likelihood ratio (LLR) [9] is recorded for is probability of its likely symbol.

The basic operation for the decoding algorithm is shown in Tanner graph in Fig 2. In the graph, the check nodes and bit nodes have interchange soft information. The variable q_{ij} is the message sent from the i^{th} bit node to j^{th} check node along a connecting edge, and r_{ji} is the message sent from j^{th} check node to the i^{th} bit node along a connection edge. The message q_{ij} is computed based on the values sent from check nodes connecting to the i^{th} bit node excluding the j^{th} check node.

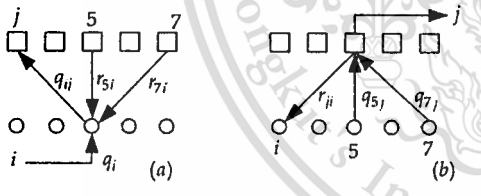


Figure 2. (a) A message is passed from *i-th* bit node to *j-th* check node and (b) A message is passed from the *j-th* check node to the *i-th* bit node.

III. PARITY CHECK MARIX DESIGN

A parity check matrix reflects directly the performance of the encoder and the corresponding decoder. Therefore, a design of a parity check matrix is one of the challenges in designing the LDPC codes. In this paper, the **H** matrix is designed by construction rows that don't have the same sub-matrix in same row. This mean that it can supports arbitrary code lengths because we can design the sub-matrix size with Non-prime. The design of the parity check matrix in this paper is based on Interleave Modified Array LDPC.

A. Some Previous Works

Eleftheriou and Olcer [4] have proposed the modified array codes (MAC) by applying cyclic shift to Fan's [7] array.

$$H = \begin{bmatrix} I & I & \cdots & I & I & \cdots & \cdots & I \\ 0 & I & \alpha & \cdots & \alpha^{(j-2)} & \alpha^{(j-1)} & \cdots & \alpha^{(k-2)} \\ 0 & 0 & I & \cdots & \alpha^{2(j-3)} & \alpha^{2(j-2)} & \cdots & \alpha^{2(k-3)} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \cdots & I & \alpha^{(j-1)} & \cdots & \alpha^{(j-1)(k-j)} \end{bmatrix} \tag{4}$$

This structure has the code rate of $R = 1-(j/k)$. MAC offers superior performance to Fan's array as it can reduce number of "1" in the lower triangle. It is simple for encoding, low error floor and capability of detecting and correcting burst error.

where I is an identity matrix of $(p\times p)$.

α is the position permutation matrix, also of $(p\times p)$.

Singhaudom[5] has proposed the interleaved modified array LDPC or IMAC by Eq. (5) by introducing the quasi cyclic matrix into the cyclic shift. This interleaved LDPC with the parity matrix given by Eq. (6). The IMAC is superior to Fan's array LDPC when the block length is particularly long.

$$H = \begin{bmatrix} I & I & I\omega & I\omega^2 & I\omega^3 & \cdots & I\omega^{j} \\ 0 & I & \alpha\omega & \alpha^2\omega^2 & \alpha^3\omega^3 & \cdots & \alpha^{(k-2)}\omega^{j} \\ 0 & 0 & I & \alpha^2\omega^2 & \alpha^4\omega^3 & \cdots & \alpha^{2(k-3)}\omega^{j} \\ \vdots & \vdots & \vdots & I & \alpha^3\omega^3 & \cdots & \vdots \\ 0 & 0 & \cdots & 0 & I & \cdots & \alpha^{(j-1)}\omega^{(k-j)} \end{bmatrix}_{(jp\times kp)} \tag{5}$$

where ω is the quasi cyclic matrix that constructed from identity matrix by cyclically-shifting of the matrix I, i.e. $\omega^{n-1} = I$.

$$I = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{(5\times 5)} \quad and \quad \omega = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{(5\times 5)} \tag{6}$$

Abematsu [8] has proposed the Size Compatible (SC) -Array LDPC Codes which the **H** matrix given in (7). The design supports arbitrary lengths achieve good error rate performance. It contains very few or no cycle-4 structure. In the SC-array LDPC code, the permutation sub-matrix is decided to eliminate all cycle-4. The proposed cyclic shift $p_{sc}(j,k)$ is expressed by Eq. (8).

$$H=\begin{bmatrix} I & I & \cdots & \cdots & I & \cdots & I \\ \alpha^{p_{sc}(2,k)} & I & \cdots & \cdots & \alpha^{p_{sc}(2,j)} & \cdots & \alpha^{p_{sc}(2,k-1)} \\ \alpha^{p_{sc}(3,k)} & \alpha^{p_{sc}(3,k-1)} & I & \cdots & \alpha^{p_{sc}(3,i)} & \cdots & \alpha^{p_{sc}(3,k-2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \cdots & \vdots \\ \alpha^{p_{sc}(j,k)} & \alpha^{p_{sc}(j,k-1)} & \cdots & \cdots & I & \cdots & \alpha^{p_{sc}(2,k-j+1)} \end{bmatrix} \quad (7)$$

where

$$p_{sc}(j,k)=(j-1)(k-1)+\left\lfloor\frac{(j-1)(k-1)}{L}\right\rfloor \quad (8)$$

This paper uses the idea of Abematsu's matrix that doesn't have the same sub-matrix in same row. It can prevent the cycle-4 when construct the **H** matrix with non-prime number. So, we will get the good performance because it has no cycle-4 structure in the **H** matrix.

B. Our Designs

The IMAC has the high code rate in particular when it is employed for long block size. One drawback is that it cannot support arbitrary code lengths. The performance of IMAC is good when designed with prime sub-matrix. But, we must add dummy bits to comply block length requirement. Non-prime sub-matrix could be designed to corporate the IMAC. However, such a straight forward implementation can result in obtaining a rather poor performance according to the appearance of cycle-4.

In our approach, we separate the design for 2 parts. The first is trying to make quasi cyclic matrix applicable to non-prime sub-matrix. The resulted modification is trying to obtain the row that contains non-identical sub-matrix. We do this by using the idea of the Size Compatible (SC)-Array LDPC Codes.

$$\omega=\begin{bmatrix} 0&1&0&0&0&0\\ 0&0&0&1&0&0\\ 0&0&0&0&0&1\\ 0&1&0&0&0&0\\ 0&0&0&1&0&0\\ 0&0&0&0&0&1 \end{bmatrix}_{(6\times6)} \quad T=\begin{bmatrix} 0&1&0&0&0&0\\ 0&0&0&1&0&0\\ 0&0&0&0&0&1\\ 1&0&0&0&0&0\\ 0&0&1&0&0&0\\ 0&0&0&0&1&0 \end{bmatrix}_{(6\times6)} \quad (9)$$

Shown in (9), the (6×6) ω matrixes make some column to 0's when we use it directly to interleave the array LDPC. It can reduce the performance of the code. So, we do left-shifting of the row 4-6 to generate a new matrix (T) with no cycle-4. Subsequently, we construct T^2 by doing right-shifting of the matrix. T^3 can be obtained similarly. They are shown in (10). Fairly obvious, $T^n=T$. The final **H** matrix then achieved as shown in (11).

In (11), we shift left Abematsu's matrix to the form of MAC to get the properties of MAC. Then, interleave the matrix with T. we show the **H** matrix for the Non-prime IMAC with non-prime sub-matrix size (j = 3, k = 12 and L = 12) by Eq. (13).

$$T^2=\begin{bmatrix} 0&0&1&0&0&0\\ 0&0&0&0&1&0\\ 1&0&0&0&0&0\\ 0&1&0&0&0&0\\ 0&0&0&1&0&0\\ 0&0&0&0&0&1 \end{bmatrix}_{(6\times6)} \quad T^3=\begin{bmatrix} 0&0&0&1&0&0\\ 0&0&0&0&0&1\\ 0&1&0&0&0&0\\ 0&0&1&0&0&0\\ 0&0&0&0&1&0\\ 1&0&0&0&0&0 \end{bmatrix}_{(6\times6)} \quad (10)$$

and

$$H=\begin{bmatrix} I & I & IT & IT^2 & \cdots & \cdots & IT^J \\ 0 & I & \alpha^{P_T(2,3)}T & \cdots & \alpha^{P_T(2,i)}T^{j-1} & \cdots & \alpha^{P_T(2,k-1)}T^J \\ 0 & 0 & I & \alpha^{P_T(3,4)}T^2 & \alpha^{P_T(3,i)}T^{j-1} & \cdots & \alpha^{P_T(3,k-2)T^J} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & \cdots & I & \cdots & \alpha^{P_T(j,k-j+1)T^J} \end{bmatrix} \quad (11)$$

where

$$p_T(j,k)=(j-1)(k-1)+\left\lfloor\frac{(j-1)(k-j)}{L}\right\rfloor \quad (12)$$

$$H=\begin{bmatrix} I & I & IT & IT^2 & IT^3 & IT^4 & IT^5 & IT^6 & IT^7 & IT^8 & IT^9 & IT^{10} \\ 0 & I & \alpha T & \alpha^2T^2 & \alpha^3T^3 & \alpha^4T^4 & \alpha^5T^5 & \alpha^6T^6 & \alpha^7T^7 & \alpha^8T^8 & \alpha^9T^9 & \alpha^{10}T^{10} \\ 0 & 0 & I & \alpha^2T^2 & \alpha^4T^4 & \alpha^6T^6 & \alpha^8T^8 & \alpha^{10}T^{10} & \alpha T^1 & \alpha^3T^3 & \alpha^5T^5 & \alpha^7T^7 \end{bmatrix} \quad (13)$$

C. Cycles of Purpose Codes

We will consider the general from of a H matrix below.

$$H=\begin{bmatrix} \alpha^{a_0,0} & \alpha^{a_0,1} & \alpha^{a_0,2} & \cdots & \alpha^{a_0,p-1} \\ 0 & \alpha^{a_1,1} & \alpha^{a_1,2} & \cdots & \alpha^{a_1,p-1} \\ 0 & 0 & \alpha^{a_2,2} & \cdots & \alpha^{a_2,p-1} \\ \vdots & \vdots & \vdots & \cdots & \cdots \\ 0 & 0 & \cdots & \cdots & \alpha^{a_{j-1},p-1} \end{bmatrix} \quad (14)$$

A cycle in the Tanner graph of purpose codes with a H matrix and block labels $a_0, a_1, \ldots, a_{j-1}$ that there exists a close path as

$$(i_1,j_1),(i_1,j_2),(i_2,j_2),(i_2,j_3),\ldots,(i_k,j_k),(i_k,j_1).$$

The length of the cycles in the permutation is described as shown below.

$$\alpha^{a_{i1},j_1}\left(\alpha^{a_{i1},j_2}\right)^{-1}\alpha^{a_{i2},j_2}\left(\alpha^{a_{i2},j_3}\right)^{-1}\ldots\alpha^{a_{ik},j_k}\left(\alpha^{a_{ik},j_1}\right)^{-1} \quad (15)$$

From the result of short cycles properties in [7], the cycles of length 4 depends on the following permutation.

$$\alpha^{a_{i1},j_1}\left(\alpha^{a_{i1},j_2}\right)^{-1}\alpha^{a_{i2},j_2}\left(\alpha^{a_{i2},j_3}\right)^{-1}=\alpha^{(a_{i1}-a_{i2})(j_1-i_2)} \quad (16)$$

which is exact when it is equal to 1. This occurs when $(a_{i1}-a_{i2})(j_1-j_2)\equiv 0$, which is clearly impossible since $i_1\neq i_2$ and $j_1\neq j_2$. Hence, there are no cycles-4 for purpose code.

IV. Performance Evaluation

We investigate our formulation by applying it to the long block length code. The 4K block length is of our interest according to its application to magnetic recording

system. The test parameters are shown in table 1 and 2 below. The codes were run for 30 iterations and the performances are observed.

TABLE 1: PARAMETERS FOR 4K BLOCK LENGTH

	j	k	p	R
Prime IMAC	5	61	67	0.918
Prime MAC	5	61	67	0.918
Non-prime IMAC	5	60	68	0.917
Non-prime MAC	5	60	68	0.917
This paper	5	60	68	0.917

TABLE 2: PARAMETERS FOR DIFFERENT NON-PRIME BLOCK LENGTH

Block Length	j	k	p	R
528	3	22	24	0.864
528	4	22	24	0.818
2016	4	42	48	0.905
4080	5	60	68	0.917

Regarding the works published by Eleftheriou and Olcer [4], and Singhaudom [5], we compare the result with that of IMAC for similar block length and code rate. The purposed matrix with non-prime IMAC has offered the similar result compared to the prime IMAC proposed by [5] and better than prime MAC proposed by [4]. BER versus E_b / N_0 plot is shown in Fig. 3.

Our proposed non-prime IMAC has superior performance over the the Non-prime MAC and Non-prime IMAC ones. Non-prime MAC and non-prime IMAC have shown rather poor performance according to generated cycle-4. The proposed matrix does not hold cycle-4 since Abematsu's matrix was interleaved by matrix T.

The modified matrix proposed in this paper was also test for different block length codes. The result is shown in Figure 4. Although we can have a desirable performance of the long codes, the purposed matrix cannot give the good result for the short block length.

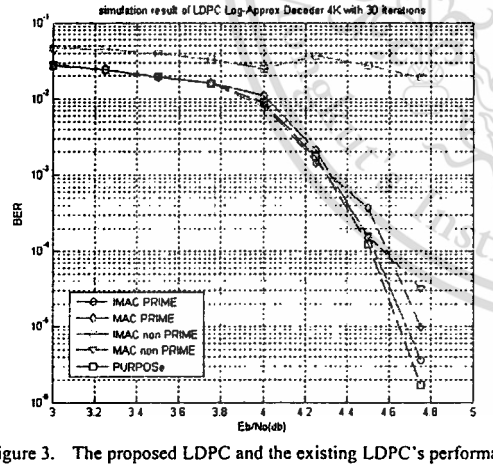


Figure 3. The proposed LDPC and the existing LDPC's performance

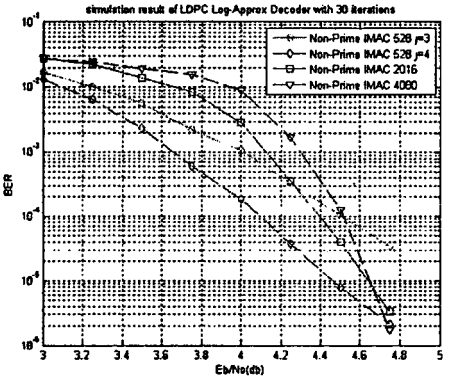


Figure 4. The proposed LDPC for different block length codes

V. CONCLUSIONS

Purpose matrix is designed by use the idea of SC-Array LDPC Codes and IMAC. We can get a parity matrix that has the good performance and can use for arbitrary code lengths in the long block length. For short block length, our idea is not good because it is not near the Shannon limit. This code has comparable performance when compared with the prime IMAC. The purpose matrix has attractive properties such as simple encoding, low error floor and capability of detecting and correcting burst error same as MAC and IMAC.

REFERENCES

[1] R. Gallager, "Low-density Parity-check Code," *IRE Trans.Information Theory*, Jan. 1962, pp.21-28.

[2] D.J.C. Mackey and R. Neal "Near Shannon limit performance of low density parity check codes," *Electronics Letters*, vol.33, pp. 457-458, Mar 1997.

[3] M. Tuchler, J. Hagenauer, E. Eleftheriou, A. Dholakia, C. Weiss, "Application of High-Rate Tail-biting Codes to Generalized Partial Response Channels," Global Telecommunications Conference, 2001

[4] E. Eleftheriou and S. Olcer, "Low-density parity check codes for digital subscriber lines," *Proc. 2002 Int. Conf. on Comm.*, pp.1752-1757., April -- May, 2002.

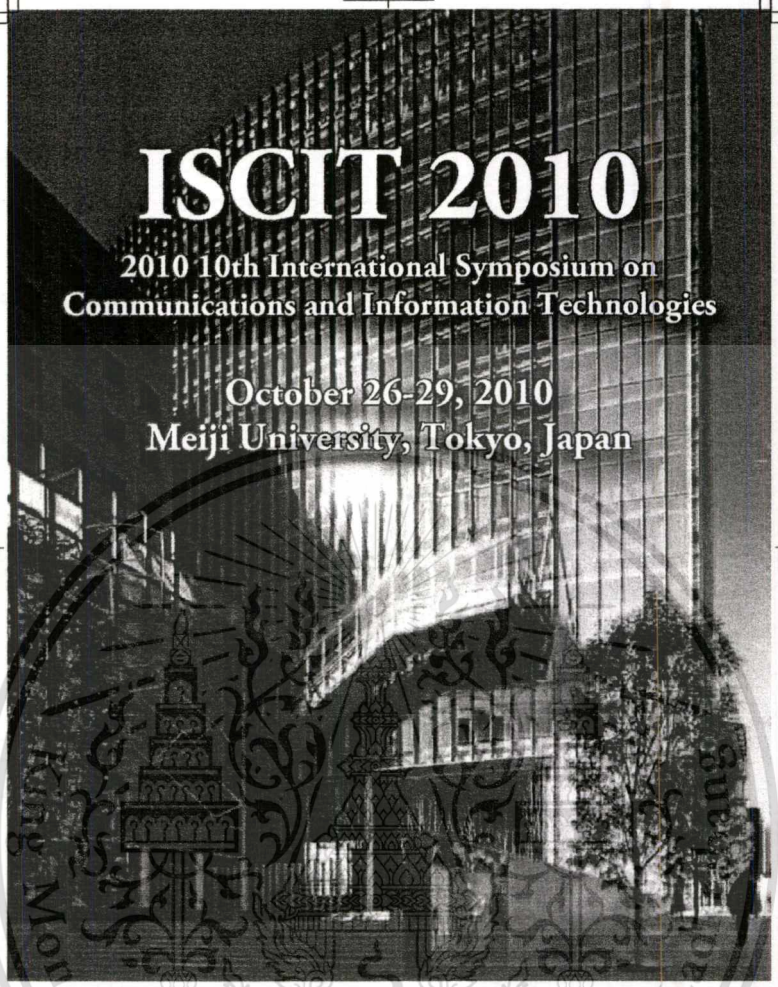
[5] W. Singhaudom, S. Noppankeepong, P. Suphithi, "Design of High-Rate Modified Array Codes for Magnetic Recording System,"*ECTI International Conference*, 2007.

[6] D.J.C. MacKay."Good error-correcting codes based on very sparse matrices". *IEEE Trans. Inform. Theory*, 45(2):pp. 399.431, March 1999

[7] J. L. Fan, "Array codes as low-density parity-check codes," in *Proc.* 2nd *Int. Symp. Turbo Codes*, Best, France, Sep 2000, pp 543-546.

[8] D. Abematsu, T. Ohtsuki, S.P.W. Jarot, T. Kashima," Size Compatible (SC)-Array LDPC Codes" *Vehicular Technology Conference*, 2007.

[9] L. Ping and W.K. Leung. "Decoding low density parity check Codes with finite quantization bits". *IEEE Comm. Letters*, 4(2):pp.62.64,February 2000.



A Design of Nonprime Block Irregular LDPC Codes via CRT

Chalit Chusin[1], Chutima Prasartkaew[2], Sekson Timakul[3], and Somsak Choomchuay[4]
[1,2,3] College of Data Storage Technology and Applications, King Mongkut's Institute of Technology Ladkrabang
BKK 10520, Thailand
[1]Tel:/Fax: + 66-3-881-5966, E-mail: ithonene@hotmail.com
[2] Tel: +66-2-329-8345 Ext.114, Fax: +66-2-329-8346, E-mail: prasartkaew@yahoo.com
[3] Tel: +66-2-329-8345 Ext.114, Fax: +66-2-329-8346, E-mail: sekson.timakul@gmail.com
[4] Faculty of Engineering, King Mongkut's Institute of Technology Ladkrabang, BKK 10520, Thailand
Tel: +66-2-329-8345 Ext.114, Fax: +66-2-329-8346, E-mail: kchsomsa@kmitl.ac.th

***Abstract*—This paper presents a new design of irregular LDPC codes that supports arbitrary block length. We propose the efficient construction method when nonprime size sub-matrices are used. The problem where $GCD(L_1, L_2) \neq 1$ that left unsolved has been tackled. We also consider the case $GCD(L_1, L_2) = 1$ but L_1 or L_2 is a nonprime number. The results show that our designed codes have superior performance compared to MAC. The resulted codes still hold the attractive properties of LDPC codes. The performances of interleaved codes are higher than noninterleaved codes and it goes higher as SNR is increased.**

***Index*— nonprime, CRT, interleave, LDPC codes, irregular**

I. INTRODUCTION

A Low Density Parity Check (LDPC) code was firstly proposed by Gallager in 1962 [1]. According to the good performance of LDPC codes that close to channel limit over AWGN channel [2] and [3], many researchers are interesting in these codes. However, in the case of LDPC codes with random H matrix, big memory size is unavoidable, in particular for the large block length code. Then, it is hard to efficiently access the memory and encoded data. A quasi cyclic structure or QC-LDPC code was taken into consideration to overcome this memory problem. With QC-LDPC codes, the storage memory size can be reduced due to their algebraic structure. The memory can be reduced by either introducing $L \times L$ circulant permutation matrices or employing zero matrices [4]. QC-LDPC codes with circulant permutation matrices are proposed by [5], [4], [6], and [7]. The performance evaluations were also made respectively.

The improved QC-LDPC codes that made use of the Chinese Remainder Theorem (CRT) has been studied lately by [4], [6] and [7]. The CRT can be used to address the problems of 4-cycle issue and/or girth reduction in case of long block length QC-LDPC coded. Reference [4] has proposed a method for constructing long length QC-LDPC codes from small length QC-LDPC codes using the CRT. And by applying the CRT method to array codes, they presented a family of high-rate regular QC-LDPC codes with no 4-cycles. Reference [6] has proposed a generalization of CRT combining method to design better QC-LDPC codes. Their results show that a BER of 10^{-6} can be obtained. Such a generalized combining method outperformed *0.5* dB of SNR compared to [4]. Recently, reference [7] has presented the remedy to two problems associated with CRT-based QC-LDPC codes: how to extend the code length code without reducing the girth, and how to design codes with a prescribed girth easily. They have proposed a method of combining QC-LDPC codes via CRT. In contrast to [4] and [6], [7] has constructed H_2 with large girth sub-matrix while H_1 still follows [8]. The resulted codes have flexible length, flexible rates, large girth and suite well iteration decoding.

Based on the prime number parameters of the matrices, [2], [8], [3] and [4] have more focused to regular LDPC codes, whilst [9] and [10] worked on irregular LDPC codes. A limitation of using of prime number parameters is that the designed codes can support only some particular code lengths. With this note, [11] has proposed regular Size Compatible (SC)-array LDPC codes using nonprime number parameters. The sub-matrices are permuted to avoid the cycle of 4. The performance of SC-array is better when compared to [8].

With above regards, the CRT has neither applied to nonprime codes nor irregular structure. In this paper we investigate the application of CRT in designing the irregular LDPC codes with nonprime block length. Although the design cannot cover prescribed girth, it suits well an arbitrary length. The rest of this paper is organized as follows: the literature reviews of the related works are given in section II. The combination of QC-LDPC codes for the longer codes using the CRT proposed by [4] is reviewed in section III. The detailed designed, the highlight of this work, is given in section IV. Examples of 4 cases are also given in this section. In section V, the designed coded are evaluated and their performance are discussed -- also in comparison with some previous works proposed by other authors. Finally, this paper is concluded in section VI.

II. RELATED WORKS

Reference [8] has introduced the array structure parity check matrix for regular LDPC codes that can offer code

performance close to a random generated parity check matrix. Other features are: low error floor and no cycle of 4, which are as same as the features of regular LDPC codes originally proposed by [1]. Fan's parity check matrix has regular structure and rewritten herewith in (1) below. This yields the code rate of $R = 1-(pj-j+1)/p^2$.

$$H(p,j,k) \triangleq \begin{bmatrix} I & I & I & \cdots & \\ I & P & P^2 & \cdots & P^{-1} \\ & & P^4 & \cdots & P^{-(j-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ I & P^{j-1} & P^{2(j-1)} & \cdots & P^{(k-1)} \end{bmatrix} \quad (1)$$

where I is an $L \times L$ identity matrix
P is a position permutation matrix.

Reference [9] has proposed the modified array codes (MAC) by applying cyclic shifting to the matrix proposed by [8]. In contrast to [8], [9] proposed an irregular LDPC. The parity check matrix of this design is given here again in (2). It offers the code rate of $R = 1-(j/k)$. Modified array codes yields superior performance compared to [8], especially for long block lengths. The reduction of number of "1" in the lower triangle of the matrix leads to simpler encoder while preserving other LDPC's features.

$$H = \begin{bmatrix} I & I & \cdots & I & I & \cdots & \cdots & I \\ 0 & I & P & \cdots & P^{(j-2)} & P^{(j-1)} & \cdots & P^{(k-2)} \\ 0 & 0 & I & \cdots & P^{2(j-3)} & P^{2(j-2)} & \cdots & P^{2(k-3)} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \cdots & & \cdots & & \end{bmatrix} \quad (2)$$

where I is an identity matrix
P are position permutation matrices.

Reference [4] has proposed a construction method of QC-LDPC codes of large length by combining QC-LDPC codes of small length as their component codes. Such a construction is based on the CRT method given by [12]. The girth of the obtained codes is always larger than or equal to that of each of the component code. By applying the similar method to array codes presented in [8], they presented a family of high-rate regular QC-LDPC codes with no 4-cycles. Simulation results show that [4] have almost the same performance as random regular LDPC codes.

Reference [6] has proposed a generalization of the CRT presented in [4]. The combining method was used to design much more and better QC-LDPC codes for given the parity check matrices of the component codes. The proposed work can be used to design a much larger class of QC-LDPC codes with similar performance by loosening the condition for determining the intermediate parameters. By permuting the block rows of the parity check matrices of the component codes, a lot of QC-LDPC codes with much less 6-cycles and better performance can be designed. At a BER of 10^{-6} some QC-LDPC codes designed by the generalized combining method outperform those designed by the CRT combining method by *0.5* dB compared to [4].

Reference [7] considered two problems associated with QC-LDPC codes. The first is how to extend the code length of a code without reducing the girth. The second is how to design a code with a prescribed girth easily. The method given in [4] and [6] cannot directly applied to construct low-rate and large girth code. Reference [7] proposed an alternative way to construct sub-matrix and used combining method presented in [4] to build a parity matrix. LDPC codes constructed with such a method have flexible lengths, flexible rates and large girth.

Reference [11] has proposed the SC-Array LDPC codes developed form the array structure of [8]. The $\boldsymbol{H}$ matrix with nonprime number parameters is given here in (3). The design supports arbitrary lengths achieve good error rate performance compared to [8]. It contains very few or no cycle-4 structure. In their SC-array regular LDPC codes, the permutation sub-matrix is decided to eliminate all cycle-4. The proposed cyclic shift $\alpha_{sc}(j,k)$ is expressed by (4).

$$H = \begin{bmatrix} I & I & \cdots & \cdots & \cdots & \\ P^{\alpha_{sc}(2,k)} & I & \cdots & \cdots & \cdots & P^{-(1)} \\ P^{\alpha_{sc}(3,k)} & P^{\alpha_{sc}(3,k-1)} & I & \cdots & \cdots & P^{(\ )} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \cdots \\ P^{\alpha_{sc}(j,k)} & P^{\alpha_{sc}(j,k-1)} & \cdots & \cdots & \cdots & P^{(j+1)} \end{bmatrix} \quad (3)$$

where

$$\alpha_{sc}(j,k) = (j-1)(k-1) + \left\lfloor \frac{(j-1)(k-1)}{L} \right\rfloor \quad (4)$$

Lately, reference [13] has proposed the design of parity check matrix to support arbitrary code length. It is based on the interleaving method suggested by [14] and the SC-array proposed by [11]. In their design, the constructed rows don't hold the same sub-matrices. The results show that even if the sub-matrix size is a nonprime number, their designed Interleaved Modified Array codes achieve the same error rate performance as the prime size sub-matrices.

The interleave method given by [13] was applied for this study as the following procedure steps.

1) The sub-matrix for interleaving can be either $\boldsymbol{T}$ or ω, where sizes of $\boldsymbol{T}$ and ω can be nonprime or prime number. The ω matrix was designed by locating '1' in the position as shown in the following matrices. Because of there are two 1's in each alternative column, then the ω matrix can be modified by a lower half left shifting and the $\boldsymbol{T}$ matrix is obtained.

$$\omega = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{6\times 6} \rightarrow T = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}_{6\times 6}$$

2) The sub-matrix, T^1 or T^2 can be obtained by right shifting the sub-matrix, T.

$$T^1=\begin{bmatrix}0&0&1&0&0&0\\0&0&0&0&1&0\\1&0&0&0&0&0\\0&1&0&0&0&0\\0&0&0&1&0&0\\0&0&0&0&0&1\end{bmatrix}_{6\times6}\rightarrow T^2=\begin{bmatrix}0&0&0&1&0&0\\0&0&0&0&0&1\\0&1&0&0&0&0\\0&0&1&0&0&0\\0&0&0&0&1&0\\1&0&0&0&0&0\end{bmatrix}_{6\times6},\ T^n=T$$

3) The interleaving is performed similar to that mentioned in [13] where the maximum exponent number of sub-matrix, T^2 (or ω^2) must be less than or equal to j.

4) Matrix element α, can be computed using CRT method given in [4]. The achieved matrix is shown in (5).

$$H=\begin{bmatrix}I&I&IT&IT^2&\cdots&\cdots&\cdot I\\0&I&P^{\alpha_T(2,3)}T&\cdots&&\cdots&\\0&0&I&P^{\alpha_T(3,4)}T^2&P^{\alpha_T(3,j)}T^{j-1}&\cdots&{}^{3,k-2)}T^j\\\vdots&\vdots&\vdots&\ddots&\vdots&\cdots&\vdots\\0&0&\cdots&\cdots&&\cdots&\end{bmatrix}\quad(5)$$

where

$$\alpha_T(j,k)=(j-1)(k-1)+\left\lfloor\frac{(j-1)(K-j)}{L}\right\rfloor\quad(6)$$

III. The Combination of QC-LDPC Codes via CRT

The parity check matrix of QC-LDPC codes shown in (7) consists of small square blocks of $L\times L$ zero matrix or circulant permutation matrices.

$$H=\begin{bmatrix}P^{\alpha_{11}}&P^{\alpha_{12}}&\cdots&P^{\alpha_{1(k-1)}}&P^{\alpha_{1k}}\\P^{\alpha_{21}}&P^{\alpha_{22}}&\cdots&P^{\alpha_{2(k-1)}}&P^{\alpha_{2k}}\\\vdots&\vdots&\cdots&\vdots&\vdots\\P^{\alpha_{j1}}&P^{\alpha_{j2}}&\cdots&P^{\alpha_{j(k-1)}}&P^{\alpha_{jk}}\end{bmatrix}_{j\times k}\quad(7)$$

Where $P^{\alpha_{mn}};(1\le m\le j,1\le n\le k)$ represents an $L\times L$ circulant permutation matrix obtained by cyclically right-shifting the identity matrix, I to the right by α_{mn} times, and $\alpha\in\{0,1,\ldots\ \infty\}$. The zero matrix of size $L\times L$ is denoted by P^{∞}. The exponent matrix, $E(H)$ of the given H in (7) is defined as

$$E(H)=\begin{bmatrix}\alpha_{11}&\alpha_{12}&\cdots&1)&\alpha_{1k}\\\alpha_{21}&\alpha_{22}&\cdots&1)&\alpha_{2k}\\\vdots&\vdots&\cdots&\vdots&\vdots\\\alpha_{j1}&\alpha_{j2}&\cdots&1)&\alpha_{jk}\end{bmatrix}\quad(8)$$

The check matrix H can be obtained by replacing each α_{mn} entry of $E(H)$ with $P^{\alpha_{mn}}$. The mother matrix, $M(H)$ of the given H is obtained by replacing zero matrix and circulant permutation matrix in H with "0" and "1", respectively. α^q_{mn} notes the exponent $(mn)^{th}$ of the matrix q.

Let $L_1, L_2, \ldots, L_t$ be pair-wise relatively prime positive integers. Let C_q be a QC-LDPC code whose binary parity check matrix, H_q with dimension of $jL_q\times kL_q$. A combining method proposed in [4] was used to construct a QC-LDPC code as the following procedures.

Step1: If $GCD(L_q,L_r)=1$ for all $q,r\le t$ and $(q\ne r)$

- If $\alpha^q_{mn}\ne\infty$ for $1\le q\le t$, then

$$\alpha_{mn}=\sum_{q=1}^{t}\alpha^q_{mn}b_qL_q^{'}\bmod L\quad(9)$$

Where

$$L=L_1L_2\ldots..L_t,\quad L_q^{'}=\frac{L}{L_q}\quad(10)$$

and $b_qL_q^{'}\equiv1\bmod L_q$. Otherwise, $\alpha_{mn}=\infty$.

Step2: The exponent matrix $E(H)$ for H is defined by $E(H)=(a_{mn})$.

Step3: The parity check matrix H can be obtained by replacing each exponent coupling of $E(H)$ with $P^{\alpha_{mn}}$.

For example, consider two QC-LDPC codes that have parity check matrices, H_1 and H_2 that $GCD(L_1,L_2)=1$, $E(H_1)=(a^1_{mn})$, $E(H_2)=(a^2_{mn})$ and $M(H_1)=M(H_2)$. When they are combined using CRT, the exponent matrix $E(H)=(c_{mn})$ is given by

$$c_{mn}=\left\{a^1_{mn}b_1L_1^{'}+a^2_{mn}b_2L_2^{'}\right\}\bmod L\quad(11)$$

where b_1 and b_2 are two integers such that $b_1L_2\equiv1\bmod L_1$ and $b_2L_1\equiv1\bmod L_2$. Then the parity check matrix, H can be obtained by exponent coupling of $E(H)$ and the $L_1L_2\times L_1L_2$ circulant permutation matrix, P where $P_{mn}=\begin{cases}1 & if\ n+1\equiv m\bmod L\\0 & otherwise\end{cases}$.

Points to be noted here; [4], [6], and [7] have designed the codes only with prime-prime number combination. Moreover, the design of $H_1, H_2, \ldots, H_t$ is quite complicate when designing nonprime–nonprime or nonprime–prime combinations. In next section we introduce the method to construct $H_1, H_2, \ldots, H_t$ that compatibles with the rules of CRT even if nonprime nonprime or non prime prime combination are the cases.

IV. Flexible Length Codes Design via CRT

The construction method with nonprime number parameters for both $GCD(L_1,L_2)=1$ and $GCD(L_1,L_2)\ne1$ will be elaborated in detail in this section. For better understanding, case examples are also given. To extend an $jL_q\times kL_q$ to $jL\times kL$, parity matrix, H can be

constructed by combining ***E(H)*** via CRT. This method can be separated into 2 cases. First case, $GCD(L_1,..,L_q)=1$, is the combination of either prime-prime such as L_7 and L_3, or prime-nonprime such as L_7 and L_6. In the second case, $GCD(L_1,..,L_q)\neq 1$, that can be either nonprime-prime such as L_6 and L_3 or nonprime-nonprime such as L_6 and L_4. The prime-prime choice is in fact the same as [4]. The design of each case is elaborated as the follows:

Case 1: Design to support $x=GCD(L_1,...,L_q)=1$

- Construct the H_1 parity matrix as (2) given in [11]
- Construct H_2 by cyclic shift down for one time and replace the 1st row with I for all a^2_{1n}
- Using CRT to combine H_1 and H_2 to form H
- Rearrange the obtained H matrix into Modified array codes structure, since Modified array codes yields attractive properties in encoding.
- Process the interleave (permutation) with the method given in (5)

Case 2: Design to support $x=GCD(L_1,...,L_q)\neq 1$

- Construct the H_1 parity matrix as (2) given in [11]
- Construct H_2 by
  - the 1st row construct with I for all a^2_{1n}
  - Construct a^2_{2n} for $2\le n\le k$ by $a^2_{2n}=(I+y)$, while $a^2_{2n}-a^1_{2n}|x$ and $y=0,1,...,x-1$
  - Construct a^2_{ij} in H_2 for $3\le m\le j$ and $2\le n\le k$ by $a^2_{mn}=a^1_{mn}+x$
  - $a^2_{mn}=I$, and $L=(\prod_{i=1}^{t}L_i)/x$
- Using CRT to combine H_1 and H_2 to form H
- Check whether matrices, H satisfy the conditions that all remainders of $a_{mn}/L_q=a^q_{mn}$ for all m,n
- Rearrange the obtained H matrix into Modified array codes structure
- Process the interleave (permutation) with the method given in (5)

In the case where $GCD(L_1,...,L_q)\neq 1$, $a^1_{mn},a^2_{mn},...,a^q_{mn}$ in each matrix, H_k must be compatible with the rules of CRT that $a^2_{mn}-a^1_{mn}|GCD(L_1,L_2)$. Therefore we can construct the H_2 with the modified version of H_1. And to be complied with CRT rules, all the remainders a_{mn} of H must satisfy the condition that $a_{mn}/L_q=a^q_{mn}$ for all m,n. Then we will show the combination matrix via CRT when $GCD(L_1,...,L_q)\neq 1$ that illustrated by the follow example

Example 1: (Prime-Prime, $GCD(L_1,L_2)=1$); To design the parity check matrix, H for 2050 codeword LDPC codes. With $j=3,k=10,L=205$, we can have $L_1=205,L_2=5$.

- Check whether $x=GCD(41,5)=1$
- Construct H_1 and H_2 as discussed above. The obtained matrices are shown in Fig.1.

	Column Index										
Row Index	H_1	1	2	3	4	5	6	7	8	9	10
	1	1	1	1	1	1	1	1	1	1	1
	2	1	1	2	3	4	5	6	7	8	9
	3	1	2	4	6	8	10	12	14	16	18

	Column Index										
Row Index	H_2	1	2	3	4	5	6	7	8	9	10
	1	1	1	1	1	1	1	1	1	1	1
	2	1	1	1	1	1	1	1	1	1	1
	3	1	1	2	3	4	1	1	2	3	4

Fig. 1. The matrix structures of H_1 and H_2

- Determine

$$L=\frac{41\times 5}{1}=205, L_1'=205/41=5, L_2'=205/5=41$$

$$b_1(5)\equiv 1 \bmod 41, b_1=33; b_2(41)\equiv 1 \bmod 5, b_2=1$$

- We then have $c_{22}\equiv\{(1\times 33\times 5)+0\} \bmod 205=165$. The obtained H matrix is given in Fig. 2.

	Column Index										
Row Index	H	1	2	3	4	5	6	7	8	9	10
	1	1	1	1	1	1	1	1	1	1	1
	2	1	165	125	85	45	5	170	130	90	50
	3	1	166	127	88	49	10	176	137	98	59

Fig. 2. The matrix structures of H (Ex.1)

- Rearrange H as shown in Fig. 3

	Column Index										
Row Index	H	1	2	3	4	5	6	7	8	9	10
	1	1	1	1	1	1	1	1	1	1	1
	2	0	1	165	125	85	45	5	170	130	90
	3	0	0	1	166	127	88	49	10	176	137

Fig. 3. The matrix structures of modified H (Ex.1)

Example 2: (Nonprime Nonprime, $GCD(L_1,L_2)\neq 1$); Consider codeword, C which has a 3×12 parity check matrix, H with $L=174$. We use two circulant permutation matrices, H_1 and H_2 with $L_1=87$ and $L_2=6$, respectively. As $GCD(H_1,H_2)=3$, we are sticking to case 2.

- Construct the H_1 parity matrix

	Column Index												
Row Index	H_1	1	2	3	4	5	6	7	8	9	10	11	12
	1	1	1	1	1	1	1	1	1	1	1	1	1
	2	1	1	2	3	4	5	6	7	8	9	10	11
	3	1	2	4	6	8	10	12	14	16	18	20	22

- H_2 is obtained with the modification of H_1

Row Index	Column Index											
	H_1	1	2	3	4	5	6	7	8	9	10	11	12
	1	1	1	1	1	1	1	1	1	1	1	1	1
	2	1	1	2	1	1	2	1	1	2	1	1	2
	3	1	5	1	3	5	1	3	5	1	3	5	1

- Use CRT to combine H_1 and H_2 to form H
  - $L = \frac{87 \times 6}{3} = 174$
  - $L_1 = 87, L_2 = 6$, $L_{11} = 29, L_{12} = 3, L_{21} = 3, L_{22} = 2$, $L_{12} = L_{21} = 3$, So, we choose L_{12} only
  - $L'_{11} = 174/29 = 6, L'_{12} = 174/3 = 58, L'_{21} = 174/2 = 86$, $b_{11}(6) \equiv 1 \bmod 29, b_{11} = 5$ $b_{12}(58) \equiv 1 \bmod 3, b_{12} = 1$ $b_{22}(87) \equiv 1 \bmod 2, b_{22} = 1$
  - $c_{22} \equiv \{(1 \times 5 \times 6) + (1 \times 1 \times 58) + (1 \times 1 \times 87)\} \bmod 174 = 1$

Row Index	Column Index											
	H_1	1	2	3	4	5	6	7	8	9	10	11	12
	1	1	1	1	1	1	1	1	1	1	1	1	1
	2	1	1	2	90	91	92	6	7	8	96	97	98
	3	1	89	91	93	95	97	99	101	103	105	107	109

- Check whether matrices, H satisfy the conditions that all remainders of $a^{'}_{mn} / L_q = a^{q}_{mn}$ for all m, n
- Rearrange the obtained H matrix into Modified array codes

Row Index	Column Index											
	H_1	1	2	3	4	5	6	7	8	9	10	11	12
	1	1	1	1	1	1	1	1	1	1	1	1	1
	2	0	1	1	2	90	91	92	6	7	8	96	97
	3	0	0	1	89	91	93	95	97	99	101	103	105

Example 3: (Prime – Nonprime, $GCD(L_1, L_2) = 1$); Let H_1 and H_2 be sub-matrices of a 3×8 parity check matrix, H with $L_1 = 43$, $L_2 = 6$ such that $L = 258$. Since $GCD(41,6) = 1$, we use case 1 to design parity check matrix, H. The resulted H of this example is shown in Fig. 4 and Fig. 5 below.

Row Index	Column Index							
	H	1	2	3	4	5	6	7
	1	1	1	1	1	1	1	1
	2	0	1	42	84	126	168	210
	3	0	0	1	43	86	129	172

Fig. 4. The matrix structures of modified H (Ex.3)

Row Index	Column Index							
	H	1	2	3	4	5	6	7
	1	1	1	1T	$1T^2$	$1T^3$	$1T^3$	$1T^3$
	2	0	1	42T	$84T^2$	$126T^3$	$168T^3$	$210T^3$
	3	0	0	1	$43T^2$	$86T^3$	$129T^4$	$172T^4$

Fig. 5. The matrix structures of interleaved H (Ex.3)

Example 4: (Nonprime–Prime, $GCD(L_1, L_2) = 1$); Choose non-prime $L_1 = 44$ for H_1 and prime $L_1 = 5$ for H_2 to construct 3×8 parity check matrix, H which $L = 220$. Case 1 is then used to construct H_1 and H_2 because $GCD(44,5) = 1$. The parity check matrices, H obtained in this example are shown in Fig.6 and Fig.7 below.

Row Index	Column Index							
	H	1	2	3	4	5	6	7
	1	1	1	1	1	1	1	1
	2	0	1	45	90	135	180	5
	3	0	0	1	46	92	138	184

Fig. 6. The matrix structures of modified H (Ex.4)

Row Index	Column Index							
	H	1	2	3	4	5	6	7
	1	1	1	IT	IT^2	IT^3	IT^3	IT^3
	2	0	1	45T	$90T^2$	$135T^3$	$180T^3$	$5T^3$
	3	0	0	1	$46T^2$	$92T^3$	$138T^4$	$184T^4$

Fig. 7. The matrix structures of interleaved H(Ex.4)

V. Performance Evaluation

To illustrate the performance of codes designed with the technique proposed in the previous section, we design the codes of 4Kbits block length with slightly high code rate, i.e. 0.81 and 0.9. The high rate is generally required by the data storage system. Designed parameters are shown in table I and II below. The channel is assumed to be AWGN. The decoder iteration limit is set to 30.

TABLE I
PARAMETERS FOR 4KBITS (R=0.81)

Type of parameter	Block Length (bits)	j	k	L
MAC: prime [9]	4179,3383	4	21	199
CRT: pri× pri [4]	4263,3451,(29,7)	4	21	203
MAC: prime [9]	4431,3587	4	21	211
CRT: nonpri× pri	4410,3570,(30,7)	4	21	210
CRT: nonpri× nonpri	4410,3570,(42,10)	4	21	210

TABLE II
PARAMETERS FOR 4KBITS (R=0.9)

Type of parameter	Block Length (bits)	j	k	L
MAC: prime [9]	4120,3708	4	40	103
CRT: nonpri× pri	4080,3672(51,2)	4	40	102
CRT: nonpri× nonpri	4080,3672(51,6)	4	40	102
CRT: nonpri× nonpri	4080,3672(51,34)	4	40	102

The available results of [9] were used to compare with our results at similar block length and code rate. Figure 8 shows the results of bit error rate (BER) performance, where $(n,k),(L_1,L_2)$ represent code length, information length, and sub-matrix size of H_1 and H_2, respectively. It can be seen that our work has superior performance compared to the original Modified array codes. The codes performance, however, seems to vary upon the selection of the pair-wise parameters. The pair of nonprime-prime yields slightly better performance compared to others.

Figure 9 shows the comparable to that offers by Modified array codes. Only one advantage we can have is that, over code can support any arbitrary length. With the consideration of interleaving and noninterleaving, as shown in Fig. 10,

interleaving can only slightly improve the performance of the code, for instance *0.1* dB at the BER of 10^{-6}.

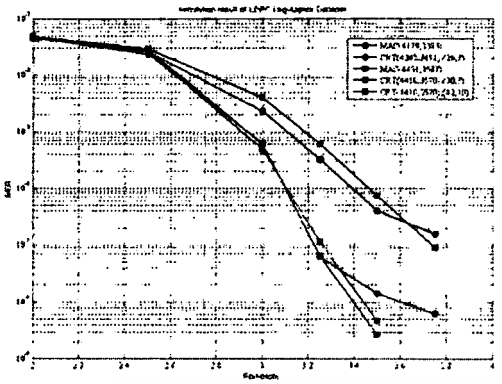


Fig. 8. Codes performance of ours and MAC's at $R = 0.81$

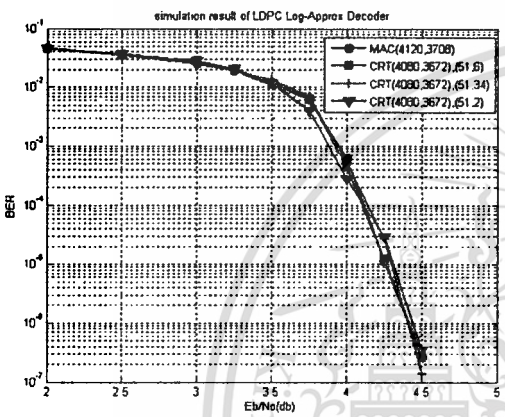


Fig. 9. Codes performance of ours and MAC's at $R = 0.9$.

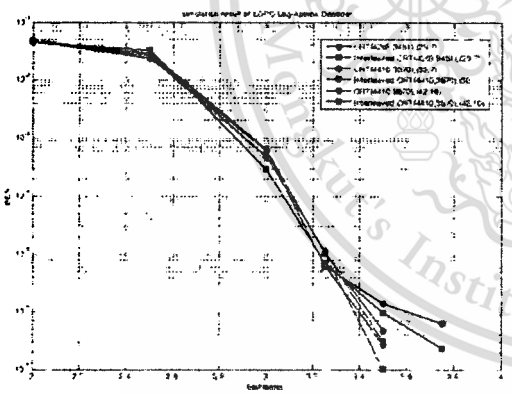


Fig. 10. Codes performance; before and after interleaving

VI. Conclusions

We propose a new design of irregular LDPC codes that can support any block length. Using the idea of Chinese Remainder Theorem, we introduced a method to extend the shorten array code length to longer code length without reducing the code performance. Codes constructed with our method differ from [4], [6], [7] in such a way that our codes can combine the matrix $\boldsymbol{H}_1$ and $\boldsymbol{H}_2$ when $GCD(L_1, L_2) \neq 1$. With this property, the code can support the arbitrary lengths because we can design matrix $\boldsymbol{H}_1$ and $\boldsymbol{H}_2$ with any sub-matrices size [11]. Our design has quite complicated compared to [6] and [7] but it doesn't generate large girth like [7]. In additions the codes still have the attractive properties of LDPC codes (such as simple encoding, low error floor and capability of detecting and correcting burst error) since the purposed matrices hold the structure as modified array codes . Our results show that even though there are some cycle-4 in $\boldsymbol{H}_2$, they do not affect on the codes performance. Our codes have superior performance compared to modified array codes. Furthermore, we used the interleave method to improve the codes performance. The results show that the performance of interleaved codes is slightly higher than non-interleaved codes and it will be much higher, when the SNR is more increased. Of its high rate, the codes are ideal for magnetic recording system where 4Kbytes block length are of interest.

References

[1] R. G. Gallager, "Low-Density parity-check Codes," *IRE Trans.Inform. Theory*, pp. 21-28, Jan. 1962.

[2] D. J. C. Mackay and R. M. Neal, "Near Shannon limit performance of low-density parity-check codes," *Electronics Lett.*, vol. 33, pp. 457-458, Mar. 1997.

[3] Chung, S.-Y., Forney, G. D., Jr. Richardson, T. J., and Urbanke, R. L., "On the design of low-density parity-check codes within 0.0045 dB of the shannon limit," *Electronics Lett.*, vol. 5, pp. 58-60, Feb. 2001.

[4] S. Myung and K. Yang, "A combing method of quasi-cyclic LDPC codes by the Chinese Remainder Theorem," *IEEE Comm. Lett.*, vol. 9, pp. 823-825, Sept. 2005.

[5] M. P. C. Fossorier, "Quasi-cyclic low-density parity- check codes from circulant permutation matrices," *IEEE trans. Inform. Theory*, vol. 50, No. 8, pp. 1788-1793, Aug. 2004.

[6] Y. Liu, X. Wangm, R. Chen, and Y. He, "Generalized Combining Method for Design of Quasi-Cyclic LDPC Codes," *IEEE comm. Lett.*, Vol. 12, No.5, pp. 392-394, May, 2008.

[7] X. Jiang and M. H. Lee, "Large Girth Quasi-Cyclic LDPC Codes Based on the Chinese Remainder Theorem," *IEEE comm. Lett.*, Vol. 13, No. 5, pp. 342-344, May, 2009.

[8] J. L. Fan, "Array Codes as Low-Density Parity-Check Codes," in *Proc.* 2nd *Int. Symp. on Turbo Codes*, Brest, France, Sept. 4-7, 2000, pp. 543-546.

[9] E. Eleftheriou and S. Olcer, "Low-density parity check codes for digital subscriber lines," *Proc. 2002 Int. Conf. on Comm.*, pp. 1752-1757, April - May, 2002.

[10] C. Prasartkaew and S. Choomchuay, "A Parity Check Matrix Design for Irregular LDPC Codes with 2K Block Length," *ISPACS Int. Symp. on Intelligent Signal Processing and Comm. Systems*, Dec. 7-9, 2009.

[11] D. Abematsu, T. Ohtsuki, S. PW Jarot, and T. Kashima, "Size Compatible (SC)-Array LDPC Codes," *IEEE Vehicular Technology Conference 2007*, pp. 1147-1151, 2007.

[12] K. H. Rosen, Elementary Number Theory and Its Applications. Reading, MA: Addition-Wesley, pp. 322-324, 2000.

[13] C. Chusin, C. Prasartkaew and S. Choomchuay, "A Design of Non-prime LDPC Based on Interleave Modified Array Codes", *Inte. Data Storage Technology Conf. DST-CON 2010*, July 2010.

[14] W. Singhaudom, S. Noppankeepong, P. Suphithi, "Design of Hign-Rate Modified Array Codes for Magnetic Recording System," *ECTI International Conference*, May 2007.

Biography

Personal Information

Name & Surname: Mr. Chalit Chusin
Nationality: Thai
Birth of date: December 02nd, 1983
Place of birth: Bangkok, Thailand

Education

Bachelor degree

Field: Electronics & Telecommunication Engineering
Duration: 2002 – 2006
Institute: Department of Electronics & Telecommunication Engineering, Faculty of Engineering, King Mongkut's University of Technology Thonburi, Thailand

Master degree

Field: Data Storage Technology
Duration: 2008 – 2013
Institute: College of Data Storage Innovation, Faculty of Engineering, King Mongkut's Institute of Technology, Ladkrabang, Thailand

Research Interests

Low density parity check code, Non binary Low density parity check code, Hard disk drive technology, Read channel in Hard disk drive.

List of International Conference and Proceeding Papers

1. Chalit Chusin , Chutima Prasartkaew and, Somsak Choomchuay, "*Non-Prime Parameters of LDPC Codes with Symmetrical Sub-matrix*", ITC-CSCC, July 4-7, 2010, Pattaya, Thailand.
2. Chalit Chusin , Chutima Prasartkaew and, Somsak Choomchuay, "*A Design of Non-prime LDPC Based on Interleave Modified Array Codes*", DST-CON, July 30 – August 1, 2010, Bangkok, Thailand.
3. Chalit Chusin, Chutima Prasartkaew, Sekson Timakul, and Somsak Choomchuay, "*A Design of Nonprime Block Irregular LDPC Codes via CRT*", ISCIT 2010, October 26-29, 2010, Tokyo, Japan