



January 2017

Multilevel Orthogonal Coded Modulation

Sunil Kumar Gaire

Follow this and additional works at: <https://commons.und.edu/theses>

Recommended Citation

Gaire, Sunil Kumar, "Multilevel Orthogonal Coded Modulation" (2017). *Theses and Dissertations*. 2216.
<https://commons.und.edu/theses/2216>

This Thesis is brought to you for free and open access by the Theses, Dissertations, and Senior Projects at UND Scholarly Commons. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of UND Scholarly Commons. For more information, please contact zeinebyousif@library.und.edu.

MULTILEVEL ORTHOGONAL CODED MODULATION

by

Sunil Kumar Gaire

Bachelor in Electronics & Communication Engineering, 2007

A Thesis

Submitted to the Graduate Faculty

of the

University of North Dakota

in partial fulfillment of the requirements

for the degree of

Master of Science

Grand Forks, North Dakota

August

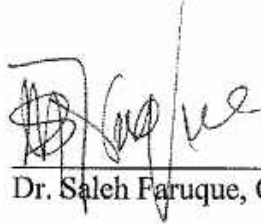
2017

Copyright 2017 Sunil Kumar Gaire

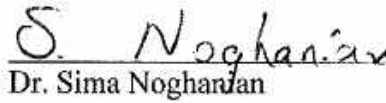
And

Dr. Saleh Faruque

This thesis, submitted by Sunil Kumar Gaire in partial fulfillment of the requirements for the Degree of Master of Science from the University of North Dakota, has been read by the Faculty Advisory Committee under whom the work has been done and is hereby approved.



Dr. Saleh Faruque, Chairperson

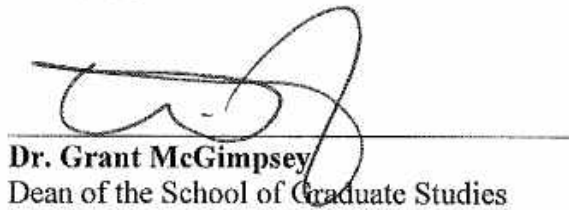


Dr. Sima Noghanian



Dr. Praksah Ranganathan

This thesis is being submitted by the appointed advisory committee as having met all of the requirements of the School of Graduate Studies at the University of North Dakota and is hereby approved.



Dr. Grant McGimpsey
Dean of the School of Graduate Studies

July 24, 2017
Date

TABLE OF CONTENTS

LIST OF FIGURES	ix
LIST OF TABLES	xii
ABBREVIATION	xiv
ACKNOWLEDGMENTS	xvii
ABSTRACT	xviii
CHAPTER	
1. INTRODUCTION	1
1.1 Research Objectives, and Contribution	2
1.2 Outline of Thesis	3
2. BACKGROUND	4
2.1 Fundamental of Digital Communication Systems	4
2.2 Shannon's Channel Capacity	6
2.3 Power Limited and Bandwidth Limited Systems	9
2.4 Bandwidth Efficiency of Modulated Systems	10
2.5 Channel Model	10
2.6 Error Control Coding	11
2.6.1 Automatic-Repeat Request (ARQ) Schemes	11

2.6.2	Forward Error-correction (FEC) Codes.....	12
2.6.2.1	Block Codes	12
2.6.2.2	Convolution Coding	15
2.6.2.3	Turbo Code	16
2.6.2.4	Low-density Parity Check (LDPC) Code	16
2.7	Multilevel Coded Modulations	17
2.7.1	Introduction	17
2.7.2	Multistage Decoding	20
2.7.3	Parallel Independent Decoding at Individual Level (PDL)	20
3.	<i>M</i>-ary PSK AND <i>M</i>-ary QAM MODULATION SCHEMES	22
3.1	<i>M</i> -ary Phase Shift Keying (<i>M</i> -ary PSK)	22
3.1.1	Introduction	22
3.1.2	Power Spectra of <i>M</i> -ary PSK Signals	24
3.1.3	Transmission Bandwidth and Bandwidth Efficiency	24
3.1.4	Error Probability of <i>M</i> -ary PSK	25
3.2	<i>M</i> -ary Quadrature Amplitude Modulation (<i>M</i> -ary QAM)	27
3.2.1	Introduction	27
3.2.2	Transmission Bandwidth and Bandwidth Efficiency	29
3.2.3	Error Probability of <i>M</i> -ary QAM	30
3.3	Comparison of Error Probabilities of <i>M</i> -ary PSK and <i>M</i> -ary QAM ..	31
4.	ERROR CONTROL USING ORTHOGONAL CODES	33

4.1	Orthogonal Codes	33
4.2	Generation of Orthogonal Codes	33
4.3	Properties of Orthogonal Codes	34
4.4	Bi-orthogonal Codes	34
4.5	Some Applications of Orthogonal Codes	35
4.6	Error Correction Capability	36
5.	MULTILEVEL ORTHOGONAL CODED MODULATION	39
5.1	Orthogonal Coded Modulation	39
5.2	Multilevel Orthogonal Coded Modulation	41
5.2.1	Different Code Rate Encoder Structures.....	42
5.2.1.1	Rate $\frac{1}{2}$ Single Level Orthogonal Coded Modulation..	42
5.2.1.2	Rate $\frac{1}{2}$ Multilevel Orthogonal Coded Modulation	43
5.2.1.3	Rate $\frac{3}{4}$ Multilevel Orthogonal Coded Modulation...	44
5.2.1.4	Rate 1 Multilevel Orthogonal Coded Modulation.....	46
5.2.2	Construction Parameters for Higher Length Orthogonal Codes	47
5.3	Parallel Independent Decoding at Individual Level for Multilevel Orthogonal Coded Modulation.....	48
5.4	Error Correction Capability for Different Code Rate	49
6.	PERFORMANCE OF MULTILEVEL ORTHOGONAL CODED MODULATION IN DIFFERENT MODULATION TECHNIQUES	51
6.1	Orthogonal Coded M -ary PSK Modulation (OMPSK) for Optical Wireless Communications	51

6.1.1	Block Diagram of <i>OptoRadio</i> System.....	52
6.1.2	Multilevel OMPSK Encoder Structure	54
6.1.3	Transmission Bandwidth and Bandwidth Efficiency.....	56
6.1.4	Error Performance of OMPSK	57
6.1.5	Simulated Results	60
6.2	Multilevel Orthogonal Coded <i>M</i> -ary QAM Modulation (OMQAM)..	69
6.2.1	Block Diagram of Multilevel OMQAM System.....	69
6.2.2	Transmission Bandwidth and Bandwidth Efficiency	71
6.2.3	Error Performance of OMQAM	72
6.2.4	Simulation Results	75
6.2.4.1	Rate $\frac{3}{4}$ Multilevel Orthogonal Coded <i>M</i> -ary QAM Encoder	75
6.2.4.2	Rate $\frac{1}{2}$ Multilevel Orthogonal Coded <i>M</i> -ary QAM Encoder	79
6.2.4.3	Rate 1 Multilevel Orthogonal Coded 16-QAM Encoder	82
6.2.4.4	Multilevel Orthogonal Coded 256-QAM Modulation	85
7.	CONCLUSION AND FUTURE WORK	87
7.1	Conclusion	87
7.2	Future Work	88
	APPENDICES	90
	REFERENCES	126

LIST OF FIGURES

Figure	Page
2.1 Block diagram of digital communication system.....	4
2.2 Channel capacity curve for AWGN channel.....	8
2.3 AWGN channel.....	10
2.4 Systematic block code.....	13
2.5 Convolution encoder.....	15
2.6 Multilevel encoding scheme.....	19
2.7 Multistage decoding.....	20
2.8 Parallel independent decoding for MLC scheme.....	21
3.1 Constellation diagram for different values of M for M -PSK modulation.....	24
3.2 Power spectra of M -ary PSK signals for $M= 2, 4, 8$ and 16	25
3.3 Bit Error rate of M -ary PSK for different values of constellation points M	27
3.4 Rectangular constellation diagram for different values of M for M -ary QAM modulation.....	29
3.6 Bit Error rate of M -ary QAM for different values of constellation points M	31
4.1 Bi-orthogonal code set combining orthogonal and antipodal codes.....	35
4.2 Distance property of orthogonal codes.....	37
5.1 General block diagram of orthogonal coded modulation transmitter.....	39
5.2 General block diagram of orthogonal coded modulation receiver.....	40

5.3	MOCM encoding structure.....	41
5.4	Rate $\frac{1}{2}$ orthogonal coded modulation with $n = 8$ bits.....	43
5.5	Rate $\frac{1}{2}$ multilevel orthogonal coded modulation with $n = 16$ bits.....	44
5.6	Rate $\frac{3}{4}$ multilevel orthogonal coded modulation with $n = 8$ bits.....	45
5.7	Rate $\frac{3}{4}$ multilevel orthogonal coded modulation with $n = 16$ bits.....	46
5.8	Rate 1 multilevel orthogonal coded modulation with $n = 8$ bits.....	47
5.9	Multilevel correlation decoding for rate $\frac{3}{4}$ with $n = 8$	48
5.10	Relation showing the number of errors corrected for different code lengths.....	50
6.1	Block diagram of <i>OptoRadio</i> implementing OMPSK.....	52
6.2	The ambient light cancellation technology.....	53
6.3	Rate $\frac{1}{2}$ multilevel orthogonal coded M -ary PSK modulation with $n = 8$	55
6.4	Theoretical SER plot of (a) coded (8-bit) QPSK with code rate $\frac{3}{4}$ and (b) coded (16-bit) QPSK with code rate $\frac{1}{2}$, compared with uncoded QPSK.....	59
6.5	Theoretical SER plot of coded (8-bit) and uncoded 16-PSK using AWGN channel	60
6.6	Error performance of uncoded QPSK and rate $\frac{3}{4}$ coded QPSK with code lengths $n = 8$ and 16 bits.....	61
6.7	Scatter plots of QPSK for various values of E_b/N_0	63
6.8	Error performance of uncoded QPSK and rate $\frac{1}{2}$ coded QPSK with code length $n = 16$ bits	64
6.9	Error performance of uncoded QPSK and rate $\frac{1}{2}$ and rate $\frac{3}{4}$ coded QPSK with code length $n = 16$ bits	65
6.10	Error performance of uncoded 16-PSK and rate 1 coded 16-PSK with code Lengths $n = 8$ and 16 bits	66
6.11	Scatter plot of 16-PSK for various values of E_b/N_0	69
6.12	General block diagram of the Orthogonal Coded M -ary QAM (OMQAM) system with transmitter and receiver	70

6.13	Theoretical error probability of uncoded 4-QAM and (a) rate $\frac{3}{4}$ with 8-bit orthogonal coded 4-QAM, (b) rate $\frac{1}{2}$ with 16-bit orthogonal coded 4-QAM	74
6.14	Theoretical plot of 16-QAM without coding and 8-bit orthogonal coding	74
6.15	Error performance of rate $\frac{3}{4}$ orthogonal coded 4-QAM with $n=8$ and $n=16$ and comparison with uncoded 4-QAM	77
6.16	Scatter plots showing the effect of AWGN for different values of E_b/N_0 for 4-QAM modulation using rate $\frac{1}{2}$ eight bit orthogonal coding.....	78
6.17	Error performance of rate $\frac{1}{2}$ orthogonal coded 4-QAM with $n = 16$ and comparison with uncoded 4-QAM	79
6.18	Error performance of rate $\frac{1}{2}$ orthogonal coded 16-QAM with $n = 32$	80
6.19	Comparison of error performance of rate $\frac{1}{2}$ and rate $\frac{3}{4}$ orthogonal coded 4-QAM with $n = 16$	81
6.20	Error performance of rate 1 orthogonal coded 16-QAM with orthogonal code lengths of 8-bit and 16-bit	84
6.21	16-QAM scatter plots for signal to noise ratios of 0, 5, 10 and 14 dB respectively	85
6.22	Error performance of 256-QAM with different code lengths and code rates	86

LIST OF TABLES

Table	Page
3.1 Transmission bandwidth and bandwidth efficiency of PSK for various values of M	26
3.2 Bandwidth efficiency of QAM for different values of M	30
3.3 SNR advantage of M -ary QAM over M -ary PSK	32
3.4 Modulation types and their required E_b/N_0 for BER of 10^{-6} and 10^{-3} , and minimum bandwidth required for intersymbol interference (ISI) free signaling	32
4.1 Error correction capabilities of orthogonal codes	38
5.1 Orthogonal code mapping table for (a) $n = 4$ and (b) $n = 8$	40
5.2 Different code rates construction parameters for different orthogonal code lengths	48
5.3 Orthogonal codes lengths and their error correction capabilities for different rates.....	50
6.1 OMPSK transmission bandwidths and bandwidth efficiencies.....	56
6.2 Simulation parameters	60
6.3 Number of bit errors due to AWGN for rate $\frac{3}{4}$, 8-bit orthogonal coded QPSK before and after decoding for different values of E_b/N_0	61
6.4 Number of bit errors due to AWGN for rate $\frac{3}{4}$, 16-bit orthogonal coded QPSK before and after decoding for different values of E_b/N_0	62
6.5 Number of bit errors for rate $\frac{1}{2}$, 16-bit orthogonal coded QPSK before and after decoding for different values of E_b/N_0	64
6.6 Number of bit errors for rate 1, 8-bit orthogonal coded 16-PSK before and after decoding for different values of E_b/N_0	66
6.7 Number of bit errors for rate 1, 16-bit orthogonal coded 16-PSK before and after decoding for different values of E_b/N_0	67

6.8	Transmission bandwidths and bandwidth efficiencies of multilevel OMQAM scheme	71
6.9	Number of bit errors for rate $\frac{3}{4}$, 8-bit orthogonal coded 4-QAM before and after decoding for different values of E_b/N_0	76
6.10	Number of bit errors for rate $\frac{3}{4}$, 16-bit orthogonal coded 4-QAM before and after decoding for different values of E_b/N_0	76
6.11	Number of bit errors for rate $\frac{1}{2}$, 16-bit orthogonal coded 4-QAM before and after decoding for different values of E_b/N_0	80
6.12	Number of bit errors for rate $\frac{1}{2}$, 32-bit orthogonal coded 16-QAM before and after decoding for different values of E_b/N_0	81
6.13	Number of bit errors for rate 1, 8-bit orthogonal coded 16-QAM before and after decoding for different values of E_b/N_0	83
6.14	Number of bit errors for rate 1, 16-bit orthogonal coded 16-QAM before and after decoding for different values of E_b/N_0	83

ABBREVIATIONS

A/D	Analog to Digital Conversion
ACK	Acknowledgment
ARQ	Automatic-Repeat Request
AWGN	Additive White Gaussian Noise
BER	Bit Error Rate
BPSK	Binary Phase Shift Keying
CDMA	Code Division Multiple Access
D/A	Digital to Analog Conversion
DC	Direct Current
HDTV	High-Definition Television
HSDPA	High-Speed Download Packet Access
HSUPA	High-Speed Uplink Packet Access
IEEE	Institute of Electrical and Electronics Engineers

ISI	Inter Symbol Interference
LDPC	Low-Density Parity Check
LOS	Line of Sight
LTE	Long Term Evolution
<i>M</i> -ary PSK	<i>M</i> -ary Phase-Shift Keying
<i>M</i> -ary QAM	<i>M</i> -ary Quadrature Amplitude Modulation
MFSK	Multi Frequency Shift Keying
NACK	Negative acknowledgment
NRZ	Non-Return to Zero
OOK	On-Off Keying
OMPSK	Orthogonal <i>M</i> -ary Phase-Shift Keying
OMQAM	Orthogonal <i>M</i> -ary Quadrature Amplitude Modulation
PAM	Pulse Amplitude Modulation
PDL	Parallel Independent Decoding at Individual Level
RF	Radio Frequency
SER	Symbol Error Rate
S/N, SNR	Signal to Noise Ratio

WER

Word Error rate

WLAN

Wireless Local Area Network

ACKNOWLEDGMENTS

First of all, I would like to express my sincere gratitude to the Department of Electrical Engineering at University of North Dakota, for giving me the opportunity to peruse master's degree. My deepest gratitude goes to my advisor, Dr. Saleh Faruque, for his support and guidance throughout my research during my time in master's program. I would also like to express my sincere appreciation to the thesis committee members, Dr. Sima Noghianian, and Dr. Prakash Ranganathan, for their support and valuable comments.

Last, but not the least, I would like to thank my family for their continuous love and support.

To my family

ABSTRACT

Multilevel orthogonal coded modulation (MOCM) technique is a combination of orthogonal channel coding and M -ary modulation, where two or more codes are used simultaneously to detect and correct multiple errors with bandwidth efficiency. The concept, principles, and simulations of the multilevel orthogonal coding combined with M -ary modulations is presented in this thesis. In multilevel orthogonal coding, both information and orthogonal code blocks are split into multiple levels. The information data in each level are mapped to the corresponding orthogonal codes. The outputs of the encoders are then grouped to form symbols suitable for input to spectrally efficient multilevel modulations. The modulated symbols are then transmitted to the channel. At the receiver, the decoding is performed using the correlative decoder. The correlative decoding is also performed in multiple levels independently. At each level of decoding, the incoming orthogonal codes with errors are cross-correlated with the known set of orthogonal code blocks. Then the code set that gives the smallest correlation value, at each level, is chosen as the desired orthogonal code output. The implementing of multilevel encoding and decoding structure increases the error correcting capability of the system significantly, allowing multiple numbers of error corrections. Also, error correction capability of orthogonal codes improves with increasing the code lengths. Combining longer codes in multilevel structures the system shows better error performance. Additionally, the multilevel encoding outputs are suitable for performing multilevel modulation (M -ary modulation), which allows the

transmission of a large number of bits per symbol. This makes the system to have minimum transmission bandwidths that increase the throughput.

The encoding and decoding structures for three code rates: rate $\frac{1}{2}$, rate $\frac{3}{4}$, and rate 1, for different orthogonal code lengths are presented and simulated. The transmission bandwidths, bandwidth efficiencies are calculated and error performance analyses of systems are conducted. M -ary Phase Shift Keying (PSK) and M -ary Quadrature Amplitude Modulation (QAM) modulation techniques are chosen as modulation techniques to simulate and analyze the performances in MATLAB. To show the application of MOCM, multilevel orthogonal coded M -ary PSK modulation transmitted in optical wireless communication systems using ambient light cancellation technology is also presented. To simulate the error performances, additive white Gaussian noise channel is used at various signal-to-noise ratios (SNR). The number of errors before decoding and after decoding is counted and tabulated. The number of bit error corrected at specified signal to noise ratio is also tabulated. Error performance curves are plotted and coding gains are observed. The results show the performance improvement of the systems compared to uncoded systems. Also, the coded systems are bandwidth efficient. Thus, the multilevel orthogonal coded modulation systems provide higher error correction capability with bandwidth efficiency.

CHAPTER 1

INTRODUCTION

The demand for efficient and reliable digital data transmission and storage system has been increasing significantly day by day. This demand has been speed up by emergence of large scale, high-speed data networks for the exchange, processing and storage of digital information in the commercial, governmental, and military applications. Thus, the major concern of data transmission and storage system designers is to control the errors during the transfer of the data from source to destination so that data can be reliably reproduced at the destination [1]. Channel encoding and decoding process are inseparable part of modern communication applications to cope with the errors that occur during transmission. The forward error correction coding or channel encoding/decoding performs the error detection and correction with the addition of redundant bits to the information data streams. The addition of redundant bits, however, increases the available bandwidth. Since the bandwidth is scarce and limited, the coding technique that minimizes the transmission bandwidth requirements is highly desirable.

In this thesis, multilevel orthogonal channel coding combined with M -ary PSK and QAM modulation techniques are presented. The encoding and decoding structures of multilevel orthogonal coded modulation (MOCM) are less complex in comparison with other coded modulation techniques. Additionally, this coded modulation format will allow high transmission data rates with larger number of error correction capability. Thus is suitable for different wired and wireless communication system applications.

1.1. Research Objectives and Contribution

The objectives of this research are:

1. To provide conceptual principles on MOCM system with parallel independent correlative decoding at individual level.
2. To verify using simulation, the theoretical analysis of the error performance of the MOCM systems under the additive white Gaussian noise (AWGN) channel.
3. To develop MOCM with bandwidth efficient multilevel modulation schemes and demonstrates one of its applications in optical communication systems.

The thesis contributes on demonstrating the conceptual principal of MOCM. In developing conceptual theory, the transmission bandwidths and bandwidth efficiencies of the systems are calculated. Further, theoretical error performance analyses are also presented. Finally, simulations are conducted to visualize the error performances and achievable coding gains.

Different encoding and decoding structures of multilevel orthogonal channel coding combined with bandwidth efficient modulation schemes with different code lengths and code rates are presented. Three different code rates, rate $\frac{1}{2}$, rate $\frac{3}{4}$ and rate 1, with various code lengths are used. The appropriate constellation points ' M ' were chosen for carefully chosen modulations.

One of the applications of MOCM in optical wireless communication systems using M -ary PSK modulations with ambient light cancellation technique is also presented. Similarly, the theoretical analyses, encoding and decoding structures, bandwidth calculations and error performances of MOCM implemented with M -ary QAM are demonstrated, simulated and results are analyzed.

1.2. Outline of Thesis

This thesis is organized as follows. Chapter 2 provides the basic background knowledge necessary for the understanding of this thesis. Chapter 3 provides the discussion of M -ary PSK and M -ary QAM with transmission bandwidths and error probability. Chapter 4 explains the concept of error control using orthogonal codes. Chapter 5 details the description of MOCM under different code rates encoding schemes. The performance of multilevel orthogonal coded modulations with M -ary PSK for optical wireless communications, and M -ary QAM modulations are presented in Chapter 6. This chapter also presents the MATLAB simulation results including the error correction capabilities and error performance analyses. Chapter 7 presents the conclusion and recommendations for further work. Lastly the implemented MATLAB coding is provided in Appendices.

CHAPTER 2

BACKGROUND

In this chapter, some backgrounds on digital communication systems implementing the error control coding are discussed. Basic concepts of digital communication systems, modulations and demodulations, channel encoders, and decoders are explained briefly. In addition to that, Shannon's channel capacity, power limited and bandwidth limited systems, bandwidth efficiency of modulated systems, and different error correction coding schemes, are also presented in short.

2.1. Fundamental of Digital Communication Systems

General block diagram of digital communication system, shown in Figure 2.1, consists of three main components: transmitter, channel, and receiver.

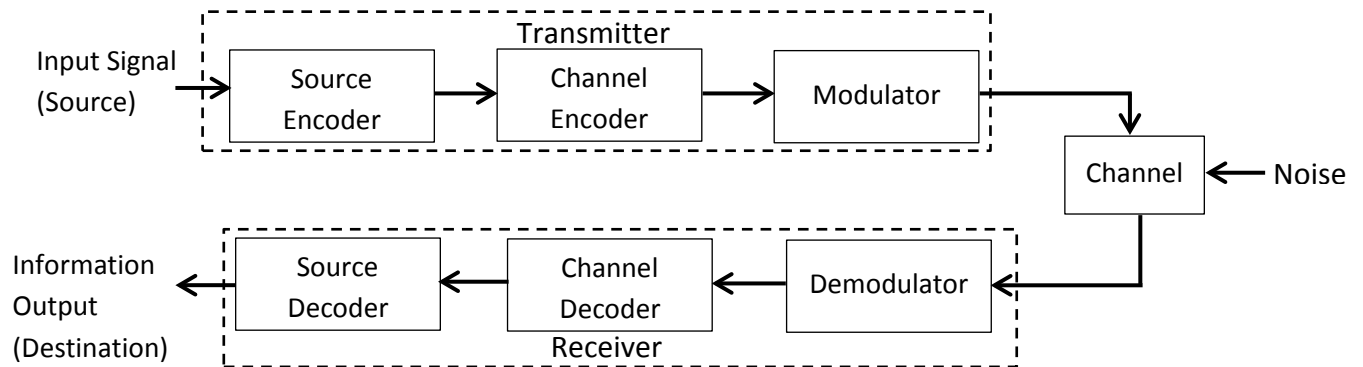


Figure 2.1 Block diagram of a digital communication system.

The input signal from the source may be voice, music or data. The source encoder performs the sampling, quantization and analog to digital (A/D) conversion processes. During this process the

analog signal is sampled: converting the signal to discrete in time and continuous in amplitude. The sampled signal is then quantized to fixed quantization levels. Then, these levels are converted into digital codewords. The source coding limits the occupied bandwidth. Techniques such as filtering, Non-Return-to-Zero (NRZ) signals are used to limit bandwidth occupancy. In this thesis, the input bit is directly taken as digital data; so, source encoding is not discussed.

The channel encoder enhances the reliability and efficiency of the digital signal transmission, and is responsible for error detection and correction. It adds controlled redundancy to the source code to produce a new stream of data bits longer than source codeword. The redundant bit added by the channel encoder doesn't carry any information but helps the receiver to detect and in most cases, correct the errors in the received message. The coded data are modulated by the modulator, which represents each set of codeword by an appropriately selected analog waveform, suitable for transmission over the communication channel. The modulated signal is then transmitted by the wired or wireless communication channel. During the transmission of the signal in the channel, information signal is often affected by noise, attenuation, interference, fading and distortions.

The demodulator at the receiver converts received modulated analog waveform into sequence of bits with minimum error. Channel decoder recovers information bearing bit sequence from the bit sequence recovered by the demodulator, by removing the redundant bit that were added in channel coding. It detects and corrects errors that are introduced in the signal during transmission. Source decoder recovers the information signal from the sequence of bits from the channel decoding. If the destination is analog, digital to analog (D/A) conversions are performed in this block. Finally, the signal transmitted from the source is utilized at the destination.

2.2 Shannon's Channel Capacity

The Shannon-Hartley theorem states that for an additive white Gaussian noise (AWGN) channel the maximum capacity is given by

$$C = B \log_2 \left(1 + \frac{S}{N} \right) \quad (1)$$

or

$$C = 3.32 B \log_{10} \left(1 + \frac{S}{N} \right) \quad (2)$$

where C is the capacity of channel in bits per second called 'Shannon's capacity limit', B is the bandwidth of the channel in Hertz, and $\frac{S}{N}$ is the signal to noise ratio. This channel capacity equation gives the number of bits that can be transmitted per second without error over a channel of bandwidth B Hz, when the signal power is limited to S Watts, and is exposed to AWGN. It also explains that, theoretically it is possible to transmit information over such channels with an arbitrarily small error probability at any rate R , where $R \leq C$, by using proper coding scheme. On the other hand, if $R > C$, error-free transmission is not possible whatever the signal processing or coding performed at the transmitter and receiver. Equation (1) gives the fundamental maximum transmission capacity that can be achieved on a channel given by any combination of coding scheme, transmission or decoding scheme, and is the best performance limit that can be achieved. This theorem clearly states what is achievable, while not explaining how it can be achieved [2].

Signal to noise ratio (SNR or S/N), is often represented as information bit energy to noise power spectral density ratio, E_b/N_0 , which allows systems with different modulations or coding schemes to be compared in fair basis. The two quantities are represented as

$$\frac{S}{N} = \frac{E_b}{N_0} \times \frac{R_b}{B} \quad (3)$$

where R_b is the bit rate and N_0 is the noise power spectral density. Let us consider we are sending binary digits across AWGN channel at a transmission bit rate equal to the channel capacity i.e. $R_b = C$, i.e. *ideal system*, then Shannon-Hartley equation can be written as

$$\frac{C}{B} = \log_2 \left(1 + \frac{E_b C}{N_0 B} \right) \quad (4)$$

where C/B represents the normalized channel capacity also called *spectrum efficiency* or *throughput*.

Rearranging, (4) we get

$$\frac{E_b}{N_0} = \frac{B}{C} \left(\frac{2^{\frac{C}{B}} - 1}{\eta} \right) \quad (5)$$

where $\eta = C/B$ is the *spectral efficiency* in bits/seconds/Hz.

$$\frac{E_b}{N_0} = \frac{B}{C} \left(\frac{2^\eta - 1}{\eta} \right) \quad (6)$$

The relation is plotted in bandwidth-efficiency diagram in Figure 2.2. Shannon's theory also states the limiting value of E_b/N_0 below which there can be no error free transmission at any data rate. Using the identity

$$\lim_{n \rightarrow \infty} (1 + x)^{\frac{1}{x}} = e \quad (7)$$

We can calculate the limiting value of E_b/N_0 by considering

$$x = \frac{E_b}{N_0} \left(\frac{C}{B} \right) \quad (8)$$

Then we can rewrite the (4) as:

$$\frac{C}{B} = x \log_2 (1 + x)^{\frac{1}{x}} \quad (9)$$

$$1 = \frac{E_b}{N_0} \log_2(1 + x)^{\frac{1}{x}} \quad (10)$$

In the limit, as $C/B \rightarrow 0$, we get

$$\frac{E_b}{N_0} = \frac{1}{\log_2 e} = 0.693 \quad (11)$$

In dB, $\frac{E_b}{N_0} = -1.59 \text{ dB}$, represents the Shannon's limit or specifically 'Shannon's power efficiency limit', which gives the minimum possible E_b/N_0 required to achieve maximum transmission capacity ($R_b = C$). It is the limit of band limited system irrespective of modulation or coding. As this capacity is approached, the system complexity will increase significantly. So, the aim of any system design is to achieve this limit.

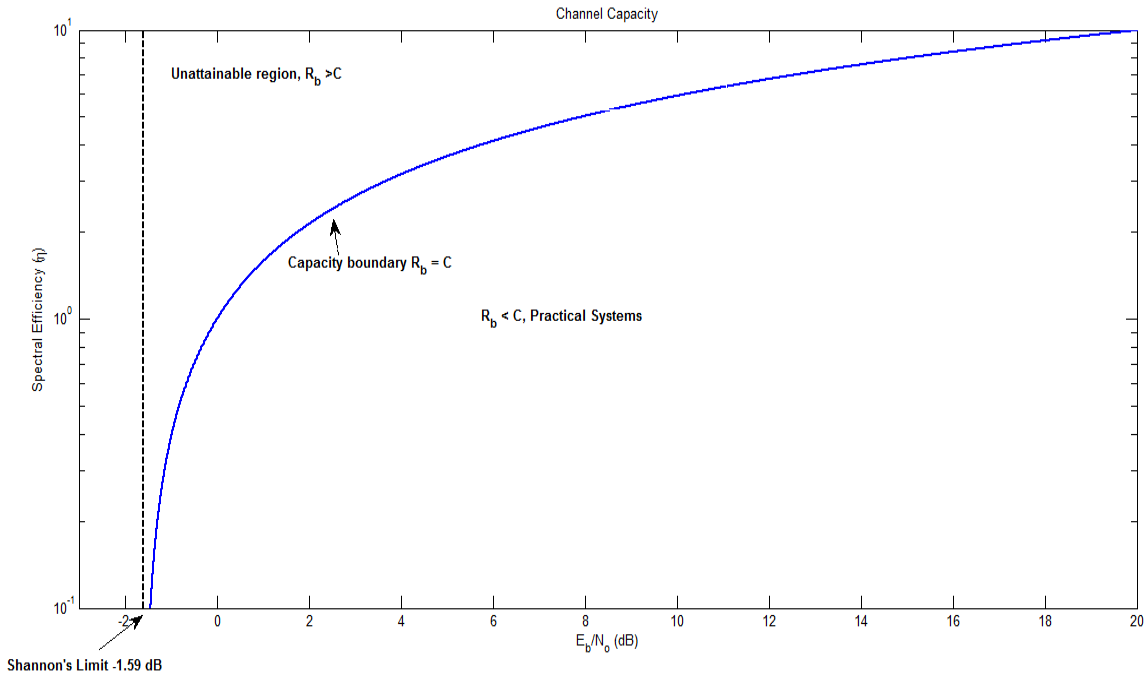


Figure 2.2 Channel capacity curve for AWGN channel.

In the Figure 2.2 capacity boundary, $R_b = C$ is the absolute Shannon's limit for AWGN channel. The $R_b > C$ region is the unattainable region, i.e., if we attempt to send the data over channel at a rate greater than the Shannon's capacity limit, the data errors at the receiver will be

irrecoverable and the data will be lost completely. The $R_b < C$ region is the practical attainable region, where the signaling rate R_b is less than absolute channel capacity limit for a channel.

Shannon's work provided a theoretical proof for the existence of codes that could be used to improve the performance of uncoded binary modulation schemes to approach the theoretical curve. Every coding scheme today tries to approach the Shannon's limit. But while using those coded modulation schemes, the communication system should consider two major resources the transmitter power and channel bandwidth.

2.3 Power Limited and Bandwidth Limited Systems

Two major resources/constraints of communication are transmitting power requirement and available channel bandwidth. In every communication system one of the resources is more precious than other. Based on these resources all communication systems are classified as power limited system or bandwidth limited system.

Power limited systems save power at the expense of bandwidth. For example, MFSK (M -ary Frequency Shift Keying) and CDMA (Code Division Multiple Access), allow more bandwidth for transmission of data and operates with minimal power. Deep space communication systems are in the category of power-limited systems [3].

Bandwidth limited systems save bandwidth at the expense of power. The available bandwidth for certain communication application is limited. The capacity of communication channel is proportional to bandwidth, and the demand for high capacity communication systems is increasing day by day. Bandwidth efficient modulation schemes such as M -PSK and M -QAM save the transmission bandwidth at the expense of more transmission power. Cellular mobile communication systems are categorized as bandwidth limited systems [3].

In this thesis, channel coding with bandwidth efficient modulation schemes are considered.

2.4 Bandwidth Efficiency of Modulated Systems

The ratio of bit rate to the required bandwidth is called the transmission bandwidth efficiency of a modulation scheme. The bandwidth efficiency of bandwidth efficient modulation systems, M -ary PSK and M -ary QAM, for ideal filtering, i.e. Nyquist filtering, is given as [4-6]

$$\eta = R/B = \log_2(M) (b/s)/Hz \quad (12)$$

As M increases, η also increases.

2.5 Channel Model

In AWGN channel model, white Gaussian noise is added to the transmitted signal. The channel can be mathematically described [2] by the relation

$$r(t) = s(t) + n(t) \quad (13)$$

where $s(t)$ is the transmitted signal, $n(t)$ is zero mean, white Gaussian noise with power spectral density of $N_0/2$, and $r(t)$ is the received signal. The channel model is shown in Figure 2.3.

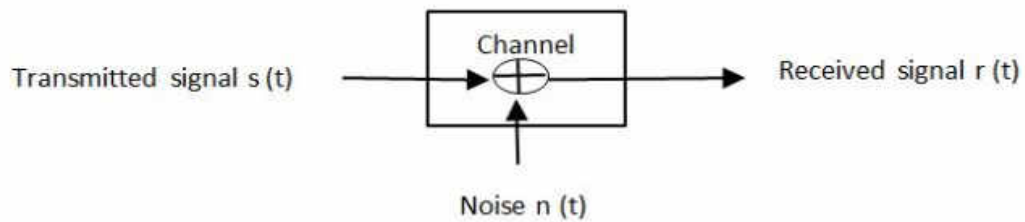


Figure 2.3 AWGN channel.

The AWGN channel model applies to broad class of communication channels and is the predominant channel model for analysis and design of communication system.

Some fading channel models for Radio Frequency (RF) wireless communications are Rayleigh, Rician, and Nakagami fading channels. The atmospheric loss mechanism in free space optics is identical to wireless for Line-Of-Sight (LOS) channels, but the fading level is higher than RF signals [7].

2.6 Error Control Coding

Error control coding is implemented to detect and correct a limited number of errors that occur during the transmission of data in noisy channels. The error is controlled by introducing a channel encoder/decoder pair in the communication systems block, as mentioned in section 2.1. The overall goal of the error control coding is to encode the digital information in such a manner that even the channel or storage medium introduces an error; the receiver can detect and correct them and recover the original transmitted information. Two broadly defined categories of error-control coding techniques are discussed below.

2.6.1 Automatic-Repeat Request (ARQ) Schemes

In ARQ scheme, upon detection of an error in a transmitted codeword, the decoder requests the transmitter for a retransmission of codeword that has error. Thus, the system needs to have a return path (i.e. feedback channel). The ARQ mechanism is based on positive acknowledgment (ACK), negative acknowledgment (NACK) messages transmitted by the receiver to the transmitter to indicate a good (ACK) or bad (NACK) reception of previous data. The NACK response triggers to resend the data to make it error-free. In addition to that there is a timeout period if no acknowledgement is received by the time it needs to resend. Because of the need to retransmit a codeword, the latency is introduced in the system, so is not feasible for systems

requiring low latency. Also, for real time information transmission such as in speech communications by wireless, ARQ is not suitable [8].

2.6.2 Forward Error-correction (FEC) Codes

The FEC coding scheme can detect and correct the errors introduced in transmitted codewords at the receiver end without being resend back to transmitter. Thus, this technique requires only one-way link between the transmitter and receiver. FEC codes have been classified mainly into two distinguished categories: block codes and convolution codes.

2.6.2.1. Block Codes

In block coding the data are encoded and decoded based on block-by-block basis. Block coding operation doesn't need memory and can be implemented using combinational logic.

In the binary block code, each data in an input sequence are converted to block size of k bits. Each data block is mapped to an n -symbol codeword, where $n > k$. The additional $n - k$ redundant bits (normally parity bits) are added to detect and correct errors. Each data block is associated with one and only one codeword from 2^k distinct codewords. The resultant code is represented as (n, k) , block code. Since each codeword contains n symbols to transmit k bits of information, the core rate of the encoder output is k/n .

The encoder will map the information sequence into codewords by properly choosing the parity bits. Decoding at the decoder is conducted by determining the most likely transmitted codewords from the received symbols.

A block code is linear if the component of the codeword can be written as linear combination of the k information bits.

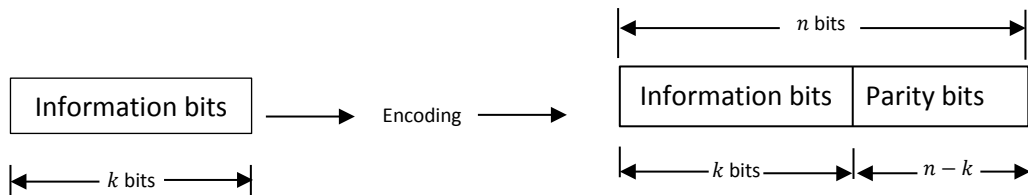


Figure 2.4 Systematic block code.

In systematic linear block code, k bits of the codewords are message symbols which are transmitted unaltered. The information bits are followed by redundant bits (parity bits) or vice versa, thus having message parts and redundant bits' parts as shown in Figure 2.4.

Some fundamental definitions are:

- Hamming weight: Hamming weight (or simply weight) of a code c is defined as the number of nonzero components of the code.
- Hamming distance: The hamming distance (or simply distance) among two codes c_1 and c_2 denoted by $d(c_1, c_2)$ is the number of positions in which they differ (modulo-2 sum of c_1 and c_2). Also, the hamming distance between two codes is equal to the hamming weight of the sum of the two codes.
- If c_1 and c_2 are both codewords of a linear block code, then sum of two codeword must be a codeword. Also, minimum distance d_{min} of a linear block code is equal to the minimum weight of its nonzero codeword. Minimum distance is also defined as the smallest hamming distance between any pair of code vectors in the code.

A block code with minimum distance d_{min} can correct up to t errors, where t is given as [2, 9]

$t \leq \lfloor (d_{min} - 1)/2 \rfloor$ where $\lfloor x \rfloor$ denotes the largest integer no greater than or equal to x .

The t error-correcting linear block code (n, k) on a memoryless Binary Symmetric Channel (BSC) with transition probability P_c , the probability that the decoder will make erroneous decoding is upper bound and is given by [2, 9]

$$P_w(C) \leq \sum_{i=t+1}^n \binom{n}{i} P_c^i (1 - P_c)^{n-i} \quad (14)$$

where $\binom{n}{i}$ represents the number of all possible patterns of i errors in an n -symbol codeword. Some of the most popular block coding are Cyclic codes, BCH (Bose-Chaudhuri-Hocuenghem) codes, Hamming codes, Reed-Muller codes, Reed Solomon (RS) codes, and Goley codes. The cyclic code is the one in which cyclic shift operation of the codeword generates another codeword. From a single n -bit codeword $n - 1$ codewords can be generated by cyclic shifts, which can be implemented by using a shift registers and small amount of additional logic. BCH code is also cyclic code and it's encoding and decoding are accomplished using shift register circuits. This code is the best constructive code for channels with errors affecting successive symbols independently. Hamming code is the special case of BCH code with a minimum distance 3. Hamming codes are used in computer memory devices [10,11]. Reed Solomon (RS) codes are non-binary BCH codes, most frequently implemented in data storages, such as in CD-ROM (Compact Disc-Read Only Memory), DVD (Digital Versatile Disc), and communication systems including space communications and terrestrial digital HDTV (High-Definition Television) transmission applications [9].

2.6.2.2. Convolution Coding

Convolution coding works on sequential data streams and its encoding and decoding operations depends upon current and previous data as well. Since convolution coding contains memory, it must be implemented using sequential logic [12].

The convolution encoder consists of three component parts: shift register, modulo-2 adders and commutator. The M shift registers are connected to n modulo-2 adders, and a commutator that serializes the outputs of the adders.

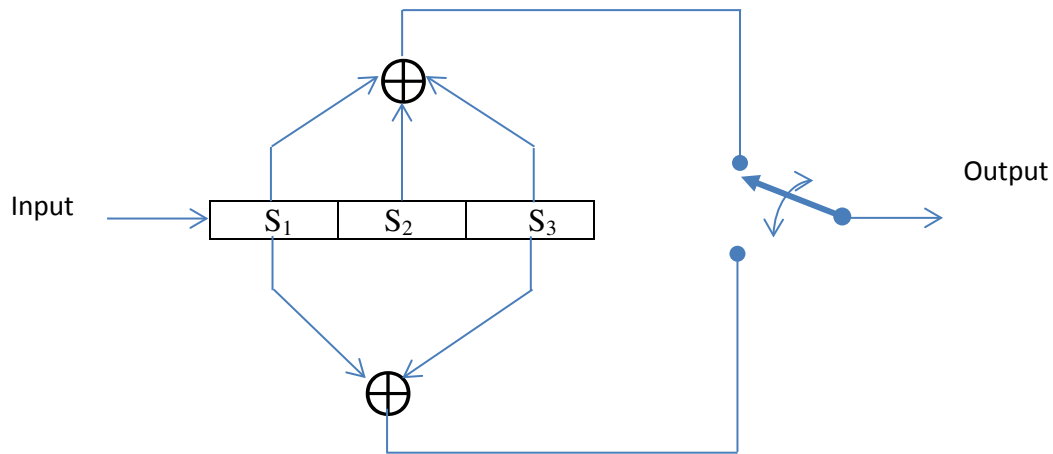


Figure 2.5 Convolution encoder.

The convolution code is described by three integers, k , n and K . The k is the input bit and n is the coded output bit which is not a block or codeword length as in block code. The constraint length K is defined as the number of shifts over which a single message bit can influence the encoder output. Since it implements shift registers, the encoder has memory, so the output of the encoder is not only the function of current input but also to previous $K - 1$ inputs. A rate $\frac{1}{2}$ convolution encoder is illustrated in Figure 2.5.

The encoder of convolution coding can be represented in several other forms such as polynomial form, state diagram, tree diagram, and trellis diagram representation. Convolution decoding can be conducted by using Maximum Likelihood Decoding (MLD), sequential decoding, Viterbi decoding, etc. Viterbi decoding algorithm has found widespread application in practical applications.

2.6.2.3. Turbo Code

Turbo codes are parallel-concatenated convolution codes with interleaving. The basic turbo encoder consists of two parallel convolution encoders separated by the interleaver. The two encoders may be either identical or different, and the code is systematic. The interleaver between the two encoders is used to scramble the bits before feeding them to the second encoder, so that the outputs of the two encoders are different. The decoding of the turbo code is conducted by using iterative decoding. As the number of iterations increases the performance of the turbo coded system improves [2, 3]. Being a capacity approaching code with excellent error correction capability it has been extensively used in mobile wireless communication including 3G and 4G wireless systems, satellite communication systems, wireless networking and various other applications. The major drawback of the turbo decoding is the decoding delay because of the use of large interleaver and complex iterative decoding algorithm [2].

2.6.2.4. Low-density Parity Check (LDPC) Code

LDPC codes are linear block codes [13] with a sparse parity check matrix with few 1's in comparison to zero. The codes are represented by the Tanner graph [14]. The code has excellent error performance capability that can achieve very close to Shannon's limit [15]. Because of strong error performance it has employed in numerous applications including Wi-Fi (Wireless

Fidelity), WiMAX (WorldWide Interoperability of Microwave Access), Ethernet and DVB-S2 (Digital Video Broadcasting-Satellite-Second Generation) standards [16].

2.7 Multilevel Coded Modulations

2.7.1 Introduction

Combining error control coding with binary modulation results in channel bandwidth expansion, because of the addition of redundant bits in the transmitted sequence to combat the error occurred in channel. To maintain the same information transmission rate $R = k/n < 1$ for error control requires bandwidths expansion by a factor of $1/R$. Therefore, coding gain is achieved at the expense of bandwidth expansion. This type of coding provides an effective trade-off between channel bandwidth and transmits power, and is suitable to operate in power limited channels, such as deep-space, satellite, and other wideband communications systems [1].

For bandwidth limited applications, such as voice band telephone, terrestrial microwave, and some satellite channels, bandwidth expansion is not desirable or even not possible. For this reason, coding by adding extra redundant bits for error correction is rarely used on band-limited channels. To overcome this problem, coding must be used in conjunction with bandwidth-efficient multilevel modulation.

For efficient communication systems, especially in band-limited channels, the coding and modulation are combined, which are called coded modulations. The concept of jointly treating coding and modulation to improve the error performance of digital communication is introduced by J.L Massey in 1974 [17]. The Massey's concept was successfully implemented by the invention of most powerful coded modulation systems known as Trellis Coded Modulation (TCM) by Ungerboeck [18], and coded modulation scheme using Multilevel Coding (MLC)

introduced by Imai and Hirakawa [19]. Both, coded modulation system uses different coding techniques to detect and correct errors in combination with multilevel modulation techniques. The core of these coded modulations is to optimize the code in Euclidean space rather than dealing with Hamming distance as in classic coding schemes [20].

Ungerboeck's TCM uses the set partitioning based mapping for coded modulation. In his approach the signal set $A = \{a_0, a_1, \dots, a_{M-1}\}$ of $M = 2^l$ ary modulation schemes is successively binary partitioned in l steps to map $X = (x^0, x^1, x^2, \dots, x^{l-1})$ signal points to a_m . Ungerboeck's set partitioning strategy, i.e. maximizing the minimum intra-subset Euclidean distance is chosen by most of the coded modulations. The binary address is usually divided into two parts in the encoder: the least significant binary symbols are convolutionally encoded and most significant binary symbols (if present) remain uncoded. To maximize the minimum distance of the coded sequences in Euclidean space, the code parameters are chosen by means of a rigorous computer search [21].

In Imai and Hirakawa's MLC each address bit x^i of the signal point is addressed by an individual binary code C^i at level i . The individual codes were chosen to maximize the minimum distance of the Euclidean space of the code. At the receiver, multistage decoding (MSD), i.e. each code C^i is decoded individually starting from lowest level and considering decision of prior decoding stages, is used. MLC approach, in contrast to Ungerboeck's TCM, provides flexible transmission rates by decoupling the dimensionality of signal constellation from the code rate. Additionally, any of the existing codes such as block codes, convolution codes, or concatenated codes, can be used as component code [21].

The concept of multilevel encoding scheme is shown in Figure 2.6. Each incoming bits are encoded and mapped to select a point for constellation. The coded data $x_i, i= 0, 1, \dots \dots, l - 1$ results from individual encoding of data symbols.

A block of K binary source data symbols $q = (q_1, \dots \dots, q_k), q_l \in \{0,1\}$, is partitioned into l blocks $q^i = (q_1^i, \dots \dots, q_{K_i}^i), i = 0,1, \dots \dots, l - 1$ of length K_i with $\sum_{i=0}^{l-1} K_i = K$. Each data block is fed into an individual binary encoder E_i generating words $x_i = (x_1^i, \dots \dots, x_N^i), x_\gamma \in \{0,1\}$, of the component code C^i . The codeword symbols $x_\gamma^i, \gamma= 1, \dots, N$ of the codeword $x^i, i = 0, \dots \dots, l - 1$ at one time instant γ , form the binary label $x_\gamma = (x_\gamma^0, \dots \dots, x_\gamma^{l-1})$ which is mapped to the signal point a_γ .

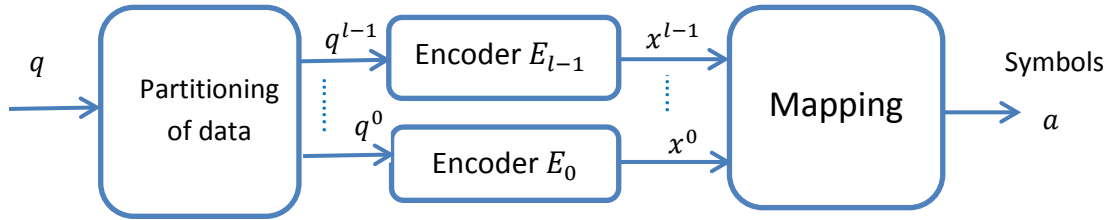


Figure 2.6 Multilevel encoding scheme [21].

The code rate R of the scheme is equal to the sum of the individual code rates $R^i=K_i/N$, namely,

$$R = \sum_{i=0}^{l-1} R_i = \frac{1}{N} \sum_{i=0}^{l-1} K_i = \frac{K}{N} \quad (15)$$

where K is the block of binary source data, K_i is the data block at each individual binary encoder E_i and N is the codeword symbol.

2.7.2 Multistage Decoding

At the receiver, the components C^i are successively decoded by the corresponding decoders D_i . Decoder D_i processes not only the received block y , but also processes the decisions of previous decoding stages. This process of multilevel decoding allows user to reduce decoding complexity [21]. The multistage decoding block is shown in Figure 2.7.

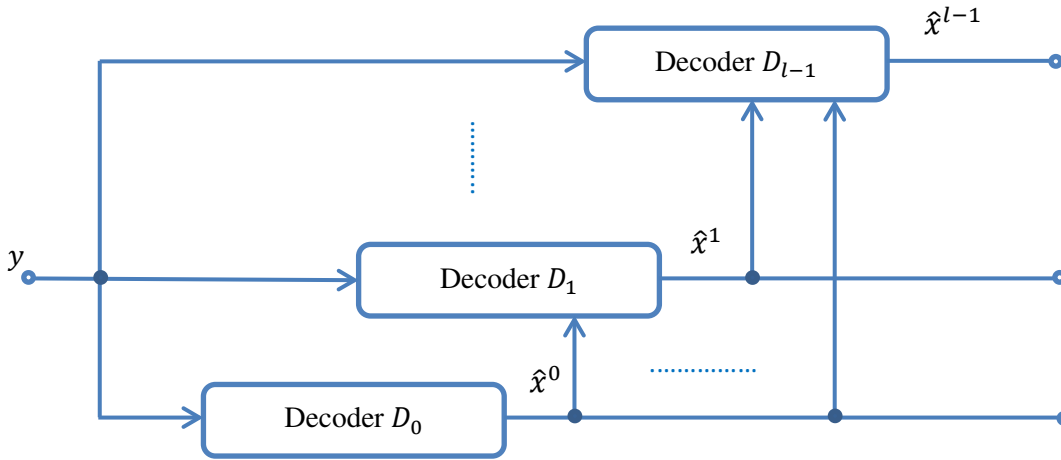


Figure 2.7 Multistage decoding [20].

2.7.3 Parallel Independent Decoding at Individual Level (PDL)

Multilevel coding can be decoded with parallel independent decoding at the individual levels (PDL) [20, 21]. In this method, the decoder D_i makes no use of the decisions of the other levels. All decoders $D_i = 0, 1, 2, \dots, l - 1$ work independently in parallel. For MLC/PDL, the transmission of each address symbol x_i $i = 0, 1, \dots, l - 1$, over equivalent i^{th} channels is based on the entire signal constellation. This decoding method may be employed to reduce the decoding delay [22], since individual decoders are working in parallel. Also, error propagations from lower levels to higher levels are avoided, since each level is decoded individually. Also,

Gray-mapped bit-interleaved coded modulation (BICM) provides mutual information (MI) very close to the channel capacity [21, 23]. For transmission over time variant channels, with static AWGN and the Rayleigh fading channel, MLC/PDL provides best robustness to both channel situations among other competing coded modulations schemes [24]. This thesis works presents the MLC/PDL using orthogonal coding and with M -ary modulation techniques.

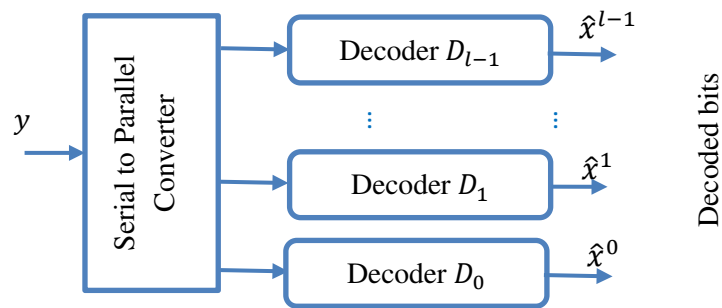


Figure 2.8 Parallel independent decoding of an MLC scheme [24].

CHAPTER 3

***M*-ary PSK and *M*-ary QAM MODULATION SCHEMES**

In *M*-ary modulation multiple number of bits are grouped into a symbol. A separate waveform for each symbol is transmitted to channel. For *k*-bit symbol, 2^k waveforms are required for transmission. Bandwidth efficient modulation schemes such as *M*-ary PSK and *M*-ary QAM are popular choices for high data rate communication applications. In this thesis, these modulations are implemented combining with the multilevel orthogonal channel encoding. The general concepts of these modulation schemes are discussed in brief in this chapter.

3.1 *M*-ary Phase Shift Keying (*M*-ary PSK)

3.1.1 Introduction

In *M*-ary PSK modulation the phase of the carrier signal takes one of the *M* possible phase values $\frac{2(i-1)\pi}{M}$, where $i = 1, 2, \dots, M$ in each of the signaling interval of duration T_s . The amplitude and frequency (f_c) of the carrier are unaltered. The modulated signal is of the form

$$S_i(t) = \sqrt{\frac{2E_s}{T_s}} \cos \left[2\pi f_c t + \frac{2\pi(i-1)}{M} \right], i = 1, 2, \dots, M, 0 \leq t \leq T_s \quad (16)$$

This can be expanded as,

$$S_i(t) = \sqrt{\frac{2E_s}{T_s}} \left[\cos\left(\frac{2\pi(i-1)}{M}\right) \cos(2\pi f_c t) - \sin\left(\frac{2\pi(i-1)}{M}\right) \sin(2\pi f_c t) \right] \quad (17)$$

$$S_i(t) = \sqrt{E_s} \cos\left(\frac{2\pi(i-1)}{M}\right) \phi_1(t) - \sqrt{E_s} \sin\left(\frac{2\pi(i-1)}{M}\right) \phi_2(t) \quad (18)$$

where, $\phi_1(t) = \sqrt{\frac{2}{T_s}} \cos(2\pi f_c t)$

and $\phi_2(t) = \sqrt{\frac{2}{T_s}} \sin(2\pi f_c t)$

Thus, each $S_i(t)$ consists of two orthogonal basis functions $\phi_1(t)$ and $\phi_2(t)$, so the signal constellation for $M > 2$ is two-dimensional, where M message points are equally spaced in the circle of radius $\sqrt{E_s}$ and centered at the origin. Figure 3.1 shows the constellation diagram of Binary Phase Shift Keying (BPSK), Quadrature Phase Shift Keying (QPSK) and 16-PSK binary and gray coded, binary signals with $\sqrt{E_s} = 1$. In BPSK the two symbols are separated by phase angle of $360^\circ/2 = 180^\circ$. In QPSK four symbols are separated by 90° apart. Similarly, in the 16-PSK the 16 symbols are separated by 22.5° . For higher values of M the phase separation of the symbol is lower and it becomes more susceptible to noise as the symbol get closer. During demodulation of M -ary PSK, the receiver determines the phase of the received symbol. For M -ary PSK, the receiver must determine the phase within the $\mp 360^\circ/(2M)$, since the phases are separated by $(360/M)^\circ$. For example, for 16-PSK, the receiver must determine the phase within $\pm 11.25^\circ$.

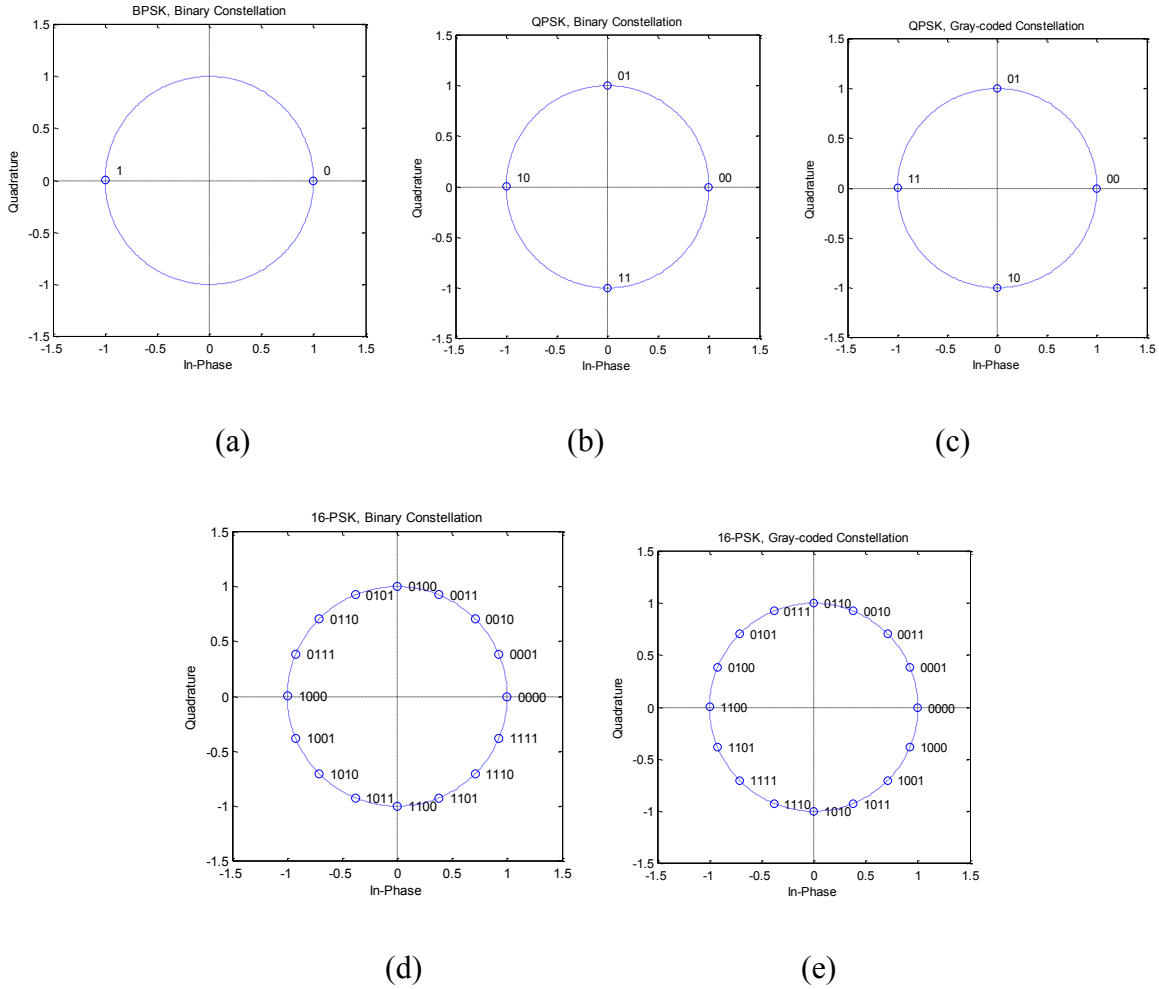


Figure 3.1 Constellation diagram for different values of M for M -PSK modulation.

3.1.2 Power Spectra of M -ary PSK Signals

The symbol duration of M -ary PSK is given by $T_s = T_b \log_2 M$, where T_b is the bit duration. The baseband power spectral density of an M -ary PSK signal is given by [25]

$$S_B(f) = 2E_s \operatorname{sinc}^2(T_s f_c) = 2E_b \log_2 M \operatorname{sinc}^2(T_b f_c \log_2 M) \quad (19)$$

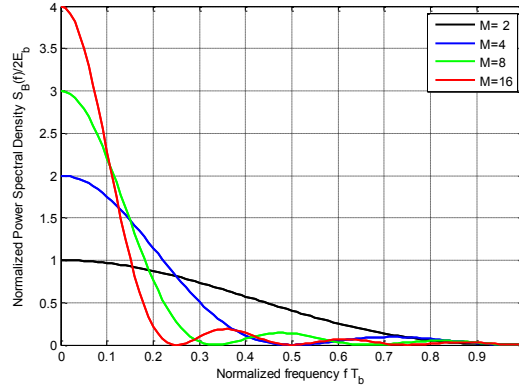


Figure 3.2 Power spectra of M -ary PSK signals for $M= 2, 4, 8$ and 16 .

3.1.3 Transmission Bandwidth and Bandwidth Efficiency

The channel bandwidth required to pass the M -ary PSK signal also referred as *null-to-null bandwidth* is given by [25]

$$B = \frac{2}{T_s} \quad (20)$$

where T_s is the symbol duration. Since, the bit rate $R_b = 1/T_b$, where T_b is the bit duration, the channel bandwidth can be given in terms of bit rate R_b as,

$$B = \frac{2R_b}{\log_2 M} = \frac{2R_b}{m} \text{ Hz} \quad (21)$$

where $m = \log_2 M$ is the bits per symbol. Using (21), the bandwidth efficiency of M -ary PSK is given as

$$\rho = \frac{R_b}{B} = \frac{\log_2 M}{2} \quad (22)$$

The calculated bandwidth and bandwidth efficiency of PSK for various values of M is presented in Table 3.1.

Table 3.1 Transmission bandwidth and bandwidth efficiency of PSK for various values of M .

M	2	4	8	16	32	64
Bandwidth	$2R_b$	R_b	$2R_b/3$	$R_b/2$	$2R_b/5$	$R_b/3$
ρ (bits/s/Hz)	0.5	1	1.5	2	2.5	3

Thus, for M -ary PSK as the number of constellation points M increases, the bandwidth efficiency is improved.

3.1.4 Error Probability of M -ary PSK

The approximated probability of symbol error P_u for M -ary PSK is given as [25]

$$P_u \approx \operatorname{erfc}\left(\sqrt{\frac{E_s}{N_0}} \sin \frac{\pi}{M}\right) \quad M \geq 4 \quad (23)$$

where E_s is symbol energy, $\frac{E_s}{N_0}$ is energy per symbol to noise spectral density and erfc is the *complementary error function* defined as

$$\operatorname{erfc}(x) = \frac{2}{\sqrt{\pi}} \int_x^{\infty} e^{-t^2} dt$$

Equivalent bit error probability for M -ary PSK is approximated as

$$P_b \approx \frac{1}{\log_2 M} P_u \quad (24)$$

The BER plot for M -ary PSK for various values of M for AWGN channel using *bertool* in MATLAB is shown in Figure 3.3. The figure shows that the bit error performances get worse as M increases.

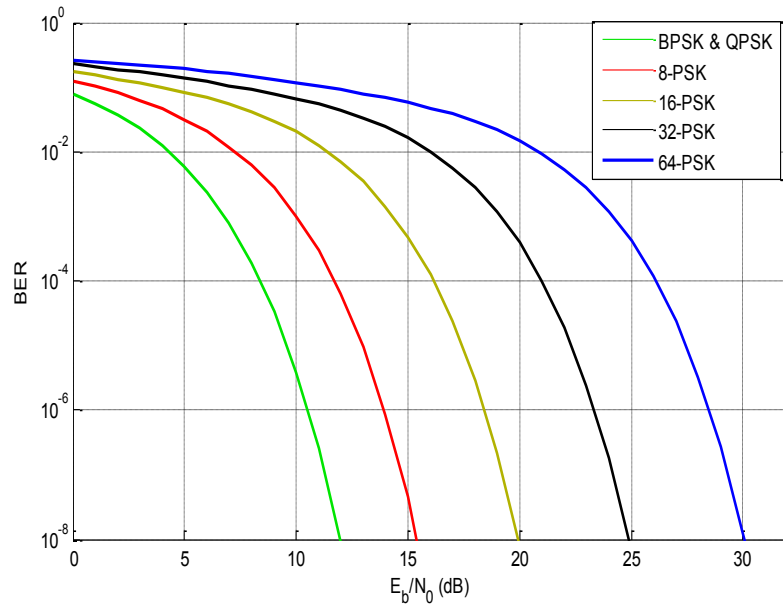


Figure 3.3 Bit error rate of M -ary PSK for different values of constellation points M .

3.2. M -ary Quadrature Amplitude Modulation (M -ary QAM)

3.2.1 Introduction

QAM is a widely-used modulation technique in digital communication systems and high data rate delivery applications. Higher order QAM has capability of transmitting more bits per symbol, enabling data to be transmitted in much smaller bandwidth, making them more spectrally efficient transmission system [25]. Higher level QAM combined with FEC coding can provide bandwidth efficiency with error correction capabilities. Some of the data transmission applications that uses M -ary QAM are high speed cable modems, particularly digital cable television and cable modem utilizes 64-QAM and 256-QAM, terrestrial and satellite broadcasting channels, microwave digital radio, WLAN IEEE 802.11 a/g, HSDPA (High-Speed Download Packet Access) & HSUPA (High-Speed Uplink Packet Access) for Long-Term Evolution) LTE Cellular systems, and so on [3, 25-30].

QAM can be viewed as combined amplitude and phase modulation. QAM signal waveform may be expressed as [2]

$$S_m(t) = A_{mi} g(t) \cos(2\pi f_c t) - A_{mq} g(t) \sin(2\pi f_c t) \quad (25)$$

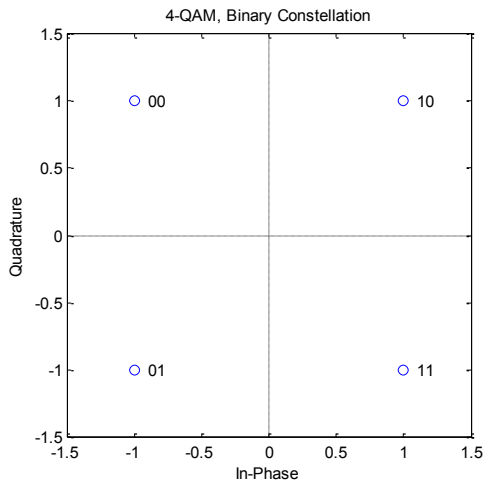
where $S_m(t)$ represents the bandpass signal chosen from the M possible waveforms, f_c is the carrier frequency, A_{mi} and A_{mq} are the information-bearing signal amplitudes (A_m) of the in-phase and quadrature carrier and $g(t)$ is the signal pulse which shape influence the spectrum of transmitted signal. The QAM signal waveform can also be represented as

$$S_m(t) = r(\cos(2\pi f_c t + \theta_m)) \quad (26)$$

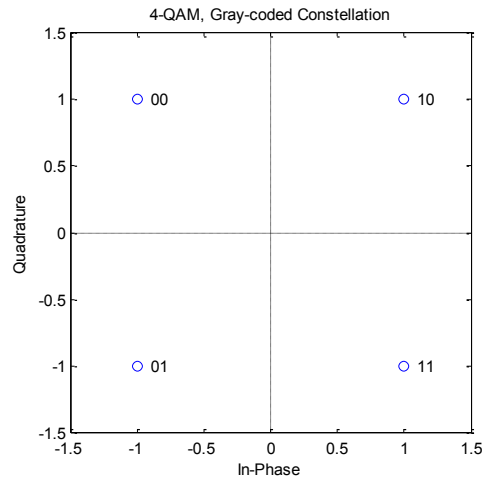
where, $r = \sqrt{A_{mi}^2 + A_{mq}^2}$ and $\theta_m = \tan^{-1}(A_{mq}/A_{mi})$. Thus, QAM signal waveforms can be observed as the combined amplitude (r) and phase (θ_m) modulation.

M -ary QAM constellations can be constructed in many ways, and they have different capacity and error characteristics. Rectangular QAM signal constellations, in which the number of bits per symbol is even, have distinct advantage of being easily generated and transmitted as two Pulse Amplitude Modulation (PAM) signals impressed on phased-quadrature carriers. Also, they are easily demodulated [2].

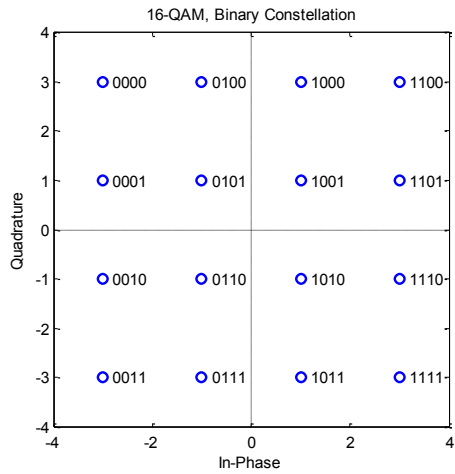
Rectangular signal constellation with binary and gray coding for 4-QAM and 16-QAM is shown in Figure 3.4. For orthogonal mapping based channel coding the output symbols from channel encoder are directly mapped to the QAM modulation as rectangular constellation.



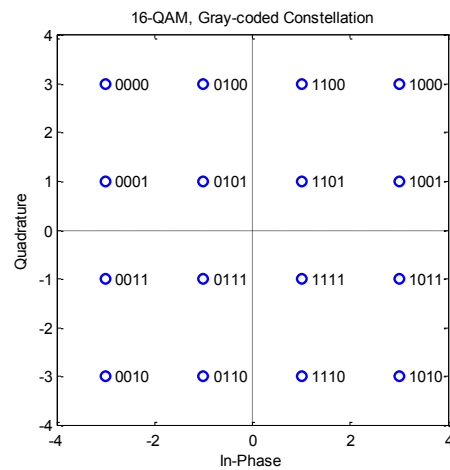
(a)



(b)



(c)



(d)

Figure 3.4 Rectangular constellation diagram for different values of M for M -ary QAM modulation.

3.2.2 Transmission Bandwidth and Bandwidth Efficiency

The transmission bandwidth (BW) of the M -ary QAM system required to pass the main lobe of signal spectrum is calculated as

$$B \approx \frac{2 R_b}{\log_2 M} \text{ (Hz)} \quad (27)$$

where, R_b is the input bit rate (bits/sec). This bandwidth is referred as *null-to-null bandwidth*. The bandwidth efficiency of the M -ary QAM modulation is increased by using higher values of M . The number of possible signals are given by $M = 2^m$, where m is an integer. The symbol duration is $T_s = m T_b$, where T_b is the bit duration [3].

The bandwidth efficiency of the M -ary QAM system can be defined as

$$\rho = \frac{R_b}{B} = \frac{\log_2 M}{2} \quad (28)$$

The bandwidth efficiency for different M -ary QAM is given in Table 3.2.

Table 3.2 Bandwidth efficiency of QAM for different values of M .

M	4	16	64	256	1024
ρ (bits/sec/Hz)	1	2	3	4	5

3.2.3 Error Probability of M -ary QAM

The approximated probability of symbol error P_u with rectangular constellation for M -ary QAM is given as [2]:

$$P_u \approx 4 \left(1 - \frac{1}{\sqrt{M}}\right) Q \left(\sqrt{\frac{3 \log_2 M E_{bavg}}{M-1 N_0}} \right) \left(1 - \left(1 - \frac{1}{\sqrt{M}}\right) Q \left(\sqrt{\frac{3 \log_2 M E_{bavg}}{M-1 N_0}} \right) \right) \quad (29)$$

$$P_u \leq 4Q \left(\sqrt{\frac{3 \log_2 M E_{bavg}}{M-1 N_0}} \right) \quad (30)$$

where \sqrt{M} represents the number of amplitude levels, $\log_2 M$ is number of bits per symbol, $Q(x)$ is the Q -function defined as

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-\frac{t^2}{2}} dt$$

Equivalent bit error probability for M -ary QAM is approximated as

$$P_b \approx \frac{1}{\log_2 M} P_u \quad (31)$$

The BER plot for M -ary QAM for various values of M with AWGN channel using *bertool* in MATLAB is shown in Figure 3.6.

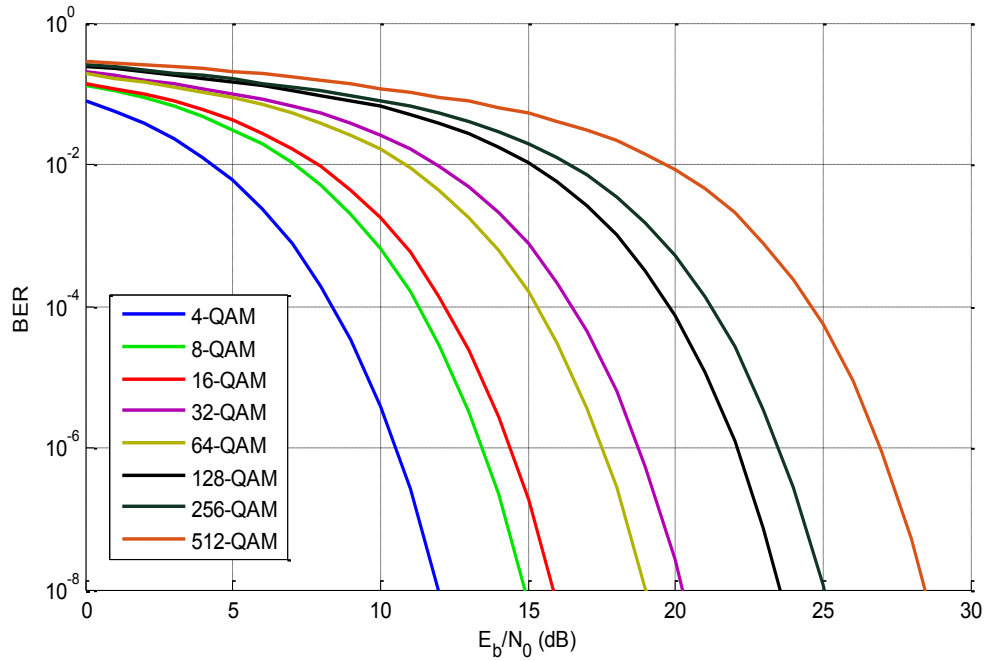


Figure 3.6 Bit error rate of M -ary QAM for different values of constellation points M .

3.3 Comparison of Error Probabilities of M -ary PSK and M -ary QAM

For a given size of constellation M , both M -ary PSK and M -ary QAM signal are two-dimensional. The ratio of symbol error probabilities of M -ary PSK and M -ary QAM as presented in (23) and (30) is [2]

$$R_M = \frac{3}{\frac{M-1}{2 \sin^2 \frac{\pi}{M}}} \quad (32)$$

Since, for $M = 1$, we get, $R_M = 1$, QPSK and 4-QAM show comparable performances for the same SNR per symbol. For $M > 4$, $R_M > 1$, so M -ary QAM shows better performance than M -ary PSK. SNR advantage of QAM over PSK for several values of M is presented in Table 3.3. For example, 16-QAM has 4.20 dB SNR advantage over 16-PSK.

Table 3.3 SNR advantage of M -ary QAM over M -ary PSK [2].

M	$10 \log R_M$ (dB)
8	1.65
16	4.20
32	7.02
64	9.95

Table 3.4 shows the values of E_b/N_0 to achieve BER= 10^{-6} and BER= 10^{-3} from *bertool* in MATLAB for bit error rates curves shown in Figure 3.3 and Figure 3.6.

Table 3.4 Modulation types and their required E_b/N_0 for BER of 10^{-6} and 10^{-3} , and minimum bandwidth required for intersymbol interference (ISI) free signaling.

Modulation types	Required E_b/N_0 (dB) BER= 10^{-6}	Required E_b/N_0 (dB) BER= 10^{-3}	Minimum channel bandwidth for ISI free signaling
BPSK	10.53	6.78	R_b
QPSK	10.53	6.78	$0.5 R_b$
8-PSK	13.95	10.01	$0.33 R_b$
16-PSK	18.44	14.35	$0.25 R_b$
32-PSK	23.36	19.14	$0.2 R_b$
4-QAM	10.53	6.78	$0.5 R_b$
16-QAM	14.4	9.65	$0.25 R_b$
64-QAM	18.78	14.76	$0.16 R_b$

CHAPTER 4

ERROR CONTROL USING ORTHOGONAL CODES

4.1 Orthogonal Codes

A pair of codes is called orthogonal if the cross-correlation value is zero [31]. For two m -bit orthogonal codes: x_1, x_2, \dots, x_m and y_1, y_2, \dots, y_m the orthogonality is given by

$$R_{xy}(0) = \sum_{i=1}^m x_i y_i = 0 \quad (33)$$

4.2 Generation of Orthogonal Codes

Walsh codes, originally developed by J.L. Walsh [32], are perfectly orthogonal binary block codes [33]. These orthogonal Walsh codes can be generated using Hadamard matrix using two-step process [31]:

Step-1: Represent a $n \times n$ matrix as 4-quadrants.

Step-2: Make 1st, 2nd, and 3rd quadrants identical and invert the 4th.

This principle can be extended to generate an $n \times n$ matrix. The Hadamard matrix H_k of dimension $2^k \times 2^k$ $k = 2, 4, 8, \dots$ (i.e. k is an even integer of 1s and 0s) for generation of orthogonal code can be represented as,

$$H_{2n} = \begin{bmatrix} H_n & H_n \\ H_n & \overline{H_n} \end{bmatrix} \quad (34)$$

where $\overline{H_n}$ denotes the complement (1s replaced by 0s and vice versa) of H_n . For example, Hadamard matrix of order 2 and 4 will be,

if $n = 1$ and $H_1 = [0]$, then

$$H_2 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \quad (35)$$

$$H_4 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \quad (36)$$

Thus, Hadamard matrices are symmetric matrices whose rows are orthogonal. The matrix contains all zeros in one row, while all other rows each contain $n/2$ zeros and $n/2$ ones. The codes on the rows of the matrix are orthogonal to each other. Also, rows differ from any other row by $n/2$ positions. Using similar method, any n -bit orthogonal code can be generated [2].

4.3 Properties of Orthogonal Codes

Two basic properties which make orthogonal codes powerful codes for error control coding are:

- (i) Distance property: each orthogonal code has equal number of 1s and 0s in it.
- (ii) Zero cross-correlation property: the cross correlation between a pair of orthogonal code is zero.

4.4 Bi-orthogonal Codes

Orthogonal codes are binary valued and have equal numbers of 1s and 0s. Antipodal codes, on the other hand, are just the complement (1s are replaced by 0s and vice versa) of orthogonal

codes. Antipodal code-set are orthogonal to themselves. Thus, an antipodal code has one row of all ones and the other rows each contains $n/2$ ones and $n/2$ zeros.

	1	2	3	4	5	6	7	8
1	0	0	0	0	0	0	0	0
2	0	1	0	1	0	1	0	1
3	0	0	1	1	0	0	1	1
4	0	1	1	0	0	1	1	0
5	0	0	0	0	1	1	1	1
6	0	1	0	1	1	0	1	0
7	0	0	1	1	1	1	0	0
8	0	1	1	0	1	0	0	1
9	1	1	1	1	1	1	1	1
10	1	0	1	0	1	0	1	0
11	1	1	0	0	1	1	0	0
12	1	0	0	1	1	0	0	1
13	1	1	1	1	0	0	0	0
14	1	0	1	0	0	1	0	1
15	1	1	0	0	0	0	1	1
16	1	0	0	1	0	1	1	0

Orthogonal

Antipodal

Figure 4.1 Bi-orthogonal code set combining orthogonal and antipodal codes.

Therefore, an n -bit orthogonal code has n -orthogonal codes and n -antipodal codes, for a total of $2n$ bi-orthogonal code set. Thus, bi-orthogonal code set consists of two sets, one of them is orthogonal code set and the other is antipodal code set [3]. For example, 8-bit orthogonal code has 16 bi-orthogonal codes as shown in Figure 4.1.

4.5 Some Applications of Orthogonal Codes

Orthogonal codes have been widely used in Code-Division Multiple Access (CDMA) systems. In CDMA applications, each user uses one of the orthogonal sequences in the set as a spreading code that provides the zero-cross correlation among all users [34]. In IS-95 CDMA, 64-bit orthogonal Hadamard codes are stored in read only memory and their purpose is to provide [31]:

1. Forward channel spreading over 1.2288 MHz band, and
2. Unique identification to a mobile user.

The chip rate (code rate) of Walsh code is 1.2288 MC/s (Mega-Chips per Second).

Since there are four different types of forward channels, they are designated as follows:

1. Pilot channel = W0 (Walsh Code-0)
2. Paging channel = W1 to W7 (Unused paging codes can be used for traffic)
3. Sync channel = W32
4. Traffic channel = W8-W31 and W33-W63

Similarly, CDMA 2000 1X uses Walsh code of 128 bits.

4.6 Error Correction Capability

Orthogonal codes have the capability of correcting errors that occur during transmission. Error correction capabilities of orthogonal codes are discussed in [35, 36]. It is presented here again for convenience. As explained in section 4.2, the distance between two orthogonal codes is $d = n/2$. Therefore, a received code with error can be detected by setting a threshold midway between two orthogonal codes as shown in Figure 4.2, where the received code is shown as a dotted line. For the code length n the threshold d_{th} which is midway between two valid orthogonal code is

$$d_{th} = \frac{n}{4} \quad (37)$$

In the decision process, the incoming impaired orthogonal code is examined for correlation with the neighboring codes for a possible match.

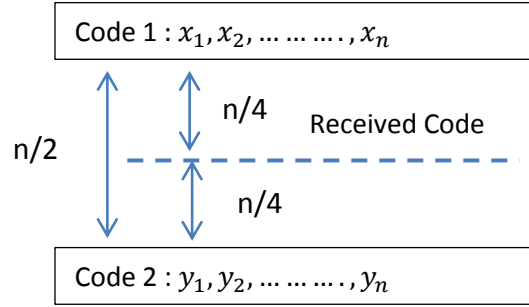


Figure 4.2 Distance property of orthogonal codes [35, 36].

Thus, the received code is examined for correlation with the neighboring codes for a possible match. If the n -bit comparison yields a good auto-correlation value for a certain code then that will be the valid code; otherwise, a false detection will occur. Mathematically, it can be shown by following correlation process, where an impaired orthogonal code is compared with a pair of n -bit orthogonal codes to yield,

$$R(x, y) = \sum_{i=1}^n x_i y_i \geq (n - d_{th}) + 1 \quad (38)$$

where x and y are two n -bit orthogonal codes and $R(x, y)$ is the auto correlation function, n is the code length. Since the threshold is in the midway between two valid codes, an additional 1-bit offset is added for reliable detection. Thus, the average number of errors that can be corrected by means of this process can be estimated by combining (37) and (38), yielding,

$$t = n - R(x, y) = \frac{n}{4} - 1 \quad (39)$$

In (39), t is the number of errors that can be corrected by means of an n -bit orthogonal code. For example, a single error-correcting orthogonal code can be constructed by means of an 8-bit orthogonal code ($n = 8$). Similarly, a three-bit error-correcting code can be constructed by means of a 16-bit orthogonal code ($n = 16$), and so on. Table 4.1 shows a few orthogonal codes and the corresponding error-correction capabilities. Also, the minimum value of n to ensure the

error correction is 8. Table 4.1 also shows that as the length of the code increases the number of bit error correction capability increases respectively.

Table 4.1 Error correction capabilities of orthogonal codes.

Code length <i>n</i>	Number of error corrected '<i>t</i>'
8	1
16	3
32	7
64	15
128	31
256	63

The FPGA implementation of orthogonal encoding and decoding without modulation is presented in [39]. The concept of orthogonal MPSK is presented in [35,37]. The application of orthogonal coded modulation for optical communications using On-Off keying is presented in [38]. In this thesis, the analysis of the orthogonal coded modulations in term of multilevel approach is performed. The application of orthogonal MPSK presented in [35,37] to laser based wireless communication systems is presented and the analysis and simulations based on multilevel approach are conducted. Also, the concept of multilevel orthogonal coded modulation for M -ary QAM modulation is presented and simulated.

CHAPTER 5

MULTILEVEL ORTHOGONAL CODED MODULATION

5.1 Orthogonal Coded Modulation

Orthogonal coded modulation [35] is a coded modulation technique where incoming message blocks are mapped to the orthogonal codes, which are then modulated. The incoming information data are converted to parallel form, which are then mapped to the orthogonal codes. The orthogonal code mapping table is presented in Table 5.1. $n = 4$ is not used for error correction but is shown to illustrate the concept. The mapped orthogonal code then will be modulated with a high frequency carrier signal. The modulated signals are then transmitted to the channel. The general block diagram of the transmitter of the orthogonal coded modulation is shown Figure 5.1.

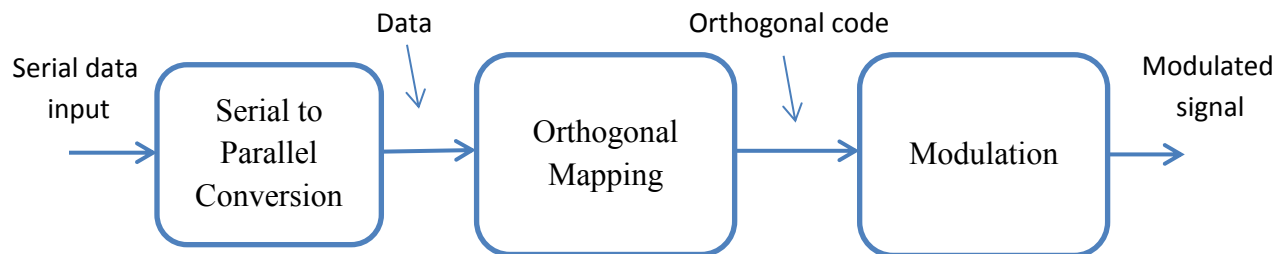


Figure 5.1 General block diagram of orthogonal coded modulation transmitter.

The general block diagram of receiver structure is shown in Figure 5.2. At the receiver end, the incoming noisy signal is first demodulated. Then, the demodulated data is cross-correlated with

n bi-orthogonal code sets generated/stored at the receiver end. The code that generates the lowest correlation value is the possible match. Once the closest approximation is achieved, the corresponding data is outputted from the lookup table. If the matching is not found along the length, the closest matches are picked.

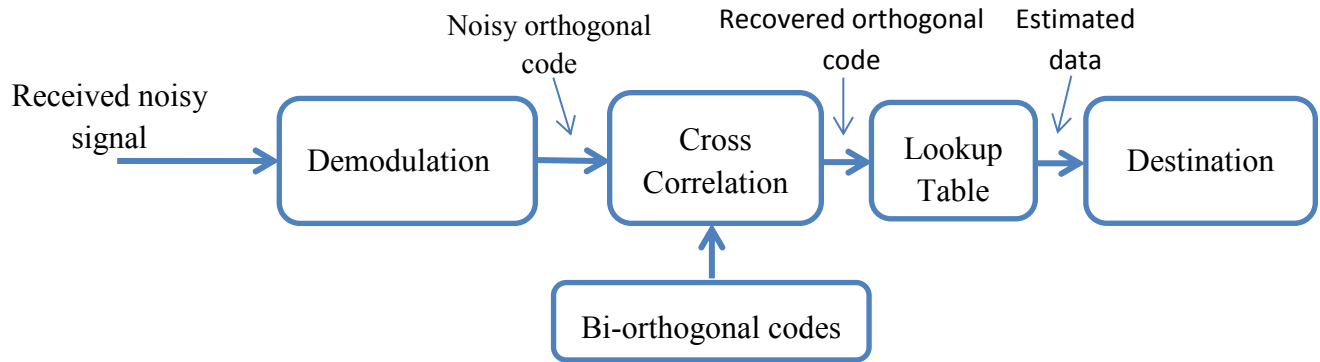


Figure 5.2 General block diagram of orthogonal coded modulation receiver.

Table 5.1 Orthogonal code mapping table for (a) $n = 4$ and (b) $n = 8$.

Data $2^3 = 8$	Mapped orthogonal code $n=4$
000	0000
001	0101
010	0011
011	0110
100	1111
101	1010
110	1100
111	1001

(a)

Data $2^4 = 16$	Mapped orthogonal code $n=8$
0000	0000 0000
0001	0101 0101
0010	0011 0011
0011	0110 0110
0100	0000 1111
0101	0101 1010
0110	0011 1100
0111	0110 1001
1000	1111 1111
1001	1010 1010
1010	1100 1100
1011	1001 1001
1100	1111 0000
1101	1010 0101
1110	1100 0011
1111	1001 0110

(b)

5.2 Multilevel Orthogonal Coded Modulation

In multilevel orthogonal coded modulation (MOCM) both incoming parallel data streams and bi-orthogonal code sets split into multiple levels, mapping each data set to the corresponding orthogonal code. Multilevel encoding structure is shown in Figure 5.3. The orthogonal code sets are generated by recursive method as explained in Section 4.2 and are stored in a Read Only Memory (ROM).

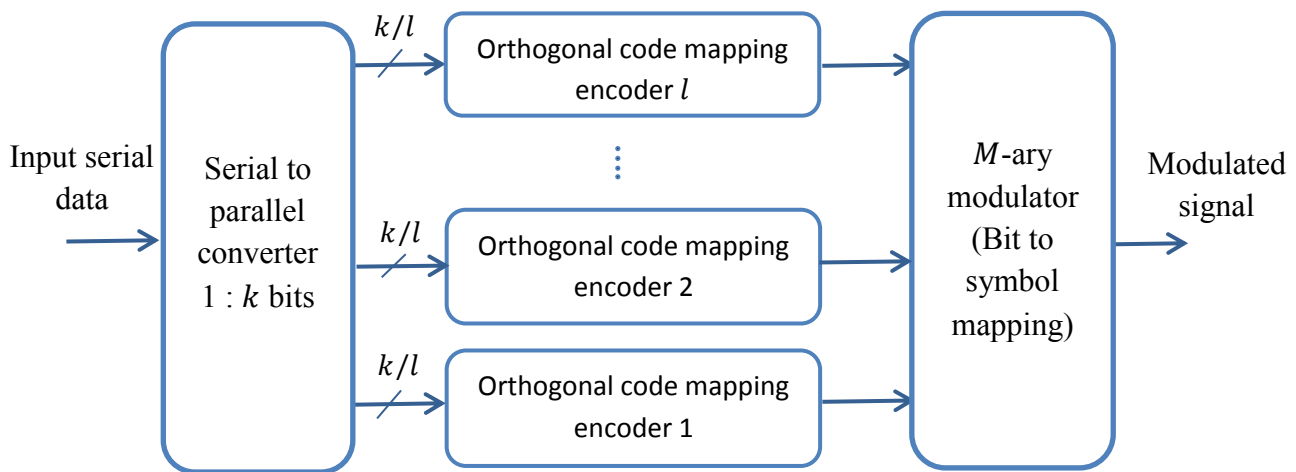


Figure 5.3 MOCM encoding Structures.

The incoming n bit serial data bits are partitioned into k bits parallel data streams. In normal single level of encoding the k -bit data are directly mapped to waveforms which requires 2^k waveforms. In multilevel orthogonal coding schemes the k -bit parallel data streams are partitioned into l sub-sets; each subset having length of k/l bits, used to address $2^{k/l}$ n -bit orthogonal codes stored in $2^{k/l} \times n$ ROMs. This arrangement reduces the number of mapping waveforms. There are total of l ROMs storing $2n$ biorthogonal codes, i.e. orthogonal and antipodal codes sets, with n -bit code lengths. The code rate of encoder is k/n . Thus, during

encoding, for transmission of k bits of data n -bit orthogonal code sets are transmitted. For example, normal single level mapping data block of $k = 8$, requires 256 waveforms but if we use multilevel mapping, splitting the data stream into 4 levels each having 2-bits, just $2^4 = 16$ waveforms are needed. These 16 waveforms are the 16-orthogonal codes of length 8. This provides the multilevel orthogonal coding a better bandwidth efficiency compared to normal mapping based coding. The partitioning of data into parallel forms and then to sub-sets allows multilevel encoding structures which reduces the required number of waveforms providing bandwidth efficiency. The output bits of the multilevel encoders at each instant from ROMs are combined to generate symbols for multilevel modulations which further enhance the bandwidth efficiency. Thus, combining multilevel orthogonal coding and multilevel M -ary modulation the system can correct multiple errors and provides bandwidth efficiency.

5.2.1 Different Code Rate Encoder Structures

In this section, we first explain the encoder structure for single level orthogonal coded modulation (rate $\frac{1}{2}$ with 8-bit orthogonal code). Being single level, it can correct single error as specified in Table 4.1 in Chapter 4. Then various other encoding structures of different code rates are presented. Coding schemes for higher code lengths can be constructed by using the parameters as specified in Table 5.2.

5.2.1.1 Rate $\frac{1}{2}$ Single Level Orthogonal Coded Modulation

Rate $\frac{1}{2}$ orthogonal coded modulation with an 8-bit orthogonal code [37] is the simplest orthogonal coded modulation with a single level encoder. This encoder can be constructed by inverse multiplexing the incoming serial data stream, into 4-parallel streams as shown in Figure 5.4. These parallel bit streams are used to address sixteen 8-bit orthogonal codes, stored in a 16

× 8 ROM. The output of the ROM is a unique 8-bit orthogonal code, which is then modulated. The modulated signal is then transmitted to channel. Since the configuration consists of single encoder the code rate is given by $R = 4/8 = 1/2$.

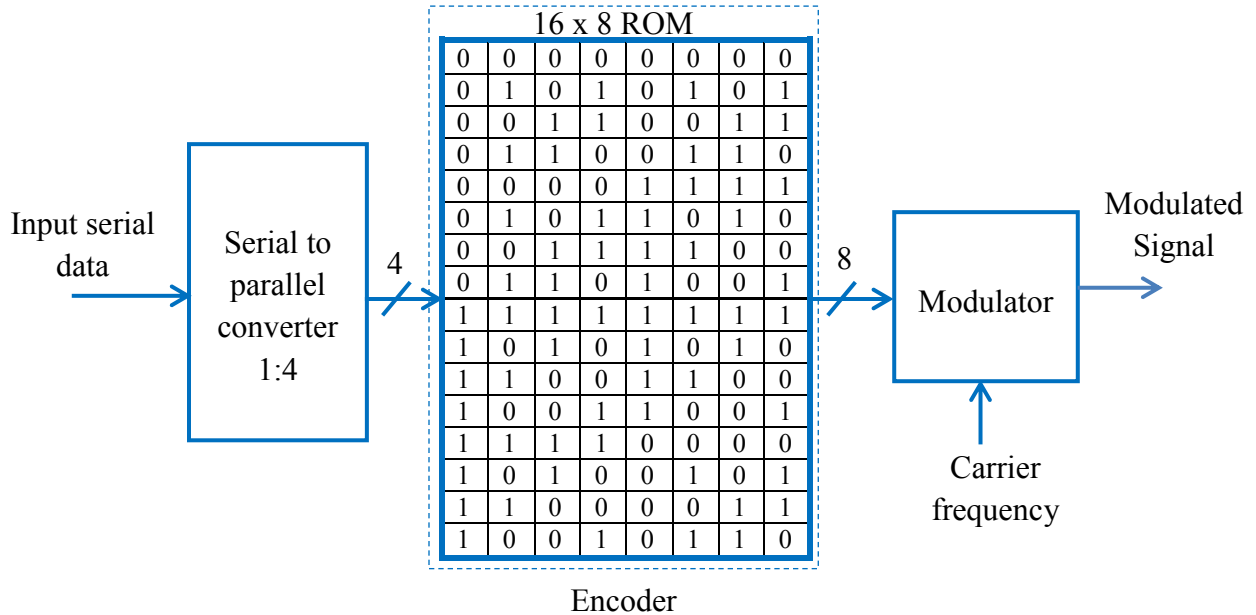


Figure 5.4 Rate 1/2 orthogonal coded modulation with $n = 8$ bits.

5.2.1.2 Rate 1/2 Multilevel Orthogonal Coded Modulation

As the code lengths increases, multilevel structures of rate 1/2 orthogonal coded modulations can be constructed. The MOCM structure with code length of $n=16$ is shown in Figure 5.5. The incoming serial data stream is converted into 8-bit parallel data streams. These parallel data streams are partitioned into two subsets, 4-bit per subset, which are used to address sixteen 16-bit orthogonal codes, stored in a pair of 16×16 ROMs. The outputs from each encoder are then taken one bit at a time resulting in a symbol of 2 bits (i.e. $m = 2$) which is modulated using 4-ary modulation (QPSK or 4-QAM) and finally transmitted to channel. Similarly, 2, 4 or 8 (i.e. $m = 4, 8, 16$) bits can be taken at each instant of time from each ROM, which can be modulated using

16-ary, 256-ary or 65,536-ary modulations. This system has two parallel encoders; each can correct three errors, so overall configuration can correct 6 errors.

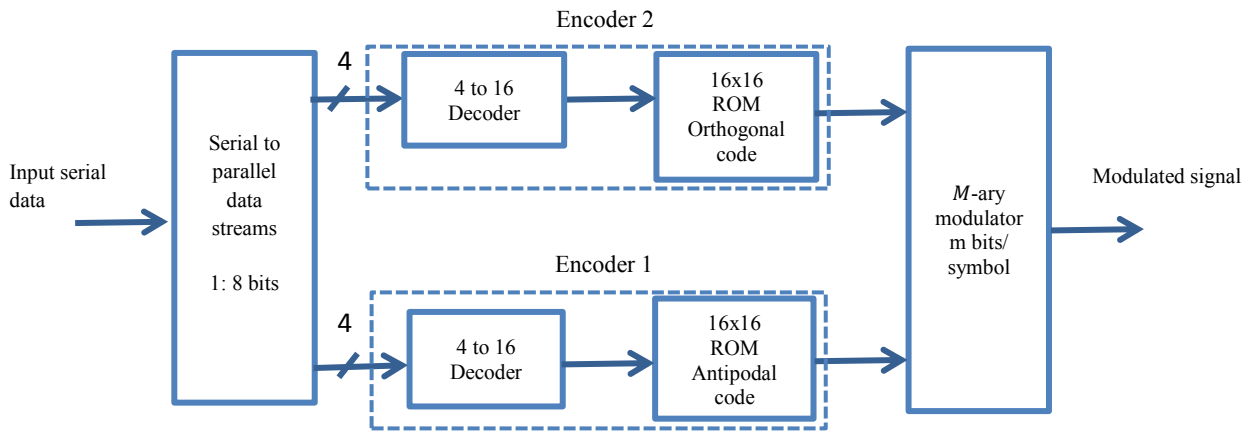


Figure 5.5 Rate $\frac{1}{2}$ multilevel orthogonal coded modulation with $n = 16$ bits.

5.2.1.3 Rate $\frac{3}{4}$ Multilevel Orthogonal Coded Modulation

Rate $\frac{3}{4}$ MOCM encoder structure with 8-bit orthogonal code has two levels as shown in Figure 5.6. For this structure, the incoming serial data stream is converted into 6-bit parallel data streams which are partitioned into two subsets, three bits per subset to address eight 8-bit orthogonal codes stored in a pair of 8x8 ROMs. The first 3 bits are used to address 8 orthogonal code set and next 3 bits are used to address 8 antipodal code sets in ROMs. Thus, we have two encoders in parallel. The code rate for each encode is $\frac{3}{8}$. Since the code rate is the sum of individual code rate the overall code rate is $\frac{3}{4}$. The multilevel output from each ROM/encoder can be taken one bit at a time resulting a symbol of 2 bits (i.e. $m = 2$ and $M=4$, where M is constellation points), having four constellation points, which can be used to modulate 4-ary modulation i.e. QPSK or 4-QAM (based on application) and the modulated waveform are finally transmitted to channel using antennas for radio frequency wireless communication or using lasers for optical communication. Similarly, 2, 4 or 8 (i.e. $m=4, 8, 16$) bits can be taken at

each instant of time from each ROM, which can be modulated using 16-ary, 256-ary or 65,536-ary modulations. This system has two parallel encoder blocks having 8-bit orthogonal codes; each can correct one error, so the configuration can correct 2 errors.

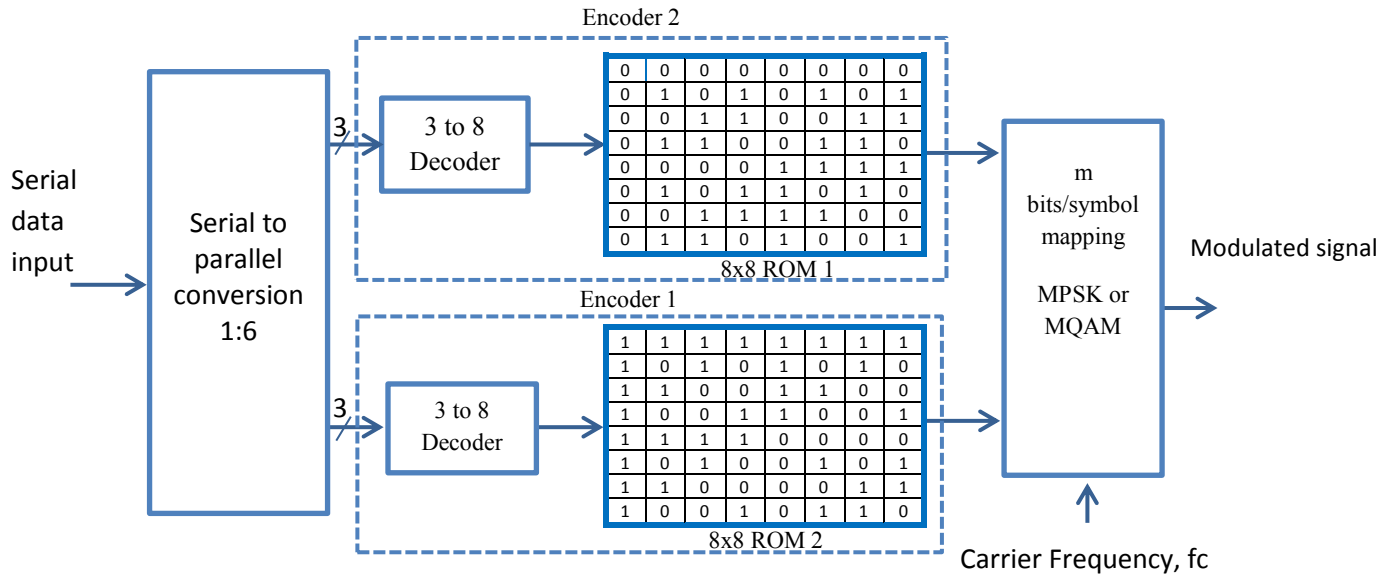


Figure 5.6 Rate $\frac{3}{4}$ multilevel orthogonal coded modulation with $n = 8$ bits.

Similarly, the rate $\frac{3}{4}$ MOCM encoding structure with orthogonal code length of $n=16$, as shown in Figure 5.7, can be constructed by converting the incoming serial data stream into 8-bit parallel data streams. These parallel data streams are partitioned into four subsets each having three bits to address eight 16-bit orthogonal codes stored in four 8×16 ROMs, one in each encoder. The code rate for each encoder is $\frac{3}{16}$ and the overall code rate is $\frac{3}{4}$. The output from each encoder is then taken one bit at a time resulting a symbol of 4 bits (i.e. $m = 4$ and $M=16$) which is modulated using M -ary modulation (16-PSK or 16-QAM) and finally transmitted to the communication channel. Similarly, taking 2, 4, or 8 (i.e. $m= 8, 16, 32$) bits from each ROM,

higher order modulated is possible. This multilevel encoding scheme has four parallel encoders; each correcting 3 error bits, hence the overall multilevel configuration can correct 12 errors.

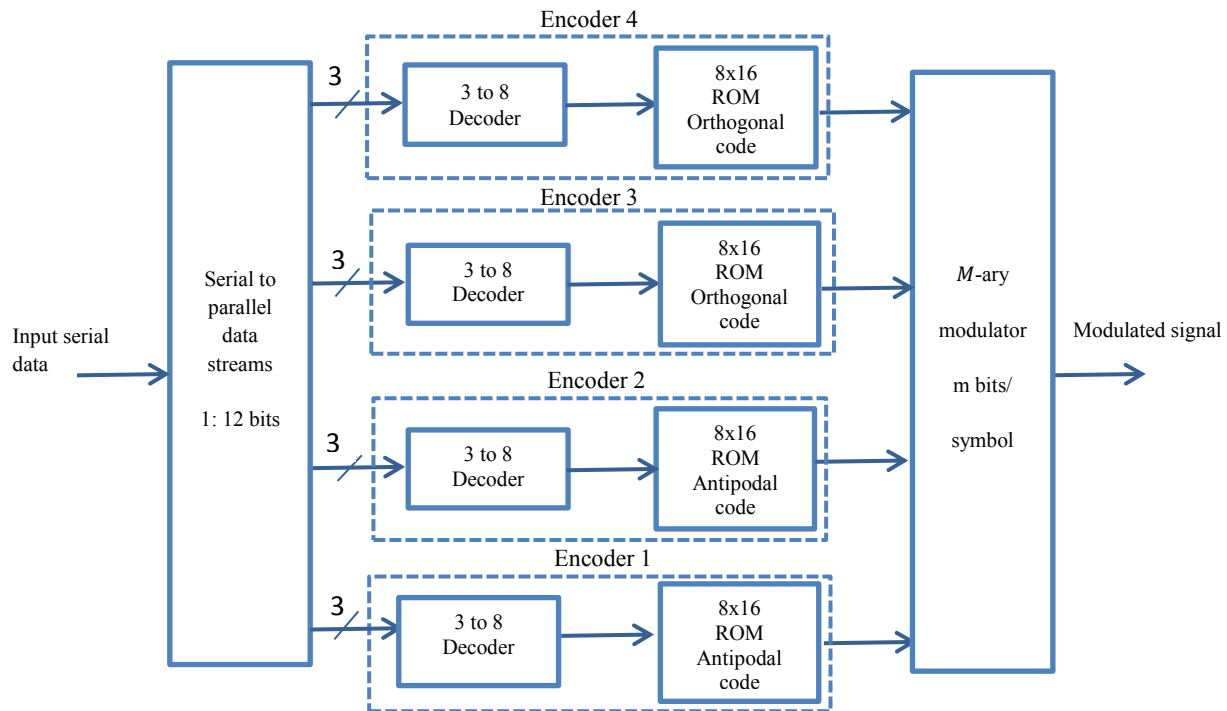


Figure 5.7 Rate $\frac{3}{4}$ multilevel orthogonal coded modulation with $n = 16$ bits.

4.2.1.4. Rate 1 Multilevel Orthogonal Coded Modulation

The rate 1 MOCM encoding structure with orthogonal code length $n=8$ is shown in Figure 5.8. It can be constructed by converting the incoming serial data stream into 8-bit parallel data streams. These parallel data streams are partitioned into four subsets, two bits per subset to address four 8-bit orthogonal codes stored in four 4×8 ROM, one in each encoder. The code rate for each encoder is $\frac{2}{8}$ and the overall code rate is 1. The output from each encoder is then mapped to the modulator using one bit at a time resulting a symbol of 4 bits (i.e. $m = 1$ and $M=16$) for M -ary modulation (16-PSK or 16-QAM) and finally transmitted to a communication channel. Similarly, 2, 4, or 8 (i.e. $m= 8, 16, 32$) bits can be taken at each instant of time from each ROM, which are

then modulated using corresponding M -ary modulations. This system has four parallel encoder blocks each can correct one error, so the multilevel configuration can correct 4 errors.

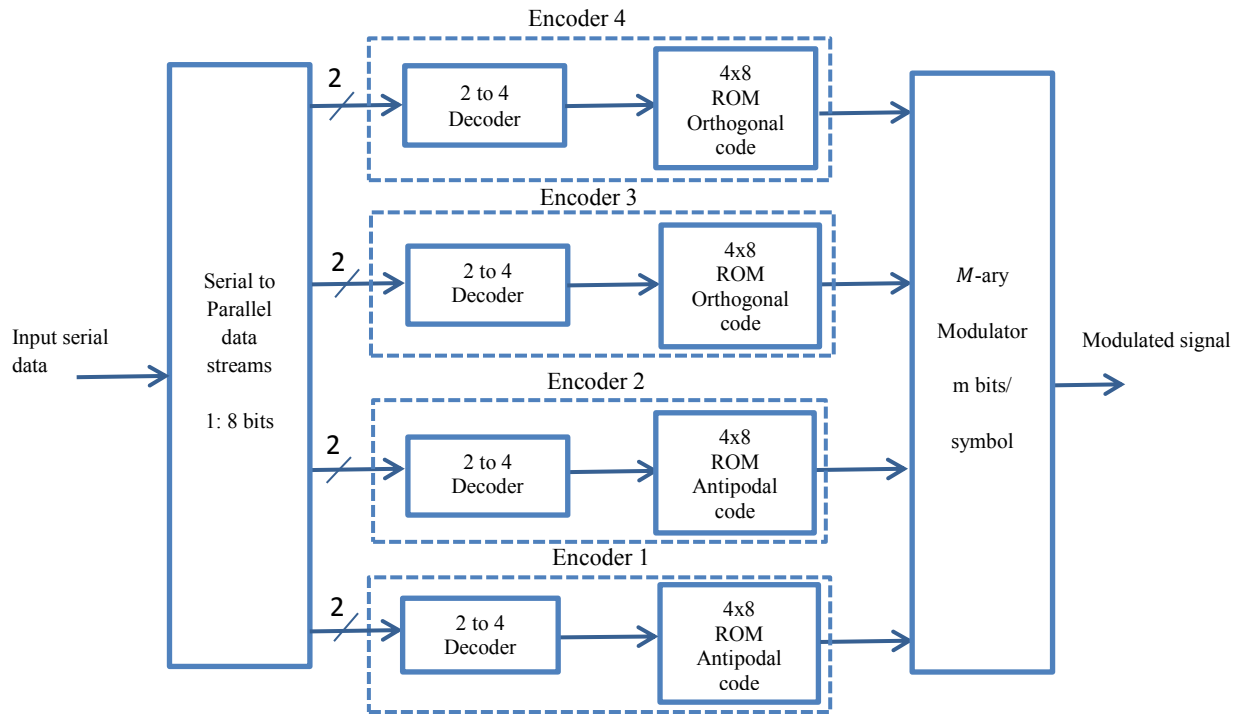


Figure 5.8 Rate 1 multilevel orthogonal coded modulation with $n = 8$ bits.

5.2.2 Construction Parameters for Higher Length Orthogonal Codes

Multilevel orthogonal coding schemes for higher code lengths can be constructed by using the parameters as specified in Table 5.2. As the code size increases, the number of level increases and complexity also increases, on the other hand the error correction capability and bandwidth efficiency also increases. Similar pattern can be used to construct the encoder for code lengths higher than 64 bits.

Table 5.2 Different code rates construction parameters for different orthogonal code lengths.

Orthogonal code length (n bits)	Code rate 1/2		Code rate 3/4		Code rate 1	
	Serial to parallel converter data length (k bits)	Number of encoder (l)	Serial to parallel converter data length (k bits)	Number of encoder (l)	Serial to parallel converter data length (k bits)	Number of encoder (l)
8	4	1	6	2	8	4
16	8	2	12	4	16	8
32	16	4	24	8	32	16
64	32	8	48	16	64	32

5.3 Parallel Independent Decoding at Individual Level for Multilevel Orthogonal Coded Modulation

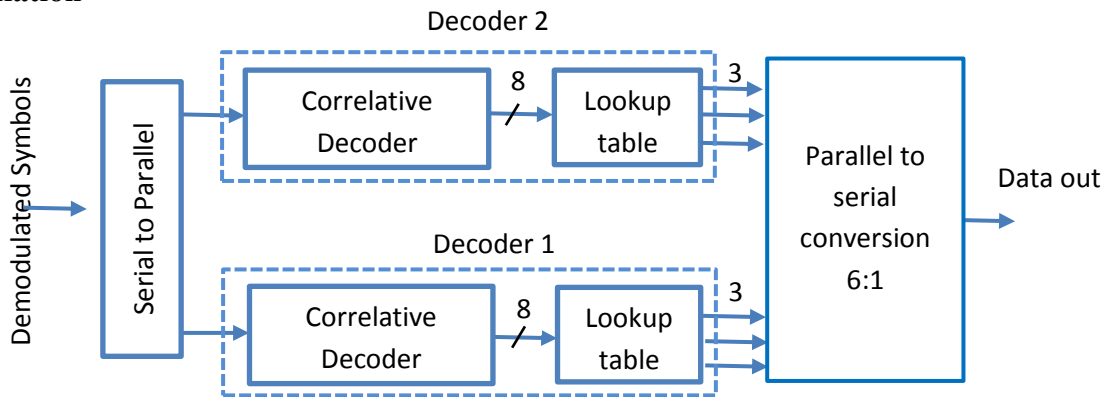


Figure 5.9 Multilevel correlation decoding for rate $3/4$ with $n = 8$.

The demodulated symbols at the receiver are grouped to form a unique impaired orthogonal codes of length n , which are then sent to the corresponding correlative decoder blocks for error detection and correction. In order to maintain the cross-correlation property of the orthogonal codes, there should be perfect synchronization in time during demodulation. The receiver consists of the same numbers of correlative decoder blocks as in encoders. The correlative decoder detects and corrects the error in the received code. First, the incoming orthogonal code

with errors is cross correlated with the known set of orthogonal codes (which are partitioned into l blocks as in the transmitter). Then, the code set from the output of the correlation process which gives smallest correlation value is chosen as a desired orthogonal code [35-41] for each decoder. These recovered orthogonal codes are then mapped to the closest original data blocks stored in the lookup table. The data block outputs from each lookup tables are combined and are then converted to serial data streams, which are finally collected at the information sink. Figure 5.9 shows the multilevel correlation decoding of rate $\frac{3}{4}$ with $n= 8$.

5.4. Error Correction Capability for Different Code Rate

As mentioned in Section 4.6 of Chapter 4, the average number of errors that can be corrected by using orthogonal codes and the code correlation process as decoding is given by ‘ t ’, which is equal to

$$t = \frac{n}{4} - 1 \quad (40)$$

where, n = orthogonal code length. As we notice in Section 5.2.1, each level can correct t errors, so, using multilevel encoding higher number of errors can be corrected depending upon the number of levels and code lengths. As the number of encoding levels as well as code lengths increases the error correction capabilities of the system also increases enabling multiple number of error corrections. The Table 5.3 shows the error correction capability of different length of orthogonal codes for different code rate. Thus, multilevel orthogonal encoding enables a large number of error correction capabilities.

Table 5.3 Orthogonal codes lengths and their error correction capabilities for different rates.

Orthogonal code length 'n' (bits)	Number of error corrected 't'		
	Rate 1/2	Rate 3/4	Rate 1
8	1	2	4
16	6	12	24
32	28	56	112
64	120	240	480

From Table 5.3, it can be observed that the multilevel orthogonal coding scheme offers significant error correction capability. For example, if rate 1 is implemented with 16-bit orthogonal code it has capability to correct 24 bit errors. The number of error corrections with vs. code length plotted in Figure 5.10. Rate 1 shows the highest number of error correction capability.

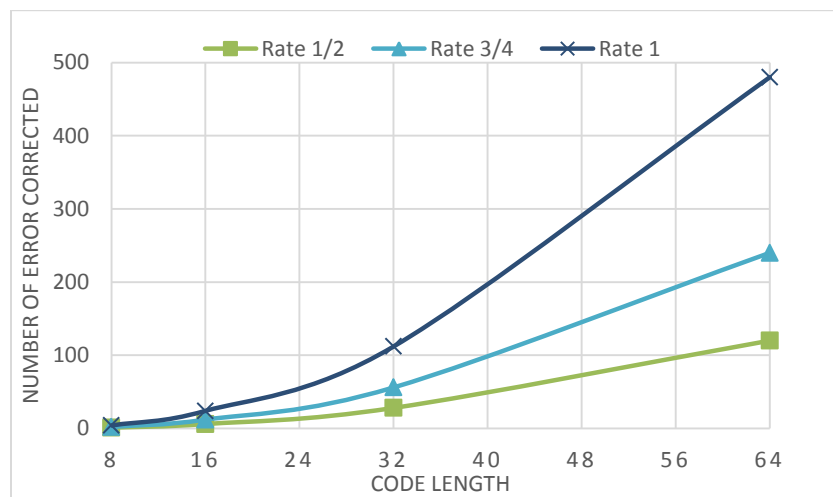


Figure 5.10 Relation showing the number of errors corrected for different code lengths.

CHAPTER 6

PERFORMANCE OF MULTILEVEL ORTHOGONAL CODED MODULATION IN DIFFERENT MODULATION TECHNIQUES

In this chapter, we present the MOCM systems performance analysis and simulation results for bandwidth efficient modulation schemes: M -ary PSK and M -ary QAM and their applications.

6.1 Orthogonal Coded M -ary PSK Modulation (OMPSK) for Optical Wireless Communications

OptoRadio is a method of laser based wireless communication system, which has wide bandwidth and high security capabilities [40]. The key concepts, underlying principles and practical realization of the *OptoRadio*, based on Orthogonal M -ary Phase Shift Keying (OMPSK) modulation is presented in this section.

OMPSK [37] is a coded multi-phase shift keying modulation technique, which map blocks of data to the blocks of bi-orthogonal code. In *OptoRadio*, mapped bi-orthogonal codes are modulated by an M -ary PSK modulator and transmitted to free space using laser beam. The modulations that can be used are BPSK, QPSK, 16-PSK and 256-PSK. At the receiver, two cross-coupled photodiodes are used to receive the laser beam so that net output power due to ambient light is close to zero. The laser signal is transmitted only in one of the photo diode. With all other signals being cancelled out, the laser signal is an overwhelmingly dominant signal [38].

Orthogonal ON-OFF Keying (O^3K) for free space laser communication is presented in [38]. Based on references [37,39], detailed analysis, MATLAB simulation of error probabilities, and

bandwidth efficiencies for implementation of orthogonal code based error control coding for subcarrier intensity modulated M -ary PSK for laser based wireless communication is conducted.

6.1.1 Block Diagram of *OptoRadio* System

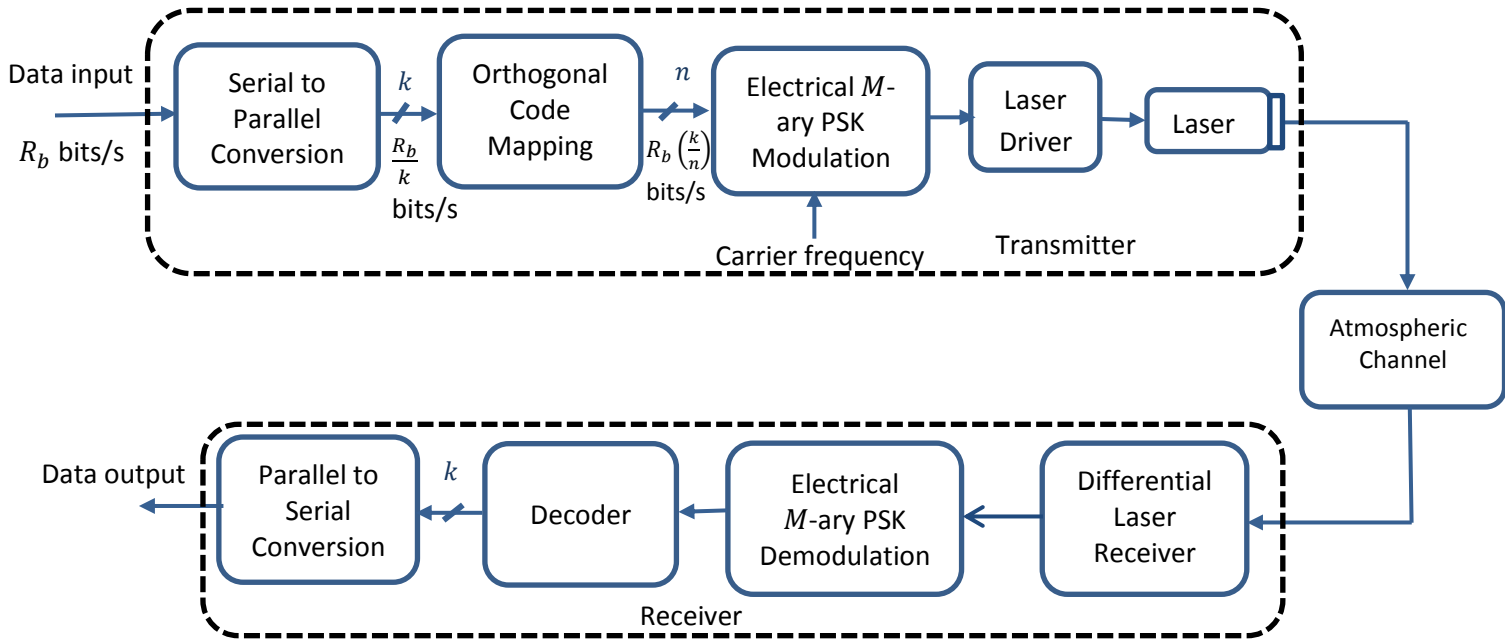


Figure 6.1 Block diagram of *OptoRadio* implementing OMPSK [40].

The general system block diagram for *OptoRadio* based on OMPSK modulation is shown in Figure 6.1. The transmitter consists of serial to parallel converter, orthogonal code mapping block, electrical M -ary PSK (MPSK) modulator, laser driver circuit, and laser diode. The incoming serial data streams are converted to k -bit parallel data stream. These k -bit parallel data are used to address the n -bit orthogonal code. The orthogonal code mapping block consists of biorthogonal codes. When k -bit of data transmission is required, n -bit orthogonal code is transmitted. The code rate at the output of encoder is k/n . The mapped orthogonal codes are modulated by electrical M -ary PSK modulator, which include radio frequency (RF) subcarrier. Pulse shape filtering such as raised cosine filtering can be used to maximize spectral efficiency

after modulation. Since the subcarrier signal is sinusoidal having both positive and negative values, a DC level is added to modulated signal before driving the laser diode [7]. The modulated signal is then used to drive the laser. The output of laser diode is used as optical source to transmit signal to atmospheric channel.

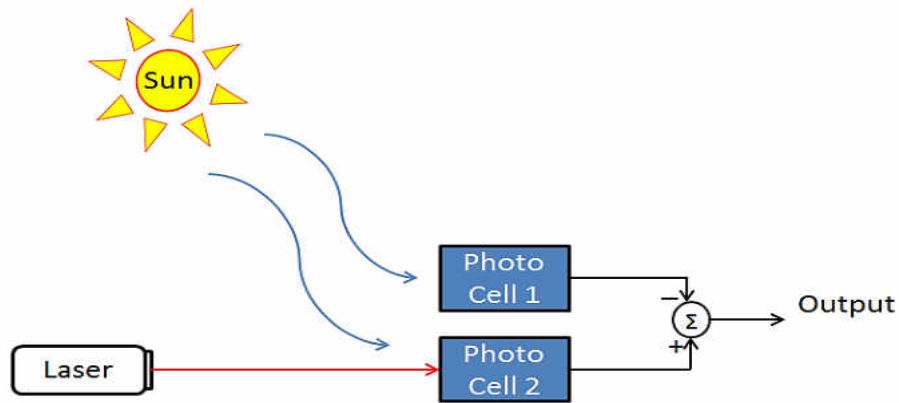


Figure 6.2 The ambient light cancellation technology [38].

At the receiver, photodetector receives the incoming optical radiations and convert them to electrical signals. One of such detection techniques is described in [38] which uses differential laser receiver that cancels the effect of sunlight and other atmospheric noises. Two cross-coupled photo-receivers' arrangement for ambient light cancellation is shown in Figure 6.2. The received electrical signal is then passed to the electrical RF M -ary PSK demodulator to recover transmitted symbols. The electrical bandpass filters are used before demodulation if multiple subcarriers are used [7]. The demodulated signal is then passed to correlative decoder, which detect and correct errors in transmitted signals during transmission. The correlative decoding is performed by using code correlation where incoming impaired orthogonal code is cross-correlated with the known set of orthogonal codes. The output of the correlation process with smallest correlation value is the desired orthogonal code. The decoded orthogonal code is then

mapped back to the parallel data stream. Finally, parallel data are converted to serial data stream and is collected at the information sink.

6.1.2 Multilevel OMPSK Encoder Structure

To illustrate the concept of coded modulation for *OptoRadio*, single level orthogonal coded modulation with rate $\frac{1}{2}$ orthogonal coded M -ary PSK modulation scheme with orthogonal codes length $n = 8$ is shown in Figure 6.3. For this encoder scheme, the incoming serial data stream with data rate R_b (b/s) is converted into 4-parallel data streams ($k = 4$). These parallel data stream, reduced in speed to $R_b/4$ (b/s) are used to address sixteen 8-bit orthogonal codes, stored in a 16×8 ROM.

Output of ROM is a unique 8-bit orthogonal code which can be taken ‘ m ’ bit at a time, where ‘ m ’ is a binary number and its relation with M can be represented as $M = 2^m$. For rate $\frac{1}{2}$ 8-bit orthogonal coded configurations, the bit taken for mapping from the ROM, m can be 1, 2, 4, and 8. Thus coded bits can be modulated using BPSK, QPSK, 16-PSK or 256-PSK subcarrier intensity modulation. Similarly, for 16-bit orthogonal code m can be 1, 2, 4, 8, and 16. So BPSK, QPSK, 16-PSK and 256-PSK subcarrier intensity modulations can be used to modulate the coded data. The electrical modulated signals, which are in orthogonal spaces, are then used to modulate the laser beam. DC bias is added before driving the laser to avoid clipping. For this scheme, since there is only one orthogonal waveform, the number of errors that can be corrected is one.

Similarly, rate $\frac{3}{4}$ can be constructed by serial to 6-bit parallel conversion of incoming data stream. These 6-bit parallel data are partitioned into two subsets, 3-bits per subset to address 16 orthogonal code blocks. The mapped orthogonal codes outputs from two 8×8 ROM can be taken as 2 bits/symbol at a time, i.e. $m=2$, which are then modulated using QPSK subcarrier

intensity modulation and finally transmitted using laser to free space. This scheme can correct two error bits.

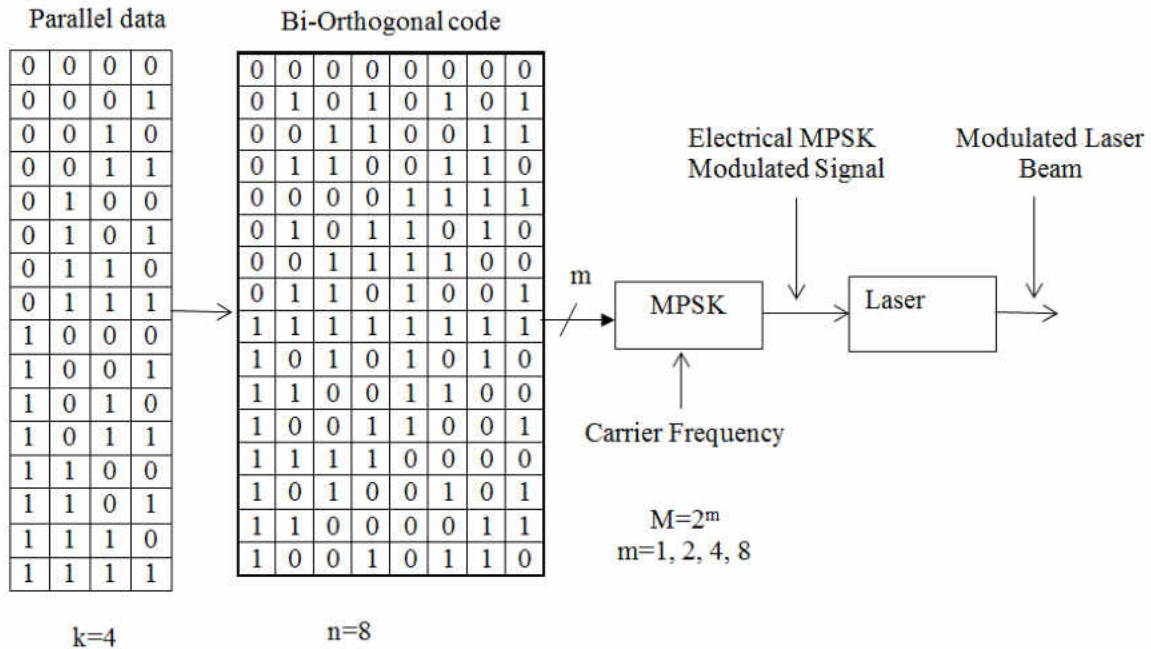


Figure 6.3 Rate 1/2 orthogonal coded M-ary PSK modulation with $n = 8$ [40].

Similarly, rate 1 orthogonal coded modulation can correct 4 errors, which can be constructed by inverse multiplexing incoming traffic into 8-bit parallel data stream, which are partitioned into three subsets, 2-bits per subset to address 16 orthogonal code. These mapped orthogonal codes can be taken 4 bits at a time and are modulated using 16-PSK subcarrier intensity modulation, which are then transmitted to free space using laser beam.

By using similar approach and using Table 5.2 various other multilevel encoding structures can be constructed.

6.1.3 Transmission Bandwidth and Bandwidth Efficiency

The proposed multilevel OMPSK scheme is bandwidth efficient. The approximate transmission bandwidth (BW) (*null-to-null bandwidth*) for orthogonal M -ary PSK can be calculated as,

$$BW \approx \left(\frac{2R_b}{m}\right) \left(\frac{n}{k}\right) \text{ Hz} \quad (41)$$

where, R_b is uncoded bit rate (bits/sec), k is number of parallel data bit streams, n is orthogonal code length, and $m = \log_2 M$ (number of bits per symbol) and its value may be 1, 2, 4, 8 for 8-bit orthogonal code, and 1, 2, 4, 8, 16 for 16-bit orthogonal code, and so on. The transmission bandwidth and bandwidth efficiency for three different code rates using M -ary PSK modulation schemes for 8-bit orthogonal code is shown in Table 6.1.

Thus, bandwidth efficiency of OMPSK increases with increase in M . Also, increasing the rate, the bandwidth efficiency can be enhanced. For instance, the same bandwidth efficiency of rate $\frac{1}{2}$ with $M = 16$ can be obtained with $M = 4$, if encoded with rate 1 coding. Similar results are valid for other data observed from Table 6.1.

Table 6.1 OMPSK transmission bandwidths and bandwidth efficiencies.

Constellation points (M)	Bandwidth (Hz)			Bandwidth Efficiency (ρ)		
	Rate 1/2	Rate 3/4	Rate 1	Rate 1/2	Rate 3/4	Rate 1
2	$4R_b$	$8R_b/3$	$2 R_b$	$\frac{1}{4}$	$\frac{3}{8}$	$\frac{1}{2}$
4	$2R_b$	$4R_b/3$	R_b	$\frac{1}{2}$	$\frac{3}{4}$	1
16	R_b	$2R_b/3$	$R_b/2$	1	$\frac{3}{2}$	2
256	$R_b/2$	$R_b/3$	$R_b/4$	2	3	4

6.1.4 Error Performance of OMPSK

As detailed in Sections 4.6 and 5.4, the average number of error that can be corrected by using orthogonal codes as a forward error correction code is given by 't'. Based on that, the number of bit errors that can be correct by using multilevel channel coding is shown in Table 5.3.

In orthogonal coded modulation, k -bit data set is replaced by an n -bit orthogonal code, where $n > k$. Then code rate is k/n . Let R_b be the uncoded bit rate, then, coded bit rate can be written as $R_c = (n/k) R_b$. Since $n > k$, coded bit rate R_c will be greater than uncoded bit rate R_b by a factor of n/k . Thus, coded bit energy (E_c) is less than uncoded bit energy (E_b), i.e. $E_c < E_b$. If S is the transmit carrier power, then the uncoded bit energy and coded bit energy are given by [2, 3]:

$$E_b = \left(\frac{S}{R_b} \right) \quad (42)$$

$$E_c = E_b \left(\frac{k}{n} \right) = \left(\frac{S}{R_b} \right) \left(\frac{k}{n} \right) \quad (43)$$

The theoretical uncoded bit error probability or BER P_u for BPSK modulation over AWGN channel without atmospheric turbulence and fading is given by:

$$P_u = \frac{1}{2} \operatorname{erfc} \left(\sqrt{\frac{E_b}{N_0}} \right) = \frac{1}{2} \operatorname{erfc} \left(\sqrt{\frac{S}{N_0 R_b}} \right) \quad (44)$$

Similarly, the probability of symbol error P_u for M -ary PSK over the AWGN channel without atmospheric turbulence and fading is given in (23).

With uncoded bit energy E_b and using (42), probability of symbol error of (23) can be written as:

$$P_u \approx \operatorname{erfc} \left(\sqrt{\frac{mE_b}{N_0}} \sin \frac{\pi}{M} \right) = \operatorname{erfc} \left(\sqrt{\frac{mS}{N_0 R_b}} \sin \frac{\pi}{M} \right) \quad (45)$$

where m is the number of bits in a symbol. The coded symbol error probability for M -ary PSK modulation can be written as

$$P_c \approx \frac{1}{2} \operatorname{erfc} \left(\sqrt{\frac{E_c}{N_0}} \right) = \frac{1}{2} \operatorname{erfc} \left[\left(\sqrt{\frac{S}{N_0 R_b}} \left(\frac{k}{n} \right) \right) \right] \quad \text{for BPSK} \quad (46)$$

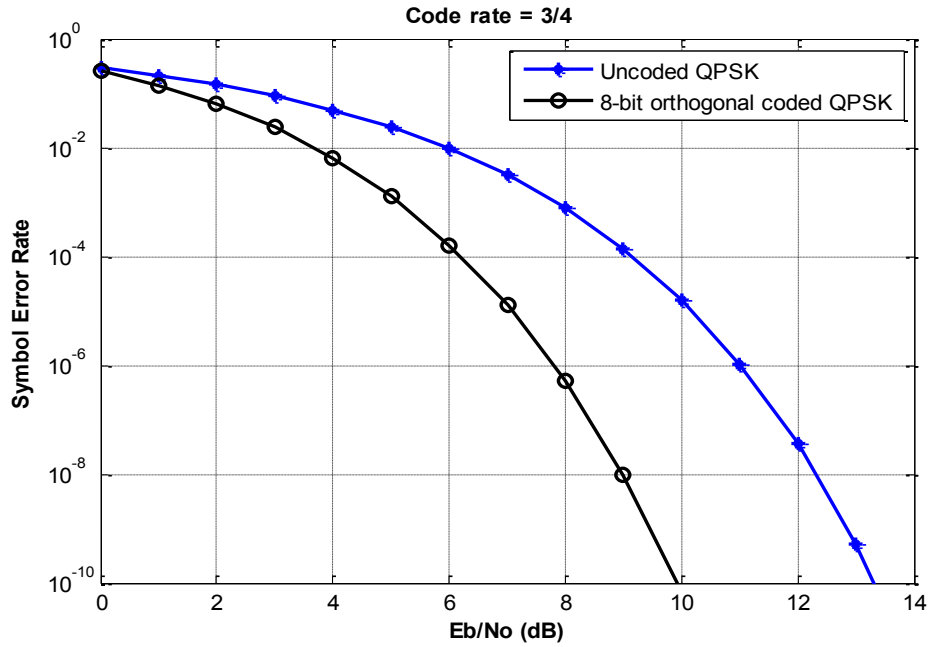
$$P_c \approx \operatorname{erfc} \left(\sqrt{\frac{mE_c}{N_0}} \sin \frac{\pi}{M} \right) = \operatorname{erfc} \left[\left(\sqrt{\frac{mS}{N_0 R_b}} \left(\frac{k}{n} \right) \right) \sin \frac{\pi}{M} \right] \quad \text{for MPSK and } M > 2 \quad (47)$$

The coding gain is the difference in $\frac{E_b}{N_0}$ between uncoded and coded word error, which can be found by using (48) and (49) for AWGN channel without atmospheric turbulence and fading as [2, 3]:

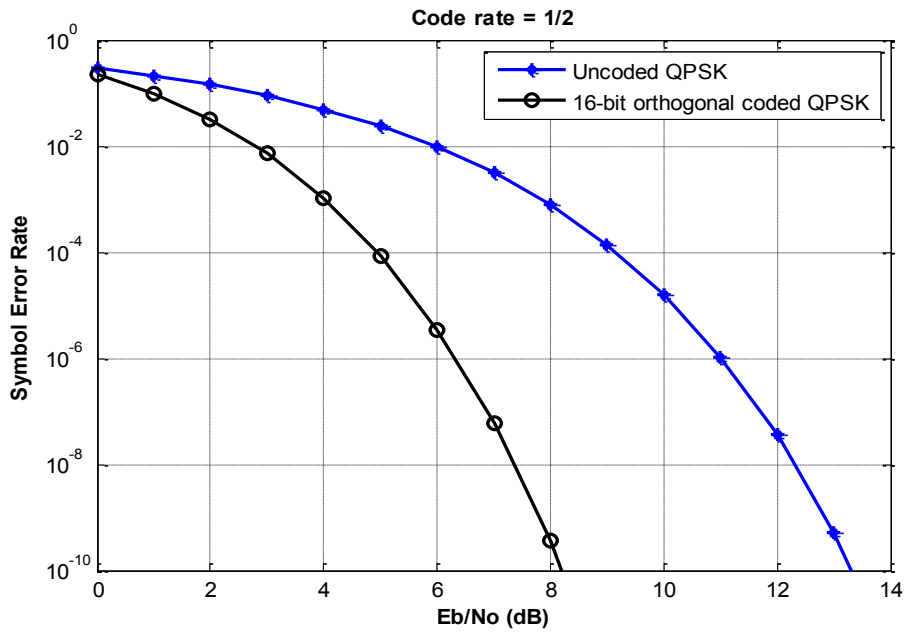
$$P_e(U) = 1 - (1 - P_u)^m \quad (48)$$

$$P_e(C) = \sum_{i=t+1}^n \binom{n}{i} P_c^i (1 - P_c)^{n-i} \quad (49)$$

where P_u and P_c are uncoded and coded symbol error rate (SER), t is the maximum error that can be corrected by the code and n is code length. The probability of symbol error for rate $\frac{3}{4}$ with QPSK modulation and rate 1 with 16-PSK modulation for 8-bit orthogonal code is shown in Figure 6.4 (a) and Figure 6.5 respectively, which are compared with the uncoded modulations. Similarly, probability of symbol error for 16-bit orthogonal coded QPSK modulation compared with uncoded QPSK is presented in Figure 6.4 (b). Significant coding gain can be observed in all cases. Coded rate 1 16-PSK modulation shows better performance than rate $\frac{3}{4}$ and rate 1.



(a)



(b)

Figure 6.4 Theoretical SER plot of (a) coded (8-bit) QPSK with code rate $\frac{3}{4}$ and (b) coded (16-bit) QPSK with code rate $\frac{1}{2}$, compared with uncoded QPSK.

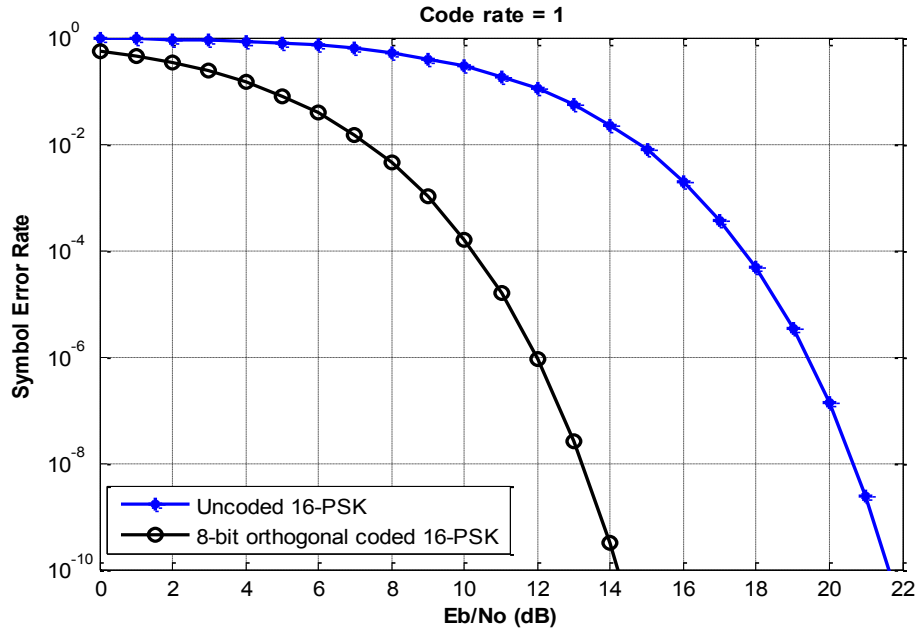


Figure 6.5 Theoretical SER plot of coded (8-bit) and uncoded 16-PSK using AWGN channel.

6.1.5 Simulation Results

Table 6.2 Simulation parameters.

Parameters	Rate 3/4	Rate 1	Rate 1/2
Constellation points (M)	4	16	4
Bits per symbol (m)	2	4	2
Modulation	QPSK	16-PSK	QPSK
Total number of bits transmitted	50,000	50,000	50,000
E_b/N_0	0 to 12 dB	0 to 19 dB	0 to 11 dB
Orthogonal code length (n -bit)	8,16	8,16	16
Fading	No	No	No
Channel	AWGN	AWGN	AWGN

The MOCM system as described in Chapter 5, combined with multilevel PSK modulation was simulated in MATLAB. First rate $\frac{3}{4}$ multilevel coded modulation is implemented using 8-bit and 16-bit orthogonal codes (Figures 5.6 and 5.7). The symbol error rate (SER) curves of the simulated system is plotted as shown in Figure 6.6. The figure shows that using 8-bit orthogonal code there is approximately 1.8 dB of coding gain and with the use of 16-bit code there is

approximately 3.8 dB of coding gain at SER of 10^{-3} . Also the number of bit errors before and after decoding, and number of errors corrected at various values of E_b/N_0 is tabulated in Tables 6.3 and 6.4.

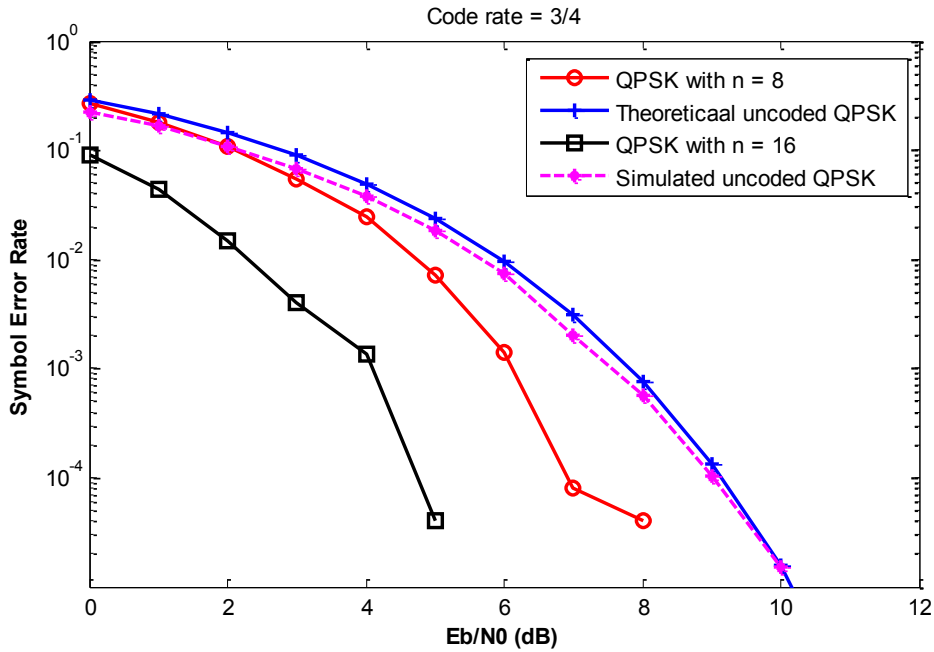


Figure 6.6 Error performance of uncoded QPSK and rate $\frac{3}{4}$ coded QPSK with code lengths $n = 8$ and 16 bits.

Table 6.3 Number of bit errors due to AWGN for rate $\frac{3}{4}$, 8-bit orthogonal coded QPSK before and after decoding for different values of E_b/N_0 .

E_b/N_0 (dB)	Number of bit errors		Number of errors corrected
	Before decoding	After decoding	
0	20374	6631	13743
1	15903	4413	11490
2	11759	2658	9101
3	8057	1362	6695
4	5222	608	4614
5	2935	176	2759
6	1380	35	1345
7	623	2	621
8	221	1	220
9	58	0	58
10	12	0	12
11	0	0	0
12	0	0	0

Table 6.4 Number of bit errors due to AWGN for rate $\frac{3}{4}$, 16-bit orthogonal coded QPSK before and after decoding for different values of E_b/N_0 .

E_b/N_0 (dB)	Number of bit errors		Number of errors corrected
	Before decoding	After decoding	
0	40620	2240	38380
1	32202	1073	31129
2	23838	365	23473
3	16327	99	16228
4	10219	34	10185
5	5744	1	5744
6	2835	0	2835
7	1227	0	1227
8	421	0	421
9	99	0	9
10	21	0	21
11	0	0	0
12	0	0	0

The scatter plot for various values of E_b/N_0 for AWGN channel with QPSK modulation is presented in Figure 6.6. The effect of AWGN can be clearly observed in Figure 6.7 (cyan colored). As E_b/N_0 values increases the constellations points are clearly separated. The result is low error in bits during demodulation.

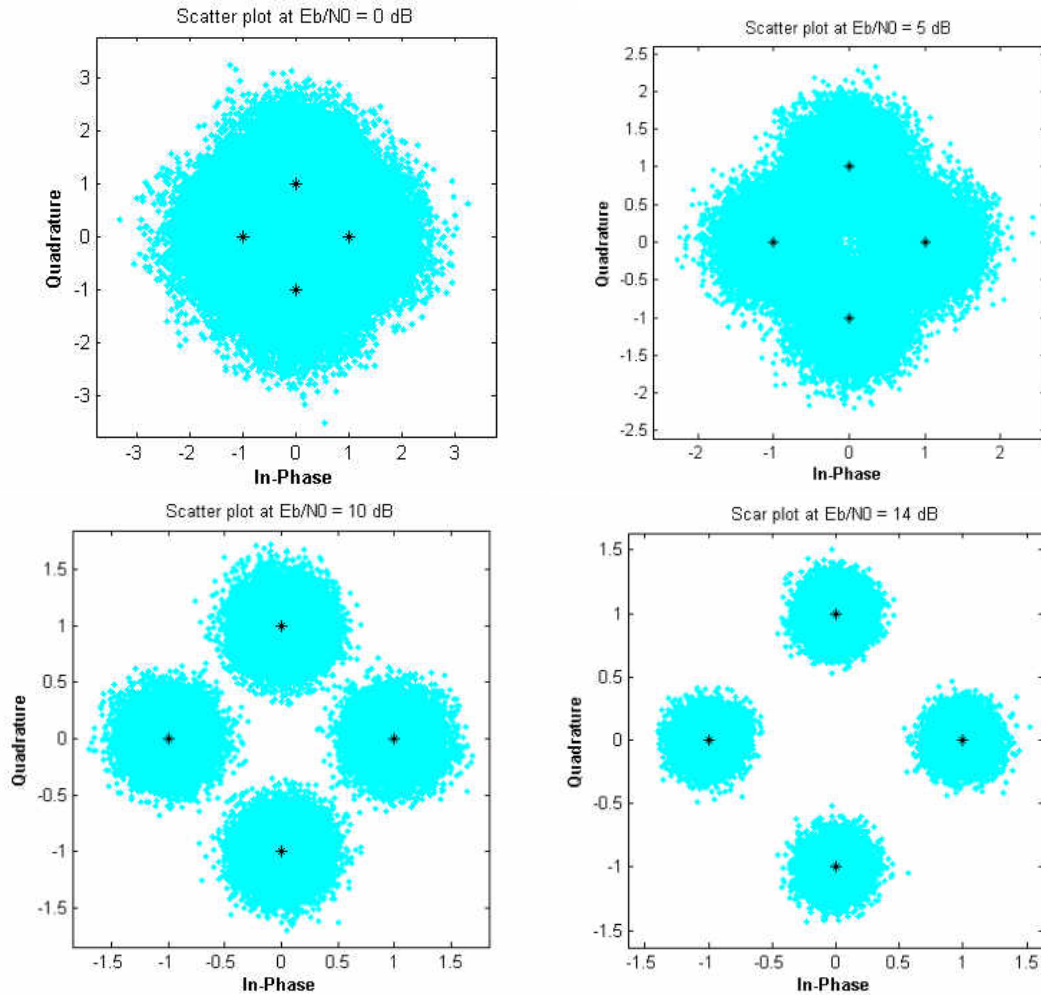


Figure 6.7 Scatter plots of QPSK for various values of E_b/N_0 .

Likewise, the simulated SER plot for multilevel 16-bit orthogonal coded rate $\frac{1}{2}$ with QPSK modulation is shown in Figure 6.8. The coding gain of approximately 1.9 dB is obtained at SER of 10^{-3} . Table 6.5 lists the counted bit errors before and after correlative decoding, and error corrected by the decoder. The 16-bit orthogonal coded QPSK modulation with rate $\frac{3}{4}$ and rate 1 are compared in Figure 6.9, showing that rate $\frac{3}{4}$ have coding gain of around 2 dB greater than rate $\frac{1}{2}$ for same code lengths.

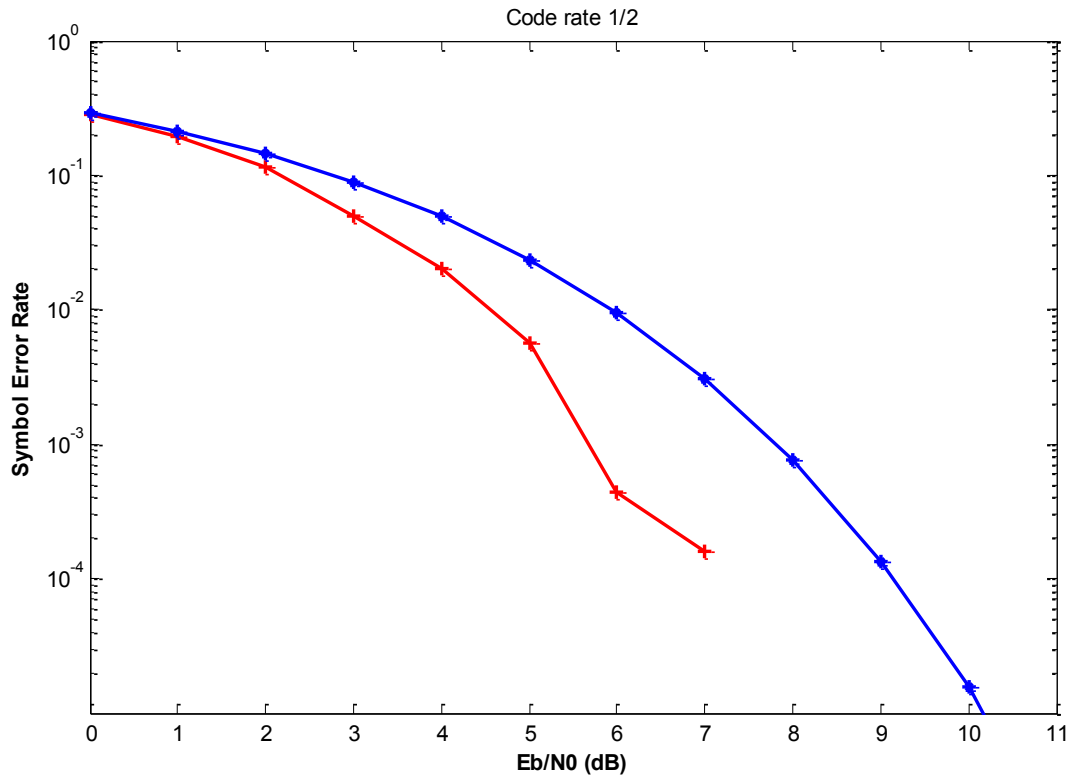


Figure 6.8 Error performance of uncoded QPSK and rate $\frac{1}{2}$ coded QPSK with code length $n = 16$ bits.

Table 6.5 Number of bit errors due to AWGN for rate $\frac{1}{2}$, 16-bit orthogonal coded QPSK before and after decoding for different values of E_b/N_0 .

E_b/N_0 (dB)	Number of bit errors		Number of errors corrected
	Before decoding	After decoding	
0	42680	7144	35536
1	35778	4798	30980
2	29359	2889	26470
3	22625	1261	21364
4	16482	531	15951
5	10926	140	10786
6	6977	11	6966
7	3777	4	3773
8	1702	0	1702
9	764	0	764
10	279	0	279
11	55	0	55

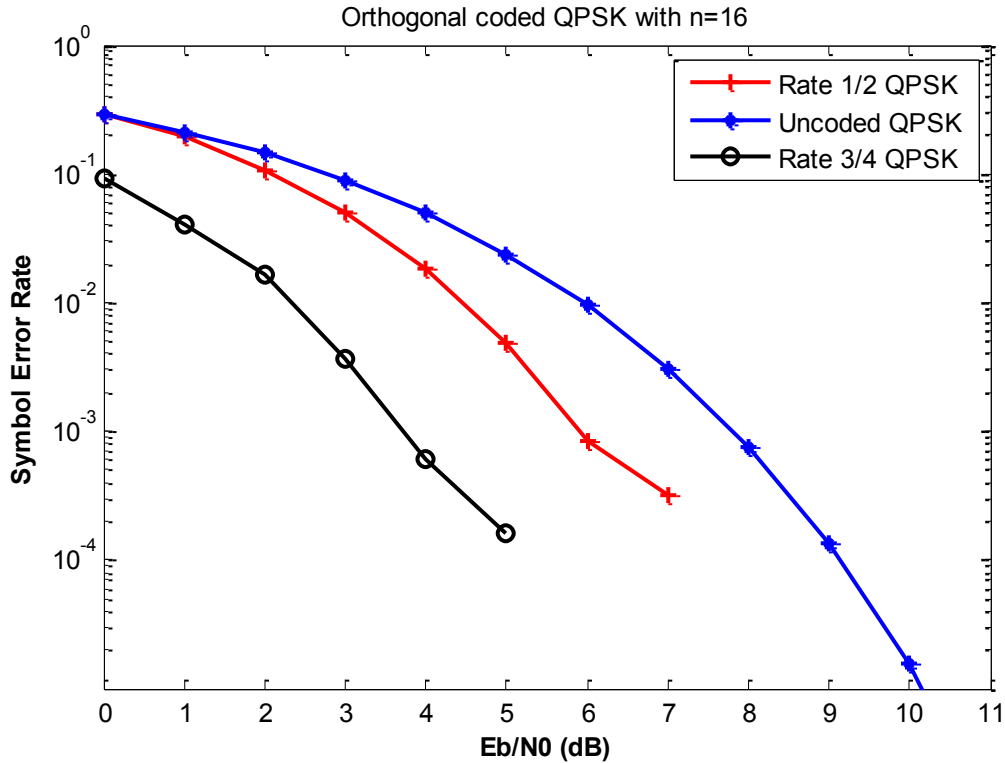


Figure 6.9 Error performance of uncoded QPSK, and rate $\frac{1}{2}$ and rate $\frac{3}{4}$ coded QPSK with code length $n = 16$ bits.

Similarly, for the rate 1 MOCM with 16-PSK modulation, the 8-bit orthogonal code shows approximately 3.2 dB of coding gain, and with the use of 16-bit code the coding gain is around 5.9 dB at SER of 10^{-3} as shown in Figure 6.10. Thus, increasing the code length from 8-bit to 16-bit additional coding gain of 2.7 dB is observed. The exact numbers of bit errors before and after decoding for rate 1 with 8-bit and 16-bit orthogonal code with 16-PSK modulation are tabulated in Tables 6.6 and 6.7 respectively. The table also shows the number of bit error corrected by demodulator.

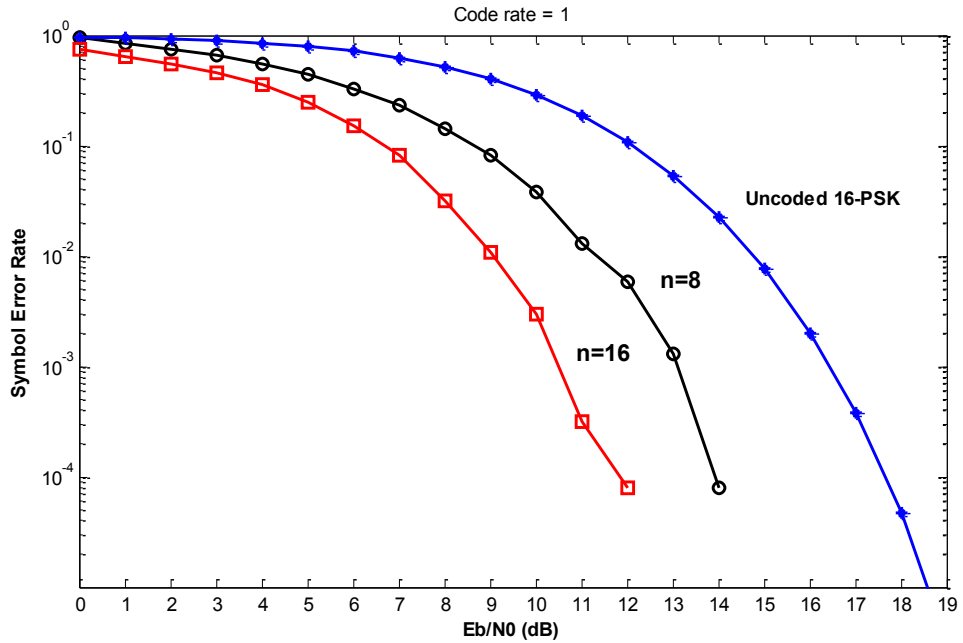


Figure 6.10 Error performance of uncoded 16-PSK and rate 1 coded 16-PSK with code lengths $n = 8$ and 16 bits.

Table 6.6 Number of bit errors for rate 1, 8-bit orthogonal coded 16-PSK before and after decoding for different values of E_b/N_0 .

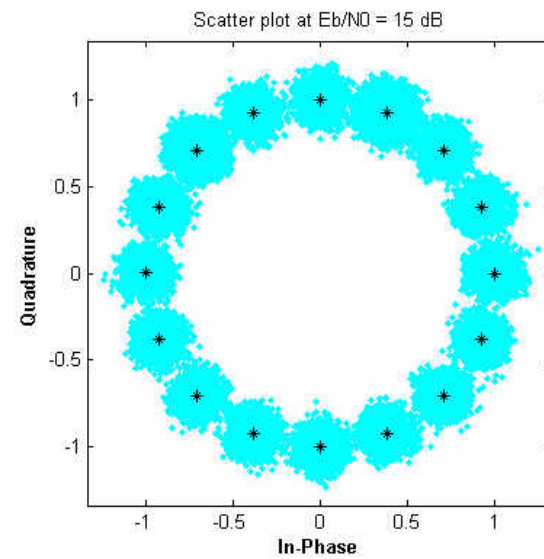
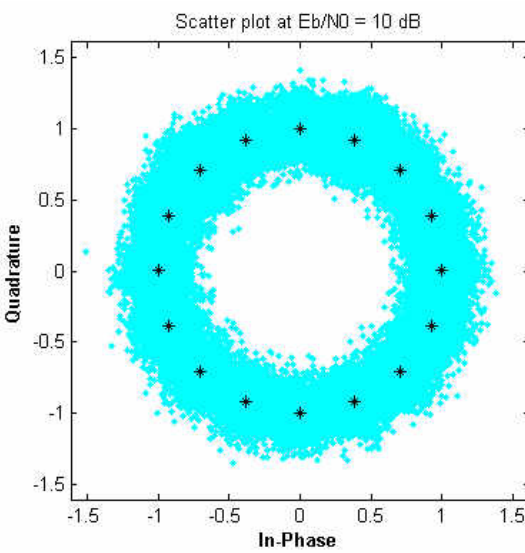
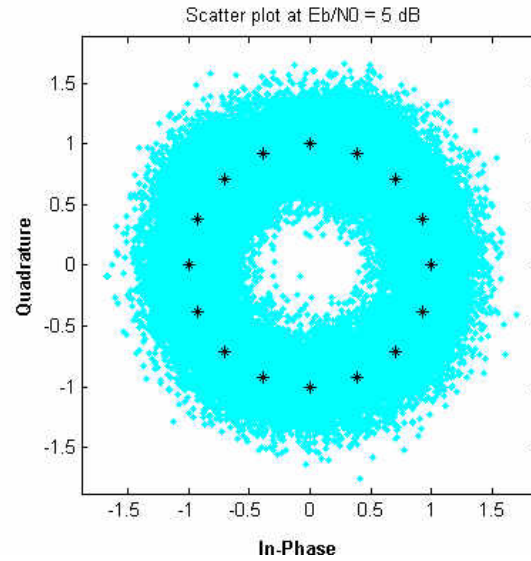
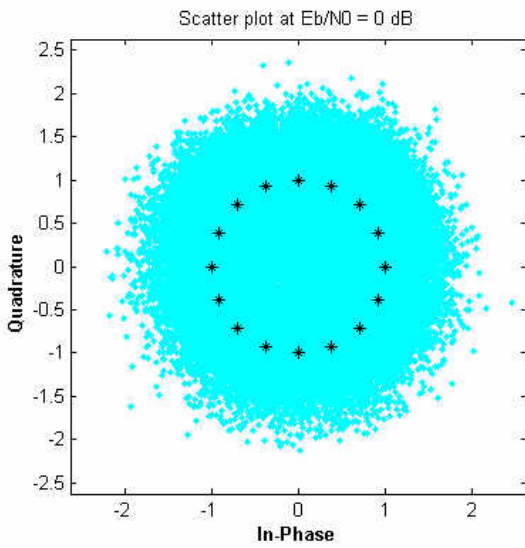
E_b/N_0 (dB)	Number of bit errors		Number of errors corrected
	Before decoding	After decoding	
0	53639	11849	41790
1	49559	10877	38682
2	44760	9572	35188
3	40316	8152	32164
4	34903	6809	28094
5	29848	5550	24298
6	24902	4110	20792
7	19823	2966	16857
8	15517	1944	13573
9	11135	1127	10008
10	7715	576	7139
11	4701	226	4475
12	2598	60	2538
13	1235	14	1221
14	509	2	507
15	213	0	213
16	39	0	39
17	14	0	14
18	0	0	0

Table 6.7 Number of bit errors for rate 1, 16-bit orthogonal coded 16-PSK before and after decoding for different values of E_b/N_0 .

E_b/N_0 (dB)	Number of bit errors		Number of errors corrected
	Before decoding	After decoding	
0	108171	9094	99122
1	100003	8057	91946
2	90612	6856	83756
3	81746	5637	76109
4	71647	4375	67272
5	60944	3053	57891
6	50610	1822	48788
7	40905	960	39995
8	31543	435	31108
9	22742	141	22601
10	15396	35	15361
11	9137	8	9129
12	5267	1	5266
13	2576	0	2576
14	1096	0	1096
15	396	0	396
16	98	0	98
17	24	0	24
18	2	0	2
19	0	0	0

The two-dimensional circular scatter plots for various values of E_b/N_0 for 16-PSK modulation are shown in Figure 6.11. The effect of AWGN is shown by the cyan color. The addition of random white Gaussian noise makes the constellation points scatter randomly from the exact transmitted constellation positions. As E_b/N_0 increases i.e. as the signal power is higher the constellations points are clearly separated. For example, at $E_b/N_0 = 20$ dB the constellation points are separated clearly, so the demodulator will be able to estimate the transmitted symbols easily, resulting in low bit/symbol errors after demodulation. Also as the value of M increases symbols are closer to each other, and there is more chance for error during demodulation

process. The bit errors after demodulations are corrected by the correlation decoder, which will minimize the errors that demodulator makes during the estimation of symbol.



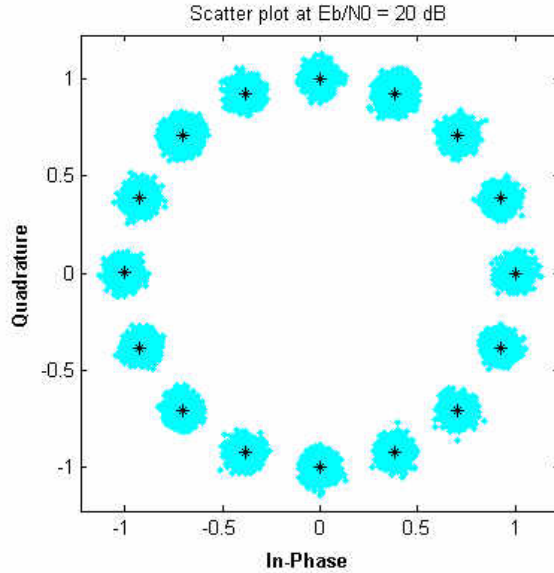


Figure 6.11 Scatter plots of 16-PSK for various values of E_b/N_0 .

6.2 Multilevel Orthogonal Coded M -ary QAM Modulation (OMQAM)

Multilevel orthogonal coded M -ary QAM [41] is the coded modulation techniques where orthogonal coded symbols are mapped to the M -ary QAM modulation and then transmitted to the channel. The multilevel orthogonal coding gives parallel outputs, these outputs are mapped directly to the constellation points of M -ary QAM modulation. The system block diagram, transmission bandwidths and bandwidth efficiencies of the system are presented in next section. Then the theoretical and simulated analyses of the error performances of the system are presented.

6.2.1 Block Diagram of Multilevel OMQAM System

The system block diagram for orthogonal coded M -ary QAM modulation is shown in Figure 6.12. At the transmitter, input serial data streams having data rate R_b bits/sec are converted to k -bit parallel data streams. These parallel data streams, which are reduced to data rate R_b/k

bits/sec will address n -bit orthogonal codes stored in ROM located at the orthogonal code mapping block. The ROM has $2n$ bi-orthogonal code sets, i.e. n orthogonal and n antipodal codes. Thus, for transmission of k -bit of data n -bit orthogonal code sets are transmitted [35-41]. The code rate is k/n . Parallel signal processing, i.e. multilevel encoding is performed during the encoding which will improve the error correction capability. The symbol outputs from the ROM are then modulated using M -ary QAM modulation. Pulse shaping filtering such as root raised cosine filtering is used to maximize spectral efficiency after modulation. Finally, the pulse shaped modulated signals, which are in orthogonal space, are transmitted to channel. The noise will affect the transmitted signal, during transmission, in channel. The AWGN channel is consider for analysis in this thesis.

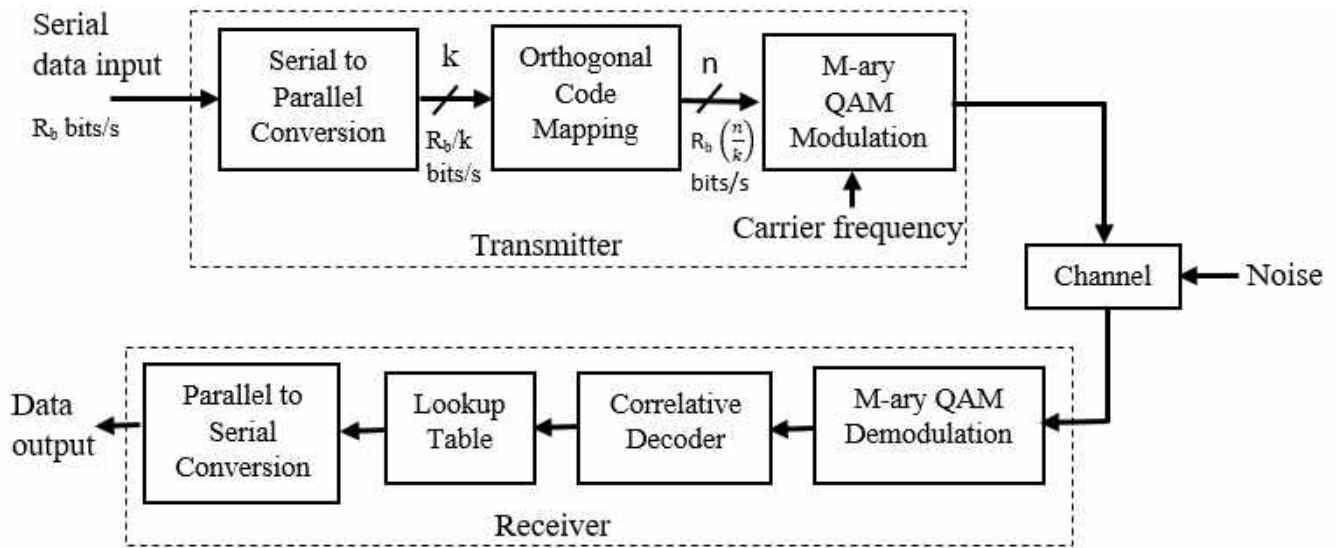


Figure 6.12 General block diagram of the Orthogonal Coded M -ary QAM (OMQAM) system with transmitter and receiver [41].

At the receiver, the received signal is passed to the M -ary QAM demodulator to recover transmitted symbols. The demodulated symbols are then passed to the correlative decoders, which detect and correct the errors introduced in the transmitted signals. The correlative

decoding is performed by using code correlation process where incoming impaired orthogonal codes are cross-correlated with the known set of orthogonal codes. The output of correlation process with the smallest correlation values are the desired orthogonal code sets. The decoded orthogonal codes are then mapped to original data in lookup table, which are then converted to serial form and are collected at the information sink.

6.2.2 Transmission Bandwidth and Bandwidth Efficiency

The proposed multilevel OMQAM is a bandwidth efficient system. The transmission bandwidth (BW) (*null-to-null bandwidth*) can be calculated as,

$$BW \approx \left(\frac{2R_b}{m}\right) \left(\frac{n}{k}\right) \quad (50)$$

where, R_b is uncoded bit rate (bits/sec), k is number of parallel data streams, n is orthogonal code length, and bits per symbol ' m ' = $\log_2 M$ and its value can be 2, 4, and 8 for 8-bit orthogonal code and 2, 4, 8, and 16 for 16-bit orthogonal code, and so on. The transmission bandwidths for three different M -ary QAM schemes for n -bit orthogonal code is shown in Table 6.8. Similarly, bandwidth efficiency in bits/second/hertz for various values of M is also presented in the Table 6.8.

Table 6.8 Transmission bandwidths and bandwidth efficiencies of multilevel OMQAM scheme.

Modulation	Bandwidth (Hz)			Bandwidth Efficiency (ρ) (bits/sec/Hz)		
	Rate 1/2	Rate $\frac{3}{4}$	Rate 1	Rate 1/2	Rate $\frac{3}{4}$	Rate 1
4-QAM	$2R_b$	$4R_b/3$	R_b	1/2	$\frac{3}{4}$	1
16-QAM	R_b	$2R_b/3$	$R_b/2$	1	$3/2$	2
256-QAM	$R_b/2$	$R_b/3$	$R_b/4$	2	3	4

Table 6.8 shows that transmission bandwidth of rate $\frac{1}{2}$ coded 16-QAM is same as that of input data rate. Similarly, rate $\frac{1}{2}$ coded 256-QAM, the transmission bandwidth is half of the input data rate. Similar results can be observed for other coding rates from the Table 6.8. Thus, multilevel OMQAM system is bandwidth efficient system and can be implemented in data transmission applications requiring higher bandwidth.

6.2.3 Error Performance of OMQAM

The probability of symbol error for uncoded QAM modulation for a rectangular constellation with $M = 2^m$ symbols where m is even, over AWGN channel is given in (29) and (30). Substituting (42) the uncoded probability of symbol error in terms of average transmit carrier power S_{avg} and uncoded bit rate R_b can be written as

$$P_u \leq 4Q \left(\sqrt{\left(\frac{3 \log_2 M}{M-1} \right) \frac{S_{avg}}{R_b N_0}} \right) \quad (51)$$

The coded symbol error probability for M -ary QAM modulation can be obtained by using (43) which can be written as,

$$P_c \leq 4Q \left(\sqrt{\left(\frac{3 \log_2 M}{M-1} \right) \frac{E_{bavg}}{N_0} \left(\frac{k}{n} \right)} \right) = 4Q \sqrt{\left(\frac{3 \log_2 M}{M-1} \right) \frac{S_{avg}}{N_0 R_b} \left(\frac{k}{n} \right)} \quad (52)$$

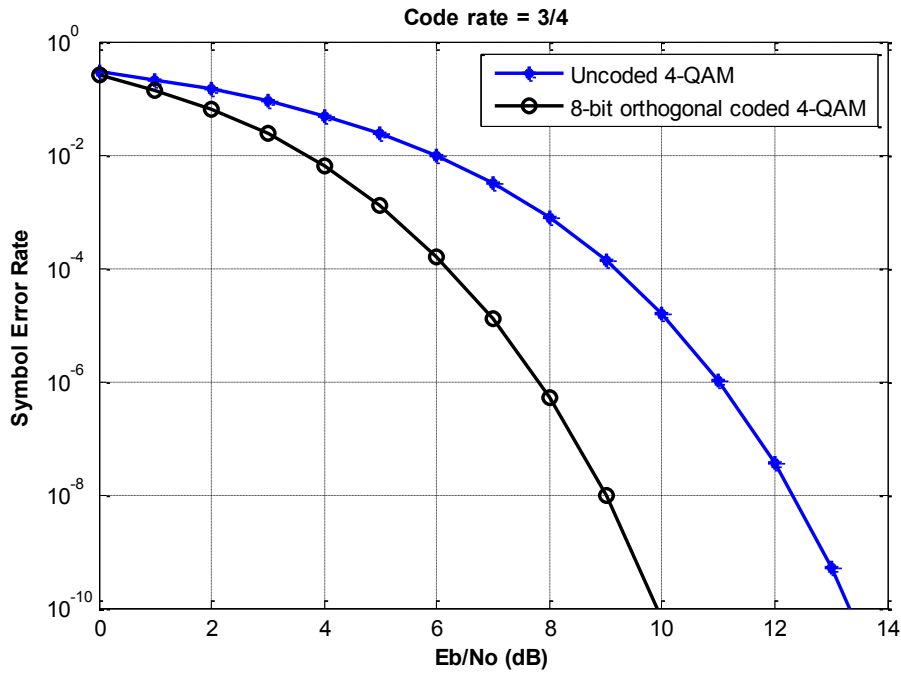
The coding gain is the difference in $\frac{E_b}{N_0}$ between uncoded and coded word error which can be obtained by using (51) and (52) for AWGN channel without fading [2,3]:

$$Pe(U) = 1 - (1 - P_u)^m \quad (53)$$

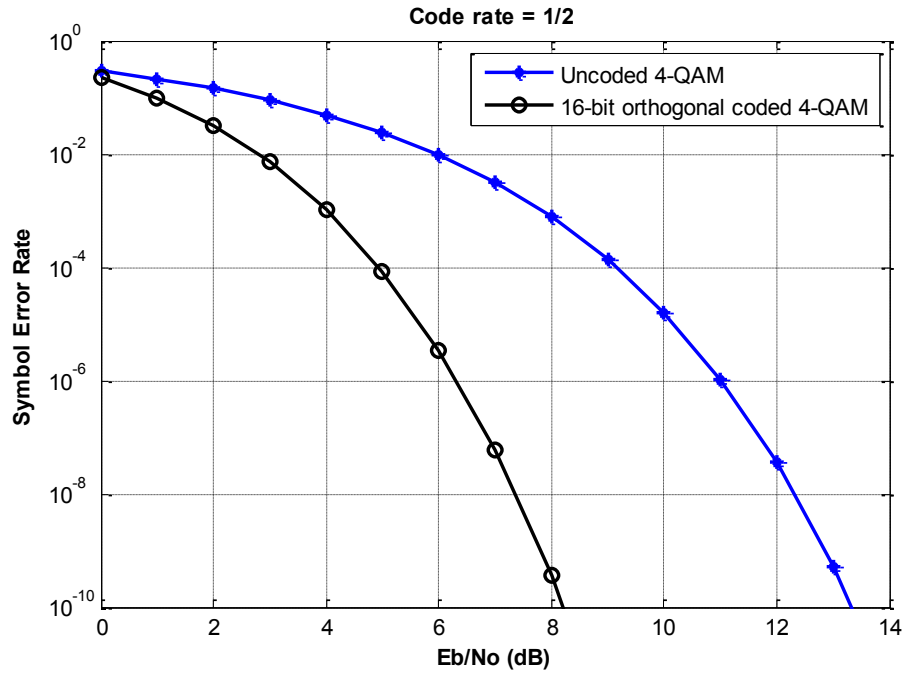
$$P_e(C) = \sum_{i=t+1}^n \binom{n}{i} P_c^i (1 - P_c)^{n-i} \quad (54)$$

where P_u and P_c are uncoded and coded symbol error rate, t is the maximum error that can be corrected by the code and n is code length.

Theoretical SER vs $\frac{E_b}{N_0}$ plot for uncoded 4-QAM and rate $\frac{3}{4}$ 8-bit coded 4-QAM system, and rate $\frac{1}{2}$ 16-bit coded 4-QAM systems are shown in Figure 6.13. At SER of 10^{-4} , the coding gain of about 3 dB is observed using rate $\frac{3}{4}$ 8-bit coding, while coding gain of about 4 dB is observed by using rate $\frac{1}{2}$ 16-bit coding. Similarly, theoretical SER plot for uncoded 16-QAM and rate 1 8-bit coded 16-QAM system is shown in Figure 6.14. The coding gain of approximately 6 dB can be achieved at SER of 10^{-4} . With coded modulation, the system shows better performance in all cases. The code rate 1 shows improved performance than code rate $\frac{3}{4}$ and rate $\frac{1}{2}$.



(a)



(b)

Figure 6.13 Theoretical error probability of uncoded 4-QAM and (a) rate $\frac{3}{4}$ with 8-bit orthogonal coded 4-QAM, (b) rate $\frac{1}{2}$ with 16-bit orthogonal coded 4-QAM.

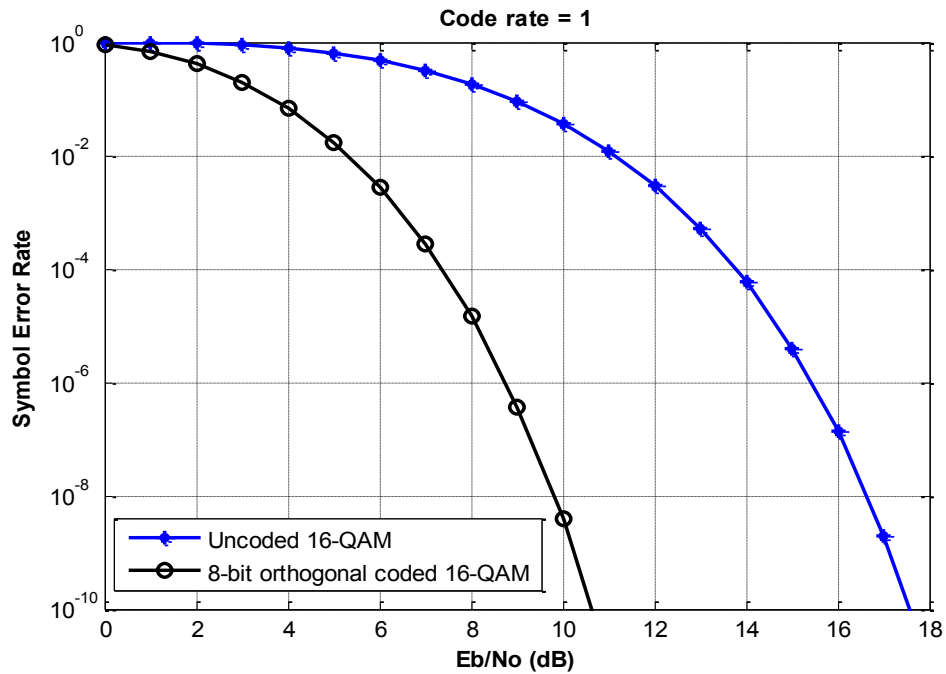


Figure 6.14 Theoretical plot of 16-QAM without coding and 8-bit orthogonal coding.

6.2.4 Simulation Results

The error performance of multilevel orthogonal coded M -ary QAM modulation techniques are investigated for different code rates and various code lengths in MATLAB. The same numbers of binary input data were encoded, modulated and transmitted over an AWGN communication channel. At the receiver, the demodulate symbols are decoded using PDL with correlative decoding at each step. The decoder estimated orthogonal codes are mapped back to binary digits and converted to the serial binary data. The estimated data are compared with the transmitted data to find the error rate of the system and plotted against the normalized electrical SNR, E_b/N_0 , where E_b is the energy of each bit, and N_0 denotes the power spectrum density of AWGN.

6.2.4.1 Rate $\frac{3}{4}$ Multilevel Orthogonal Coded M -ary QAM Encoder

Error performance of the rate $\frac{3}{4}$ orthogonal coded modulations was first implemented using 8-bit orthogonal code and later using 16-bit orthogonal code. For 8-bit orthogonal mapping, randomly generated 50,000 bits of binary serial data stream is first converted to parallel form of 6-bit each, which are further split into two parallel data streams each having 3 bits. These 3-bit are used to address the 8-bit orthogonal code. Thus, each data block in each encoder is mapped to 8-bit orthogonal code. Thus, the outputs of the encoders are two parallel orthogonal code stream of 8-bit. For mapping coded data to the 4-QAM modulation, one bit from each parallel encoder is taken at a time. So, for six input bits there are 8 output symbols, which are modulated and transmitted to the channel with E_b/N_0 varying from 0 to 10 dB. At receiver, the detected symbols are demodulated and then decoded using PDL. The estimated codes are mapped back to estimate the original data. The number of bit errors for each E_b/N_0 after demodulation and after correlative decoding are counted and is presented in Table 6.9 and Table 6.10. These tables also

present the number of bit errors corrected. And the plots for symbol error rate to the normalized E_b/N_0 are presented in Figure 6.10. The coding gain of approximately 2 dB and 4.5 dB are achieved using 8-bit and 16-bit orthogonal codes at the SER of 10^{-3} . The theoretical and simulated uncoded 4-QAM is also plotted.

Table 6.9 Number of bit errors for rate $\frac{3}{4}$, 8-bit orthogonal coded 4-QAM before and after decoding for different values of E_b/N_0 .

E_b/N_0 (dB)	Number of r bit errors		Number of errors corrected
	Before decoding	After decoding	
0	14737	3739	10998
1	11019	2285	8734
2	8181	1298	6883
3	5523	681	4842
4	3522	264	3258
5	1971	75	1896
6	999	24	975
7	399	0	399
8	145	0	145
9	33	0	373
10	7	0	0

Table 6.10 Number of bit errors for rate $\frac{3}{4}$, 16-bit orthogonal coded 4-QAM before and after decoding for different values of E_b/N_0 .

E_b/N_0 (dB)	Number of bit errors		Number of errors corrected
	Before decoding	After decoding	
0	29236	608	28628
1	22555	249	22306
2	16604	77	16527
3	11188	20	11168
4	7004	2	7002
5	3967	0	3967
6	1962	0	1962
7	851	0	851
8	282	0	282
9	74	0	74
10	18	0	18

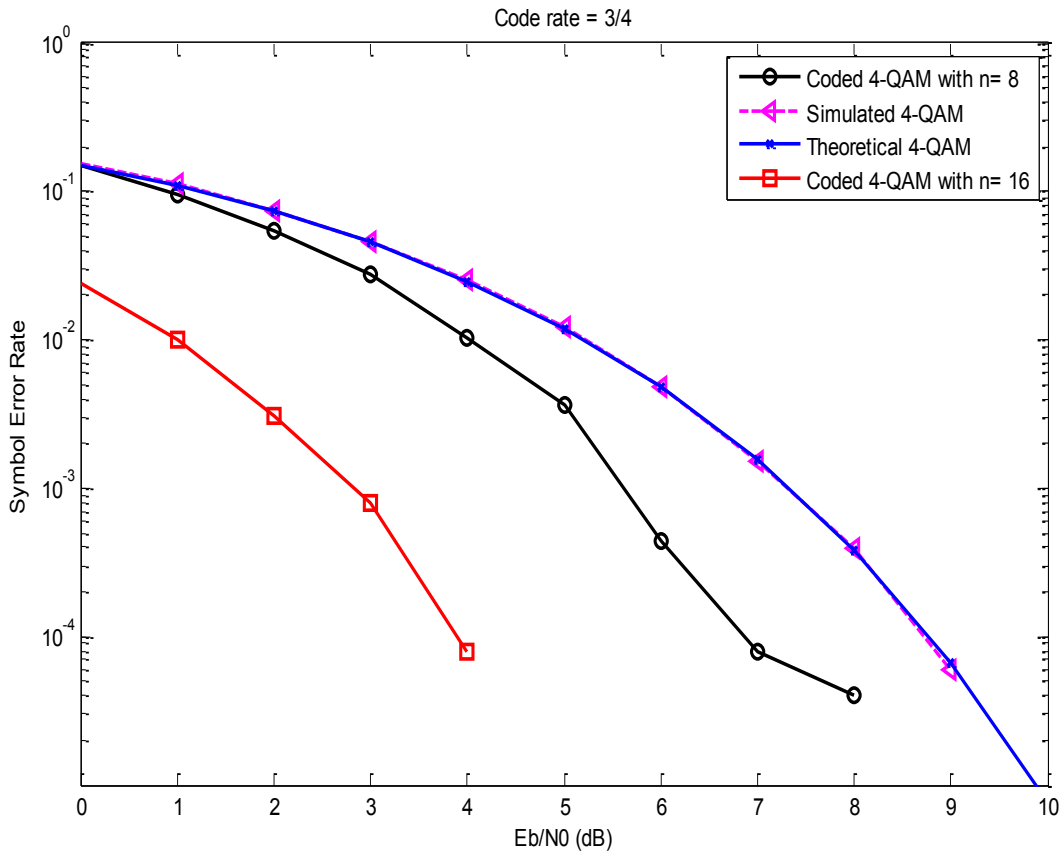


Figure 6.15 Error performance of rate $\frac{3}{4}$ orthogonal coded 4-QAM with $n=8$ and $n=16$ and comparison with uncoded 4-QAM.

The scatter plots for 4-QAM with different values of E_b/N_0 are shown in Figure 6.16. The transmitted symbol constellation points (black marks) and the received symbols at the receiver end with the effect of AWGN (cyan colored), are clearly depicted in figure. As the noise level is higher there are likely to be more errors at the demodulation than with lower noise levels.

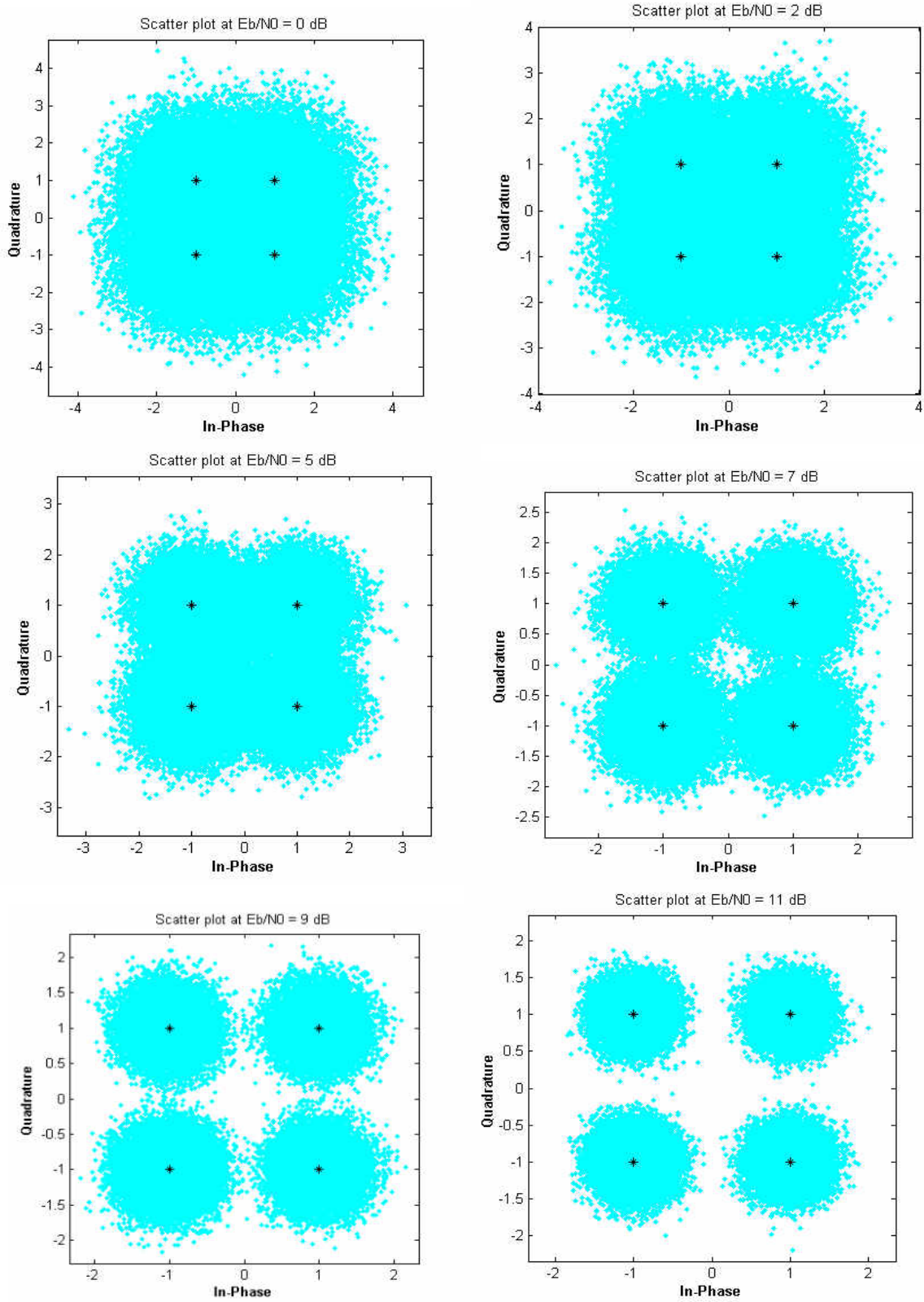


Figure 6.16 Scatter plots showing the effect of AWGN for different values of E_b/N_0 for 4-QAM modulation using rate $\frac{1}{2}$ eight bit orthogonal coding.

6.2.4.2 Rate 1/2 Multilevel Orthogonal Coded M -ary QAM Encoder

The code rate half multilevel OMQAM with orthogonal code length of 16-bit have two encoding levels and can be directly modulated using 4-QAM modulation. The simulated SER curve shows the coding gain of approximately 2.5 dB at SER of 10^{-3} as shown in Figure 6.17. Similarly, the code rate half with 32-bit orthogonal coding have four encoding levels and can be directly modulated using 16-QAM modulation with 4 bits per symbols. The simulated SER curve is shown in Figure 6.18. The coding gain of approximately 4.25 dB is observed at SER of 10^{-3} in this encoding scheme. The numbers of errors before and after the decoding, and number of errors corrected for both cases are tabulated in Table 6.11 and Table 6.12 respectively. We can observe that large number of errors are corrected using the longer code length and with higher level of encoding.

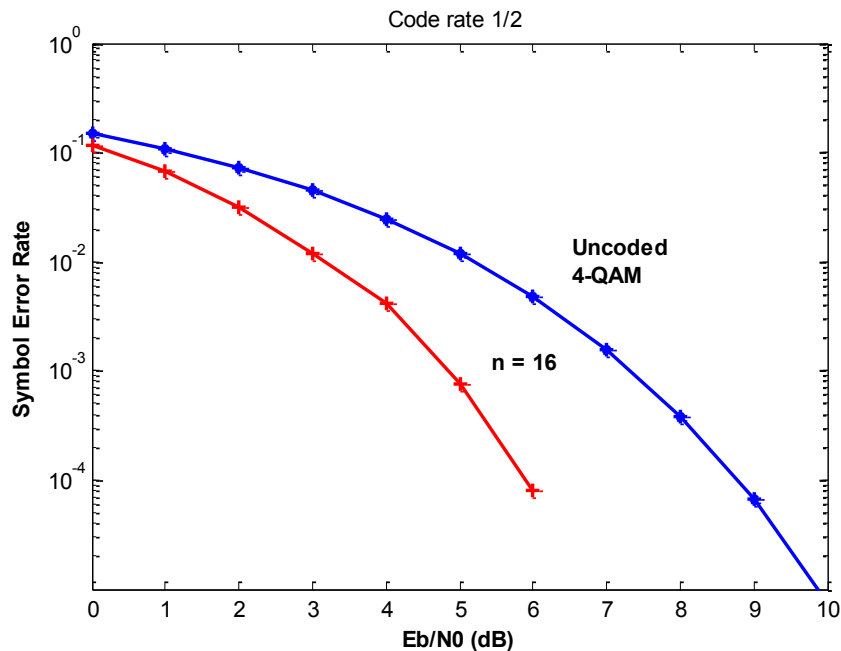


Figure 6.17 Error performance of rate 1/2 orthogonal coded 4-QAM with $n = 16$ and comparison with uncoded 4-QAM.

Table 6.11 Number of bit errors for rate 1/2, 16-bit orthogonal coded 4-QAM before and after decoding for different values of E_b/N_0 .

E_b/N_0 (dB)	Number of bit errors		Number of errors corrected
	Before decoding	After decoding	
0	31862	2927	28935
1	26057	1627	24430
2	20759	795	19964
3	15797	313	15484
4	11235	104	11131
5	7461	19	7442
6	4567	2	4565
7	2454	0	2454
8	1201	0	1201
9	498	0	498
10	174	0	174

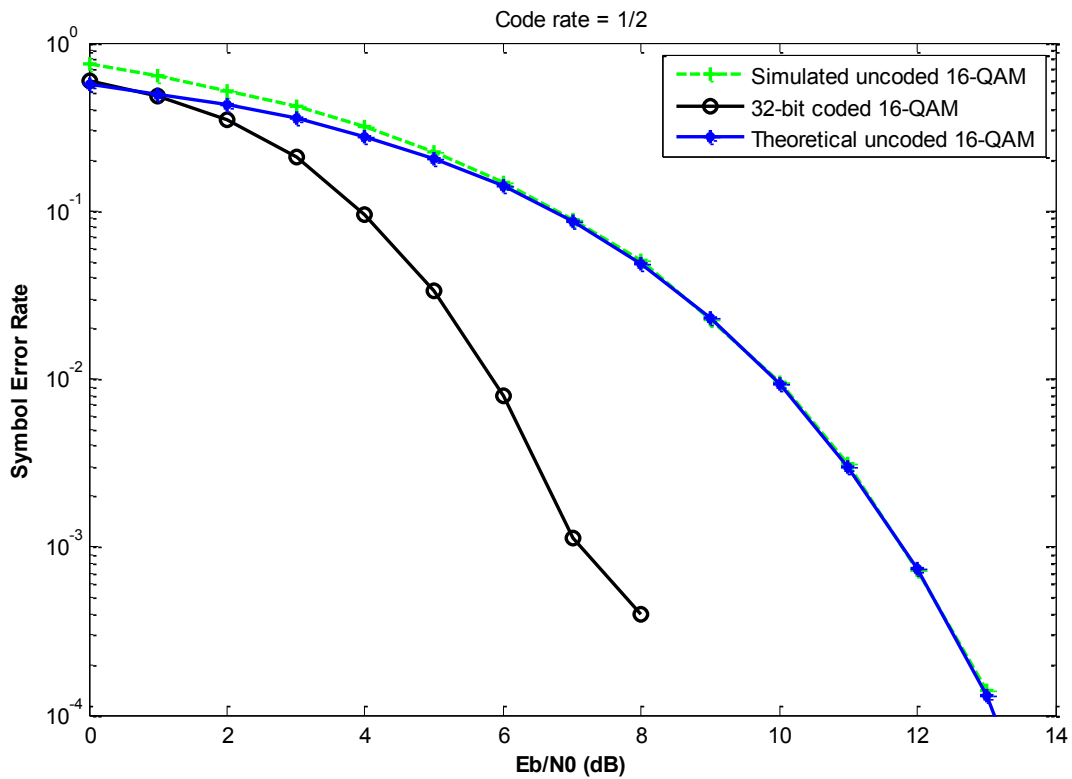


Figure 6.18 Error performance of rate 1/2 orthogonal coded 16-QAM with $n = 32$.

Table 6.12 Number of bit errors for rate 1/2, 32-bit orthogonal coded 16-QAM before and after decoding for different values of E_b/N_0 .

E_b/N_0 (dB)	Number of bit errors		Number of errors corrected
	Before decoding	After decoding	
0	102588	7355	95233
1	93615	6024	87591
2	84568	4385	80183
3	74278	2630	71648
4	62930	1175	61755
5	52041	422	51619
6	41035	99	40936
7	31173	14	31159
8	22753	5	22748
9	14692	0	14692
10	9143	0	9143
11	5080	0	5050
12	2366	0	2366
13	938	0	938
14	261	0	261

The comparison of error performance of 4-QAM for two different code rates using the same 16-bit orthogonal code length is presented in Figure 6.19. This figure shows that for same code length the code rate $\frac{3}{4}$ encoding schemes shows better performance than that of code rate $\frac{1}{2}$ encoding. Additional about 2.1 dB of coding gain is achieved by using code rate $\frac{3}{4}$ encoding configuration. However, the code rate $\frac{3}{4}$ has four levels and there will be more delay compared to code rate half, since only two levels output symbols are taken at a time for modulation, and remaining two level symbols will be hold and taken at different time, but error performance will be better.

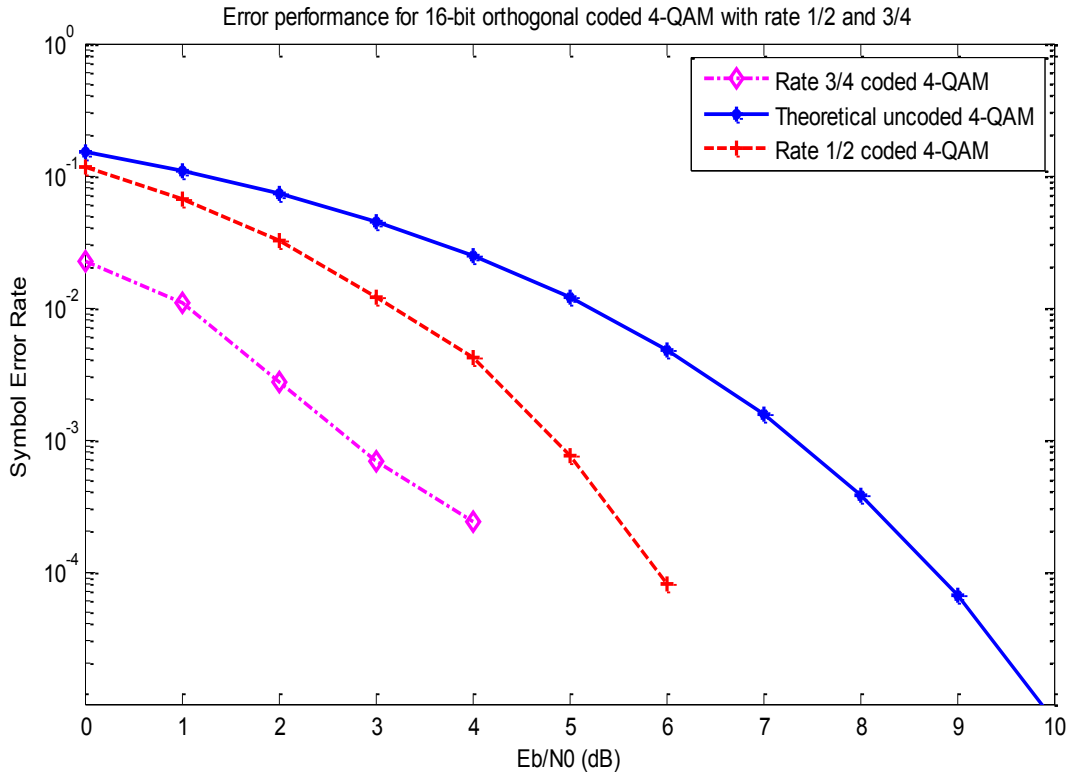


Figure 6.19 Comparison of error performance of rate $\frac{1}{2}$ and rate $\frac{3}{4}$ orthogonal coded 4-QAM with $n=16$.

6.2.4.3 Rate 1 Multilevel Orthogonal Coded 16-QAM Encoder

As explained in Chapter 5, the simulation results of rate 1 MOCM with four parallel encoders can be directly mapped to the 16-QAM, is presented in this section. First the number of bit errors before and after performing the correlative PDL, and number of bit error corrected by using PDL are tabulated in Table 6.13 and Table 6.14 for 8-bit and 16-bit code length. Then simulated SER curves are presented in Figure 6.20. The coding gains achieved are approximately 3 dB and 5.75 dB respectively at SER of 10^{-3} .

Table 6.13 Number of bit errors for rate 1, 8-bit orthogonal coded 16-QAM before and after decoding for different values of E_b/N_0 .

E_b/N_0 (dB)	Number of bit errors		Number of errors corrected
	Before decoding	After decoding	
0	37137	6737	30400
1	31561	5390	26171
2	26114	3927	22187
3	20620	2638	17982
4	15491	1728	13763
5	11107	850	10257
6	7498	397	7101
7	4649	178	4252
8	2422	55	2367
9	1142	12	1130
10	483	3	480
11	134	0	134
12	42	0	42
13	6	0	6
14	0	0	0
15	0	0	0

Table 6.14 Number of bit errors for rate 1, 16-bit orthogonal coded 16-QAM before and after decoding for different values of E_b/N_0 .

E_b/N_0 (dB)	Number of bit errors		Number of errors corrected
	Before decoding	After decoding	
0	117671	9330	108341
1	110250	8393	101857
2	102482	7190	95292
3	93781	6011	87770
4	84406	4726	79680
5	73728	3223	70505
6	63149	2128	61021
7	52006	1145	50861
8	41316	524	40792
9	30926	246	30680
10	22436	63	22373
11	14783	16	14767
12	9212	1	9211
13	4957	0	4957
14	2365	0	2365
15	897	0	897

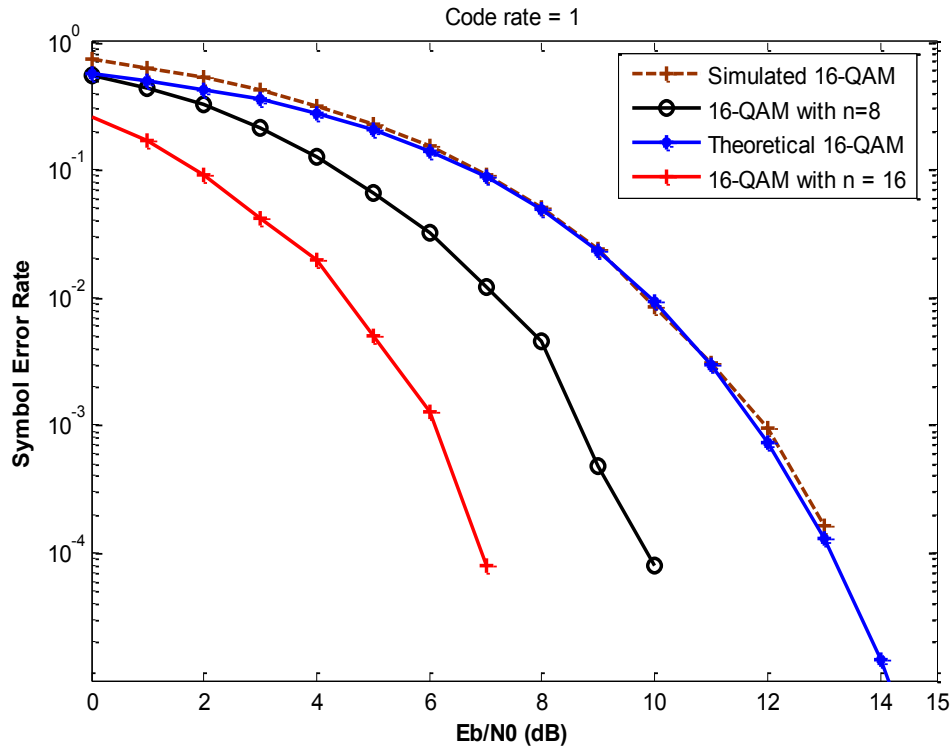


Figure 6.20 Error performance of rate 1 orthogonal coded 16-QAM with orthogonal code lengths of 8-bit and 16-bit.

Figure 6.21 shows the plots of the rectangular constellation points for different SNRs for multilevel OMQAM with $M = 16$. The effect of random AWGN noise can be clearly shown by the cyan color. For higher values of E_b/N_0 constellation points are separated clearly resulting in the improvement of the error performance of system.

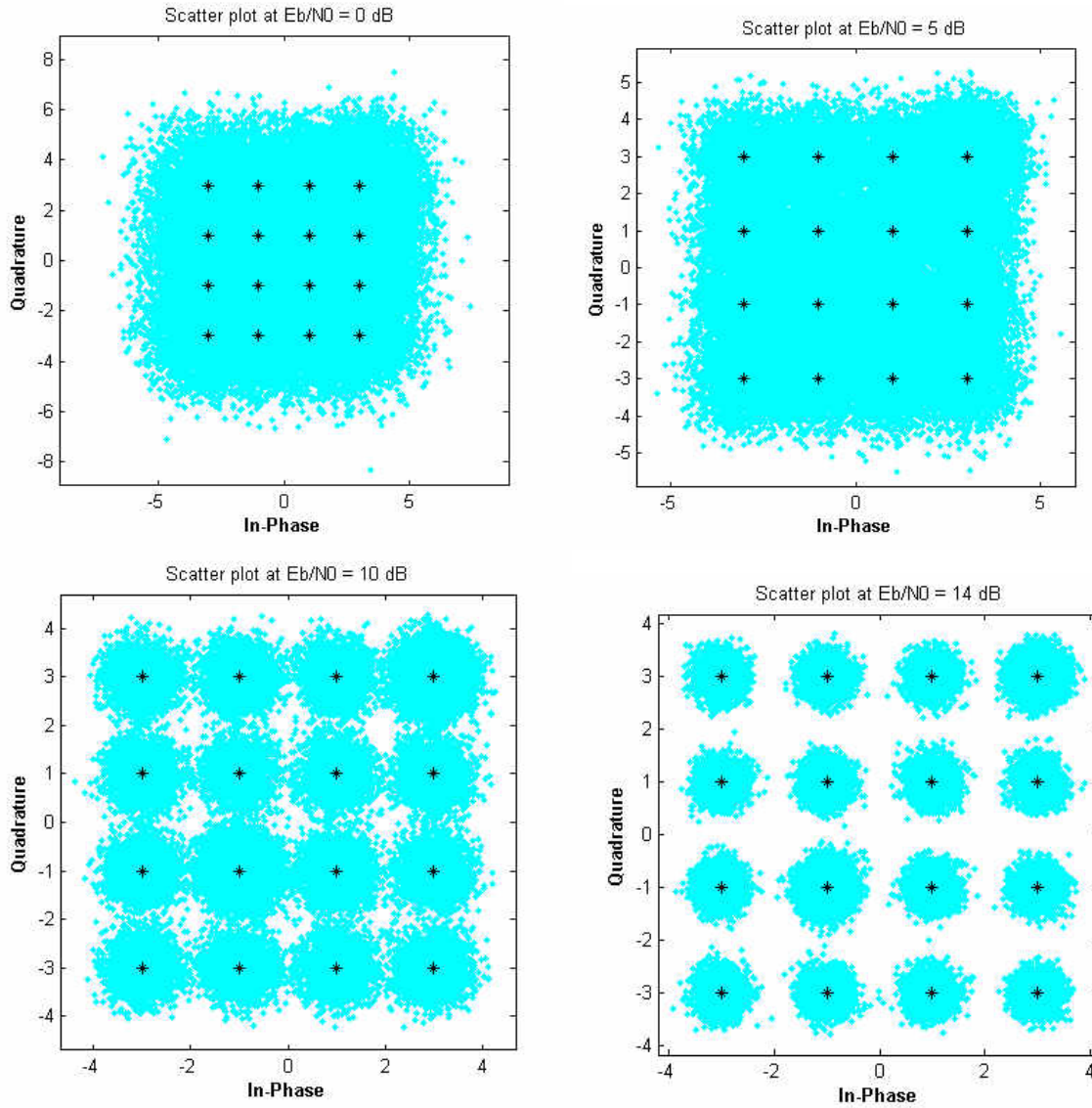


Figure 6.21 16-QAM scatter plots for signal to noise ratios of 0, 5, 10 and 14 dB respectively.

6.2.4.4 Multilevel Orthogonal Coded 256-QAM Modulation

The multilevel orthogonal coding for higher length of codes with eight levels can be directly modulated using 256-QAM with constellation points of 8-bit per symbol. This system will be more bandwidth efficient since eight bits can be transmitted per symbol. Figure 6.22 shows the simulated SER plotted against $\frac{E_b}{N_0}$ for the systems implementing 256-QAM for different code

rates and different code lengths. The code lengths are chosen from Table 5.2 having eight levels, which can be directly implemented for 256-QAM. Rate 1 with 16-bit code, rate $\frac{1}{2}$ with 64-bit and rate $\frac{3}{4}$ with 32-bit codes can be used to map directly with 256-QAM. The error performance plot for Rate 1 with 16-bit and rate $\frac{3}{4}$ with 32-bit encoding schemes show the similar performance but rate $\frac{3}{4}$ with 32-bit shows slightly better performance for higher values of E_b/N_0 . Even with higher code lengths code rate $\frac{1}{2}$ shows slightly poor error performance than that of others. However, the system shows excellent error performance of approximately 7 dB coding gain at SER of 10^{-2} for each case.

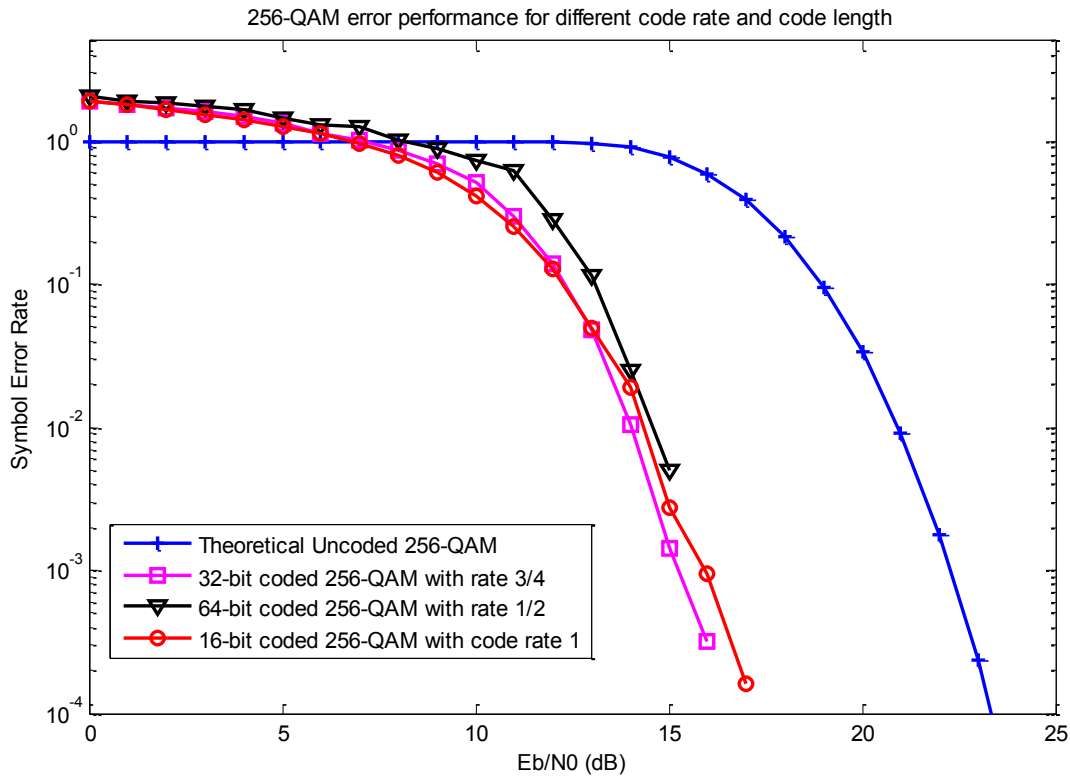


Figure 6.22 Error performance of 256-QAM with different code lengths and code rates.

CHAPTER 7

CONCLUSION AND FUTURE WORK

This chapter concludes the thesis and recommends future extensions for this research work.

7.1 Conclusion

The bandwidth efficient multilevel coded modulation approach using orthogonal channel coding and M -ary modulation scheme was successfully presented and simulated using MATLAB. The encoding and decoding structures, transmission bandwidths, bandwidth efficiencies, theoretical error performances and simulated error performance results for the MOCM system are presented. At encoder information data are mapped to the multiple orthogonal codes at multiple levels simultaneously. The outputs of the multilevel encoders at each instant are symbols for modulation. These symbols are used as constellation points for M -ary PSK and M -ary QAM modulations. The AWGN channel is taken as channel for analysis. The correlative decoding is performed independently at each level at the receiver after demodulation. This decoding arrangement reduces decoding delay and avoids the error propagations at different level.

The error correction capability of the MOCM depends on parameters such as encoding levels, orthogonal code lengths and coding rate. In fact the error performance increases with the increase in the number levels of encoder, longer orthogonal codes and with higher code rates. Among the three code rates, the code rate 1 show better performance than that of rate $\frac{3}{4}$ and rate $\frac{1}{2}$. Various arrangements for longer code lengths, higher encoding levels, and various code rates

encoding and decoding structures are described and simulated. Using multilevel structures allows reduced number of transmission waveforms. Similarly, more number of bits per symbol can be mapped to constellation points for M -ary modulation. This reduced the bandwidth of the system and makes the system bandwidth efficient.

The simulated symbol error rate vs. signal to noise ratio plot shows that coded M -ary PSK and QAM systems shows similar results. Improved coding gains were obtained for higher code lengths, higher code rates and higher values of constellation points M .

OptoRadio, optical wireless communication using orthogonal coded M -ary PSK system with ambient light cancellation technology, is also presented as one of the applications of MOCM for laser based optical wireless communications for secure, high data rate and long distance outdoor applications.

In summary, this thesis presented the coded modulation system that provides simple, cost effective and efficient encoding and decoding structures that detects and corrects multiple errors with bandwidth efficiency. The system is suitable for implementation in various modern spectral-efficient communication systems applications.

7.2 Future Work

In this thesis the analysis of MOCM system in AWGN channel model is conducted. In future, the system analysis can be performed in various channel models to estimate the actual performance in different channel environments. It would be interesting to observe the performance in fading channels for RF wireless communications. Similarly, for optical wireless communication performance analysis can be conducted for different atmospheric turbulence models [7].

The hardware implementation of encoders and decoders can also be conducted in future to observe the real time performance. The single level orthogonal encoding and decoding implementation (without modulation) in FPGA was already conducted in [38]. The implementation of MOCM system performance using FPGA system using multilevel modulation can also be performed in future.

This thesis shows the analysis of the system with single carrier single input single output (SISO) systems. Performance analysis in multicarrier systems and multiple input multiple output (MIMO) systems can also be conducted.

Similarly, comparison with other well-known coded modulation such as turbo coding, bit interleaved coded modulations (BICM), LDPC codes, etc. can be conducted in future. Further, the MOCM implementation with other modulation techniques can also be conducted.

APPENDICES

APPENDIX A

MULTILEVEL ORTHOGONAL CODED M-ARY PSK

Appendix A contains the main MATLAB code for multilevel OMPSK for *OptoRadio*. The simulation parameters such as code rate, orthogonal code size and number of input data bits are specified by the user. This program generates the random input binary data bits, and calls function for different code rate. Finally, plot the theoretical and simulated error performance curves.

```
*****
% Multilevel Orthogonal Coded M-ary PSK Modulation for OptoRadio MATLAB code.
% Program for Rate 1/2 : n = 16-bit, Modulation: QPSK
%           Rate 3/4 : n = 8-bit and 16-bit, Modulation: QPSK
%           Rate 1   : n = 8-bit and 16-bit, Modulation: 16-PSK
% Note: The input data bits must be multiple of parallel data bits in order to make sure that no
%       error in comparison during bit error checking.
% *****
clc;
clear all;
close all;
% Simulation parameters
coderate= input('Enter the code rate choose between 1/2,3/4 and 1:');
codeSize= input('Enter the code size must be power of 2:');
num_bit =input('Enter the number of input data bits:');
in_data=randint(1,num_bit);
% Perform channel coding for different code rates
if coderate == 1/2
    [Recovered_data,ser,SNR,M]=channelCodingratehalfMPSK(codeSize,coderate,in_data)
elseif coderate == 3/4
    [Recovered_data,ser,SNR,M]=channelCodingrate3by4MPSK(codeSize,coderate,in_data)
elseif coderate == 1
    [Recovered_data,ser,SNR,M]=channelCodingrate1MPSK(codeSize,coderate,in_data)
else
    disp('Enter valid code rate : 1/2,3/4 or 1');
end
% Plot of the error performance
t=log2(M); % t is number of bits per symbol
semilogy(SNR,ser,'red','Marker','+', 'LineWidth',2);
SER_theo = erfc(sqrt(t*10.^(SNR/10))*sin(pi/M));
hold on;
WERu = 1-(1-SER_theo).^(t);
semilogy(SNR,WERu,'Blue','Marker','o', 'LineWidth',2);
grid on;
xlabel('Eb/N0 (dB)');
ylabel('Symbol Error Rate');
title(['Code rate = ' num2str(rats(coderate))]);
```


APPENDIX B

DIFFERENT CODE RATE FOR MULTILEVEL OMPSK

Appendix B contains the MATLAB functions for different code rates. Based on code rates and code lengths the numbers of encoding level are defined. The function also call the multilevel encoding functions which return the mapped orthogonal codes, prepares symbols for modulation and send symbols to the functions for modulation, add noise, demodulate and decode the data.

%channelCodingratehalfMPSK function prepares for code rate $\frac{1}{2}$ multilevel OMPSK encoding and prepares the symbols for modulation after receiving the mapped data. It also calls the function that performs modulation, demodulation and decoding.

function

```
[Recovered_data,ser,SNR,M]=channelCodingratehalfMPSK(codeSize,coderate,in_data)
data_size=codeSize*coderate;
num_levels=data_size/4;
data_set=data_size/num_levels;
```

% Serial to parallel conversion of input binary data.

```
[B,padded]=vec2mat(in_data,data_size);% Convert data vector into matrix and zero padding the
%remaining data at last if required.
```

% Perform the multilevel encoding, here we have only two levels.

```
[ x_1, x_2 H A ] = multenc(B,data_set,data_size,codeSize) ;
```

% Converting orthogonal code matrix to single array.

```
y_1 = reshape(x_1',1,numel(x_1));
```

```
y_2= reshape(x_2',1,numel(x_2));
```

```
sym=[y_1;y_2];
```

```
data_in_sym=bi2de(sym,'left-msb');
```

%Function that performs M -ary PSK modulation, noise addition, demodulation and decoding

```
[Recovered_data,ser,SNR,M]=
```

```
ts_pskmod_decod(data_in_sym,H,A,data_size,coderate,y_1,y_2,in_data,codeSize);
```

end

%channelCodingrate3by4MPSK function prepares for code rate $\frac{3}{4}$ multilevel OMPSK encoding and prepares the symbols for modulations after receiving the mapped data. It also calls the function that performs modulation, demodulation and decoding.

function

```
[Recovered_data,ser,SNR,M]=channelCodingrate3by4MPSK(codeSize,coderate,in_data)
```

```

data_size=codeSize*coderate;
num_levels=data_size/3;
data_set=data_size/num_levels;

% Serial to parallel conversion of input data.
[B,padded]=vec2mat(in_data,data_size

% Perform the multilevel encoding for different levels.
if num_levels==2
 [ x_1, x_2 H A] = multenclev2(B,data_set,data_size,codeSize)

% Converting orthogonal code matrix to single array.
y_1 = reshape(x_1',1,numel(x_1));
y_2= reshape(x_2',1,numel(x_2));
sym=[y_1;y_2];
data_in_sym=bi2de(sym,'left-msb');

% M-ary PSK Modulations, addition of noises, demodulations and decoding.
[Recovered_data,ser,SNR,M] =
ts_pskmod_decod(data_in_sym,H,A,data_size,coderate,y_1,y_2,in_data,codeSize);

elseif num_levels==4
 [ x1, x2, x3, x4, H1,H2,A1,A2] = multenclev4(B,data_set,data_size,codeSize)
y1 = reshape(x1',1,numel(x1));
y2 = reshape(x2',1,numel(x2));
y3 = reshape(x3',1,numel(x3));
y4 = reshape(x4',1,numel(x4));
sc1 =[y1;y3];
sc2= [y2;y4];
data_in_sym1=bi2de(sc1,'left-msb');
data_in_sym2=bi2de(sc2,'left-msb');
sc= reshape([data_in_sym1, data_in_sym2]', 1, []);
s=de2bi(sc,2,'left-msb');
sc1 = s(:);

% M-ary PSK modulations, addition of noises, demodulations and decoding.
[Recovered_data,ser,SNR,M]=ts_pskmod_decodlv4(sc,data_size,coderate,y1,y2,y3,y4,in_data,c
odeSize,sc1,H1,H2,A1,A2,data_set);
End

*****
%channelCodingrate1MPSK function prepares for code rate 1 multilevel OMPSK encoding and
prepares symbols for modulation after receiving the mapped data. It also calls the function that
performs modulation, demodulation and decoding.
*****

```

```

function [Recovered_data,ser,SNR,M]= channelCodingrate1MPSK(codeSize,coderate,in_data)
data_size=codeSize*coderate;
num_levels=data_size/2;
data_set=data_size/num_levels;

% Serial to parallel conversion of input data.
[B,padded]=vec2mat(in_data,data_size);% Convert data vector into matrix and zero padding the
remaining data at last row if required.

% Perform the multilevel encoding for different levels.
if num_levels==4
[x1, x2, x3, x4, H1,H2,A1,A2] = multenclev4(B,data_set,data_size,codeSize)
y1 = reshape(x1',1,numel(x1));% Converting orthogonal matrix to single array
y2 = reshape(x2',1,numel(x2));
y3 = reshape(x3',1,numel(x3));
y4 = reshape(x4',1,numel(x4));
yc=[y1;y2;y3;y4]';
yc1 = yc(:);
[Recovered_data,ser,SNR,M]=ts_pskmod_decodlv14_16PSK(yc,data_size,coderate,in_data,codeS
Size,H1,H2,A1,A2,yc1,data_set);

elseif num_levels==8
[x1,x2,x3,x4,x5,x6,x7,x8,H1,H2,H3,H4,A1,A2,A3,A4] =
multenclev8(B,data_set,data_size,codeSize)

% Preparing symbols for 16-PSK
y1 = reshape(x1',1,numel(x1));
y2 = reshape(x2',1,numel(x2));
y3 = reshape(x3',1,numel(x3));
y4 = reshape(x4',1,numel(x4));
y5 = reshape(x5',1,numel(x5));
y6 = reshape(x6',1,numel(x6));
y7 = reshape(x7',1,numel(x7));
y8 = reshape(x8',1,numel(x8));
s1 = [y1;y2;y3;y4];
s2= [y5;y6;y7;y8];
s= vertcat(s1',s2');
[Recovered_data,ser,SNR,M]=ts_pskmod_decodlv18_16PSK(s,data_size,coderate,in_data,codeS
ize,H1,H2,H3,H4,A1,A2,A3,A4,y1,data_set)
else
disp('Higher level not available');
end
end

```

APPENDIX C

MULTILEVEL ENCODING

Appendix C contains the MATLAB code for different level of encoding for MOCM. At each encoding level the binary data are mapped to the orthogonal codes.

```
*****  
% multenclev2 function maps the binary data to orthogonal codes with two level of encoding.  
*****
```

```
function [x_1 x_2 H A] = multenclev2(B,data_set,data_size,codeSize)  
B1=B(:,1:data_set);  
B2=B(:,data_set+1:data_size);  
d1=bi2de(B1,'left-msb');  
row_d1=size(d1,1);  
d2=bi2de(B2,'left-msb');  
  
% Adding +1 to get decimal digit start from 1 (to perform row wise mapping)  
for i=1:1:size(d1,1);  
    e_row1(i)=d1(i);  
end  
E1=transpose(e_row1)+1;  
  
for i=1:1:size(d2,1);  
    e_row2(i)=d2(i);  
end  
E2=transpose(e_row2)+1;  
  
% Generation of bi-orthogonal codes of required size.  
H = OrthogonalCodegeneration(codeSize);  
A = AntipodalCodegeneration(codeSize);  
  
% Orthogonal code mapping.  
%Level 1  
for i=1:1:size(E1);  
    for k = 1:1:codeSize;  
        x_1(i,k)= H(E1(i),k);  
    end  
end  
% Level 2  
for i=1:1:size(E2);  
    for k = 1:1:codeSize;  
        x_2(i,k)= A(E2(i),k);  
    end  
end
```

```

end
*****
% multenclev4 function maps the binary data to orthogonal codes with four level of encoding.
*****
function [x1, x2, x3, x4, H1, H2, A1, A2] = multenclev4(B,data_set,data_size,codeSize)
B1=B(:,1:data_set);
B2=B(:,data_set+1:2*data_set);
B3=B(:,2*data_set+1:3*data_set);
B4=B(:,3*data_set+1:data_size);

d1=bi2de(B1,'left-msb');% Converts binary number to decimal
d2=bi2de(B2,'left-msb');
d3=bi2de(B3,'left-msb');
d4=bi2de(B4,'left-msb');

% Generation of bi-orthogonal codes
H = OrthogonalCodegeneration(codeSize);
A = AntipodalCodegeneration(codeSize);

mapsize = power(2,data_set);
% Splitting the orthogonal and antipodal code matrices into required levels.
H1=H(1:mapsize,:);
H2=H(mapsize+1:2*mapsize,:);
A1=A(1:mapsize,:);
A2=A(mapsize+1:2*mapsize,:);
row_H1=size(H1,1);

% Making the decimal number from 1.i.e. making the rows from 1 not from 0.Later at receiver it
% will be subtracted to recover correct value.
d1T=d1+1;
d2T=d2+1;
d3T=d3+1;
d4T=d4+1;

% Orthogonal code mapping
% Level 1
for i=1:size(d1T,1);
    for k = 1:codeSize;
        x1(i,k)= H1(d1T(i),k);
    end
end
% Level 2
for i=1:size(d2T,1);
    k=0;
    for k = 1:codeSize;
        x2(i,k)= H2(d2T(i),k);
    end
end

```

```

    end
end
% Level 3
for i=1:1:size(d3T,1);
    k=0;
    for k = 1:1:codeSize;
        x3(i,k)= A1(d3T(i),k);
    end
end
% Level 4
for i=1:1:size(d4T,1);
    k=0;
    for k = 1:1:codeSize;
        x4(i,k)= A2(d4T(i),k);
    end
end

*****
% multenclev8 function maps the binary data to orthogonal code having eight encoding level.
*****

function [x1,x2,x3,x4,x5,x6,x7,x8,H1,H2,H3,H4,A1,A2,A3,A4] =
multenclev8(B,data_set,data_size,codeSize)
B1=B(:,1:data_set);
B2=B(:,data_set+1:2*data_set);
B3=B(:,2*data_set+1:3*data_set);
B4=B(:,3*data_set+1:4*data_set);
B5=B(:,4*data_set+1:5*data_set);
B6=B(:,5*data_set+1:6*data_set);
B7=B(:,6*data_set+1:7*data_set);
B8=B(:,7*data_set+1:data_size);

d1=bi2de(B1,'left-msb');
d2=bi2de(B2,'left-msb');
d3=bi2de(B3,'left-msb');
d4=bi2de(B4,'left-msb');
d5=bi2de(B5,'left-msb');
d6=bi2de(B6,'left-msb');
d7=bi2de(B7,'left-msb');
d8=bi2de(B8,'left-msb');

% Generation of bi-orthogonal codes
H = OrthogonalCodegeneration(codeSize);
A = AntipodalCodegeneration(codeSize);

% Splitting bi-orthogonal codes to different levels.

```

```

mapsize = power(2,data_set);
H1=H(1:mapsize,:);
H2=H(mapsize+1:2*mapsize,:);
H3=H(2*mapsize+1:3*mapsize,:);
H4=H(3*mapsize+1:4*mapsize,:);
A1=A(1:mapsize,:);
A2=A(mapsize+1:2*mapsize,:);
A3=A(2*mapsize+1:3*mapsize,:);
A4=A(3*mapsize+1:4*mapsize,:);

% Making the row that starts from 1.
d1T=d1+1;
d2T=d2+1;
d3T=d3+1;
d4T=d4+1;
d5T=d5+1;
d6T=d6+1;
d7T=d7+1;
d8T=d8+1;

% Mapping the data to orthogonal code.
% Level 1
for i=1:1:size(d1T,1);
    for k = 1:1:codeSize;
        x1(i,k)= H1(d1T(i),k);
    end
end

% Level 2
for i=1:1:size(d2T,1);
    k=0;
    for k = 1:1:codeSize;
        x2(i,k)= H2(d2T(i),k);
    end
end

% Level 3
for i=1:1:size(d3T,1);
    k=0;
    for k = 1:1:codeSize;
        x3(i,k)= H3(d3T(i),k);
    end
end

% Level 4
for i=1:1:size(d4T,1);

```

```
    k=0;
    for k = 1:1:codeSize;
        x4(i,k)= H4(d4T(i),k);
    end
end
```

```
% Level 5
for i=1:1:size(d5T,1);
    for k = 1:1:codeSize;
        x5(i,k)= A1(d5T(i),k);
    end
end
```

```
% Level 6
for i=1:1:size(d6T,1);
    k=0;
    for k = 1:1:codeSize;
        x6(i,k)= A2(d6T(i),k);
    end
end
```

```
% Level 7
for i=1:1:size(d7T,1);
    k=0;
    for k = 1:1:codeSize;
        x7(i,k)= A3(d7T(i),k);
    end
end
```

```
% Level 8
for i=1:1:size(d8T,1);
    k=0;
    for k = 1:1:codeSize;
        x8(i,k)= A4(d8T(i),k);
    end
end
end
```


APPENDIX D

ORTHOGONAL AND ANTIPODAL CODE GENERATION

Appendix D contains the MATLAB code for generating orthogonal and antipodal code matrix of required code size [42]. These codes are generated by recursive method using Hadamard matrix.

% OrthogonalCodegeneration function generates the orthogonal code of size n.

```
function H = OrthogonalCodegeneration(codeSize)
```

```
N=4;
```

```
H=[0 0 ; 0 1];
```

```
if bitand(codeSize,codeSize-1)==0
```

```
while(N~=codeSize*2)
```

```
    N=N*2;
```

```
    H= repmat(H,[2,2]);
```

```
    [m,n]=size(H);
```

```
    for i=m/2+1:m,
```

```
        for j=n/2+1:n,
```

```
            H(i,j)=~H(i,j);
```

```
        end
```

```
    end
```

```
end
```

```
else
```

```
disp(' CODE SIZE is not valid !! The code size must be a power of 2.');
```

```
end
```

% AntipodalCodegeneration function generates the antipodal code of size n.

```
function A = AntipodalCodegeneration(codeSize)
```

```
N1=4;
```

```
A=[1 1 ; 1 0];
```

```
if bitand(codeSize,codeSize-1)==0
```

```
while(N1~=codeSize*2)
```

```
    N1=N1*2;
```

```
    A= repmat(A,[2,2]);
```

```
    [m,n]=size(A);
```

```
    for i=m/2+1:m,
```

```
        for j=n/2+1:n,
```

```
            A(i,j)=~A(i,j);
```

```
        end
```

```
    end
```

```
end
```

```
else
```

```
disp(' CODE SIZE is not valid !!The code size must be a power of 2.');
```

```
end
```

APPENDIX E

***M*-ary PSK MODULATION, ADDITION OF NOISE, DEMODULATION AND CALCULATE THE BER AND SER**

Appendix E contains the MATLAB code for *M*-ary PSK modulations, transmission of data to AWGN channel, *M*-ary PSK demodulation and send the decoded data for correlative PDL. This function also obtains the number of received bit errors and calculates BER/SER values.

```
*****  
% ts_pskmod_decod function performs the QPSK modulations, addition of white Gaussian noise  
and demodulation. It sends the demodulated data for decoding. It also calculates the bit error  
rates and symbol error rates.  
*****
```

```
function [Recovered_data,ser,SNRdB,M] =  
ts_pskmod_decod(data_in_sym,H,A,data_size,coderate,y_1,y_2,in_data,codeSize)  
M=4;
```

```
%QPSK modulation.  
dataMod=pskmod(data_in_sym,M);  
MaxsnrdB = input('Enter the maximum value of Eb/No in dB: ');  
SNRdB =0:MaxsnrdB;  
t=log2(M);
```

```
for ite=1:length(SNRdB)  
    SNR(ite) = SNRdB(ite)+10*log10(t*coderate);  
    receivedSignal = awgn(dataMod,SNR(ite),'measured');
```

```
%Scatter plot of the received data at receiver end.  
splotfig=scatterplot(receivedSignal,1,0,'c');  
hold on;  
scatterplot(dataMod,1,0,'k*',splotfig);  
hold off;
```

```
% Demodulation of QPSK.  
Demod_dataSymbols = pskdemod(receivedSignal,M);  
Demod_dataOut = de2bi(Demod_dataSymbols,t,'left-msb');
```

```
% Converting demodulated symbol to parallel forms for performing PDL.  
y1_dec = Demod_dataOut(:,1);  
y2_dec = Demod_dataOut(:,2);
```

```
% Function to perform decoding and obtaining the recovered data.  
[Recovered_data] = Decoding_level_2(y1_dec,y2_dec, codeSize, H, A, data_size);  
% Count the number of errors and calculate the BER and SER.
```

```

[numerror(ite),ber(ite)] = biterr(in_data,Recovered_data);
ser = ber *t;
end

*****
% ts_pskmod_decodlvl4 function performs QPSK modulation, addition of white Gaussian noise,
demodulation, and sends demodulated data for decoding. It also calculates the bit error rates and
symbol error rates.
*****

function [Recovered_data,ser,SNR,M]=
ts_pskmod_decodlvl4(sc,data_size,coderate,y1,y2,y3,y4,in_data,codeSize,sc1,H1,H2,A1,A2,data
_set)
M=4;
t=log2(M);
Y = pskmod(sc,M);

MaxsnrdB = input('Enter the maximum value of Eb/No in dB: ');
SNR = 0:MaxsnrdB;
k = 0;
for k=1:length(SNR)

    SNRdB(k) = SNR(k) +10*log10(coderate) +10*log10(t);
    rxSig = awgn(Y,SNRdB(k),'measured');
    Demod_datasyms = pskdemod(rxSig,M);
    demod = Demod_datasyms';

% Converting demodulated symbol to parallel form for performing PDL
demod1 = demod(1:2:end,:);
demod2 = demod(2:2:end,:);
rc1 = de2bi(demod1,2,'left-msb');
rc2 = de2bi(demod2,2,'left-msb');
y1_rec= rc1(:,1);
y2_rec= rc2(:,1);
y3_rec= rc1(:,2);
y4_rec= rc2(:,2);

[Recovered_data] =
Decoding_level_4(y1_rec,y2_rec,y3_rec,y4_rec,codeSize,H1,H2,A1,A2,data_set)

[numerror(k),ber(k)] = biterr(in_data,Recovered_data);
ser = ber *t;

end

```

```

*****
% ts_pskmod_decodlv14_16PSK function performs 16-PSK modulation, addition of white
Gaussian noise, demodulation and sends demodulated data for decoding. It also calculates the bit
error rates and symbol error rates.
*****

```

```
function
```

```
[Recovered_data,ser,SNRdB,M]=ts_pskmod_decodlv14_16PSK(yc,data_size,coderate,in_data,codeSize,H1,H2,A1,A2,yc1,data_set)
```

```
M=16;
```

```
t=log2(M);
```

```
y = bi2de(yc,'left-msb');
```

```
Ymod = pskmod(y,M);
```

```
MaxsnrdB = input('Enter the maximum value of Eb/No in dB: ');
```

```
SNRdB =0:MaxsnrdB;
```

```
for ite=1:1:length(SNRdB)
```

```
    SNR(ite) = SNRdB(ite)+10*log10(coderate) +10*log10(t);
```

```
    rxSig = awgn(Ymod,SNR(ite),'measured');
```

```
    rx= pskdemod(rxSig,M);
```

```
z= de2bi(rx,'left-msb');
```

```
size_z=size(z,2);
```

```
y1_rec=z(:,1);
```

```
y2_rec=z(:,2);
```

```
y3_rec=z(:,3);
```

```
y4_rec=z(:,4);
```

```
% Decoding
```

```
[Recovered_data] =
```

```
Decoding_level_4(y1_rec,y2_rec,y3_rec,y4_rec,codeSize,H1,H2,A1,A2,data_set)
```

```
[biterror(ite),ber(ite)] = biterr(in_data,Recovered_data);
```

```
ser= ber*t; % Symbol error rate
```

```
end
```

```

*****
% ts_pskmod_decodlv18_16PSK function performs 16-PSK modulation, addition of white
Gaussian noise, demodulation, and sends demodulated data for decoding. It also calculates the
bit error rates and symbol error rates.
*****

```

```
function
```

```
[Recovered_data,ser,SNRdB,M]=ts_pskmod_decodlv18_16PSK(s,data_size,coderate,in_data,codeSize,H1,H2,H3,H4, A1,A2,A3,A4,y1,data_set)
```

```
M=16;
```

```
t=log2(M);
```

```

y = bi2de(s,'left-msb');
Ymod = pskmod(y,M,0);
MaxsnrdB = input('Enter the maximum value of Eb/No in dB: ');
SNRdB =0:MaxsnrdB;

for ite=1:1:length(SNRdB)
    SNR(ite) = SNRdB(ite)+10*log10(coderate) +10*log10(t);
    rxSig = awgn(Ymod,SNR(ite),'measured');
    rx= pskdemod(rxSig,M);
    z= de2bi(rx,'left-msb');
    size_z=size(z,1);
    y1_=z(1:size_z/2,:);
    y2_=z(size_z/2+1:size_z,:);
    y1_rec = y1_(:,1);
    y2_rec = y1_(:,2);
    y3_rec = y1_(:,3);
    y4_rec = y1_(:,4);
    y5_rec = y2_(:,1);
    y6_rec = y2_(:,2);
    y7_rec = y2_(:,3);
    y8_rec = y2_(:,4);

    [Recovered_data] =
    Decoding_level_8(y1_rec,y2_rec,y3_rec,y4_rec,y5_rec,y6_rec,y7_rec,y8_rec,codeSize,H1,H2,H
    3,H4,A1,A2,A3,A4,data_set)

    [biterror1(ite),ber(ite)] = biterr(in_data,Recovered_data);
    ser= ber*t;
end

```

APPENDIX F

PARALLEL INDEPENDENT DECODING AT INDIVIDUAL LEVEL (PDL) FOR MOCM

Appendix F contains the MATLAB code for Parallel Independent Decoding at Individual Level (PDL) for MOCM. At each decoding level, the demodulated impaired orthogonal codes are cross correlated to known set of orthogonal code to estimate the transmitted binary data.

```
*****  
% Decoding_level_2 function performs the correlative decoding independently at two levels to  
estimate the binary information data.  
*****
```

```
function [Recovered_data] = Decoding_level_2(y1_dec,y2_dec,codeSize,H,A,data_size)  
%Preparation for decoding.  
[HE_1,padded]=vec2mat(y1_dec,codeSize);% Convert vector into matrix and zero padding the  
%remaining data at last to make data divisible by codeSize.  
[HE_2,padded]=vec2mat(y2_dec,codeSize);  
col_HE_1=size(HE_1,2);  
row_HE_1=size(HE_1,1);  
  
col_HE_2=size(HE_2,2);  
row_HE_2=size(HE_2,1);  
  
[rows1 cols1]=size(H);  
[rows2 cols2]=size(A);  
  
% Recovering the transmitted data i.e. decoding at each level independently.  
%Level 1  
i=0;  
j=0;  
for i= 1:row_HE_1  
    R_HE_1=HE_1(i,:);  
    for j=1:rows1  
        H1=H(j,:);  
        corrval(j,:)=xor(R_HE_1,H1);  
        CS(j)=sum(corrval(j,:));  
        [m_1,i_1]=min(CS);  
        RS(i,:)=H(i_1,:);  
        RST= RS(i,:);  
        m1(i)= find(ismember(H,RST,'rows'));  
    end  
end  
  
% Level 2
```

```

i=0;
j=0;
for i= 1:row_HE_2
    R_HE_2=HE_2(i,:);
    for j=1:rows2
        A1=A(j,:);
        corrval1(j,:)=xor(R_HE_2,A1);
        CS1(j)=sum(corrval1(j,:));
        [m_2,i_2]=min(CS1);
        RS1(i,:)=A(i_2,:);
        RST1=RS1(i,:);
        m2(i)=find(ismember(A,RST1,'rows'));
    end
end

% Getting original row from the recovered row.
M1=m1'-1;
M2=m2'-1;
b1=de2bi(M1,data_size/2,'left-msb');
b2=de2bi(M2,data_size/2,'left-msb');
decod=horzcat(b1,b2); % Combining two parallel recovered data
Recovered_data=reshape(decod',1,[]); % Converting parallel data to serial data

*****
% Decoding_level_4 function performs the correlative decoding independently at four levels to
estimate the information data.
*****
function [Recovered_data] =
Decoding_level_4(y1_rec,y2_rec,y3_rec,y4_rec,codeSize,H1,H2,A1,A2,data_set)

% Convert vector into matrix and zero padding the remaining data at last to make data divisible
%by codeSize.
R0=received orthogonal code
[RO_1,padded]=vec2mat(y1_rec,codeSize);
[RO_2,padded]=vec2mat(y2_rec,codeSize);
[RO_3,padded]=vec2mat(y3_rec,codeSize);
[RO_4,padded]=vec2mat(y4_rec,codeSize);
[r1 c1]=size(RO_1);
[r2 c2]=size(RO_2);
[r3 c3]=size(RO_3);
[r4 c4]=size(RO_4);

% Rows and columns of orthogonal and antipodal code matrices.
[rowsH1 colsH1]=size(H1);
[rowsH2 colsH2]=size(H2);
[rowsA1 colsA1]=size(A1);

```

```

[rowsA2 colsA2]=size(A2);

%PDL
% Level 1.
for i= 1:r1
    R1=RO_1(i,:);
    for j=1:rowsH1
        H11=H1(j,:);
        corrval1(j,:)=xor(R1,H11);
        CS1(j)=sum(corrval1(j,:));
        [m_1,i_1]=min(CS1);
        RS1(i,:)=H1(i_1,:);
        RST1= RS1(i,:);
        m1(i)= find(ismember(H1,RST1,'rows'));
    end
end

%Level 2.
for i= 1:r2
    R2=RO_2(i,:);
    for j=1:rowsH2
        H22=H2(j,:);
        corrval2(j,:)=xor(R2,H22);
        CS2(j)=sum(corrval2(j,:));
        [m_2,i_2]=min(CS2);
        RS2(i,:)=H2(i_2,:);
        RST2= RS2(i,:);
        m2(i)= find(ismember(H2,RST2,'rows'));
    end
end

%Level 3.
for i= 1:r3
    R3=RO_3(i,:);
    for j=1:rowsA1
        A11=A1(j,:);
        corrval3(j,:)=xor(R3,A11);
        CS3(j)=sum(corrval3(j,:));
        [m_3,i_3]=min(CS3);
        RS3(i,:)=A1(i_3,:);
        RST3= RS3(i,:);
        m3(i)= find(ismember(A1,RST3,'rows'));
    end
end

%Level 4.

```



```

for i= 1:r4
    R4=RO_4(i,:);
    for j=1:rowsA2
        A22=A2(j,:);
        corrval4(j,:)=xor(R4,A22);
        CS4(j)=sum(corrval4(j,:));
        [m_4,i_4]=min(CS4);
        RS4(i,:)=A2(i_4,:);
        RST4= RS4(i,:);
        m4(i)= find(ismember(A2,RST4,'rows'));
    end
end
M1=m1'-1;
M2=m2'-1;
M3=m3'-1;
M4=m4'-1;
b1=de2bi(M1,data_set,'left-msb');
b2=de2bi(M2,data_set,'left-msb');
b3=de2bi(M3,data_set,'left-msb');
b4=de2bi(M4,data_set,'left-msb');
dec_interleave=horzcat(b1,b2,b3,b4);
Recovered_data=reshape(dec_interleave',1,[]);

```

```

*****
% Decoding_level_8 function performs the correlative decoding independently at eight levels to
estimate the information data.
*****

```

```

function [Recovered_data] =
Decoding_level_8(y1_rec,y2_rec,y3_rec,y4_rec,y5_rec,y6_rec,y7_rec,y8_rec,codeSize,H1,H2,H
3,H4,A1,A2,A3,A4,data_set)

```

```

[RO_1,padded]=vec2mat(y1_rec,codeSize);
[RO_1,padded]=vec2mat(y1_rec,codeSize);
[RO_2,padded]=vec2mat(y2_rec,codeSize);
[RO_3,padded]=vec2mat(y3_rec,codeSize);
[RO_4,padded]=vec2mat(y4_rec,codeSize);
[RO_5,padded]=vec2mat(y5_rec,codeSize);
[RO_6,padded]=vec2mat(y6_rec,codeSize);
[RO_7,padded]=vec2mat(y7_rec,codeSize);
[RO_8,padded]=vec2mat(y8_rec,codeSize);

```

```

[r1, c1]=size(RO_1);
[r2, c2]=size(RO_2);
[r3, c3]=size(RO_3);
[r4, c4]=size(RO_4);

```

```
[r5, c5]=size(RO_5);
[r6, c6]=size(RO_6);
[r7, c7]=size(RO_7);
[r8, c8]=size(RO_8);
```

```
[rowsH1, colsH1]=size(H1);
[rowsH2, colsH2]=size(H2);
[rowsH3, colsH3]=size(H3);
[rowsH4, colsH4]=size(H4);
[rowsA1, colsA1]=size(A1);
[rowsA2, colsA2]=size(A2);
[rowsA3, colsA3]=size(A3);
[rowsA4, colsA4]=size(A4);
```

% Perform decoding at each levels and finding the correct row to recover the data.

% Level 1

```
for i= 1:r1
    R1=RO_1(i,:);
    for j=1:rowsH1
        H11=H1(j,:);
        corrval1(j,:)=xor(R1,H11);
        CS1(j)=sum(corrval1(j,:));
        [m_1,i_1]=min(CS1);
        RS1(i,:)=H1(i_1,:);
        RST1= RS1(i,:);
        m1(i)= find(ismember(H1,RST1,'rows'));
    end
end
```

% Level 2

```
for i= 1:r2
    R2=RO_2(i,:);
    for j=1:rowsH2
        H22=H2(j,:);
        corrval2(j,:)=xor(R2,H22);
        CS2(j)=sum(corrval2(j,:));
        [m_2,i_2]=min(CS2);
        RS2(i,:)=H2(i_2,:);
        RST2= RS2(i,:);
        m2(i)= find(ismember(H2,RST2,'rows'));
    end
end
```

% Level 3

```
for i= 1:r3
```

```

R3=RO_3(i,:);
for j=1:rowsH3
    H33=H3(j,:);
    corrval3(j,:)=xor(R3,H33);
    CS3(j)=sum(corrval3(j,:));
    [m_3,i_3]=min(CS3);
    RS3(i,:)=H3(i_3,:);
    RST3=RS3(i,:);
    m3(i)=find(ismember(H3,RST3,'rows'));
end
end

```

```

% Level 4
for i= 1:r4
    R4=RO_4(i,:);
    for j=1:rowsH4
        H44=H4(j,:);
        corrval4(j,:)=xor(R4,H44);
        CS4(j)=sum(corrval4(j,:));
        [m_4,i_4]=min(CS4);
        RS4(i,:)=H4(i_4,:);
        RST4=RS4(i,:);
        m4(i)=find(ismember(H4,RST4,'rows'));
    end
end

```

```

% Level 5
for i= 1:r5
    R5=RO_5(i,:);
    for j=1:rowsA1
        A11=A1(j,:);
        corrval5(j,:)=xor(R5,A11);
        CS5(j)=sum(corrval5(j,:));
        [m_5,i_5]=min(CS5);
        RS5(i,:)=A1(i_5,:);
        RST5=RS5(i,:);
        m5(i)=find(ismember(A1,RST5,'rows'));
    end
end

```

```

% Level 6
for i= 1:r6
    R6=RO_6(i,:);
    for j=1:rowsA2
        A22=A2(j,:);
        corrval6(j,:)=xor(R6,A22);
    end
end

```

```

        CS6(j)=sum(corrval6(j,:));
        [m_6,i_6]=min(CS6);
        RS6(i,:)=A2(i_6,:);
        RST6= RS6(i,:);
        m6(i)= find(ismember(A2,RST6,'rows'));
    end
end

```

```

% Level 7
for i= 1:r7
    R7=RO_7(i,:);
    for j=1:rowsA3
        A33=A3(j,:);
        corrvall7(j,:)=xor(R7,A33);
        CS7(j)=sum(corrval7(j,:));
        [m_7,i_7]=min(CS7);
        RS7(i,:)=A3(i_7,:);
        RST7= RS7(i,:);
        m7(i)= find(ismember(A3,RST7,'rows'));
    end
end

```

```

% Level 8
for i= 1:r8
    R8=RO_8(i,:);
    for j=1:rowsA4
        A44=A4(j,:);
        corrvall8(j,:)=xor(R8,A44);
        CS8(j)=sum(corrval8(j,:));
        [m_8,i_8]=min(CS8);
        RS8(i,:)=A4(i_8,:);
        RST8= RS8(i,:);
        m8(i)= find(ismember(A4,RST8,'rows'));
    end
end

```

```

M1=m1'-1;
M2=m2'-1;
M3=m3'-1;
M4=m4'-1;
M5=m5'-1;
M6=m6'-1;
M7=m7'-1;
M8=m8'-1;

```

```

b1=de2bi(M1,data_set,'left-msb');

```

```
b2=de2bi(M2,data_set,'left-msb');
b3=de2bi(M3,data_set,'left-msb');
b4=de2bi(M4,data_set,'left-msb');
b5=de2bi(M5,data_set,'left-msb');
b6=de2bi(M6,data_set,'left-msb');
b7=de2bi(M7,data_set,'left-msb');
b8=de2bi(M8,data_set,'left-msb');

dec_interleave=horzcat(b1,b2,b3,b4,b5,b6,b7,b8);
Recovered_data=reshape(dec_interleave',1,[]);
```

APPENDIX G

MULTILEVEL ORTHOGONAL *M*-ARY QAM

Appendix G contains the main MATLAB code for Multilevel Orthogonal *M*-ary QAM. The simulation parameters such as code rate, orthogonal code size and number of input data bits are entered by user. This program generates the random input binary data bits, and performs the encoding and decoding for corresponding code rates and plots the SER curve of the system.

```
*****
% Orthogonal coded M-ary QAM modulation MATLAB code.
% Program for Rate 1/2: n = 16-bit and 32-bit, Modulation: 4-QAM, 16-QAM, 256-QAM
% Rate 3/4: n = 8-bit, 16-bit and 32-bit, Modulation: 4-QAM, 16-QAM, 256-QAM
% Rate 1 : n = 8-bit, 16-bit and 32-bit, Modulation 16-QAM, 256-QAM
% Note: The input data bits must be multiple of parallel data bits in order to make sure that no
% error in comparison during bit error checking.
*****
clc;
clear all;
close all;

% Simulation parameters
coderate= input('Enter the code rate choose between 1/2,3/4 and 1:');
codeSize= input('Enter the code size must be power of 2:');
num_bit =input('Enter the number of input data bits:');

%Generation of random input binary data.
in_data=randint(1,num_bit);

%Perform channel coding for different code rates.
if coderate == 1/2
    [Recovered_data,ser,SNRdB,M]=channelCodingratehalfMQAM(codeSize,coderate,in_data)
elseif coderate == 3/4
    [Recovered_data,ser,SNRdB,M]=channelCodingrate3by4MQAM(codeSize,coderate,in_data)
elseif coderate == 1
    [Recovered_data,ser,SNRdB,M]= channelCodingrate1MQAM(codeSize,coderate,in_data)
else
    disp('Enter Valid code rate : 1/2,3/4 or 1');
end
% Plot of the error performance
t=log2(M); % Bits per symbol.
semilogy(SNRdB,ser,'red','Marker','+','LineWidth',2);
hold on;
if M==4
    ser_theo = 0.5*erfc(sqrt((10.^(SNRdB/10))));
```

```
else
ser_theo= (4)*qfunc(sqrt((3*t/(M-1))*(10.^(SNRdB/10))));
end
WERu = 1-(1-ser_theo).^(t);
semilogy(SNRdB,WERu,'blue','Marker','*','LineWidth',2);
grid on;
xlabel('Eb/N0 (dB)');
ylabel('Symbol Error Rate');
title(['Code rate =' num2str(rats(coderate))]);
```

APPENDIX H

DIFFERENT CODE RATE FOR MULTILEVEL ORTHOGONAL *M*-ARY QAM

Appendix H contains the MATLAB functions for different code rates. Based on input code rate and code size the numbers of encoding level are defined. The function also calls the multilevel encoding operations. It obtains the coded data, prepares symbols for modulation and sends symbols to the functions that modulate, add noise, demodulate and decode the data.

```
*****  
% channelCodingratehalfMQAM function prepares for code rate 1/2 multilevel encoding. It also  
prepares the symbols for modulations after receiving the mapped data.  
*****
```

function

```
[Recovered_data,ser,SNRdB,M]=channelCodingratehalfMQAM(codeSize,coderate,in_data)  
data_size=codeSize*coderate;  
num_levels=data_size/4;  
data_set=data_size/num_levels;
```

% Serial to parallel conversion of input data.

```
[B,padded]=vec2mat(in_data,data_size); % Convert data vector into matrix and zero padding the  
remaining data at last if required.
```

if num_levels==2

```
[x_1, x_2 H, A] = multilevel2(B,data_set,data_size,codeSize) ;  
% Converting orthogonal code matrix to single array  
y_1 = reshape(x_1',1,numel(x_1));  
y_2= reshape(x_2',1,numel(x_2));  
sym=[y_1;y_2];
```

```
data_in_sym=bi2de(sym,'left-msb');
```

%*M*-ary QAM Modulation, noise addition, demodulation and decoding.

```
[Recovered_data,ser,SNRdB,M]=tx_qammod_demod_decod(data_in_sym,H,A,data_size,coderate,y_1,y_2,in_data,codeSize);
```

elseif num_levels==4

```
disp('Level 4, Modulation :16-QAM');  
M=16;
```

```
[x1, x2, x3, x4, H1,H2,A1,A2] = multenclev4(B,data_set,data_size,codeSize) ;  
y1 = reshape(x1',1,numel(x1));  
y2 = reshape(x2',1,numel(x2));  
y3 = reshape(x3',1,numel(x3));  
y4 = reshape(x4',1,numel(x4));
```



```

yc=[y1;y2;y3;y4];

[Recovered_data,ser,SNRdB,M]=ts_mod_decodlvl4_16QAM(M,yc,coderate,in_data,codeSize,H
1,H2,A1,A2,data_set);

elseif num_levels==8
display('256-QAM Modulate available');
M=input('Enter the value of M :');
if M ==256
    [x1,x2,x3,x4,x5,x6,x7,x8,H1,H2,H3,H4,A1,A2,A3,A4] =
    multenclev8(B,data_set,data_size,codeSize)
    y1 = reshape(x1',1,numel(x1));
    y2 = reshape(x2',1,numel(x2));
    y3 = reshape(x3',1,numel(x3));
    y4 = reshape(x4',1,numel(x4));
    y5 = reshape(x5',1,numel(x5));
    y6 = reshape(x6',1,numel(x6));
    y7 = reshape(x7',1,numel(x7));
    y8 = reshape(x8',1,numel(x8));
    s1=[y1;y2;y3;y4;y5;y6;y7;y8];
    data_in_sym= bi2de(s1,'left-msb');

[Recovered_data,ser,SNRdB,M]=tx_qammod_decodlvl8_MQAM(M,data_in_sym,coderate,in_d
ata,codeSize,H1,H2,H3,H4,A1,A2,A3,A4,data_set)
else
    disp('Other Modulation not available now!');
end
else
    disp('Higher level not available now !');
end
end

```

```

*****
% channelCodingrate3by4MQAM function prepares for code rate 3/4 multilevel encoding and
prepares symbols for modulation after receiving the mapped data.
*****

```

```

function
[Recovered_data,ser,SNRdB,M]=channelCodingrate3by4MQAM(codeSize,coderate,in_data)
data_size=codeSize*coderate;
num_levels=data_size/3;
data_set=data_size/num_levels;

```

% Serial to parallel conversion of input data.

```
[B,padded]=vec2mat(in_data,data_size);% Convert data vector into matrix and zero padding the
remaining data at last if required.
```

```
if num_levels==2
[x_1, x_2 H A] = multilevel2(B,data_set,data_size,codeSize) ;
y_1 = reshape(x_1',1,numel(x_1));
y_2= reshape(x_2',1,numel(x_2));
sym=[y_1;y_2]; % Modulated signal Vector
data_in_sym=bi2de(sym,'left-msb');
```

```
[Recovered_data,ser,SNRdB,M]=tx_qammod_demod_decod(data_in_sym,H,A,data_size,coderate,
y_1,y_2,in_data,codeSize);
```

```
elseif num_levels==4
[x1, x2, x3, x4, H1,H2,A1,A2] = multenclev4(B,data_set,data_size,codeSize);
y1 = reshape(x1',1,numel(x1));
y2 = reshape(x2',1,numel(x2));
y3 = reshape(x3',1,numel(x3));
y4 = reshape(x4',1,numel(x4));
display('Modulate using 4-QAM or 16-QAM?');
M=input('Enter the value of M (4 or 16):');
```

```
if M ==4
sc1 =[y1;y3];
sc2= [y2;y4];
data_in_sym1=bi2de(sc1,'left-msb');
data_in_sym2=bi2de(sc2,'left-msb');
data_in_sym= reshape([data_in_sym1, data_in_sym2]', 1, []);
```

```
[Recovered_data,ser,SNRdB,M]=ts_mod_decod_lev4_4QAM(M,data_in_sym,data_size,coderate,
e,in_data,codeSize,H1,H2,A1,A2,data_set);
```

```
elseif M==16
yc=[y1;y2;y3;y4];
ts_mod_decodlvl4_16QAM(M,yc,coderate,in_data,codeSize,H1,H2,A1,A2,data_set)
```

```
else
display('Higher order modulations not available now');
end
```

```
elseif num_levels==8
display('256-QAM Modulate available');
M=input('Enter the value of M :');
if M==256
[x1,x2,x3,x4,x5,x6,x7,x8,H1,H2,H3,H4,A1,A2,A3,A4] =
multenclev8(B,H,A,data_set,data_size,codeSize)
```

```

y1 = reshape(x1',1,numel(x1));
y2 = reshape(x2',1,numel(x2));
y3 = reshape(x3',1,numel(x3));
y4 = reshape(x4',1,numel(x4));
y5 = reshape(x5',1,numel(x5));
y6 = reshape(x6',1,numel(x6));
y7 = reshape(x7',1,numel(x7));
y8 = reshape(x8',1,numel(x8));
s1=[y1;y2;y3;y4;y5;y6;y7;y8];
data_in_sym= bi2de(s1,'left-msb');

[Recovered_data,ser,SNRdB,M]=tx_qammod_decodlvl8_MQAM(M,data_in_sym,coderate,in_d
ata,codeSize,H1,H2,H3,H4,A1,A2,A3,A4,data_set)
else
    disp('No other modulation available. ');
end
else
    disp('Higher level not available');
end
end

*****
% channelCodingrate1MQAM function prepares for code rate 1 multilevel encoding and
prepares symbols for modulation after receiving the mapped data.
*****

function
[Recovered_data,ser,SNRdB,M]=channelCodingrate1MQAM(codeSize,coderate,in_data)
data_size=codeSize*coderate;
num_levels=data_size/2;
data_set=data_size/num_levels;

% Serial to parallel conversion of input data.
[B,padded]=vec2mat(in_data,data_size);

if num_levels==4
disp('Level 4, Modulation :16-QAM');
    M= 16;
    [ x1, x2, x3, x4, H1,H2,A1,A2] = multenclev4(B,data_set,data_size,codeSize);
    y1 = reshape(x1',1,numel(x1));
    y2 = reshape(x2',1,numel(x2));
    y3 = reshape(x3',1,numel(x3));
    y4 = reshape(x4',1,numel(x4));

% M-ary QAM nodulation, addition of noise, demodulation and decoding

```

```

yc=[y1;y2;y3;y4];

[Recovered_data,ser,SNRdB,M]=ts_mod_decodlvl4_16QAM(M,yc,coderate,in_data,codeSize,H
1,H2,A1,A2,data_set);

elseif num_levels==8
display('Modulate using 16-QAM or 256-QAM?');
M=input('Enter the value of M (16 or 256):');

if M ==16
[x1,x2,x3,x4,x5,x6,x7,x8,H1,H2,H3,H4,A1,A2,A3,A4] =
multenclev8(B,data_set,data_size,codeSize)
%Preparing for 16-QAM
y1 = reshape(x1',1,numel(x1));
y2 = reshape(x2',1,numel(x2));
y3 = reshape(x3',1,numel(x3));
y4 = reshape(x4',1,numel(x4));
y5 = reshape(x5',1,numel(x5));
y6 = reshape(x6',1,numel(x6));
y7 = reshape(x7',1,numel(x7));
y8 = reshape(x8',1,numel(x8));
s1 = [y1;y2;y5;y6];
s2= [y3;y4;y7;y8];
data_in_sym1=bi2de(s1,'left-msb');
data_in_sym2=bi2de(s2,'left-msb')
dataSymbolsIn= reshape([data_in_sym1, data_in_sym2]', 1, []);

[Recovered_data,ser,SNRdB,M]=tx_qammod_decodlvl8_MQAM(M,dataSymbolsIn,coderate,in
_data,codeSize,H1,H2,H3,H4,A1,A2,A3,A4,data_set)

elseif M==256
[x1,x2,x3,x4,x5,x6,x7,x8,H1,H2,H3,H4,A1,A2,A3,A4] =
multenclev8(B,data_set,data_size,codeSize)
y1 = reshape(x1',1,numel(x1));
y2 = reshape(x2',1,numel(x2));
y3 = reshape(x3',1,numel(x3));
y4 = reshape(x4',1,numel(x4));
y5 = reshape(x5',1,numel(x5));
y6 = reshape(x6',1,numel(x6));
y7 = reshape(x7',1,numel(x7));
y8 = reshape(x8',1,numel(x8));
s1=[y1;y2;y3;y4;y5;y6;y7;y8];
data_in_sym= bi2de(s1,'left-msb');

[Recovered_data,ser,SNRdB,M]=tx_qammod_decodlvl8_MQAM(M,data_in_sym,coderate,in_d
ata,codeSize,H1,H2,H3,H4,A1,A2,A3,A4,data_set)

```

```
else
    disp('Other Modulation not available now!');
end
else
    disp('Higher level not available now!');
end
end
```

APPENDIX I

M-ARY QAM MODULATION, ADDITION OF NOISE, DEMODULATION, ERROR COUNT AND FIND SER VALUES

Appendix I contain the MATLAB code for M -ary QAM modulations, transmissions of data to AWGN channel, M -ary QAM demodulations and sends demodulated data for decoding at each level. This also calculates the number of bit errors by comparing the recovered data with input data bits. Finally it calculates the numerical values of SER.

```
*****
% tx_qammod_demod_decod function performs the 4-QAM modulation, transmit data to the
AWGN channel, demodulation, and sends demodulated data to decoding. It also calculates the
bit error in recovered data and finds the bit error rate and symbol error rates.
*****

function
[Recovered_data,ser,SNRdB,M]=tx_qammod_demod_decod(data_in_sym,H,A,data_size,coderate,
y_1,y_2,in_data,codeSize)
M=4;
t=log2(M);
% QAM Modulation
dataMod=qammod(data_in_sym,M);

MaxsnrdB = input('Enter the maximum value of Eb/No in dB: ');
SNRdB =0:MaxsnrdB;
for ite=1:length(SNRdB)

    SNR(ite) = SNRdB(ite)+10*log10(t*coderate);
    receivedSignal = awgn(dataMod,SNR(ite),'measured');

% Scatter plot of the received data at receiver end
% subplotfig=scatterplot(receivedSignal,1,0,'c. ');
% hold on;
% scatterplot(dataMod,1,0,'k*',subplotfig);

% Demodulation of 4-QAM
Demod_datasyms = qamdemod(receivedSignal,M);
Demod_dataOut = de2bi(Demod_datasyms,t,'left-msb');

% Converting demodulated symbol to parallel form for performing PDL.
y1_dec = Demod_dataOut (:,1);
y2_dec = Demod_dataOut (:,2);

% Correlative decoding for 2 levels
[Recovered_data] = Decoding_level_2(y1_dec,y2_dec,codeSize,H,A,data_size)
```

```

% Computing the bit/symbol error rate
[numerror(ite),ber(ite)] = biterr(in_data,Recovered_data);
ser = ber *t;

```

```
end
```

```

*****
% ts_mod_decod_lev4_4QAM function performs the 4-QAM modulation, transmission of
modulated data to AWGN channel, demodulation, and sends the demodulated data for decoding.
It also counts the number of errors in recovered bits and finds the bit error rate and symbol error
rates. This function works for four encoding levels and different from tx_qammod_demod_decod
function which works for two encoding levels.
*****

```

```
function
```

```

[Recovered_data,ser,SNRdB,M]=ts_mod_decod_lev4_4QAM(M,data_in_sym,data_size,coderate
e,in_data,codeSize,H1,H2,A1,A2,data_set)
%M=4;
t=log2(M);
dataMod = qammod(data_in_sym,M);

```

```

MaxsnrdB = input('Enter the maximum value of Eb/No in dB: ');
SNRdB =0:MaxsnrdB;

```

```
for ite=1:length(SNRdB)
```

```

    SNR(ite) = SNRdB(ite)+10*log10(t*coderate);
    receivedSignal = awgn(dataMod,SNR(ite),'measured');

```

```

% Demodulation of 4-QAM
Demod_datasympols = qamdemod(receivedSignal,M);
dataSymbol = Demod_datasympols';

```

```

demod1 = dataSymbol(1:2:end,:);
demod2 = dataSymbol(2:2:end,:);
rc1 = de2bi(demod1,2,'left-msb');
rc2 = de2bi(demod2,2,'left-msb');
y1_rec= rc1(:,1);
y2_rec= rc2(:,1);
y3_rec= rc1(:,2);
y4_rec= rc2(:,2);

```

```

[Recovered_data] =
Decoding_level_4(y1_rec,y2_rec,y3_rec,y4_rec,codeSize,H1,H2,A1,A2,data_set)

```

```
[biterror(ite),ber(ite)] = biterr(in_data,Recovered_data);
ser= ber*t;
```

```
end
```

```
*****
% ts_mod_decodlvl4_16QAM function performs the 16-ary QAM modulation, transmits data to
AWGN channel, demodulation, and sends demodulated data for decoding. It also counts the
number of errors in decoded data and computes numerical values of the BER and SER.
*****
```

```
function
```

```
[Recovered_data,ser,SNRdB,M]=ts_mod_decodlvl4_16QAM(M,yc,coderate,in_data,codeSize,H
1,H2,A1,A2,data_set)
```

```
%M= 16;
```

```
t = log2(M);
```

```
% Preparing for QAM modulation
```

```
dataSymbolsIn = bi2de(yc'); % Convert to integers
```

```
dataMod = qammod(dataSymbolsIn,M);
```

```
MaxsnrdB = input('Enter the maximum value of Eb/No in dB: ');
```

```
SNRdB =0:MaxsnrdB;
```

```
for ite=1:1:length(SNRdB)
```

```
    snr0_2(ite) = SNRdB(ite) + 10*log10(t)+10*log10(coderate);
```

```
    receivedSignal = awgn(dataMod, snr0_2(ite), 'measured');
```

```
% Scatter plot
```

```
% subplotfig=scatterplot(receivedSignal,1,0,'c.');
```

```
% hold on;
```

```
% scatterplot(dataMod,1,0,'k*',subplotfig);
```

```
% hold off;
```

```
% Demodulation of 16-QAM
```

```
Demod_dataSymbols= qamdemod(receivedSignal,M);
```

```
Demod_dataOut = de2bi(Demod_dataSymbols,t);
```

```
% Preparing demodulated data to correlative decoding at each level
```

```
y1_rec= Demod_dataOut(:,1);
```

```
y2_rec= Demod_dataOut(:,2);
```

```
y3_rec= Demod_dataOut(:,3);
```

```
y4_rec= Demod_dataOut(:,4);
```

```
% Performing correlative decoding at each-level.
```

```
[Recovered_data] =
```

```
Decoding_level_4(y1_rec,y2_rec,y3_rec,y4_rec,codeSize,H1,H2,A1,A2,data_set)
```

```
% Calculate the number of errors, bit error rate and symbol error rate.
```



```
[biterror(ite),ber(ite)] = biterr(in_data,Recovered_data);
ser= ber*t;
end
```

```
*****
% tx_qammod_decodlvl8_MQAM function performs the 16-QAM and 256-QAM modulations,
transmission of data in AWGN channel, demodulations and sends demodulated data for
decoding. Errors in decoded data are counted and BER and SER are computed.
*****
```

```
function
```

```
[Recovered_data,ser,SNRdB,M]=tx_qammod_decodlvl8_MQAM(M,data_in_sym,coderate,in_d
ata,codeSize,H1,H2,H3,H4,A1,A2,A3,A4,data_set)
```

```
%For M= 16 or 256
```

```
t=log2(M);
```

```
%MQAM modulation
```

```
dataMod = qammod(data_in_sym,M,0);
```

```
MaxsnrdB = input('Enter the maximum value of Eb/No in dB: ');
```

```
SNRdB =0:MaxsnrdB;
```

```
for ite=1:1:length(SNRdB)
```

```
    % Addition of white noise to modulated symbols
```

```
    receivedSignal = awgn(dataMod, snr0_2(ite), 'measured');
```

```
    % Demodulation of MQAM
```

```
    Demod_datasyms = qamdemod(receivedSignal,M);
```

```
    if M== 16
```

```
        dataSymbol = Demod_datasyms';
```

```
        demod1 = dataSymbol(1:2:end,:);
```

```
        demod2 = dataSymbol(2:2:end,:);
```

```
        rc1 = de2bi(demod1,4,'left-msb');
```

```
        rc2 = de2bi(demod2,4,'left-msb');
```

```
        y1_rec= rc1(:,1);
```

```
        y2_rec= rc1(:,2);
```

```
        y3_rec= rc2(:,1);
```

```
        y4_rec= rc2(:,2);
```

```
        y5_rec= rc1(:,3);
```

```
        y6_rec= rc1(:,4);
```

```
        y7_rec= rc2(:,3);
```

```
        y8_rec= rc2(:,4);
```

```
[Recovered_data] =  
Decoding_level_8(y1_rec,y2_rec,y3_rec,y4_rec,y5_rec,y6_rec,y7_rec,y8_rec,codeSize,H1,H2,H  
3,H4,A1,A2,A3,A4,data_set)
```

```
elseif M==256  
dataSymbol = Demod_datasymbols;  
rc1 = de2bi(dataSymbol,t,'left-msb');
```

```
% Separating symbols to individual array for each level to decode.
```

```
y1_rec= rc1(:,1);  
y2_rec= rc1(:,2);  
y3_rec= rc1(:,3);  
y4_rec= rc1(:,4);  
y5_rec= rc1(:,5);  
y6_rec= rc1(:,6);  
y7_rec= rc1(:,7);  
y8_rec= rc1(:,8);
```

```
[Recovered_data] =  
Decoding_level_8(y1_rec,y2_rec,y3_rec,y4_rec,y5_rec,y6_rec,y7_rec,y8_rec,codeSize,H1,H2,H  
3,H4,A1,A2,A3,A4,data_set)
```

```
else  
display('Error: Other modulations not available !!');  
end
```

```
[biterror1(ite),ber(ite)] = biterr(in_data,Recovered_data);  
ser= ber *t;  
end
```

REFERENCES

- [1] S. Lin, D. J. Costello, Jr, *Error Control Coding*, 2nd ed., Prentice Hall, 2004.
- [2] J. G. Proakis, M. Salehi, *Digital Communications*, 5th ed., McGraw Hill, 2008.
- [3] B. Sklar, *Digital communications fundamentals and applications*, 2nd ed., Prentice Hall, 2001.
- [4] K. Pahlavan, A. H. Levesque, *Wireless Information Networks*, 2nd ed., 2005.
- [5] J. C. Whitaker, Ed., *The Electronics Handbook*, 2nd ed., CRC Press, 2005.
- [6] J. D. Gibson, Ed., *The Mobile Communications Handbook*, 3rd ed., CRC Press, 2013.
- [7] Z. Ghassemblooy, W. Popoola, and S. Rajbhandari, *Optical wireless communications, system and channel modelling with MATLAB*, Press, Boca Raton, London & New York, 1-34, 346-394 (2013).
- [8] S. Haykin, M. Moher, *Modern Wireless Communications*, Pearson Prentice Hall, Upper Saddle River, NJ, 2005.
- [9] R. H. Morelos-Zaragoza, *The Art of Error Correcting Coding*, 2nd ed., John Wiley & Sons. Ltd, 2006.
- [10] "Hamming codes for NAND flash memory devices", Micron technology, 2005.
- [11] D. R. Shier and K. T. Wallenius, *Applied Mathematical Modeling: A Multidisciplinary Approach*, Chapter 14, Chapman & Hall/CRC Press, 1999.
- [12] Y. Jiang, *A practical guide to error control coding using MATLAB*, Artech House, 2010.

- [13] R. G. Gallager, “Low-density parity-check codes,” *IRE Transactions on Information Theory*, pp. 21–28, Jan. 1962.
- [14] R. M. Tanner, “A recursive approach to low complexity codes”, *IEEE Trans. Information Theory*, pp. 533-547, Sept. 1981.
- [15] D. MacKay, “Good error correcting codes based on very sparse matrices,” *IEEE Trans. Information Theory*, pp. 399-431, March 1999.
- [16] R.G. Maunder, “A vision for 5G channel coding”, AccelerComm White Paper, September, 2016.
- [17] J. L. Massey, “Coding and modulation in digital communications”, *Zurich Seminar*, 1974.
- [18] G. Ungerboeck, “Channel coding with multilevel/phase signals”, *IEEE Trans. Inform. Theory*, vol. IT-28, pp. 55–67, Jan. 1982.
- [19] H. Imai & S. Hirakawa, “A new Multilevel coding method using error-correcting codes”, *IEEE Trans. on Information Theory*, vol. IT-23, no.3, pp. 371-377, May 1977.
- [20] A. Ghaith, A. Alaeddine, “Multi-level Coding and Multi-Stage Decoding for Modulations with Hexagonal Constellation”, *European Scientific Journal*, Oct. 2014, vol.10, no.30.
- [21] U. Wachsmann, R. F. H. Fischer, and J. B. Huber, “Multilevel Codes: Theoretical Concepts and Practical Design Rules”, *IEEE Trans. on information theory*, vol. 45, no. 5, July 1999.
- [22] R. Y. S. Tee, Iterative decoding of multilevel and Bit-interleaved codes, [Online]. Available: http://www-mobile.ecs.soton.ac.uk/comms/seminars/mini-viva/mini-viva_RonaldYeeSiongTee.pdf (26 July 2017).
- [23] J. Hou, P. H. Siegel, L. B. Milstein, H. D. Pfister, “Capacity-approaching bandwidth-efficient coded modulation schemes based on Low-Density Parity-Check Codes”, *IEEE Trans. on Information Theory*, vol. 49, no. 9, Sept. 2003.

- [24] D. Yuan, et.al. “Optimum design criterion and multilevel coding for radio systems over AWGN and Rayleigh fading channels”, *Wirel. Commun. Mob. Comput.* 2003; 3:617–628 (DOI: 10.1002/wcm.145).
- [25] S. Haykin, *Communication Systems*, 4th ed., John Wiley & Sons, 2001.
- [26] D. Bryan, “QAM for terrestrial and cable transmission”, *IEEE Trans. Consumer Electronics*, vol. 41, no.3, 1995, pp. 383-391.
- [27] G. Karam, K. Maalej, V. Paxal, and H. Sari, “Variable symbol-rate demodulators for cable and satellite TV broadcasting”, *IEEE Trans. Broadcasting*, vol. 42, no.2, 1996, pp. 102-109.
- [28] L. D’Luna et al., “A single chip universal cable set-top box/modem transceiver”, *IEEE Journal of Solid-State Circuits*, vol. 34, no.11, 1999, pp. 1647-1660.
- [29] A. Saadani, J.B. Landre, “Realistic performance of HSDPA evolution 64-QAM in macro-cell environment”, *IEEE VTC Spring*, 2009.
- [30] UMTS Networks, Enhanced High-Speed packet access HSPA+, [Online]. Available: https://www.tu-ilmenau.de/fileadmin/public/iks/files/lehre/UMTS/13_UMTS-HSPA+_ws11.pdf (29 January 2017).
- [31] J. L. Walsh, “A closed set of Normal orthogonal functions”, *American Journals of Mathematics*, vol. 45, no. 1, pp.5-24, Jan. 1923.
- [32] A.N. Akansu, R.Poluri, “Walsh-Like nonlinear phase orthogonal codes for direct sequence CDMA communications”, *IEEE Trans. on Signal Processing*, vol. 55. No. 7, July 2007.
- [33] S. Faruque, *Cellular Mobile System Engineering*, Artech House Publishers, 1996.
- [34] W. Stallings. *Orthogonal Sequences* (August 1 2001), [Online]. Available: <http://www.drdoobbs.com/orthogonal-sequences/184404738?pgno=1> (25 December 2016).

- [35] S. Faruque, P. Maveddat, "Battlefield wideband transceivers based on combined N-ary orthogonal signaling and M-ary PSK modulation", *Proc. SPIE 3709, Digitation of Battlespace IV*, 1999, pp. 123-128.
- [36] S. Faruque, "Error control coding based on orthogonal codes", *Wireless Preceedings*, vol.2, 2004, pp.608-615.
- [37] S. Faruque, "Orthogonal MPSK (OMPSK) With Bandwidth Efficiency", *Innovations on Communication Theory, INCT-2012*, Booklet (4), 143-147, 2012.
- [38] S. Faruque, Sh. Faruque, and W. Semke, "Orthogonal On-Off Keying for Free-Space Laser Communications with Ambient Light Cancellation", *SPIE Optical Engineering Journal*, vol.52, (9), Sept. 26, 2013.
- [39] N. Kaabouch, A. Dhirde, S. Faruque, "Improvement of the orthogonal codes convolution based on FPGA implementation", *IEEE Electro/Information Technology Proceedings*, 337-341, 2007.
- [40] S. K. Gaire, S. Faruque, M. M. Ahamed, "*OptoRadio*: A method of wireless communication using orthogonal M -ary PSK (OMPSK) modulation", *Proc. SPIE 9979, Laser Communication and Propagation through the Atmosphere and Oceans V*, 99790D, Sept., 2016.
- [41] S. K. Gaire, S. Faruque, "Performance analysis of orthogonal coded M -ary QAM modulation technique", *CISS*, May, 2017.
- [42] M. Viswanathan (2011, March), Walsh Hadamard code-Matlab simulation. [Online]. Available: <http://www.gaussianwaves.com/2011/03/walsh-hadamard-code-matlab-simulation-2/> (26 July, 2017).