THE SIMULATION OF WIMAX – WORLD WIDE INTEROPERABILITY FOR MICROWAVE ACCESS – PHYSICAL LAYER.

By:

CSUO NAUMAN RAUF
TC ZEESHAN SHAHID
TC PIR FEROZ SHAH
TC AHMAD ALI

PROJECT SUPERVISOR

Asst Prof Maj(r) Fazal Ahmed
Electrical Engineering Department
Military College of Signals
NUST

Project Report Submitted in partial fulfillment of the requirements of MCS/ NUST for the award of B.E. degree in Telecommunication Engineering

DEPARTMENT OF ELECTRICAL ENGINEERING
MILITARY COLLEGE OF SIGNALS
RAWALPINDI
MARCH, 2008

DEDICATION

To our Parents

for there love and prayers without whom we could do nothing

To our Coursemates

For there support and help whenever we needed it.

To the Faculty of MCS

For there Knowledge and experience that will guide us for the rest of our lives

ACKNOWLEDGEMENTS

All praises to the Almighty Allah, who gave us strength and entrusted us with such knowledge to accomplish the project goals that we had set for ourselves at the beginning of the project, and to the successful completion of this degree.

We would like to take this opportunity to thank all the people who guided us and helped us accomplish our project. We would like to thank Mr. Abdur Rehman Usmani for his constant guidance and help he extended to us whenever we needed it. We would specially like to thank our Project supervisor Asst Prof Maj (r) Fazal Ahmed who has extended help to us whenever we needed it.

ABSTRACT

IEEE 802.16d is a wireless standard used to provide broad band access over a large area. We are analyzing the BER for the PHY layer of IEEE 802.16d in the context of Adaptive modulation and coding scheme, in the attempt to analyze the WiMAX, so that by using the BER result for different modulation schemes, Coding rates and Cyclic Prefix (G) value, the appropriate value of Data rate can be achieved with efficient use of power.

Table of contents

List of Figures

CHAPTER 1:

1 INTRODUCTION

We live in a world where everyday we are required to have the right information at the right time. The amount of information transported over communication systems grows rapidly. Not only does the file size increases, but also the bandwidth required for applications such as video on demand and video conferencing require increasing data rates to transfer the information in a reasonable amount of time or to establish real-time connections. To support this kind of services, broadband communication systems are required.

There are a lot of places where large Data rates and transfer rate. In places such as hospitals where the right knowledge may be essential for saving lives. Broad band and high data rates finds its uses in almost every aspect of life, be it a student in university, or an employ working in an office all need access to the Internet. Another case that needs to be incorporated in to this is the factor of mobility. In today's fast changing world we can no longer stay at one place to access the info that is required.

1.1 WiMAX

WiMAX, the Worldwide Interoperability for Microwave Access, is a wireless technology aimed at providing wireless data over long distances with high data rate.It is based on the IEEE -802.16 standard, which is also called Wireless MAN.

There are two system profiles in the standard

1. Fixed WiMAX

2. Mobile WiMAX

1 .2 Motivation for WiMAX

DSL/cable technologies require telephone/cable lines to be laid over long distances to serve customers. A viable complement to DSL/cable based service is WiMAX or wireless broadband, which connects users to the Internet, even in places where the infrastructure might not be as developed. Because of the construction limitations, every one want to get rid of cable network and WiMAX is a wireless technology, so is the replacement of cable network. Because of very fast life, people want to shift to a technology which provides high data so that they don't have to wait for a long to download a large size document. Because investors want to cover a large area with less equipment to avoid the maintenance and administrative cost and to make the world a global village, WiMAX will be helpful to connect those people, who are living in rural area. WiMAX will also provide the communication in those areas where cable network is very difficult to deploy, like hilly areas. WiMAX can achieve data rates up to 75Mbps and a theoretical 30 mile reach, however, in typical deployment scenarios, data rates fall with increasing reach. Geographically WiMAX is flexible and can improve yield due to wiring/labor cost savings. It is very likely that service providers will use the tiered pricing approach and service contracts that they currently employ for DSL/cable.

1.3 Working of WiMAX:

A WiMAX system consists of two parts:

1.3.1 *WiMAX tower:* similar in concept to a cell-phone tower. A single WiMAX tower can provide coverage to a very large area; as big as 3,000 square miles (~8,000 square km).

1.3.2 *WiMAX receiver*: The receiver and antenna could be a small box or PCMCIA card, or they could be built into a laptop the way Wife access is today.

WIMAX TOWER WIMAX RECEIVER

Figure 1.1 WiMAX Transmitter and receiver.

A WiMAX tower station can connect directly to the Internet using a high-bandwidth, wired connection (for example, a T3 line). It can also connect to another WiMAX tower using a line-of-sight, microwave link. This connection to a second tower (often referred to as a backhaul), along with the ability of a single tower to cover up to 3,000 square miles, is what allows WiMAX to provide coverage to remote rural areas.

1.4 MODES OF OPERATION:

1.4.1 Non-Line of sight (NLOS): Uses a lower frequency range.

i.e. 2 → 11 GHZ

There is the non-line-of-sight, WiFi sort of service, where a small antenna on your computer connects to the tower. In this mode, WiMAX uses a lower frequency range; 2 GHz to 11 GHz (similar to WiFi). Lower-wavelength transmissions are not as easily disrupted by physical obstructions, they are better able to diffract, or bend, around obstacles.

1.4.2 Line of sight (LOS): Uses a higher frequency range.

I.e. 10 → 66 GHZ

The line-of-sight connection is stronger and more stable, so it's able to send a lot of data with fewer errors. Line-of-sight transmissions use higher frequencies, with ranges reaching a possible 66 GHz. At higher frequencies, there is less interference and lots more bandwidth.

1.5 Duplexing modes:

1.5.1 Frequency division duplex (FDD):

Here transmission and reception is done at the same time but using different frequencies. Due to the limited frequency resource, it is not preferred.

1.5.2 Time division duplex (TDD):

Here transmission and reception is done by using same frequency but at different time. Here transmitter and receiver are simpler to construct than in case of FDD.

1.6 BENEFITS OF WIMAX:

- **High data**:

  Provides data rates up to 70 MBPS in ideal conditions but in practical conditions provides data rates up to 10 MBPS.

- **Wireless**

  Not having to lay cables reduces cost, so is the alternative to cable and DSL networks

- **Broad Coverage.**

  Provides network coverage up to 50 km.

- **Providing a diverse source of Internet connectivity.**
- **Mobility.**
- **Flexibility.**
- **Scalability**.

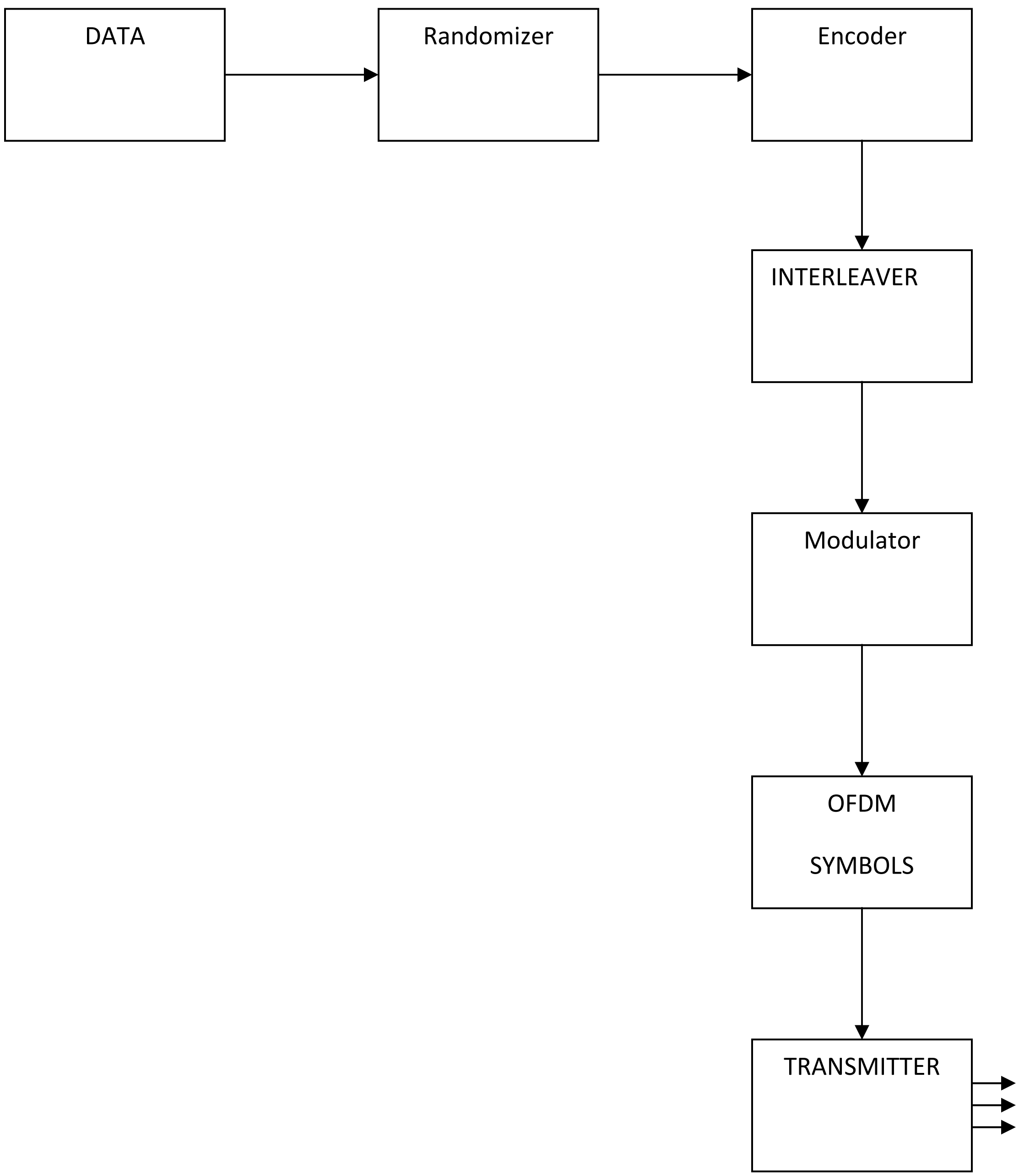
  Scale to work in different channelizations from 1.25 to 20 MHz

- **OFDM based physical layer provides resistance to multipath.**
- **Adaptive modulation and coding scheme.**
- **Support for TDD and FDD.**

SCHEMETIC DIAGRAMS:

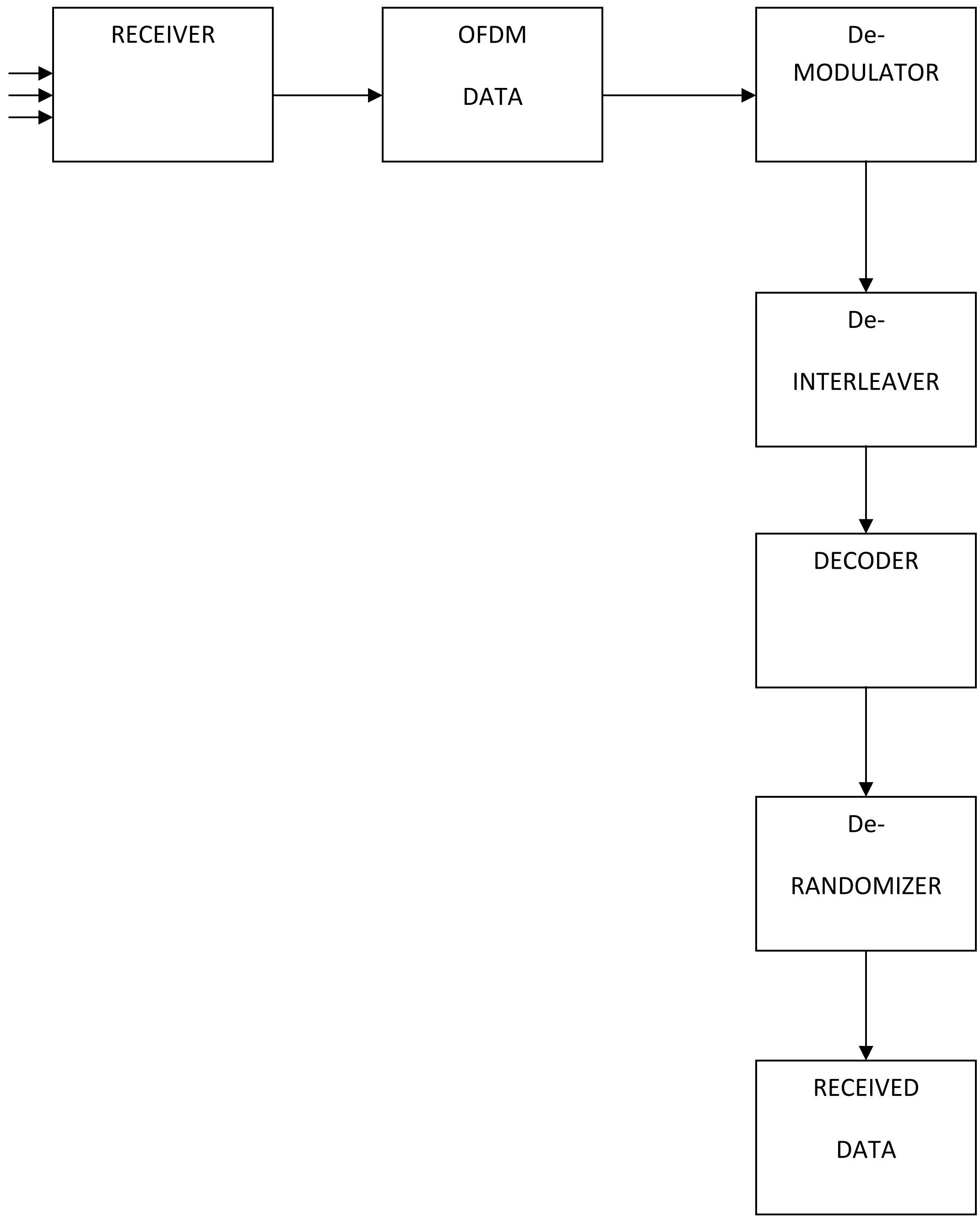
We have implemented the physical layer of 802.16d in the following way:

Transmitter Blocks:



SCHEMETIC DIAGRAMS:

Receiver Blocks:



CHAPTER 2:

Channel coding:

Channel coding is a viable method to reduce information rate through the channel and increase reliability. This goal is achieved by adding redundancy to the information.

First data is generated according to the coding rate and modulation scheme used. Then it is sent to the channel coding block. Channel code is a broadly used term mostly referring to the forward error correction code in communication. FEC in WiMAX consists of

1. Randomizer

2. Encoder

3. Interleaver

2.1. Randomizer:

Randomization is done on each block independently. Randomization vector is of 15 bits.

First 4 bits=base station ID, then we place two 1's, next 4 bits= down link identifier unique code, next we place a bit equal to binary 1 and then 4 bits=frame number.

We take the XOR of 14th and 15th bit and we take the XOR of this value and incoming bit, along with this we also send the output of XOR of 14th and 15th bit at the position of first bit of randomizing vector and previous values of randomizing vector are shifted 1 bit to its right so that previous 14th bit is now a5th bit. Same technique of randomization is used on transmitter and receiver side.

2.2. Encoder:

Two main types of encoders are

Block encoder

Convolutional encoder

2.2.1. RS-encoder:

This is a block code; here information sequence is divided into blocks of length “k”. Each block is mapped into channel inputs of length“n”. The mapping is independent from previous blocks, that is, there is no memory from one block to another.

Reed-Solomon codes are *nonbinary cyclic* codes with symbols made up of *m*-bit sequences, where *m* is any positive integer having a value greater than 2. R-S (*n*, *k*) codes on *m*-bit symbols exist for all *n* and *k* for which

$$0 < k < n < 2m + 2$$

Where k is the number of data symbols being encoded

n is the total number of code symbols in the encoded block. For the most conventional

R-S (*n*, *k*) code:

$$(n, k) = (2m - 1, 2m - 1 - 2t) \qquad (2.1)$$

where t is the symbol-error correcting capability of the code

$n - k = 2t$ is the number of parity symbols.

2.2.1.1. Finite Fields

In order to understand the encoding and decoding principles of nonbinary codes, such as Reed-Solomon (R-S) codes, it is necessary to venture into the area of finite fields known as *Galois Fields* (GF). For any prime number, *p*, there exists a finite field denoted GF (p) that contains *p* elements. It is possible to extend GF (p) to a field of *pm* elements, called an *extension field* of GF (p), and denoted by GF (pm), where *m* is a nonzero positive integer. Note that GF (pm) contains as a subset the elements of GF (p). Symbols from the extension field GF (2*m*) are used in the construction of Reed-Solomon (R-S) codes. The binary field GF(2) is a subfield of the extension field GF(2*m*), in much the same way as the real number field is a subfield of the complex number field. Besides the numbers 0 and 1, there are additional unique elements in the extension field that will be represented with a new symbol α. Each nonzero element in GF (2*m*) can be represented by a power of α. An *infinite* set of elements, *F*, is formed by starting with the elements {0, 1, α}, and generating additional elements by progressively multiplying the last entry by α, which yields the following:

F = {0, 1, α, α2, …, α *j*, …} = {0, α0, α1, α2, …, α *j*, …} (2.2)

To obtain the *finite* set of elements of GF (2*m*) from *F*, a condition must be imposed on *F* so that it may contain only 2*m* elements and is closed under multiplication. The condition that closes the set of field elements under multiplication is characterized by the irreducible polynomial shown below:

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

Or equivalently

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

Using this polynomial constraint, any field element that has a power equal to or greater than 2*m* - 1 can be reduced to an element with a power less than 2*m* - 1, as

Follows:

$$\alpha^{(2^m + n)} = \alpha^{(2^m - 1)} \alpha^{n+1} = \alpha^{n+1}$$

2.2.1.2. The Extension Field GF (23)

Consider an example involving a primitive polynomial and the finite field that it defines. Table contains a listing of some primitive polynomials. We choose the first one shown, *f*(*X*) = 1 + *X* + *X* 3, which defines a finite field GF (2*m*), where the degree of the polynomial is *m* = 3. Thus, there are 2*m* = 23 = 8 elements in the field defined by *f*(*X*). Solving for the roots of *f*(*X*) means that the values of *X* that correspond to *f*(*X*) = 0 must be found. The familiar binary elements, 1 and 0, do not satisfy (are not roots of) the polynomial *f*(*X*) = 1 + *X* + *X* 3, since *f* (1) = 1 and *f* (0) = 1 (using modulo-2 arithmetic). Yet, a fundamental theorem of algebra states that a polynomial of degree *m* must have precisely *m* roots. Therefore for this example, *f*(*X*) = 0 must yield three roots. Clearly a dilemma arises, since the three roots do not lie in the same finite field as the coefficients of *f*(*X*). Therefore, they must lie somewhere else; the roots lie in the extension field, GF (23). Let α, an element of the extension field is defined as a root of the polynomial *f*(*X)*.

Therefore, it is possible to write the following:

F (α) = 0

1 + α + α3 = 0 (16)

α3 = –1 – α

Since in the binary field +1 = −1, α3 can be represented as follows:

α3 = 1 + α

Thus, α3 is expressed as a weighted sum of α-terms having lower orders. In fact all powers of α can be so expressed. For example, consider α4, where we obtain

$$\alpha^4 = \alpha \cdot \alpha^3 = \alpha \cdot (1 + \alpha) = \alpha + \alpha^2$$

Now, consider α5, where

$$\alpha^5 = \alpha \cdot \alpha^4 = \alpha \cdot (\alpha + \alpha^2) = \alpha^2 + \alpha^3$$

α5 = 1 + α + α2

Now, for α6, we obtain

$$\alpha^6 = \alpha \cdot \alpha^5 = \alpha \cdot (1 + \alpha + \alpha^2) = \alpha + \alpha^2 + \alpha^3 = 1 + \alpha^2$$

And for α7, we obtain

$$\alpha^7 = \alpha \cdot \alpha^6 = \alpha \cdot (1 + \alpha^2) = \alpha + \alpha^3 = 1 = \alpha^0$$

Note that α7 = α0 and therefore the eight finite field elements of GF (23) are

{0, α0, α1, α2, α3, α4, α5, α6}

	Basis elements		
Field elements	X^0	X^1	X^2
0	0	0	0
α^0	1	0	0
α^1	0	1	0
α^2	0	0	1
α^3	1	1	0
α^4	0	1	1
α^5	1	1	1
α^6	1	0	1
α^7	1	0	0

And so we obtain

The generating polynomial for an R-S code takes the following form:

$$g(X) = g_0 + g_1 X + g_2 X^2 + \ldots + g_{2t-1} X^{2t-1} + X^{2t} \quad (2.3)$$

The degree of the generator polynomial is equal to the number of parity symbols. R-S codes are a subset of the Bose, Chaudhuri, and Hocquenghem (BCH) codes; hence, it should be no surprise that this relationship between the degree of the generator polynomial and the number of parity symbols holds, just as for BCH codes. Since the generator polynomial is of degree $2t$, there must be precisely $2t$ successive powers of α that are roots of the polynomial. We designate the roots of $g(X)$ as $\alpha, \alpha^2, \ldots, \alpha^{2t}$. It is not necessary to start with the root α; starting with any power of α is possible. Consider as an example the (7, 3) double-symbol-error correcting R-S code. We describe the generator polynomial in terms of its roots, as follows:

$$g(X)=(X-\alpha 1)(X-\alpha 2)(X-\alpha 3)(X-\alpha 4) \quad (2.4)$$

multiplying and Following the low order to high order format, and changing negative signs to positive, since in the binary field +1 = –1, g(*X*) can be expressed as follows:

$$g(X) = \alpha 3 + \alpha 1 X + \alpha 0 X 2 + \alpha 3 X 3 + X 4 \quad (2.5)$$

2.2.1.3. Systematic Encoding with an (*n - k*)–Stage Shift Register

Using circuitry to encode a three-symbol sequence in systematic form with the (7, 3) R -S code described by g(*X)* in Equation. It can easily be verified that the multiplier terms in Figure, taken from left to right, correspond to the coefficients of the polynomial in Equation (low order to high order).

This encoding process is the nonbinary equivalent of cyclic encoding [3]. Here, corresponding to Equation, the (7, 3) R-S nonzero codewords are made up of

2*m* - 1 = 7 symbols,

and each symbol is made up of *m* = 3 bits.

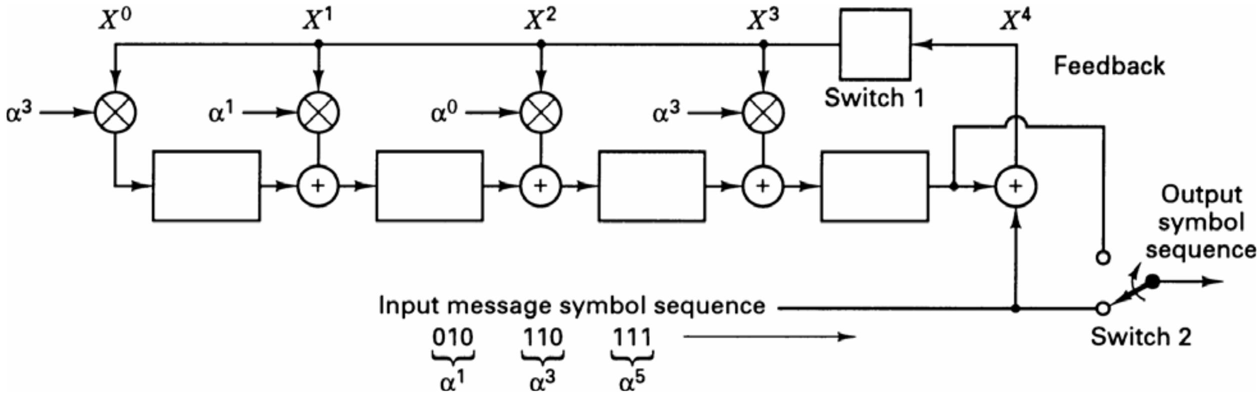


Fig 2.1: Reed Solomon Coding.

Here the example is nonbinary, so that each stage in the shift register of Figure holds a 3-bit symbol. In the case of binary codes, the coefficients labeled *g*1, *g*2, and so on are binary. Therefore, they take on values of 1 or 0, simply dictating the presence or absence of a connection in the LFSR. However in Figure, since each coefficient is specified by 3-bits, it can take on one of eight values. The nonbinary operation implemented by the encoder of Figure, forming codewords in a systematic format, proceeds in the same way as the binary one. The steps can be described as follows:

1. Switch 1 is closed during the first *k* clock cycles to allow shifting the message symbols into the (*n* - *k*)–stage shift register.

2. Switch 2 is in the down position during the first *k* clock cycles in order to allow simultaneous transfer of the message symbols directly to an output register.

3. After transfer of the *k*th message symbol to the output register, switch 1 is opened and switch 2 is moved to the up position.

4. The remaining (*n* - *k*) clock cycles clear the parity symbols contained in the shift register by moving them to the output register.

5. The total number o f clock cycles is equal to *n*

$$\underbrace{010}_{\alpha^1} \quad \underbrace{110}_{\alpha^3} \quad \underbrace{111}_{\alpha^5}$$

We use the same symbol sequence that was chosen as a test message earlier: where the rightmost symbol is the earliest symbol and the rightmost bit is the earliest bit. The operational steps during the first $k = 3$ shifts of the encoding circuit

INPUT QUEUE			CLOCK CYCLE	REGISTER CONTENTS				FEEDBACK
α^1	α^3	α^5	0	0	0	0	0	α^5
	α^1	α^3	1	α^1	α^6	α^5	α^1	α^0
		α^1	2	α^3	0	α^2	α^2	α^4
		—	3	α^0	α^2	α^4	α^6	—

After the third clock cycle, the register contents are the four parity symbols, α0, α2, α4, and α6, as shown. Then, switch 1 of the circuit is opened, switch 2 is toggled to the up position, and the parity symbols contained in the register are shifted to the output. Therefore the output codeword, U(*X)*, written in polynomial form, can be expressed as follows:

$$\mathbf{U}(X) = \sum_{n=0}^{6} u_n X^n$$

$$\mathbf{U}(X) = \alpha^0 + \alpha^2 X + \alpha^4 X^2 + \alpha^6 X^3 + \alpha^1 X^4 + \alpha^3 X^5 + \alpha^5 X^6$$

$$= (100) + (001)\,X + (011)\,X^2 + (101)\,X^3 + (010)\,X^4 + (110)\,X^5 + (111)\,X^6$$

The process of verifying the contents of the register at various clock cycles is somewhat more tedious than in the binary case. Here, the field elements must be added and multiplied .The roots of a generator polynomial, g(*X)*, must also be the roots of the codeword generated by g(*X)*, because a valid codeword is of the following form:

U(*X)* = m(*X)* g(*X)*

Therefore, an arbitrary codeword, when evaluated at any root of g(*X)*, must yield zero. It is of interest to verify that the codeword polynomial does indeed yield zero when evaluated at the four roots of g(*X)*. In other words, this means checking that

U (α) = U (α2) = U (α3) = U (α4) = 0

According to standard we are using

m=8, for which primitive polynomial is

$$F(x) = 1 + X^2 + X^3 + X^4 + X^8$$

2.2.2. Convolutional encoder:

Each block of k bits is mapped into a block of n bits but these n bits are not only determined by the present k information bits but also by the previous information bits. This dependence can be captured by a finite state machine.

k=input bits to encoder

n= output bits to encoder

K=constraint length which is equal to number of shift registers

Rate=k/n

To convolutionally encode data, start with *k* memory registers, each holding 1 input bit. All memory registers start with a value of 0. The encoder has *n* modulo-2 adders, and *n* generator polynomials — one for each adder. An input bit m_1 is fed into the leftmost register. Using the generator polynomials and the existing values in the remaining registers, the encoder outputs *n* bits. Now bit shift all register values to the right (m_1 moves to m_0, m_0 moves to m_{-1}) and wait for the next input bit. If there are no remaining input bits, the encoder continues output until all registers have returned to the zero state.

The figure below is a rate 1/3 (*m*/*n*) encoder with constraint length (*k*) of 3. Generator polynomials are $G_1 = (1,1,1)$, $G_2 = (0,1,1)$, and $G_3 = (1,0,1)$. Therefore, output bits are calculated (modulo 2) as follows:

$n_1 = m_1 + m_0 + m_{-1}$

$n_2 = m_0 + m_{-1}$

$n_3 = m_1 + m_{-1}$.

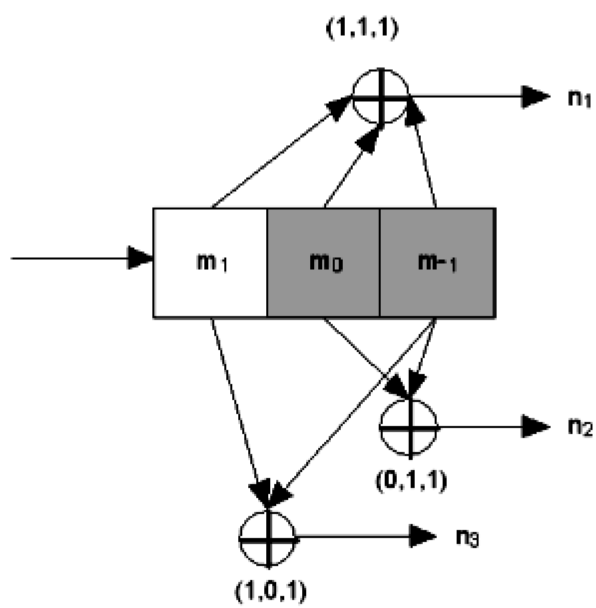


Fig 2.2 Convolutional Encoder

Rate 1/3, non-systematic convolutional encoder with constraint length 3

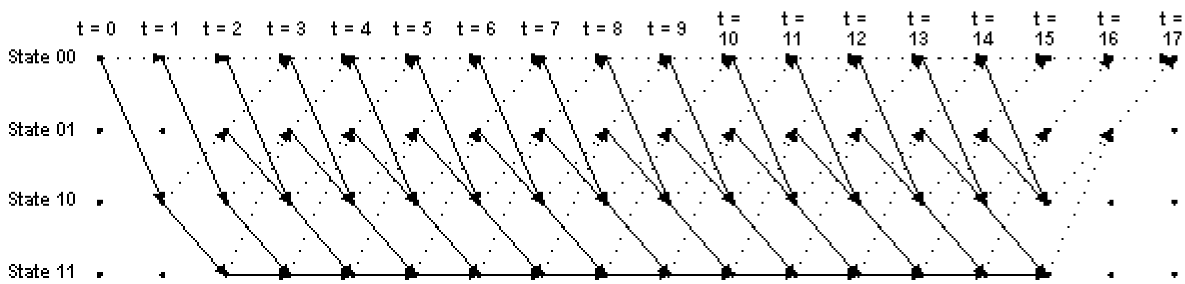
But according to standard we are using

Rate=1/2

Generator polynomials= [1 1 1 1 0 0 1 ; 1 0 1 1 0 1 1]

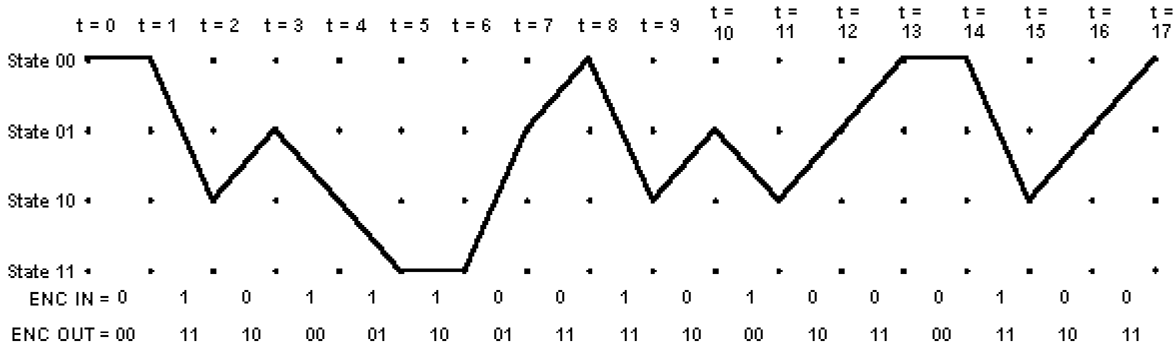
2.3 Veterbi decoding:

It is used for convolutional decoder. This is implemented by using the Trellis diagram. Let us take an example to understand its algorithim. Here, rate ½, K = 3 convolutional encoder, for a 15-bit message

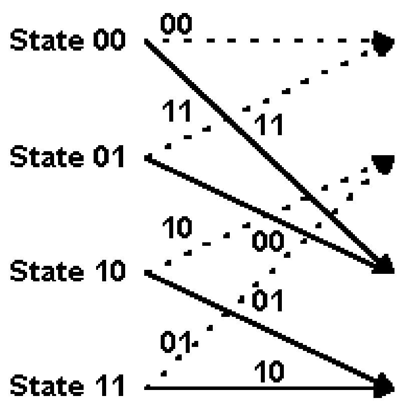


The four possible states of the encoder are depicted as four rows of horizontal dots. There is one column of four dots for the initial state of the encoder and one for each time instant during the message. For a 15-bit message with two encoder memory flushing bits, there are 17 time instants in addition to t = 0, which represents the initial condition of the encoder. The solid lines connecting dots in the diagram represent state transitions when the input bit is a one. The dotted lines represent state transitions when the input bit is a zero. Notice the correspondence between the arrows in the trellis diagram and the state transition table discussed above. Also notice that since the initial condition of the encoder is State 00_2, and the two memory flushing bits are zeroes, the arrows start out at State 00_2 and end up at the same state.

The following diagram shows the states of the trellis that are actually reached during the encoding of our example 15-bit message:

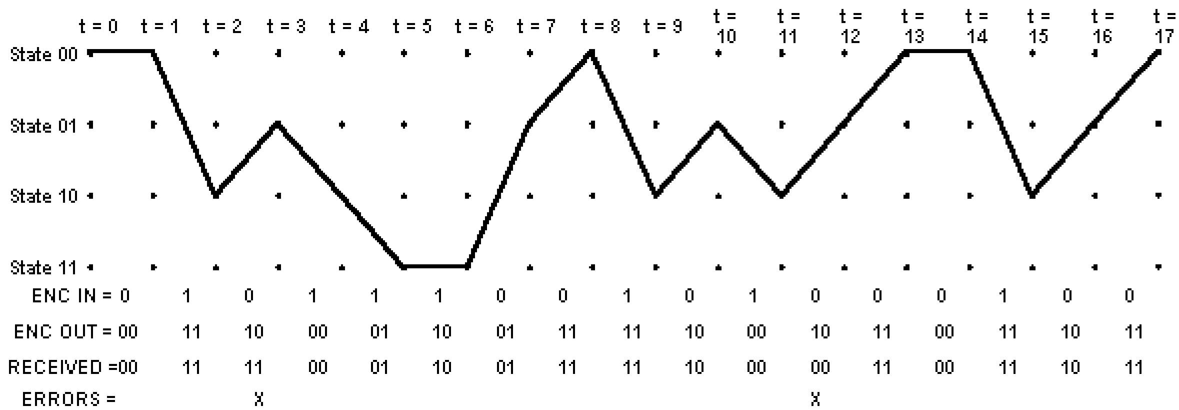


The encoder input bits and output symbols are shown at the bottom of the diagram. Notice the correspondence between the encoder output symbols and the output table discussed above. Let's look at that in more detail, using the expanded version of the transition between one times instant to the next shown below:



The two-bit numbers labeling the lines are the corresponding convolutional encoder channel symbol outputs. Remember that dotted lines represent cases where the encoder input is a zero, and solid lines represent cases where the encoder input is a one. (In the figure above, the two-bit binary numbers labeling dotted lines are on the left, and the two-bit binary numbers labeling solid lines are on the right.)

OK, now let's start looking at how the Viterbi decoding algorithm actually works. For our example, we're going to use hard-decision symbol inputs to keep things simple. (The example source code uses soft-decision inputs to achieve better performance.) Suppose we receive the above encoded message with a couple of bit errors:

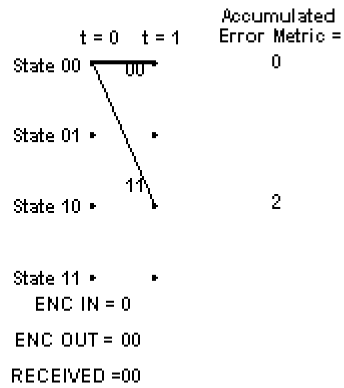


Each time we receive a pair of channel symbols, we're going to compute a metric to measure the "distance" between what we received and all of the possible channel symbol pairs we could have received. Going from t = 0 to t = 1, there are only two possible channel symbol pairs we could have received: 00_2, and 11_2. That's because we know the convolutional encoder was initialized to the all-zeroes state, and given one input bit = one or zero, there are only two states we could transition to and two possible outputs of the encoder. These possible outputs of the encoder are 00_2 and 11_2.

The metric we're going to use for now is the Hamming distance between the received channel symbol pair and the possible channel symbol pairs. The Hamming distance is computed by simply counting how many bits are different between the received channel symbol pair and the possible channel symbol pairs. The results can only be zero, one, or two. The Hamming distance (or other metric) values we compute at each time instant for the paths between the states at the previous time instant and the states at the current time instant are called branch metrics. For the first time instant, we're going to save these results as "accumulated error metric" values, associated with states. For the second time instant on, the accumulated error metrics will be computed by adding the previous accumulated error metrics to the current branch metrics.

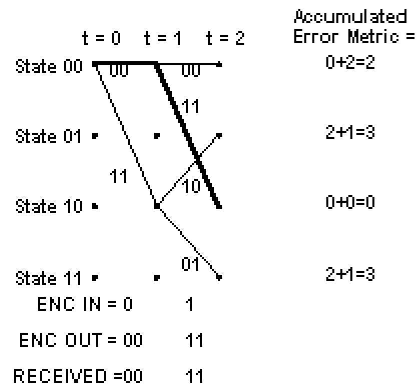
At t = 1, we received 00_2. The only possible channel symbol pairs we could have received are 00_2 and 11_2. The Hamming distance between 00_2 and 00_2 is zero. The Hamming distance between 00_2 and 11_2 is two. Therefore, the branch metric value for the branch from State 00_2 to State 00_2 is zero, and for the branch from State 00_2 to State 10_2 it's two. Since the previous accumulated error metric values are equal to zero, the accumulated metric values for State 00_2 and for State

10_2 are equal to the branch metric values. The accumulated error metric values for the other two states are undefined. The figure below illustrates the results at t = 1:



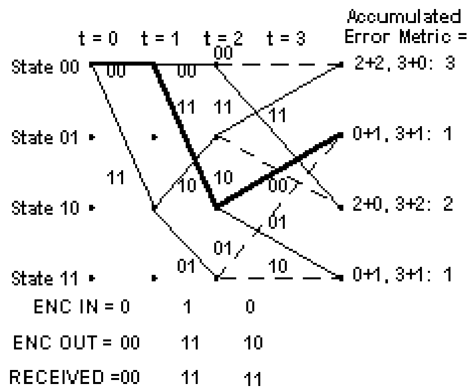
Note that the solid lines between states at t = 1 and the state at t = 0 illustrate the predecessor-successor relationship between the states at t = 1 and the state at t = 0 respectively. This information is shown graphically in the figure, but is stored numerically in the actual implementation. To be more specific, or maybe clear is a better word, at each time instant t, we will store the number of the predecessor state that led to each of the current states at t.

Now let's look what happens at t = 2. We received a 11_2 channel symbol pair. The possible channel symbol pairs we could have received in going from t = 1 to t = 2 are 00_2 going from State 00_2 to State 00_2, 11_2 going from State 00_2 to State 10_2, 10_2 going from State 10_2 to State 01_2, and 01_2 going from State 10_2 to State 11_2. The Hamming distance between 00_2 and 11_2 is two, between 11_2 and 11_2 is zero, and between 10_2 or 01_2 and 11_2 is one. We add these branch metric values to the previous accumulated error metric values associated with each state that we came from to get to the current states. At t = 1, we could only be at State 00_2 or State 10_2. The accumulated error metric values associated with those states were 0 and 2 respectively. The figure below shows the calculation of the accumulated error metric associated with each state, at t = 2.



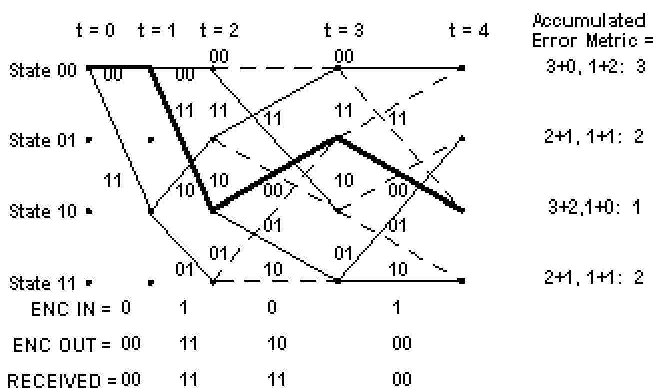
That's all the computation for t = 2. What we carry forward to t = 3 will be the accumulated error metrics for each state, and the predecessor states for each of the four states at t = 2, corresponding to the state relationships shown by the solid lines in the illustration of the trellis.

Now look at the figure for t = 3. Things get a bit more complicated here, since there are now two different ways that we could get from each of the four states that were valid at t = 2 to the four states that are valid at t = 3. So how do we handle that? The answer is, we compare the accumulated error metrics associated with each branch, and discard the larger one of each pair of branches leading into a given state. If the members of a pair of accumulated error metrics going into a particular state are equal, we just save that value. The other thing that's affected is the predecessor-successor history we're keeping. For each state, the predecessor that survives is the one with the lower branch metric. If the two accumulated error metrics are equal, some people use a fair coin toss to choose the surviving predecessor state. Others simply pick one of them consistently, i.e. the upper branch or the lower branch. It probably doesn't matter which method you use. The operation of adding the previous accumulated error metrics to the new branch metrics, comparing the results, and selecting the smaller (smallest) accumulated error metric to be retained for the next time instant is called the add-compare-select operation. The figure below shows the results of processing t = 3:

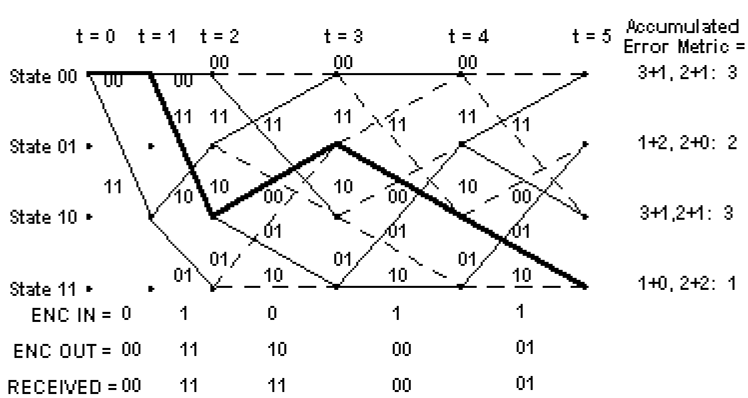


Note that the third channel symbol pair we received had a one-symbol error. The smallest accumulated error metric is a one, and there are two of these.

Let's see what happens now at t = 4. The processing is the same as it was for t = 3. The results are shown in the figure:

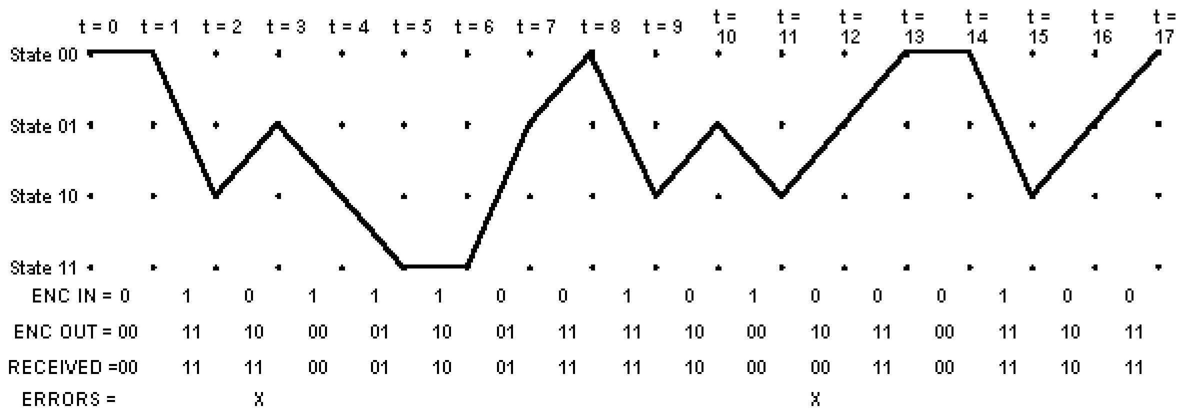


Notice that at t = 4, the path through the trellis of the actual transmitted message, shown in bold, is again associated with the smallest accumulated error metric. Let's look at t = 5:



At t = 5, the path through the trellis corresponding to the actual message, shown in bold, is still associated with the smallest accumulated error metric. This is the thing that the Viterbi decoder exploits to recover the original message.

Perhaps you're getting tired of stepping through the trellis. I know I am. Let's skip to the end. At t = 17, the trellis looks like this, with the clutter of the intermediate state history removed:



The decoding process begins with building the accumulated error metric for some number of received channel symbol pairs, and the history of what states preceded the states at each time instant t with the smallest accumulated error metric. Once this information is built up, the Viterbi decoder is ready to recreate the sequence of bits that were input to the convolutional encoder when the message was encoded for transmission. This is accomplished by the following steps:

First, select the state having the smallest accumulated error metric and save the state number of that state.

Iteratively perform the following step until the beginning of the trellis is reached: Working backward through the state history table, for the selected state, select a new state which is listed in the state history table as being the predecessor to that state. Save the state number of each selected state. This step is called trace back.

Now work forward through the list of selected states saved in the previous steps. Look up what input bit corresponds to a transition from each predecessor state to its successor state. That is the bit that must have been encoded by the convolutional encoder.

The following table shows the accumulated metric for the full 15-bit (plus two flushing bits) example message at each time t:

t =	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
State 00_2		0	2	3	3	3	3	4	1	3	4	3	3	2	2	4	5	2
State 01_2			3	1	2	2	3	1	4	4	1	4	2	3	4	4	2	
State 10_2		2	0	2	1	3	3	4	3	1	4	1	4	3	3	2		
State 11_2			3	1	2	1	1	3	4	4	3	4	2	3	4	4		

It is interesting to note that for this hard-decision-input Viterbi decoder example, the smallest accumulated error metric in the final state indicates how many channel symbol errors occurred.

The following state history table shows the surviving predecessor states for each state at each time t:

t =	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
State 00_2	0	0	0	1	0	1	1	0	1	0	0	1	0	1	0	0	0	1
State	0	0	2	2	3	3	2	3	3	2	2	3	2	3	2	2	2	0

01_2																		
State 10_2	0	0	0	0	1	1	1	0	1	0	0	1	1	0	1	0	0	0
State 11_2	0	0	2	2	3	2	3	2	3	2	2	3	2	3	2	2	0	0

The following table shows the states selected when tracing the path back through the survivor state table shown above:

t =	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
	0	0	2	1	2	3	3	1	0	2	1	2	1	0	0	2	1	0

Using a table that maps state transitions to the inputs that caused them, we can now recreate the original message. Here is what this table looks like for our example rate 1/2 K = 3 convolutional code:

	Input was, Given Next State =			
Current State	$00_2 = 0$	$01_2 = 1$	$10_2 = 2$	$11_2 = 3$

$00_2 = 0$	0	x	1	x
$01_2 = 1$	0	x	1	x
$10_2 = 2$	x	0	x	1
$11_2 = 3$	x	0	x	1

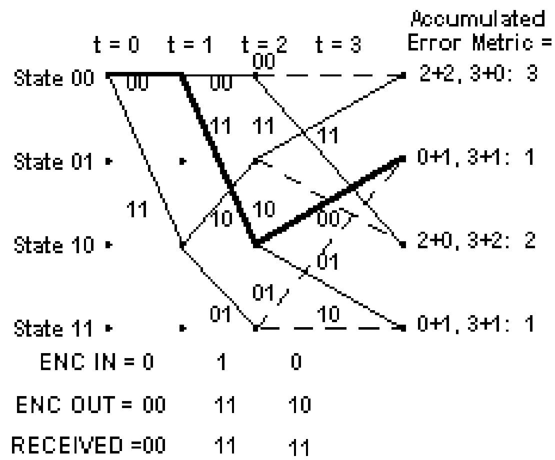
Note: In the above table, x denotes an impossible transition from one state to another state.

So now we have all the tools required to recreate the original message from the message we received:

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

The two flushing bits are discarded.

Here's an insight into how the trace back algorithm eventually finds its way onto the right path even if it started out choosing the wrong initial state. This could happen if more than one state had the smallest accumulated error metric, for example. I'll use the figure for the trellis at t = 3 again to illustrate this point:



See how at t = 3, both States 01_2 and 11_2 had an accumulated error metric of 1. The correct path goes to State 01_2 -notice that the bold line showing the actual message path goes into this state. But suppose we choose State 11_2 to start our trace back. The predecessor state for State 11_2 , which is State 10_2 , is the same as the predecessor state for State 01_2! This is because at t = 2, State 10_2 had the smallest accumulated error metric. So after a false start, we are almost immediately back on the correct path.

2.3.1. Puncturing:

Puncturing is done to achieve the desired code rates according to the pattern described by the standard. Here we reject few bits from the incoming message.

2.4. INTERLEAVER:

Interleaver is used to shuffle the bits without repeating or missing any bit. It is used to avoid burst errors. There are two types of interleavers Block interleavers and Convolutional interleavers. But we are using block interleaver according to the standard

Ncpc=no. of coded bits/sub carrier

Ncbps=no. of coded bits per OFDM symbol

BPSK	QPSK	16-QAM	64-QAM
Ncpc=1	Ncpc=2	Ncpc=4	Ncpc=6
Ncbps=192;	Ncbps=384;	Ncbps=768;	Ncbps=1152;

s=ceil (Ncpc/2);

K =index bit BEFORE the first permutation

Mk = index that bit of the second permutation BEFORE and AFTER the first

Jk = index after the second permutation, just before the mapping of the signal.

```
k = 0: Ncbps-1;
mk = ((Ncbps/12)*mod (k,12))+floor(k/12);
jk = s*floor (mk/s)+mod(mk+Ncbps-floor(12*mk/Ncbps),s);
```

CHAPTER 3:

3.1. Modulation:

Modulation is the process of varying some characteristic of a periodic wave with an external signal. Modulation is utilized to send an information bearing signal over long distances. Radio communication superimposes this information bearing signal onto a carrier signal. These high frequency carrier signals can be transmitted over the air easily and are capable of traveling long distances. The characteristics (amplitude, frequency, or phase) of the carrier signal are varied in accordance with the information bearing signal. In the field of communication engineering, the information bearing signal is also known as the modulating signal. The modulating signal is a slowly varying signal - as opposed to the rapidly varying carrier frequency.

3.1.1. Analog Modulation

In analog modulation, the modulation is applied continuously in response to the analog information signal. The aim of analog modulation is to transfer an analog low pass signal, for example an audio signal or TV signal, over an analog band pass channel, for example a limited radio frequency band or a cable TV network channel. Common analog modulation techniques are:

Angular modulation

Phase modulation (PM)

Frequency modulation (FM)

Amplitude modulation (AM)

3.1.2. Digital Modulation

In digital modulation, an analog carrier signal is modulated by a digital bit stream. This can be described as a form of digital-to-analog conversion. The changes in the carrier signal are chosen from a finite number of alternative symbols. The aim of digital modulation is to transfer a digital bit stream over an analog band pass channel, for example over the public switched telephone network (where a filter limits the frequency range to between 300 and 3400 Hz) or a limited radio frequency band. These are the most fundamental digital modulation techniques: In the case of PSK, a finite number of phases are used. In the case of FSK, a finite number of frequencies are used. In the case of ASK, a finite number of amplitudes are used. In the case of QAM, Here signal is a combination of PSK and ASK, with a finite number of at least two phases, and a finite number of at least two amplitudes.

3.2. FREQUENCY SHIFT KEYING (FSK)

Frequency-shift keying (FSK) is a form of frequency modulation in which the modulating signal shifts the output frequency between predetermined values. Usually, the instantaneous frequency is shifted between two discrete values termed the mark frequency and the space frequency. Continuous phase forms of FSK exist in which there is no phase discontinuity in the modulated signal. The example shown at right is of such a form. For FSK OR *frequency-shift modulation* and *frequency-shift signaling*.

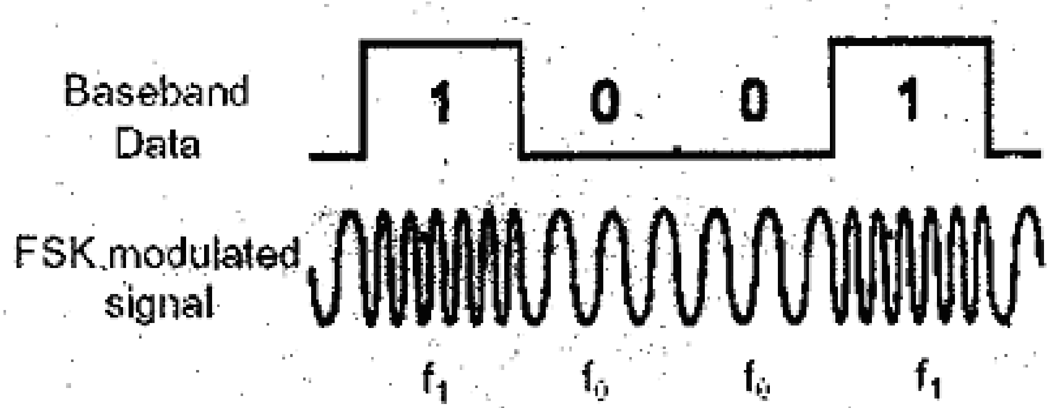


Figure: 3.1 Frequency Shift Keying

3.3. Amplitude-shift keying (ASK)

Amplitude-shift keying (ASK) is a form of modulation that represents digital data as variations in the amplitude of a wave. The amplitude of an analog carrier signal varies in accordance with the bit stream (modulating signal), keeping frequency and phase constant. The level of amplitude can be used to represent binary logic 0s and 1s. We can think of a carrier signal as an ON or OFF switch. In the modulated signal, logic 0 is represented by the absence of a carrier, thus giving OFF/ON keying operation and hence the name given.

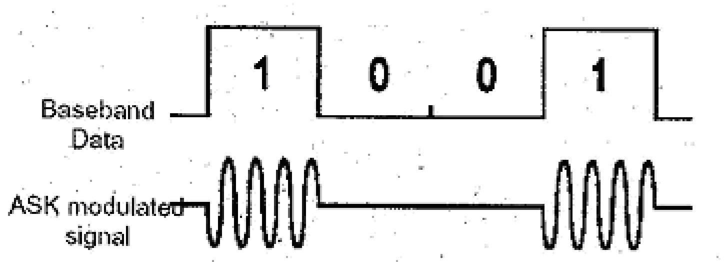


Figure :3.2 Amplitude Shift Keying

3.4. Phase-shift keying (PSK)

Phase-shift keying (PSK) is a digital modulation scheme that conveys data by changing, or modulating, the phase of a reference signal (the carrier wave).Any digital modulation scheme uses a finite number of distinct signals to represent digital data. In the case of PSK, a finite number of phases are used. Each of these phases is assigned a unique pattern of binary bits. Usually, each phase encodes an equal number of bits. Each pattern of bits forms the symbol that is represented by the particular phase. The demodulator, which is designed specifically for the symbol-set used by the modulator, determines the phase of the received signal and maps it back to the symbol it represents, thus recovering the original data.

This requires the receiver to be able to compare the phase of the received signal to a reference signal — such a system is termed coherent.

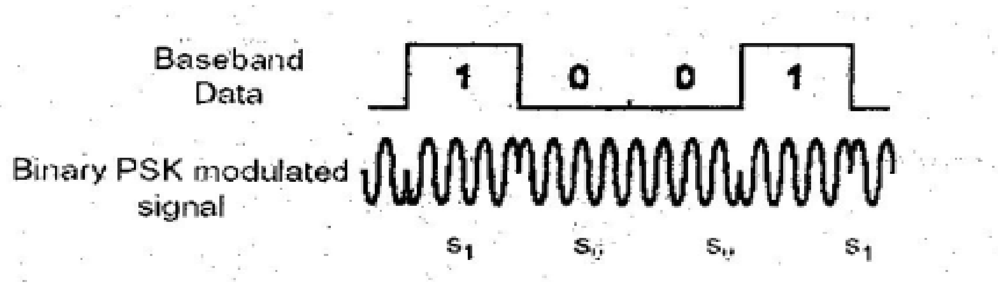


Figure 3.3: Phase Shift Keying

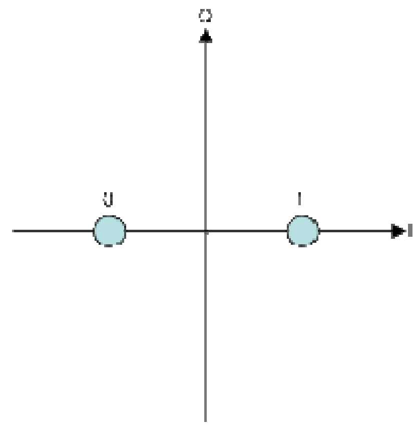
Phase-shift keying (PSK) is a digital modulation scheme that conveys data by changing, or modulating, the phase of a reference signal (the carrier wave).

Any digital modulation scheme uses a finite number of distinct signals to represent digital data. PSK uses a finite number of phases, each assigned a unique pattern of binary bits. Usually, each phase encodes an equal number of bits. Each pattern of bits forms the symbol that is represented by the particular phase

A convenient way to represent PSK schemes is on a constellation diagram. This shows the points in the Argand plane where, in this context, the real and imaginary axes are termed the in-phase and quadrature axes respectively due to their 90° separation. Such a representation on perpendicular axes lends itself to straightforward implementation. The amplitude of each point along the in-phase axis is used to modulate a cosine (or sine) wave and the amplitude along the quadrature axis to modulate a sine (or cosine) wave.

In PSK, the constellation points chosen are usually positioned with uniform angular spacing around a circle. This gives maximum phase-separation between adjacent points and thus the best immunity to corruption. They are positioned on a circle so that they can all be transmitted with the same energy. In this way, the moduli of the complex numbers they represent will be the same and thus so will the amplitudes needed for the cosine and sine waves. Two common examples are "binary phase-shift keying" (BPSK) which uses two phases, and "quadrature phase-shift keying" (QPSK) which uses four phases, although any number of phases may be used. Since the data to be conveyed are usually binary, the PSK scheme is usually designed with the number of **Binary phase-shift keying (BPSK)**

3.5. Binary phase-shift keying (BPSK)



Constellation diagram for BPSK.

BPSK is the simplest form of PSK. It uses two phases which are separated by 180° and so can also be termed 2-PSK. It does not particularly matter exactly where the constellation points are positioned, and in this figure they are shown on the real axis, at 0° and 180°. This modulation is the most robust of all the PSKs since it takes serious distortion to make the demodulator reach an incorrect decision. It is, however, only able to modulate at 1 bit/symbol (as seen in the figure) and so is unsuitable for high data-rate applications when bandwidth is limited.

BPSK has only one bit per symbol, this is also the symbol error rate.

Binary data is often conveyed with the following signals:

$$s_0(t) = \sqrt{\frac{2E_b}{T_b}}\cos(2\pi f_c t + \pi) = -\sqrt{\frac{2E_b}{T_b}}\cos(2\pi f_c t)$$

for binary "0"

$$s_1(t) = \sqrt{\frac{2E_b}{T_b}} \cos(2\pi f_c t)$$

for binary "1"

where f_c is the frequency of the carrier-wave.

E_b = Energy-per-bit

E_s = Energy-per-symbol = kE_b with k bits per symbol

T_b = Bit duration

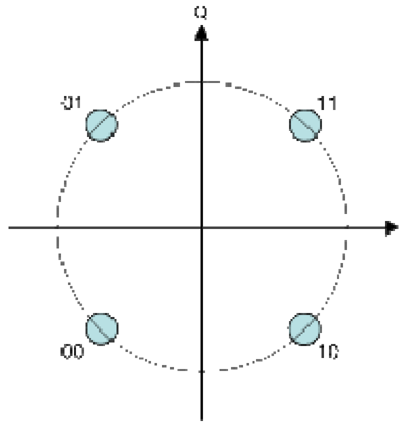
T_s = Symbol duration

Hence, the signal-space can be represented by the single basis function

$$\phi(t) = \sqrt{\frac{2}{T_b}} \cos(2\pi f_c t)$$

Where 1 is represented by $\sqrt{E_b}\phi(t)$ and 0 is represented by $-\sqrt{E_b}\phi(t)$. This assignment is, of course, arbitrary.

3.6. Quadrature phase-shift keying (QPSK)



Constellation diagram for QPSK with Gray coding.

Each adjacent symbol only differs by one bit.

Sometimes known as quaternary or quadriphase PSK or 4-PSK, QPSK uses four points on the constellation diagram, equispaced around a circle. With four phases, QPSK can encode two bits per symbol, shown in the diagram with Gray coding to minimize the BER — twice the rate of BPSK. Analysis shows that this may be used either to double the data rate compared to a BPSK system while maintaining the bandwidth of the signal or to maintain the data-rate of BPSK but halve the bandwidth needed.

Although QPSK can be viewed as a quaternary modulation, it is easier to see it as two independently modulated quadrature carriers. With this interpretation, the even (or odd) bits are used to modulate the in-phase component of the carrier, while the odd (or even) bits are used to modulate the quadrature-phase component of the carrier. BPSK is used on both carriers and they can be independently demodulated.

The implementation of QPSK is more general than that of BPSK and also indicates the implementation of higher-order PSK. Writing the symbols in the constellation diagram in terms of the sine and cosine waves used to transmit them:

$$s_i(t) = \sqrt{\frac{2E_s}{T}} \cos\left(2\pi f_c t + (2i-1)\frac{\pi}{4}\right), \quad i = 1, 2, 3, 4.$$

This yields the four phases $\pi / 4$, $3\pi / 4$, $5\pi / 4$ and $7\pi / 4$ as needed.

This results in a two-dimensional signal space with unit basis functions

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

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

The first basis function is used as the in-phase component of the signal and the second as the quadrature component of the signal.

Hence, the signal constellation consists of the signal-space 4 points

$$\left(\pm\sqrt{E_s/2}, \pm\sqrt{E_s/2}\right).$$

The factors of $1 / 2$ indicate that the total power is split equally between the two carriers.

Comparing these basis functions with that for BPSK shows clearly how QPSK can be viewed as two independent BPSK signals. Note that the signal-space points for BPSK do not need to split the symbol (bit) energy over the two carriers in the scheme shown in the BPSK constellation diagram.

QPSK systems can be implemented in a number of ways. An illustration of the major components of the transmitter and receiver structure are shown below.

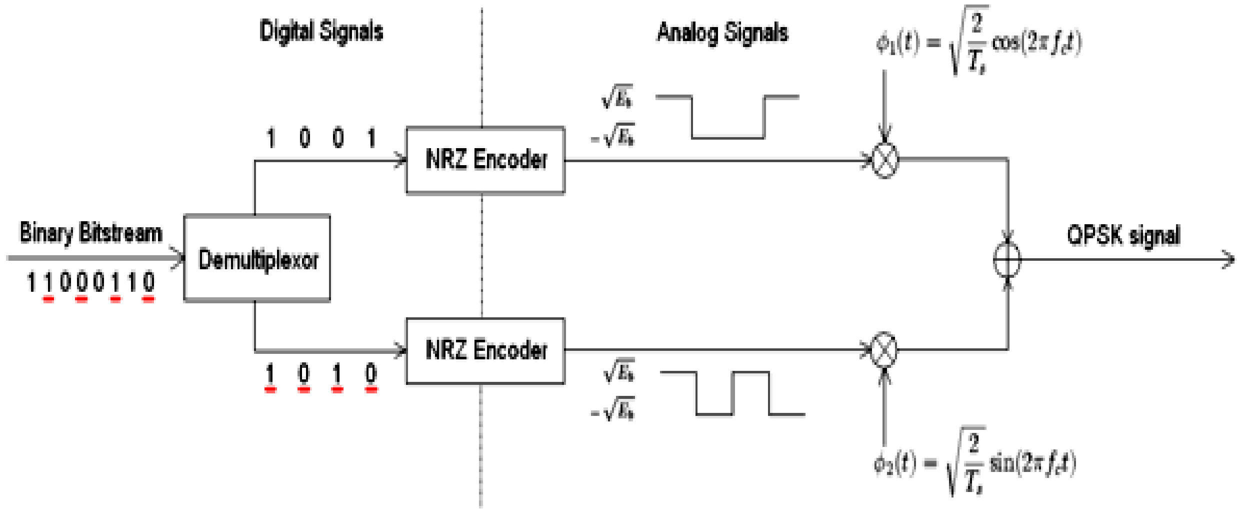


Fig: 3.5 Conceptual transmitter structure for QPSK.

The binary data stream is split into the in-phase and quadrature-phase components. These are then separately modulated onto two orthogonal basis functions. In this implementation, two sinusoids are used. Afterwards, the two signals are superimposed, and the resulting signal is the QPSK signal.

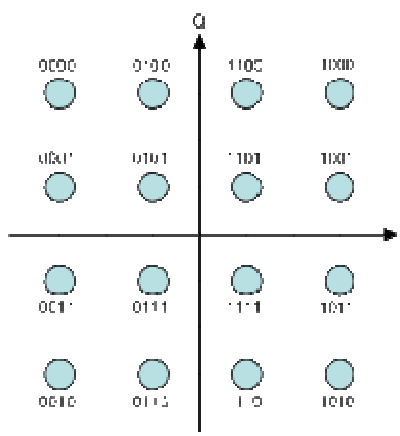
3.7. Quadrature amplitude modulation:

As with many digital modulation schemes, the constellation diagram is a useful representation. In QAM, the constellation points are usually arranged in a square grid with equal vertical and horizontal spacing, although other configurations are possible (e.g. Cross-QAM). Since in digital telecommunications the data is usually binary, the number of points in the grid is usually a power of 2 (2, 4, 8 …). Since QAM is usually square, some of these are rare—the most common forms are 16-QAM, 64-QAM, 128-QAM and 256-QAM. By moving to a higher-order constellation, it is possible to transmit more bits per symbol. However, if the mean energy of the constellation is to remain the same (by way of making a fair comparison), the points must be closer together and are thus more susceptible to noise and other corruption; this results in a higher bit error rate and so higher-order QAM can deliver more data less reliably than lower-order QAM, for constant mean constellation energy.

If data-rates beyond those offered by 8-PSK are required, it is more usual to move to QAM since it achieves a greater distance between adjacent points in the I-Q plane by distributing the points more evenly. The complicating factor is that the points are no longer all the same amplitude and so the demodulator must now correctly detect both phase and amplitude, rather than just phase.

64-QAM and 256-QAM are often used in digital cable television and cable modem applications. In the US, 64-QAM and 256-QAM are the mandated modulation schemes for digital cable (see QAM tuner) as standardized by the SCTE in the standard ANSI/SCTE 07 2000. Note that many marketing people will refer to these as QAM-64 and QAM-256. In the UK, 16-QAM and 64-QAM are currently used for digital terrestrial television

3.7.1. Rectangular QAM



Constellation diagram for rectangular 16-QAM.

Rectangular QAM constellations are, in general, sub-optimal in the sense that they do not maximally space the constellation points for a given energy. However, they have the considerable advantage that they may be easily transmitted as two pulse amplitude modulation

(PAM) signals on quadrature carriers, and can be easily demodulated. The non-square constellations, dealt with below, achieve marginally better bit-error rate (BER) but are harder to modulate and demodulate.

The first rectangular QAM constellation usually encountered is 16-QAM, the constellation diagram for which is shown here. A Gray coded bit-assignment is also given. The reason that 16-QAM is usually the first is that a brief consideration reveals that 2-QAM and 4-QAM are in fact binary phase-shift keying (BPSK) and quadrature phase-shift keying (QPSK), respectively. Also, the error-rate performance of 8-QAM is close to that of 16.

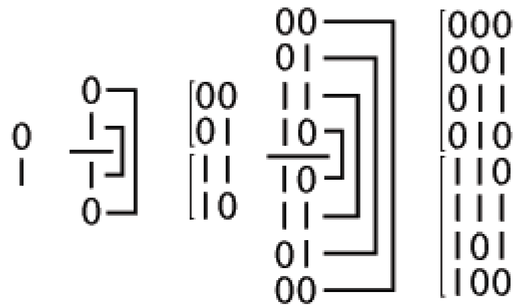
3.8 Gray code

The reflected binary code, also known as Gray code after Frank Gray, is a binary numeral system where two successive values differ in only one digit.

Gray codes play an important role in error correction. For example, in a digital modulation scheme such as QAM where data is typically transmitted in symbols of 4 bits or more, the signal's constellation diagram is arranged so that the bit patterns conveyed by adjacent constellation points differ by only one bit. By combining this with forward error correction capable of correcting single-bit errors, it is possible for a receiver to correct any transmission errors that cause a constellation point to deviate into the area of an adjacent point. This makes the transmission system less susceptible to noise.

3.8.1 Constructing an n-bit gray code:

The binary-reflected Gray code for *n* bits can be generated by reflecting the bits (i.e. listing them in reverse order and concatenating the reverse list onto the original list), prefixing the original bits with a binary 0 and then prefixing the reflected bits with a binary 1. The base case, for *n=1* bit, is the most basic Gray code, *G = {0, 1}*. The first few steps of the above-mentioned reflect-and-prefix method are:



These characteristics suggest a simple and fast method of translating a binary value into the corresponding Gray Code.

CHAPTER 4:

4 ORTHOGONAL FREQUENCY DIVISION MULTIPLEXING (OFDM)

OFDM is a special case of multicarrier transmission, where a single data stream is transmitted over a number of lower rate subcarriers. OFDM can be seen as either a modulation technique or a multiplexing technique. One of the main reasons to use OFDM is to increase the robustness against frequency selective fading or narrowband interference. In a single carrier system, a single fade or interferer can cause the entire link to fail, but in a multicarrier system, only a small percentage of the subcarriers will be affected

4.1 Advantages of OFDM.

The OFDM transmission scheme has the following key advantages:

4.1.1 Robust against multipath

OFDM is an efficient way to deal with multipath; for a given delay spread, the implementation complexity is significantly lower than that of a single carrier system with an equalizer.

4.1.2 Maximizes throughput

In relatively slow time-varying channels, it is possible to significantly enhance the capacity by adapting the data rate per subcarrier according to the signal-to- noise ratio of that particular subcarrier.

4.1.3 Robust against Narrowband Interference.

OFDM is robust against narrowband interference, because such interference affects only a small percentage of the subcarriers.

1.4.4 Establishes Single-frequency networks

OFDM makes single-frequency networks possible, which is especially attractive for broadcasting applications.

4.1.5 Frequency Diversity.

OFDM is the best place to employ Frequency Diversity. In fact, in a combination of OFDM and CDMA called the MC-CDMA transmission technique, frequency diversity is inherently present in the system. (i.e., it is available for free). Even though, OFDM provides a lot advantages for Wireless Transmission, it has a few serious disadvantages that must be overcome for this technology to become a success.

4.2. Disadvantages

On the other hand, OFDM also has some drawbacks compared with single- carrier modulation:

4.2.1. The Peak Power Problem in OFDM

One of the most serious problems with OFDM transmission is that, it exhibits a high peak-to-average ratio. In other words, there is a problem of extreme amplitude excursions of the transmitted signal. The OFDM signal is basically a sum of N complex random variables, each of which can be considered as a complex modulated signal at different frequencies. In some cases, all the signal components can add up in phase and produce a large output and in some cases, they may cancel each other producing zero output. Thus the peak-to-average ratio (PAR) of the OFDM system is very large. The problem of Peak-To-Average Ratio is more serious in the transmitter. In order to avoid clipping of the transmitted waveform, the power-amplifier at the transmitter front end must have a wide linear range to include the peaks in the transmitted waveform. Building power amplifiers with such wide linear ranges is a costly affair. Further, this also results in high power consumption. There has been a lot of research put into the study of overcoming the PAR problem in

4.2.2. Frequency and Phase offsets

OFDM is more sensitive to frequency offset and phase noise. This is a significant problem in the sense that, frequency and phase offsets in OFDM cause ISI (Inter-symbol interference) and ICI (inter-carrier interference). ISI and ICI reduces the orthogonally property causing the carriers to interfere

4.3. PRINCIPLES OF OFDM

OFDM is a multi-carrier modulation scheme where the carriers are orthogonal to each other. A single stream of digital data is split into several parallel streams of low data rate, each of the parallel stream then rides a carrier frequency within the original bandwidth such that all the carriers remain orthogonal to each other. The carriers are multiplexed to form a single OFDM carrier. It is also known as *Multi-carrier modulation* (MCM) or *Discrete Multi Tone modulation* (DMT).

4.3.1. The importance of orthogonality

The main concept of OFDM is the orthogonality of sub-carriers. Since the carriers are all sine/cosine waves, we know that the area under one period of a sine or a cosine is zero, as shown in fig-4.1

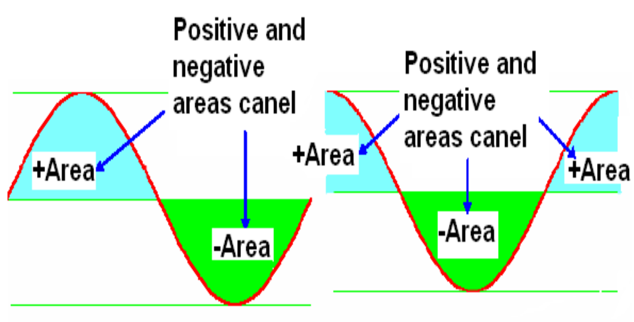


Fig-4.1 The area under a sine and a cosine wave over one period is always zero.

If we take a sine wave of frequency m and multiply it by a sinusoid (sine or cosine) of a frequency n, where both m and n are integers. The integral or the area under this product is given by:

$$f(t) = \sin m\omega t \quad \sin n\omega t \qquad (4.1)$$

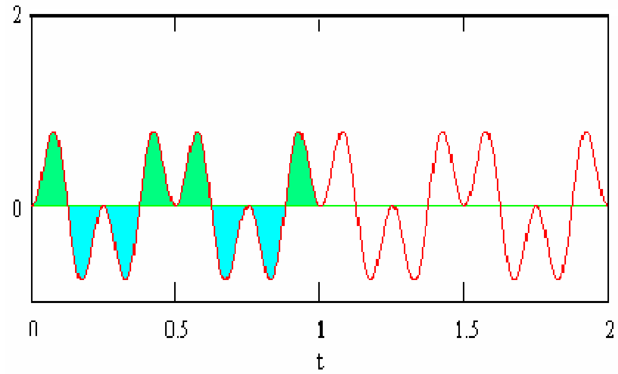


Fig-4.2 Shows the signal f (t). The area under a sine wave multiplied by its own harmonic is always zero.

By the simple trigonometric relationship, this is equal to a sum of two sinusoids of frequencies (n-m) and (n+m), so f(t) becomes

$$f(t) = \frac{1}{2}\cos(m-n)\omega t - \frac{1}{2}\cos(m+n)\omega t \qquad (4.2)$$

These two components are each a sinusoid, so the integral is equal to zero over one period.

$$\int_0^{2\pi} f(t)\,dt = \int_0^{2\pi} \frac{1}{2}\cos(m-n)\omega t\,dt - \int_0^{2\pi}\frac{1}{2}\cos(m+n)\omega t\,dt \qquad (4.3)$$

$$= 0 - 0$$

We conclude that when we multiply a sinusoid of a frequency n by a sinusoid of frequency m/n, the area under the product is zero. In general for all integers *n* and *m*, sin *mx*, sin *nx,* cos *nx and* cos *mx* are all orthogonal to each other. These frequencies are called harmonics. This is a key idea in the concept of OFDM.

The orthogonality allows simultaneous transmission on a lot of sub-carriers in a tight frequency space without interference from each other.

4.3.2. OFDM a special case of FDM

OFDM is a special case of Frequency Division Multiplexing (FDM). As an analogy a FDM channel is like water flow out of a faucet, in contrast OFDM signal is like a shower. In a faucet all of the water comes out as one big stream, putting thumb over the faucet hole would disrupt the complete water flow. But in case of shower putting thumb on it would not disrupt the entire flow. The response of OFDM to interference in comparison with FDM is also similar.

Fig-4.3 a) A regular FDM signal with all data on a single big data stream. b) An OFDM signal with multiple parallel bit streams carrying same amount of data.

In FDM if we have a bandwidth that goes from a to b, it can be divided into equal spaces, in fig.4.3 it is divided into four equal channel spaces. Fig.4.4 shows frequency modulated carriers in frequency domain. The frequencies a and b could have any value, moreover the carrier frequencies do not have any specific relationship with each other.

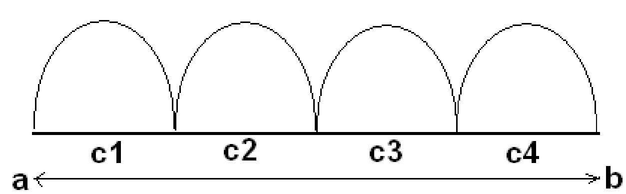


Fig-4.4 FDM carriers placed next to each other.

But if frequency *c1* and *cn* were such that for any integer n the following relationship holds

$$Cn = n \times C1 \qquad (4.4)$$

So that

$$C2 = 2\ C1$$

$$C3 = 3\ C1$$

$$C4 = 4\ C1$$

Then all of these frequencies are harmonics of c1. In this case, since these carriers are orthogonal to each other, when added together they do not interfere with each other. In FDM the carrier frequencies do not follow the above relationship so they get interference from neighboring carriers. To prevent adjacent channel interference the signals are moved farther apart.

4.4. OFDM Model

The binary data stream to be transmitted is first forward error correction (FEC) encoded. Its purpose is to detect and correct any error in the data at the receiver end, caused by channel impairment. OFDM generally employs convolutional encoding for FEC. The coded version of OFDM is also called Coded OFDM (COFDM) The data stream is then demultiplexed into N parallel data streams. Each of the parallel data stream has a bit rate which is 1/N of the bit rate of the initial data stream. Each of the parallel data stream modulates a sub-carrier using QPSK, QAM or any other digital modulation scheme. All the (N) sub-carriers are orthogonal to each other as discussed above. In OFDM, sinc -shaped pulses are used as subcarrier spectra. Zero crossings of sinc -pulses, are located at the multiples of 1/T (T is the symbol interval). Subcarrier orthogonality is maintained by selecting these frequencies according to the equation (5).

$$f_i = i / T \qquad i = 1, 2, \ldots N \qquad (5)$$

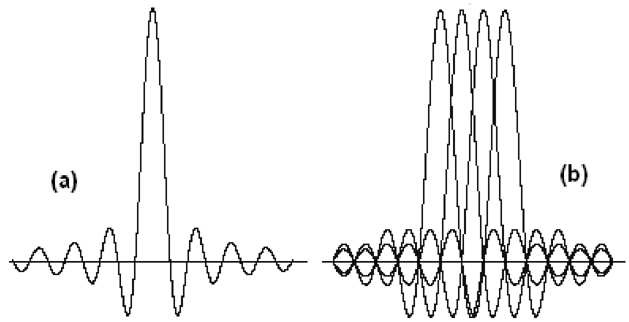


Fig-4.5(a) Single sub-carrier spectra (b) Orthogonal sub-carriers spectra

The modulated orthogonal sub-carriers are combined together to form a composite OFDM signal. This is done by taking Inverse Discrete Fourier Transform (IDFT) of the samples. The IDFT transforms a set of samples in the frequency domain into an equivalent set of samples in the time domain. The DFT performs the reverse operation at the receiver end. This transform pair could be expressed mathematically as

$$\text{DFT:} \quad b_n = \sum_{k=0}^{N-1} B_k \exp(-j\,2\pi kn/N) \qquad n = 0,1.....,N-1$$

(6)

$$\text{IDFT:} \quad B_k = 1/N \sum_{n=0}^{N-1} b_n \exp(j\,2\pi kn/N) \qquad k = 0,1.....,N-1$$

where the sequences b_n and B_k are the frequency domain and time domain samples respectively. In practical implementation the above process is performed by an efficient (digital signal processing) computational tool called the *Fast Fourier Transform / Inverse Fast Fourier Transform* (FFT/IFFT) pair. For FFT/IFFT implementation N should be a power of 2. Direct computation of N point DFT requires $N*N$ iterations, whereas a typical FFT/IFFT algo can do the same calculation in (3/2 N) $log\ N$ iterations (Cooley and Tukey algorithim). The basic strategy that is used in the FFT algorithm is one of "divide and conquer," which involves decomposing an N-point DFT into successively smaller DFT's.

Forward FFT takes a random signal, multiplies it successively by complex exponentials over the range of frequencies, sums each product and plots the results as a coefficient of that frequency. The coefficients are called a spectrum and represent how much of that frequency is present in the composite signal. IFFT quickly computes the time domain signal instead of having to do it one carrier at a time and then adding. In OFDM the input of the IFFT is also a time domain signal instead of a frequency domain signal. But IFFT being a mathematical concept treats the incoming block of bits as a spectrum, and then produces the correct time domain result. The OFDM signal can be represented mathematically as Equation (7) which is basically an equation for IFFT.

$$s(k) = \sum_{n=0}^{N-1} b_n \exp(j\, 2\pi kn/N) \qquad n = 0,1..,N-1 \qquad (4.7)$$

The output of the IFFT consists of, N time samples of a complex envelope for each period T, which are parallel to serial converted and then finally digital to analog converted to facilitate transmission of the incoming data stream over the wireless channel. The received signal goes through similar steps in the reverse order as shown in fig (6 b).

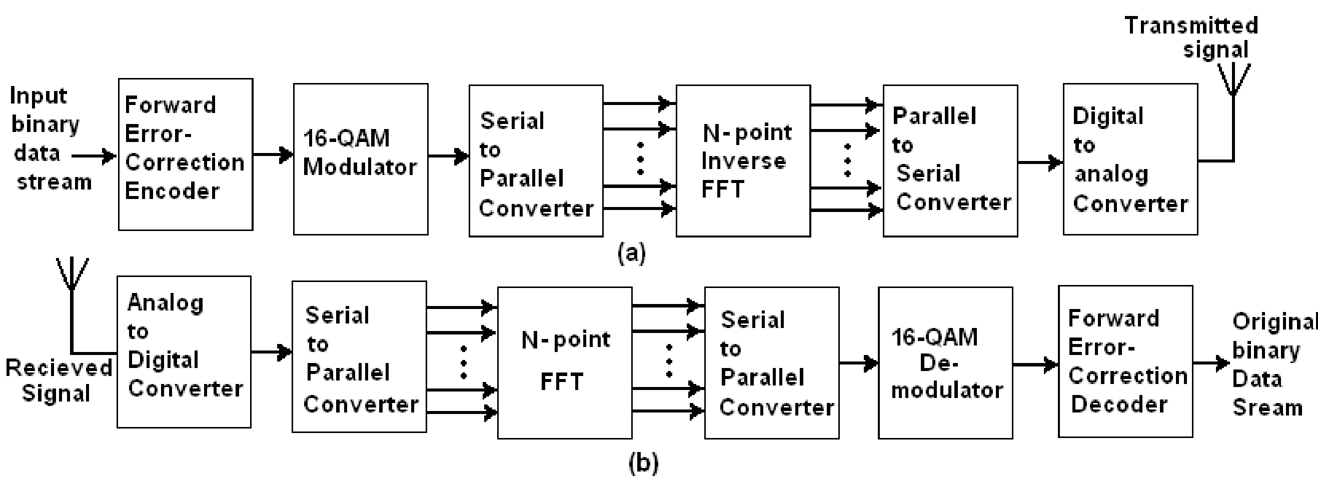
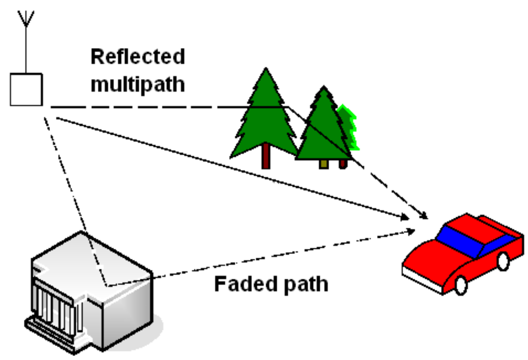


Fig4.6 Block diagram of (a) OFDM transmitter (b) OFDM receiver.

4.5. Fading tolerance in OFDM

If the path from the transmitter to the receiver either has reflections or obstructions, we can get fading effects. In this case the signal reaches the receiver from many different routes, each a copy of the original signal but with different delay and slightly different gain. The timedelays result in phase shifts which added to main signal causes the signal to degrade. Fig -4.7 Shows multipath environment



$$h(t) = \sum_{k=0}^{k-1} a_k \, \delta(t - \tau_k)$$

a_k = complex path gain

τ_0 = Normal path delay with reference to Line of sight

$\Delta k = \tau_k - \tau_0$ difference in path time

In fading, the reflected signals that are delayed add to the main signal and cause either gains in the signal strength or deep fades. The deep fades almost wipe out the signal where they occur. The signal level gets so small that the receiver cannot decide what was there.

Fig 4.7 shows the fading response of a channel, dotted lines show the spectrum of a single carrier frequency with wide frequency band. We note that at some frequencies in the band the channel does not allow any information to pass, so they are called deep fade frequencies. This type of channel frequency response is called *frequency selective fading* because it does not occur uniformly across the band, but occurs at selected frequencies only. These frequencies are selected by environment, if there are moving factors in the environment like vehicles etc, then these frequencies may vary, and the receiver must have some way of dealing with it. On the other hand if fading occurs uniformly across the entire band then it is called *Rayleigh fading*.

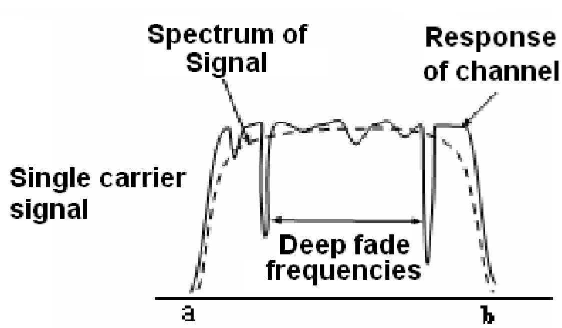


Fig-4.8 Shows the spectrum of a single carrier affected by deep fade frequencies of the channel

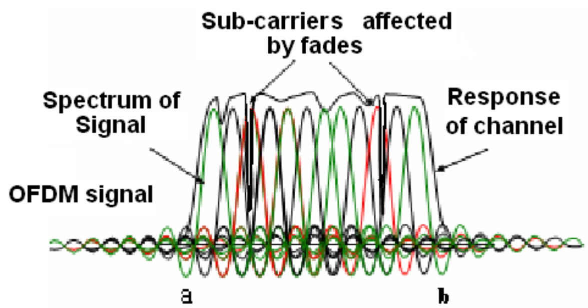


Fig-4.9 Shows OFDM signal in which a subset of carrier frequencies are affected by deep fades of the channel

An OFDM signal offers an advantage in a channel that has a frequency selective fading response. As fig-4.8 shows that when OFDM signal is subjected to frequency selective response of a channel, only two sub-carriers are affected. Instead of whole symbol being knocked out, only a subset of (1/N) bits is lost. With proper coding even these could be recovered.

The bit error rate (BER) performance of an OFDM signal in a fading channel is much better than the performance of QPSK/FDM which is a single carrier wideband signal. The advantage here is coming from the diversity of the multi-carrier such that the fading applies only to a small subset of carriers. However the response of OFDM is similar to other modulation techniques (PSK, FDM etc) in Gaussian channel.

4.6. Delay spread and the use of cyclic prefix in OFDM

In a multi-path environment, copies of the original signal reach the receiver with different time delays. The maximum time delay that occurs is called *delay spread* of the signal. This delay spread can be shorter or larger than the symbol duration. Both types cause different types of signal degradations.

This phenomenon could be explained with the analogy of two cars driving in rain one after the other. The front car splashes water on the rear car. What would the rear car do to avoid the splash. It would increase the distance from the front car to get out of the range of the water splash. If we equate the reach of splash to the delay spread of a splashed signal then we have a better picture of the way to avoid it.

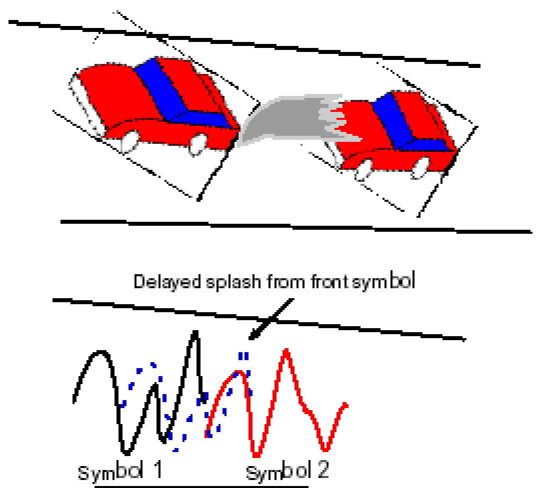


Fig-4.10 Inter symbol interference caused by the splash of the front symbol

Fig-10 shows the symbol and its splash. The reach of the splash is same as delay spread of the signal. In composite, these splashes become noise and affect the beginning of the next symbol.

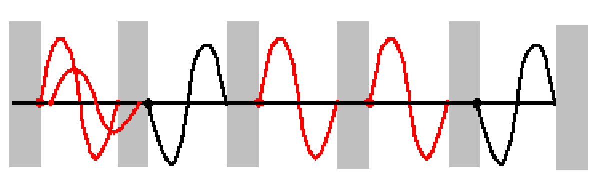


Fig-4.11 The signal is moved out of the grey region so that the front symbol does not interfere with it.

To eliminate this noise at the front of the symbol, we will move our symbol further away from the region of delay spread as shown in fig-12,a small blank space is added between the symbols to avoid the delay spread of the front symbol, this blank space is called *guard interval*. But practically these blank spaces cannot be added between the symbols, because the hardware tends to generate signals continuously. So something has to fill in this gap. For this purpose we slide the symbol to start at the edge of the delay spread time and then fill the guard space with a copy of the tail end of the symbol. In reality all we are doing is adjusting the starting phase and making the symbol period longer by 1.25 times. This is same as attaching the tail end of the symbol at its front.

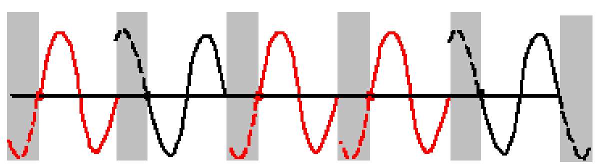


Fig-4.12 The tail of the symbol is added at the front of the symbol to fill the guard interval.

This procedure is called adding a cyclic prefix. Since OFDM, has a lot of carriers, we should do this to each and every carrier. But since OFDM signal is a linear combination therefore in reality we add cyclic prefix just once to the composite OFDM signal. The prefix is any where between10% to 25% of the symbol time.

The prefix is added once to the composite signal after doing the IFFT operation. At the receiver, firstly prefix is removed to get back the periodic signal, and then FFT operation is done on the composite signal. However, the addition of cyclic prefix which mitigates the effects of link fading and inter symbol interference, increases the bandwidth.

4.7. Inter-carrier interference

Moving, transmitter or receiver cause Doppler shift. In OFDM, Doppler-shift causes sub-carriers to shift on adjacent sub-carrier. This phenomenon is called *inter carrier interference* (ICI). ICI is 'crosstalk' between different sub-carriers, which means that they are no longer orthogonal. E.g. scatterers moving 120 km/h causes 250 Hz Doppler-shift. As sub-channel bandwidths are 312,5 kHz, Doppler-shift has no significant meaning. Conversion from fast serial data stream into N slower parallel data streams enables possibility to use longer symbol periods. Longer transmission times allow more delay spread than shorter symbol durations. This property makes OFDM suitable for difficult multi-path environments, because longer symbol times make OFDM robust against Inter Symbol Interference (ISI).

4.8. Non ideal Effects in OFDM system

OFDM system is adversely affected by offsets in phase and frequency. If the local oscillator (LO) frequency gets mismatched between receiver and transmitter it causes loss of orthogonality of the sub-carriers. This results in loss of information at the receiver end. It is also possible to have an LO phase offset, separate from an LO frequency offset. The two offsets can occur in conjunction or one or the other can be present by itself. As the name suggests, an LO phase offset occurs when there is a difference between the phase of the LO output and the phase of the received signal. Often pilot tones are served in the sub-carrier space. These are used to lock on phase and to equalize the channel.

Another important problem inherent with OFDM systems is of high *peak to average power ratio (PAPR).* If all N signals in OFDM signal combine at their maximum point at a particular instant then the PAPR would be very high for the system to handle. The large amplitude variations increase the bit error rate when the signal has to go through amplifier non-linearities. Large back off is required in this case. This makes OFDM problematic in satellite links or for mobile systems where power is a crucial factor. Several methods are used to tackle high PAPR, one of the methods is called *clipping*. The high amplitude of the signal is clipped at certain desired power level. This reduces PAPR but introduces other distortions.

CHAPTER 5

5.1 Project Details:

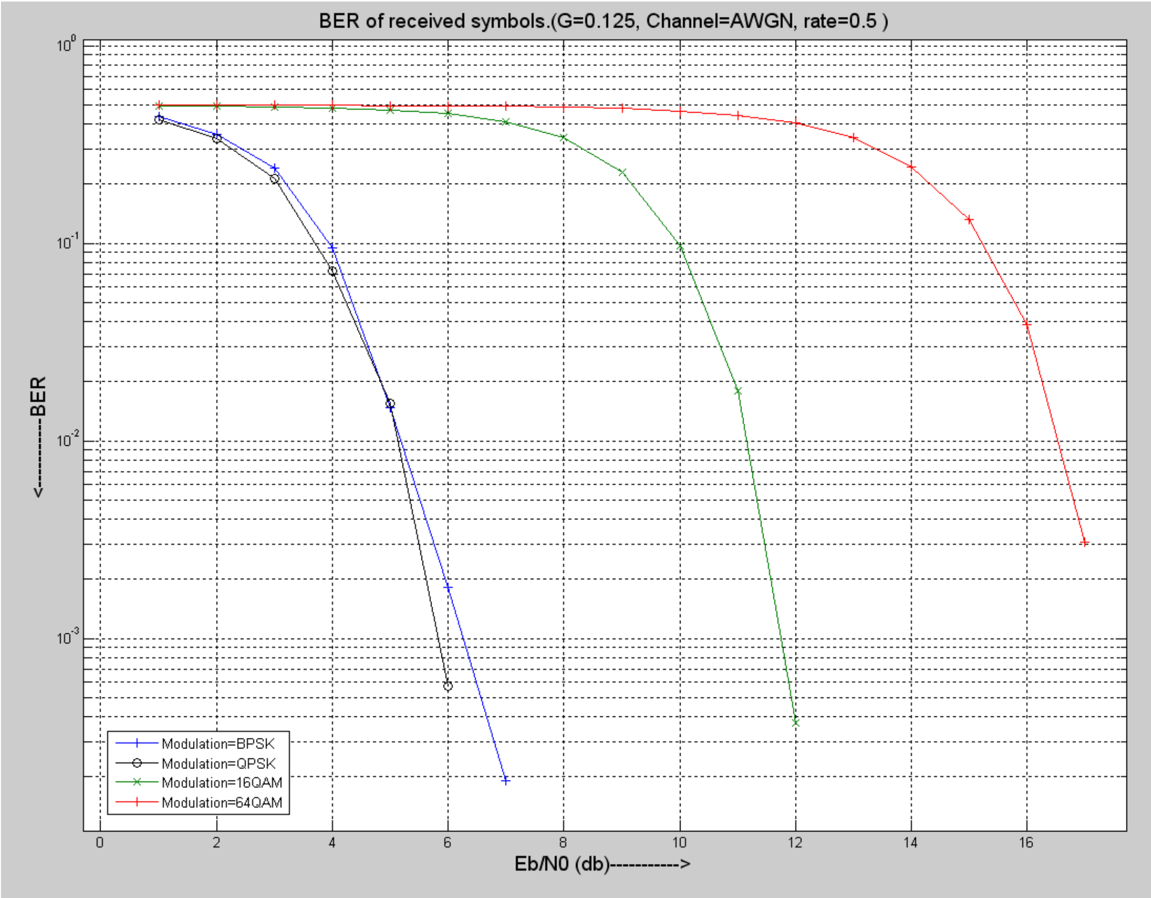
We have implemented the Physical layer of WiMAX 802.16d standard which is also termed as fixed WiMAX. We have computed the BER for the following cases:

Constant coding rate, constant modulation scheme and varying cyclic prefix.

Constant coding rate, constant cyclic prefix and varying modulation scheme.

Constant modulation scheme, constant cyclic prefix and varying coding rate.

5.2 Simulation results:

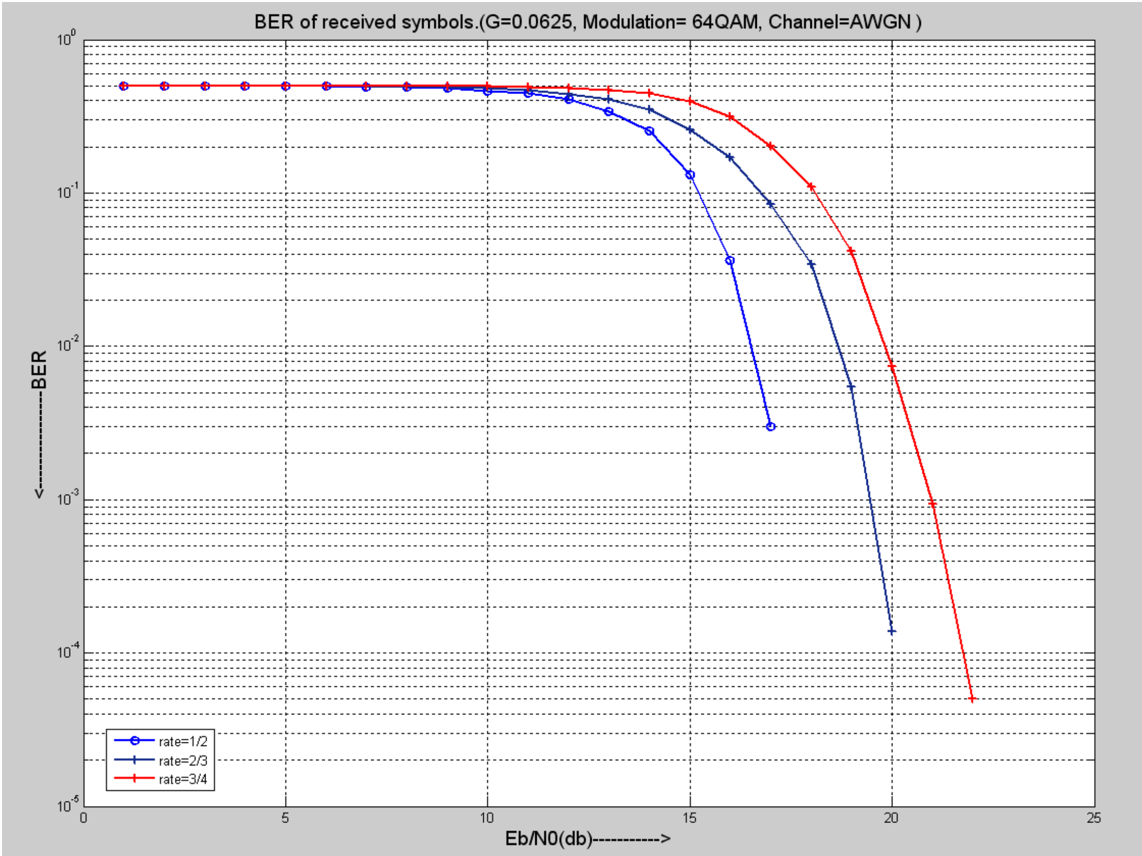


G= .125

Rate: ½

Channel :AWGN

In the above graph we have plotted curves of different modulation schemes which show clearly that lower the modulation scheme better the BER performance. If the channel quality is good then WiMAX adopts a higher modulation scheme which provides a higher data rate but on the other hand more transmitter output power is required because of the higher signal to noise ratio.

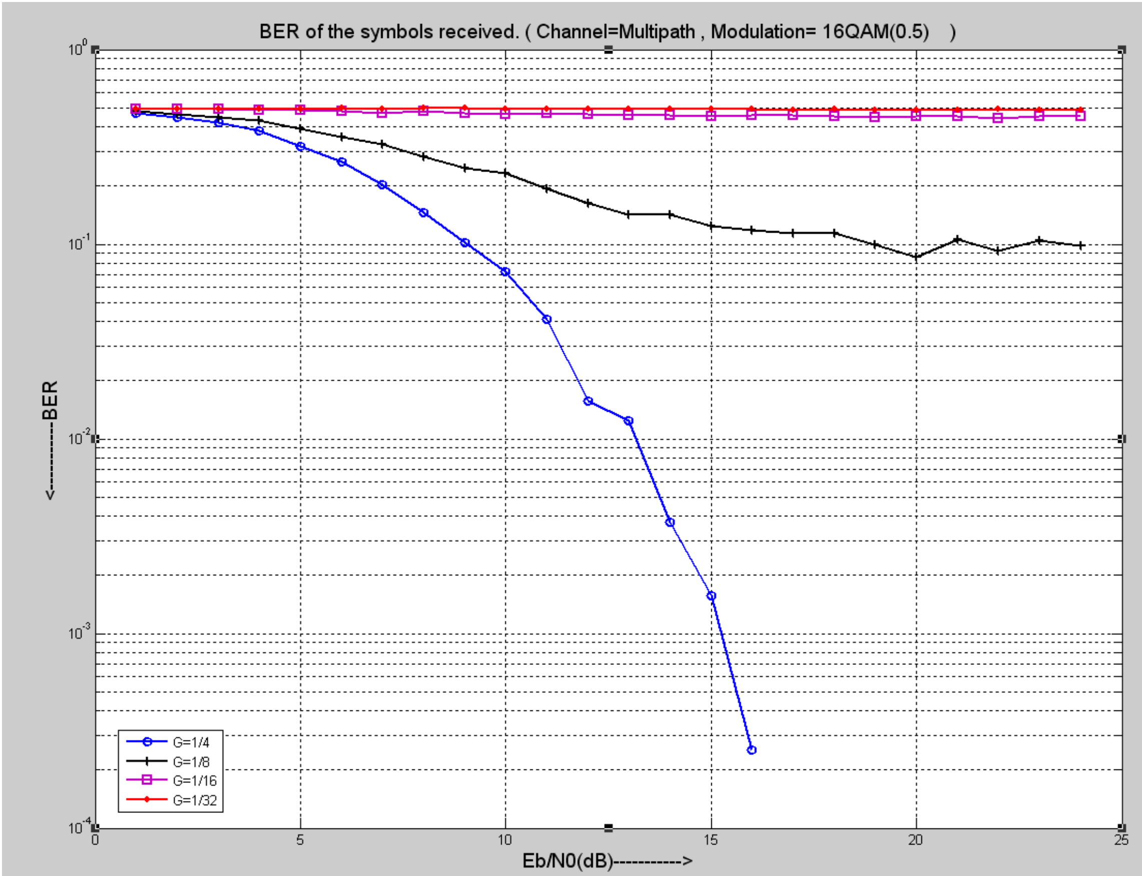


G= .0625

Modulation: 64 QAM

Channel: AWGN

In the above graph we have plotted curves of different coding rates which indicates that lower the code rate better will be BER performance because there are more redundancy bits but on the other hand we lose data rate because effective data rate is less as compared to high code rates.

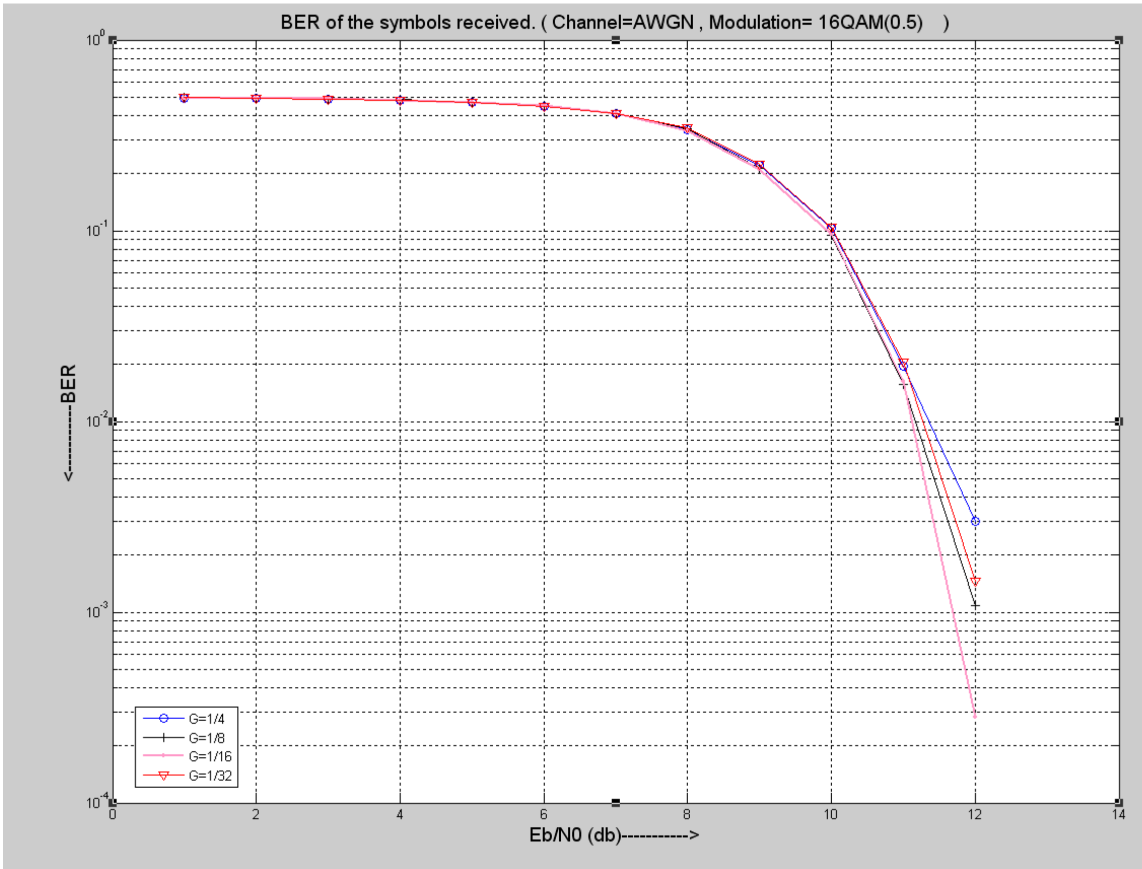


Modulation: 16 QAM

Channel: multipath

Rate ½

In the above graph we have plotted curves of different values of cyclic prefix ‘G’. The graph clearly indicates that as the value of ‘G’ decreases the BER also decreases because when the value of G becomes less than delay spread the BER increases.



Modulation: 16 QAM

Channel: AWGN

Rate ½

In the above plot it is clearly shown that the value of the cyclic prefix does not have any effect on the AWGN channel because it caters the effect of multi path channel.

References:

- IEEE Standard 802.16d
- Modern Digital and Analog Communication by BP Lathi,3rd edition
- Prentice.Hall.Fundamentals.of.WiMAX.Feb.2007 by Jeffrey G. Andrews, Arunaba Ghosh, Rias Muhamed.
- WIRELESS COMMUNICATIONS by Andrea Goldsmith, Stanford University
- Commiunication System by Simon Haykin
- Simulation and Software radio for Mobile Communication by Hiroshi Harada, Ramjee Prasad.
- Digital Communication by SKLAR
- OFDM for Wireless Multimedia Communications by Richard Van Nee, Ramjee Prasad.
- Orthogonal Frequency Division Multiplexing (OFDM) for Wireless Communications by Jian Sun, Wireless Communication Research Laboratory Lane Dept. of Comp. Sci. and Elec. Engr. West Virginia University
- Convolutional Coding and Viterbi Algorithm by Er Liu, Communication Laboratory,Helsinki University of Technology.*S-72.333 Postgraduate Seminar on Radio Communications*
- Proakis, John G. (1995). *Digital Communications*. Singapore: McGraw Hill.
- Couch, Leon W. III (1997). *Digital and Analog Communications*. Upper Saddle River, NJ: Prentice-Hall