o Channel Coding Methods:

Channel coding works to add binaries (Additional data), which known as redundancy binaries, to make the data transfer more immune especially in the channels of communication with noise. For example, mobile phones use powerful encoding techniques, in order to correct the noise and the fading in high-frequency channels. Channel coding is a successful way to reduce the rate of information through the channel and increase its reliability, where the channel encoding is used to protect data sent over the channel even with the existence of noise (mistakes), in which data can be stored or retrieved correctly. One of the most common ways is Hamming Coding.

Hamming Coding Method

Hamming code is a set of error-correction codes that can be used to detect and correct bit errors that can occur when computer data is moved or stored. Hamming code is named for R. W. Hamming of Bell Labs.

Like other error-correction code, Hamming code makes use of the concept of parity and parity bits, which are bits that are added to data so that the validity of the data can be checked when it is read or after it has been received in a data transmission. Using more than one parity bit, an error-correction code can not only identify a single bit error in the data unit, but also its location in the data unit.

In data transmission, the ability of a receiving station to correct errors in the received data is called forward error correction (FEC) and can increase throughput on a data link when there is a lot of noise present. To enable this, a transmitting station must add extra data (called *error correction bits*) to the transmission. However, the correction may not always represent a cost saving over that of simply resending the information. Hamming codes make FEC less expensive to implement through the use of a *block parity* mechanism.

Note1: Correct in Even, that means we are going to count number of "1s" in the received code, if it's Even that means we got a correct code else that means we got a wrong code.

Note2: Correct in Odd, that means we are going to count number of "1s" in the received code, if it's Odd that means we got a correct code else that means we got a wrong code.

Hamming Code method can be summarized by using the following algorithms:

Algorithm 1:

First : From The Side Of Transmitter Station.

- Mark all bit positions that are of (2^N, where n >= 0) as parity bits. (Positions 1, 2, 4, 8, 16, 32, 64, etc.)
- All other bit positions are for the data to be encoded. (Positions 3, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, 17, etc.

- Each parity bit calculates the parity for some of the bits in the code word. The position of the parity bit determines the sequence of bits that it alternately checks and skips.

Position 1: check 1 bit, skip 1 bit, check 1 bit, skip 1 bit, etc. (1,3,5,7,9,11,13,...)
Position 2: check 2 bits, skip 2 bits, check 2 bits, skip 2 bits, etc.
(2,3,6,7,10,11,14,15,...)
Position 4: check 4 bits, skip 4 bits, check 4 bits, skip 4 bits, etc.
(4,5,6,7,12,13,14,15,20,21,22,23,...)
Position 8: check 8 bits, skip 8 bits, check 8 bits, skip 8 bits, etc.
(8-15,24-31,40-47,...)
Position 16: check 16 bits, skip 16 bits, check 16 bits, skip 16 bits, etc.
(16-31,48-63,80-95,...)
Position 32: check 32 bits, skip 32 bits, check 32 bits, skip 32 bits, etc.
(32-63,96-127,160-191,...) etc.

- Set a parity bit to **1** if the total number of ones in the positions it checks is **odd**. Set a parity bit to **0** if the total number of ones in the positions it checks is **even**.

***Example*:** An 8 bit byte with binary value 10101111 is to be encoded using an even-parity Hamming code. What is the binary value after encoding?

	1	2	3	4	5	6	7	8	9	10	11	12
	-	-	1	-	0	1	0	-	1	1	1	1
P1	-		1		0		0		1		1	
	1	-	1	-	0	1	0	-	1	1	1	1
P2		-	1			1	0			1	1	
	1	**0**	1	-	0	1	0	-	1	1	1	1
P4				-	0	1	0					1
	1	**0**	1	**0**	0	1	0	-	1	1	1	1
P8								-	1	1	1	1
	1	**0**	1	**0**	0	1	0	**0**	1	1	1	1

So the binary value after encoding is: 1 0 1 0 0 1 0 0 1 1 1 1

The given 8 bit byte with a binary value of 10101111 yields a parity value of 101001001111.

Second : From The Side Of Receiver Station .

Upon arrival the code word to the recipient the recipient scans every bit control to see whether the Parity is true or not. It will be "0" if the parity true.

***Example*:** Correct in even parity, the following bits patterns (101001001111) were received?

Parity	1	2	3	4	5	6	7	8	9	10	11	12
Corr. IN Even	1	0	1	0	0	1	0	0	1	1	1	1
P1 = 0	1		1		0		0		1		1	
P2 = 0		0	1			1	0			1	1	
P4 = 0				0	0	1	0					1
P8 = 0							0	1	1	1	1	

Algorithm 2:

Suppose the transmitter will send a code of 8 bits by using Hamming code methods which is based on sending an extra bits called (helping bit or Parity bit) in the positions (1,2,4,8,16,…etc). For example if the sender sends 4 bits that's mean it will send 3 extra bits (parity bit) in the position (1,2,4) the result code it will be of 7 bits. To know the value of each extra bit we will change each position of bit has value "1" to the binary system and apply the XOR gate with each other, then we put the result in the position of extra bit (parity bit) respectively.

***Example*:** An 8 bit byte with binary value 10110101 is to be encoded using an even-parity Hamming code. What is the binary value after encoding?

Position	1	2	3	4	5	6	7	8	9	10	11	12
Value	-	-	**1**	-	0	**1**	**1**	-	0	**1**	0	**1**

Table -1-

P3 0011

P6 0110

P7 0111

P10 1010

P12 1100

Apply XOR Gate _____

Results 0100 Parity Bits

P1 = 0 P2=0 P4=1 P8=0

Position	1	2	3	4	5	6	7	8	9	10	11	12
Value	0	0	**1**	1	0	**1**	**1**	0	0	**1**	0	**1**

Table -2-

According to the above example, assume that the recipient received the code without errors. In general, the recipient will know and identify error by converting the positions their value "1" in the code to the binary system and then apply the XOR gate between them. And then apply the XOR gate again between results and the parity bits, as shown in Tables 5 and 6. If the result is 0, this means that the code did not get any error during the transmission process.

P3	0011	
P6	0110	
P7	0111	
P10	1010	
P12	1100	
Apply XOR Gate	_____	
Results	0100	
Parity Bits	0100	
Apply XOR gate again	0000	Correct Transmitting