# 6- Simple Codes

## A) Binary Repetition Code (BRC) (Example of Correction Code or FEC)

-BRC code parameters; k=1, n= odd >1 , R<sub>c</sub> =1/n and BW<sub>exp. factor</sub> = nx100%

-Encoder algorithm (repeat message bit for n times), or in the form of code table;
 For simple case where; n=3

M= [m <sub>1</sub> ]	$C = [c_1 \ c_2 \ c_3]$
0	000
1	111

-The channel will add random errors (0,1,2, or 3 errors) to encoder output. R = C +Errors

#### -Decoder Algorithm

- -The possible received words R are shown in the next table. Also shown in the table the decoded codeword  $\hat{c}$ .
- -This decoding uses the fact that if 000 (or 111) is encoded the nearest C is chosen based on Majority Logic Decision.
- -If the number of zeros and ones in R being N₀ and N₁, then The Majority Logic Decision (or BRC decoding algorithm) is;

If 
$$N_0 > N_1$$
 then  $\hat{C} = [000]$  and hence  $\widehat{M} = [0]$   
If  $N_0 < N_1$  then  $\hat{C} = [111]$  and hence  $\widehat{M} = [1]$   
Since n=odd, there is no condition of  $N_0 = N_1$ .

Decoding Table			
No.	R =	Ĉ	
	[r <sub>1</sub> r <sub>2</sub> r <sub>3</sub> ]	·	
1	000	000	
2	001	000	
3	010	000	
4	100	000	
5	111	111	
6	110	111	
7	101	111	
8	011	111	



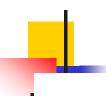
#### B) Parity Check Codes (PCC) (even-PCC and Odd-PCC) (Example of Detection Code)

-PCC code parameters; n-k=1, or k= n-1,  $R_c = (n-1)/n$  and  $BW_{exp. factor} = n/(n-1)$ Here there is only one check bit.

-Encoder algorithm for Even-PCC (let n=3 for example)  $c_1 = m_1 \ c_2 = m_2 \ and \ c_3 = m_1 + m_2 \ (+ here is XOR or Mod-2 addition)$ 

Encoding Table Even-PCC n=3			
M= [m <sub>1</sub> m <sub>2</sub> ]	$C = [c_1 \ c_2 \ c_3]$		
00	000		
10	101		
01	011		
11	110		

- -The algorithm use mod-2 addition (XOR) of m<sub>1</sub> and m<sub>2</sub> (which are c<sub>1</sub> and c<sub>2</sub>, respectively) and put the resulting bit in c<sub>3</sub>. The resultant codeword C consists of even number of binary ones always. Hence the name Even-PCC.
- There is Odd-PCC in which the check bit c<sub>n</sub> is determined by ( XOR ) which results in odd number of ones in C.



### -Decoder algorithm

- The decoder will determine whether the received word being correct or not. Using the decoding table, one may say that the blue words are correct while the others are incorrect. Since k=2 here we need to decode each bit and determined whether it is correct or not.

Unfortunately, this is not possible here. The reason for that: any correct received word with 2 errors is also correct received word (see encoder table). Thus, PCC is unable to find error position exactly, instead it may decide on the whole R to be correct or not. PCC is actually error detection code (not error correction code)

Decoding Table		
No.	R =	
	[r <sub>1</sub> r <sub>2</sub> r <sub>3</sub> ]	
1	000	
2	011	
3	101	
4	110	
5	001	
6	010	
7	100	
8	111	



The detection rule (decoding algorithm) is given by one of the following approaches:

i-Find  $S = r_{1} + r_{2} + r_{3}$  if S = 1 (R is incorrect due to odd number of errors in R) else if S = 0 (R is either correct or there is even number of errors in R)

ii- The above also can be performed by examining the number of ones in R ( $N_1$ ).

If  $N_1$ =Odd (R is incorrect due to odd number of errors in R),

else If  $N_1$ =Even (R is either correct or there is even number of errors in R)

As a result, there is no clear decision about the received word being correct or not. Such code actually is not considered for transmission over information channel due to its correction limitation. It is only used for the acse where error may affect one bit (low P<sub>e</sub>) or in ARQ system.

Regarding error probability calculation of PCC, it cannot be determined exactly since there is no correction (just detection).



## Remarks on BRC and PCC

In previous two examples, we have seen two codes with following properties:

Parameter	BRC	PCC	Comment
k	1	n-1	Large k in PCC (Good)
No. of Check bits (n-k)	(n-1)	1	Large check bits in BRC (Bad)
Rc	1/n	(n-1)/n	For very large n: BRC $R_{c}>0$ (Bad) while PCC $R_{c}>1$ (Good)
BW <sub>exp</sub>	100n%	100(n/(n-1))%	Large in BRC (Bad) Small in PCC (Good)
Decoding Performance	correction	just detection	Also the detection of PCC is weak!

BRC and PCC represent the two extreme cases of ECC. The practical codes in general have parameters:

$$1/n < R_c < (n-1)/n$$

with acceptable error correction.