5

$mBnL$ — m: Binary Bits; n: pulses; L: level of pulses

- B : Binary (2)
- T : Ternary (3)
- Q : Quaternary (4)

Coding rate $= \frac{m}{n}$

$2^m \le L^n$

E.g. 2B1Q ✓

coding rate = 2

used in DSL system

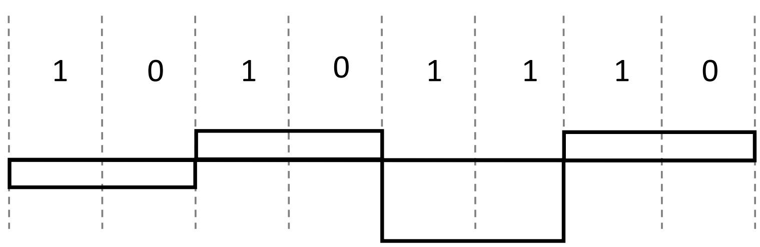
4B3T

$2^4 = 16 < 27 = 3^3$

2B*1*Q Coding* & *m*B*n*L Coding

2B*1*Q Encoding

previous level: positive



2B*1*Q

	Previous level: positive	Previous level: negative
Next bits	Next level	Next level
"00"	+A/2	-A/2
"01"	+3A/2	-3A/2
"10"	-A/2	+A/2
"11"	-3A/2	+3A/2

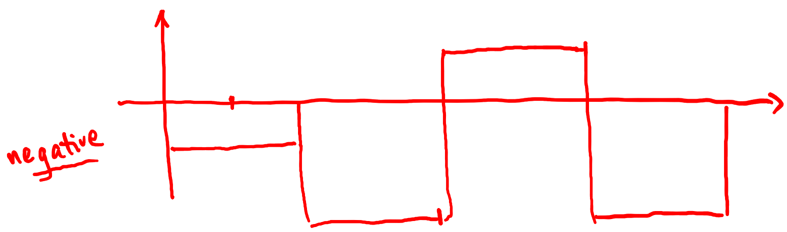
*m*B*n*L

- Maps block of *m* information bits into *n* pulses
- There is a total of L different levels of pulses

** This version of 2B1Q coding is based on "Data Communications and Networking" by Behrouz A. Forouzan.*

	Previous level: positive	Previous level: negative
Next bits	Next level	Next level
“00”	+A/2	-A/2 ✓
“01”	+3A/2	-3A/2 ✓
“10”	-A/2	+A/2 ✓
“11”	-3A/2 ✓	+3A/2

0 0 0 1 1 0 1 1



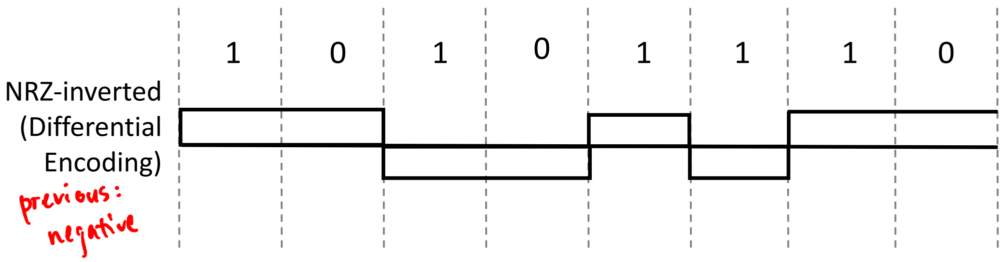
2B1Q

bit rate = Baud rate * coding rate

10 Mbps = 5 M Hz (2)

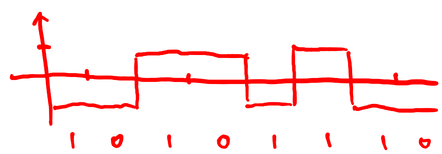
6

NRZ-Inverted Coding

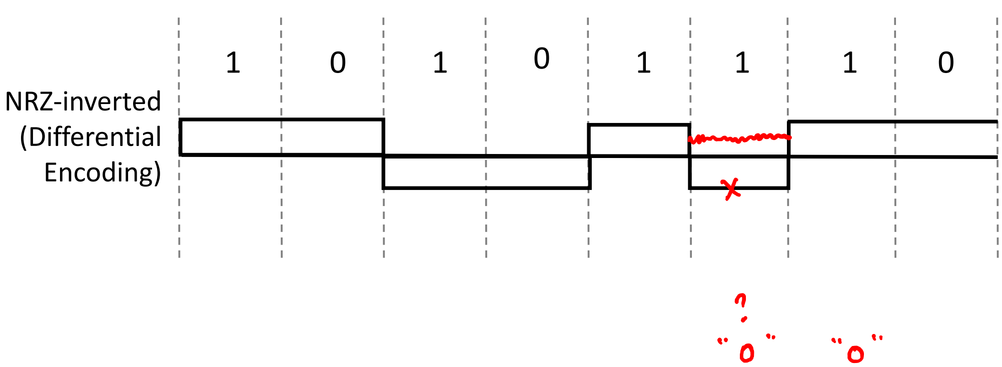


- Two signal formats: {–A/2, +A/2}
- "1" maps to transition in signal format at beginning of the bit interval
- "0" maps to no transition
- Differential line coding { dependancy between adjacent bit intervals (signal formats)
  reference signal format

* "polarity insensitive"
* systematic error in polarity



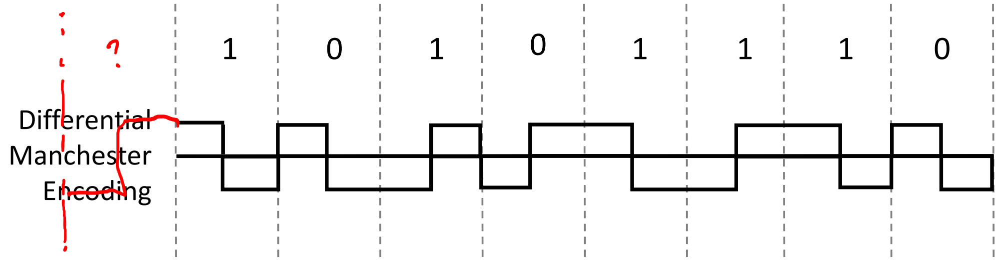
NRZ-Inverted Coding



* Errors occur in pairs
* Long string of "0" → timing issue

1

Differential Manchester Coding

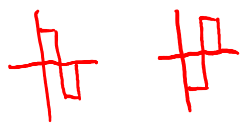


- Systematic error in polarity (i.e., + become – and vice versa) is possible
  - Manchester Coding can not handle this type of error
- Differential Manchester Coding provides robustness to this type of error
  - "1" maps to transition in signal format at beginning of the bit interval
  - "0" maps to no transition
  - Another type of differential line coding
  - Errors occur in pairs

difference in signal format

entire bit interval

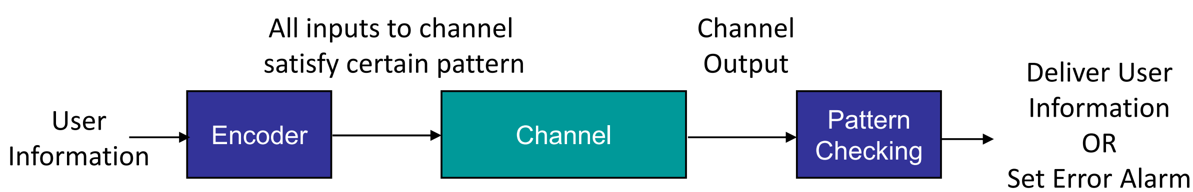
* two signal formats:



Topic 3: Error Detection and Recovery

General Error Detection System

- Transmitter (encoder) adds redundancy to user information to become codewords and transmit codewords over communication channel
- All transmitted codewords satisfy certain pattern that is agreed upon between transmitter and receiver
- If a received codeword doesn't satisfy the pattern, it is in error
  - Error detected!



even

Example: Single Parity Check Code

- Append an overall parity check bit to k information bits

Information Bits: $(i_{k-1}, \ldots, i_1, i_0)$ arbitrary

k

Parity Check Bit: $c = (i_{k-1} + \ldots + i_1 + i_0) \bmod 2$

$n-k=1$

Codeword: $(i_{k-1}, \ldots, i_1, i_0, c)$

$n=k+1$

"pattern"

* even # 1's in the codeword

info bits : k

redundancy bits : n-k

codeword bits : n

* even # 1's in info bits

⇒ c = 0

odd # 1's

⇒ c = 1

E.g.

$k = 4, \quad n = 5$

$$\left[\ \underline{1 \quad 0 \quad 1 \quad 1} \ \vdots \ 1 \ \right]_{1\times 5}$$

① Error: V V X X V

$$\left[\ 1 \quad 0 \quad 0 \quad 0 \ \vdots \ 1 \ \right]_{1\times 5}$$ undetectable

② X V X V X

$$\left[\ 0 \quad 0 \quad 0 \quad 1 \ \vdots \ 0 \ \right]_{1\times 5}$$ detectable

Fraction of Undetectable Errors (FUE)

$$= \frac{\text{total \# undetectable errors}}{\text{total \# possible valid errors}}$$ (as small as possible)

Error Vector

"pattern"

- Suppose we transmit a codeword that has n bits

$[\_ \ \_ \ \_ \ \cdots \ \_]_{1\times n}$

- Define the error vector $\underline{e} = [e_{n-1}, \ldots, e_1, e_0]_{1\times n}$ where
  - $e_i = 1$ if error occurs to the i^{th} bit position
  - $e_i = 0$ otherwise

$e = [1 \ 1 \ 0 \ \cdots \ 0]$

2-bit error

$e = [1 \ 0 \ 0 \ \cdots \ 0 \ 1]$

- Fraction of Undetectable Errors (FUE)
  - FUE = total # undetectable errors / total # valid errors