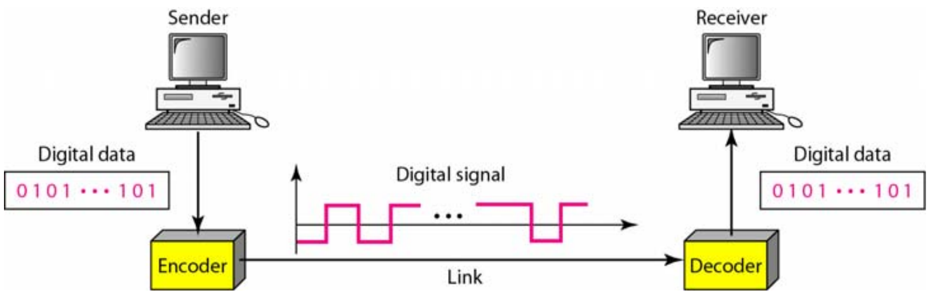
Line Coding: Design Consideration

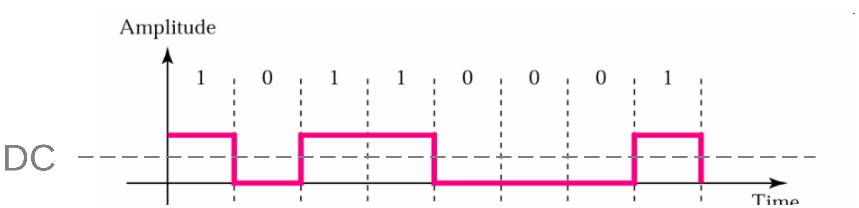
Line Coding – **process of converting binary data (sequence of bits) to a digital signal**

- **digital signal depends 'linearly' on information bits - bits are transmitted 'one-by-one' - different from block coding**

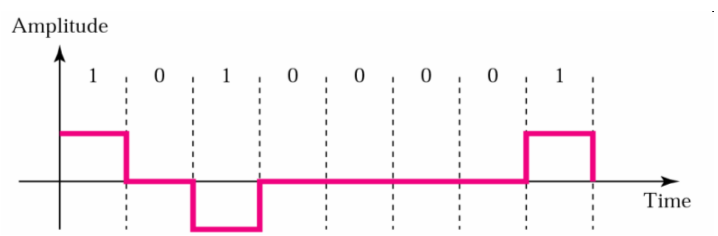


Data vs. Signal Level

- **data levels** – **number of values / levels used to represent data (typically only two: 0 and 1)**
- **signal levels** – **number of values / levels allowed in a particular signal**



Two signal levels, two data levels.



Three signal levels, two data levels.

Line Coding: Design Consideration (cont.)

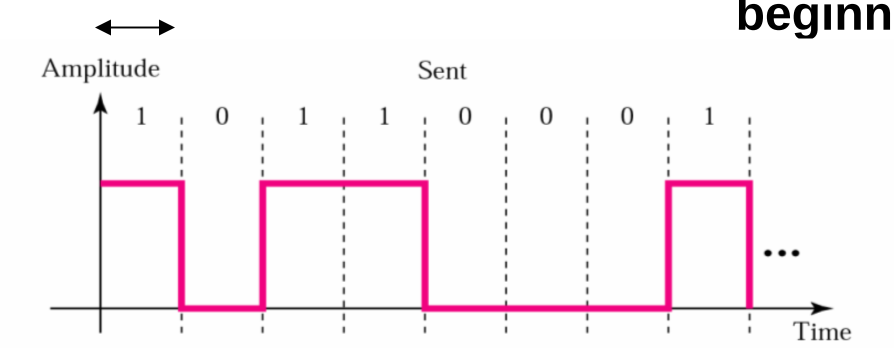
DC Component in Line Coding – **some line coding schemes have a residual (DC) component, which is generally undesirable**

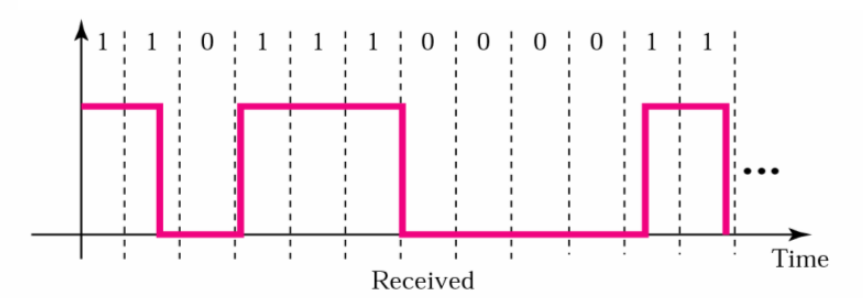
- **transformers do not allow passage of DC component**
- **DC component ⇒ extra energy – useless!**

Self-Synchronization (Clocking) – **to correctly interpret signal received from sender receiver's bit interval must exactly correspond to sender's bit intervals**

- **if receiver clock is faster/slower, bit intervals not matched ⇒ receiver misinterprets signal**
- **self-synchronizing digital signals include timing information in itself, to indicate the beginning & end of each pulse** **(see pp. 8-10)**

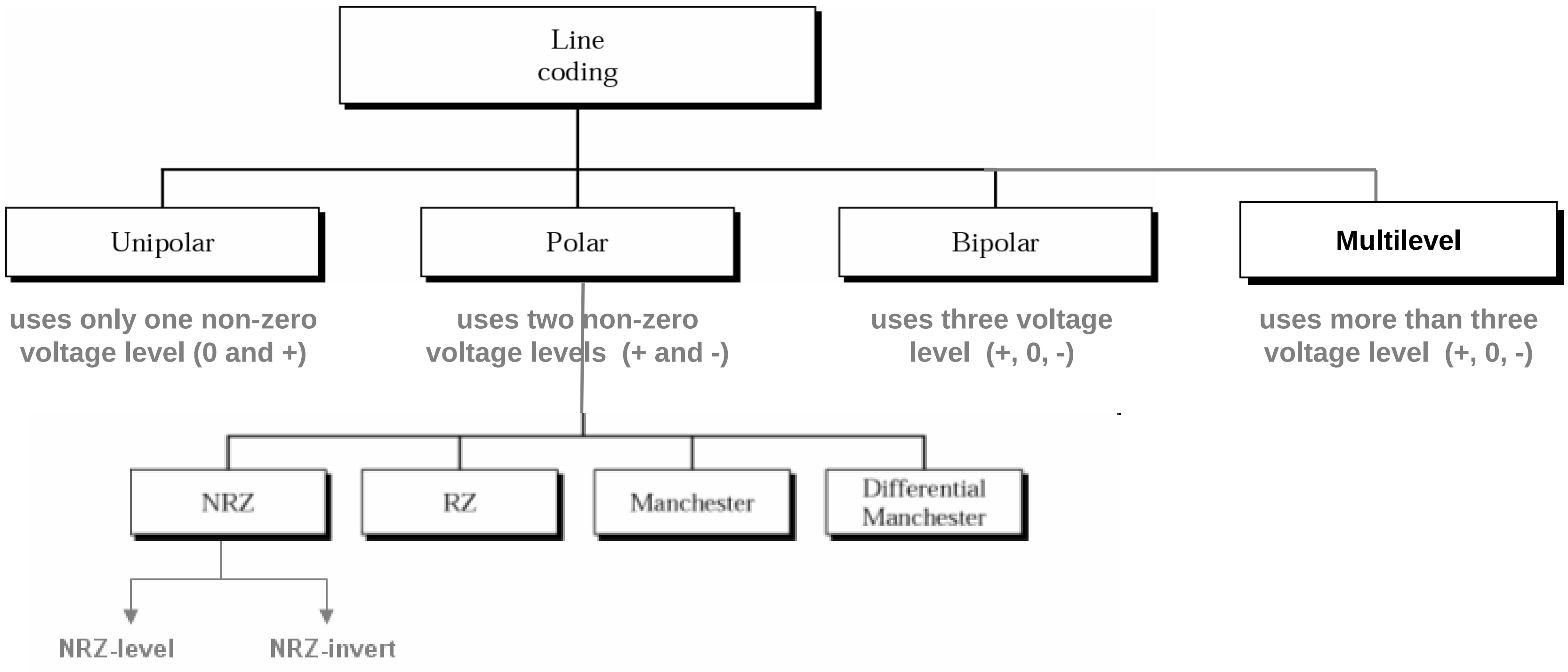
56 kbps ⇒ 0.0178 ms





Line Coding: Design Consideration (cont.)

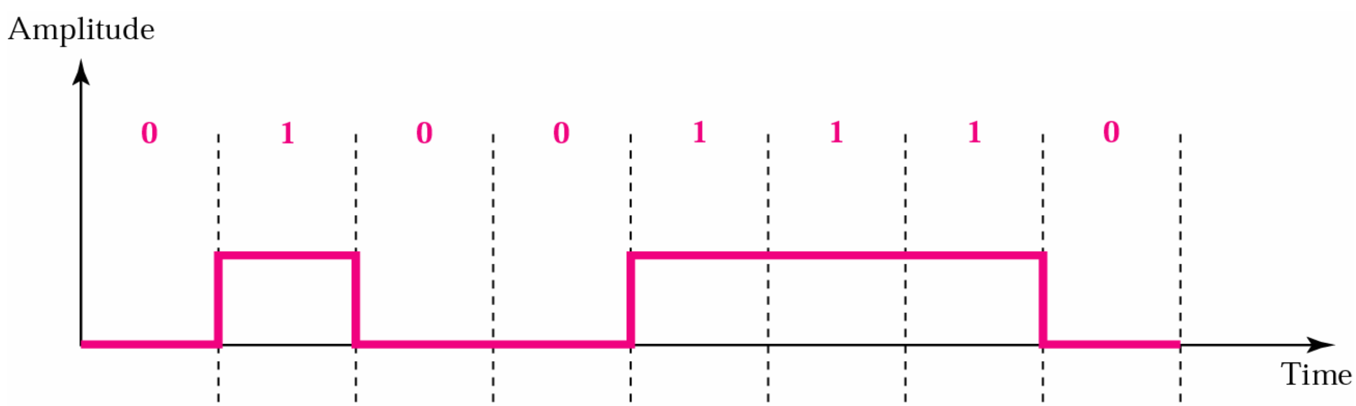
Line Coding Schemes – can be divided into four broad categories



Line Coding: Unipolar

Unipolar Line Coding – **uses only <u>one non-zero</u> and one zero voltage level**

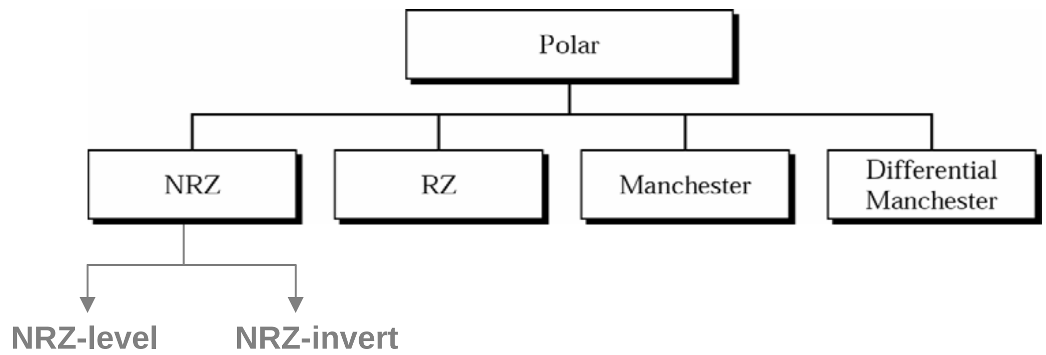
- **(e.g.) 0 = zero level, 1 = non-zero level**
- **simple to implement, but obsolete due to two main problems:**
  - **DC component present ☹**
  - **lack of synchronization for long series of 1-s or 0-s ☹**



Line Coding: Polar

Polar Line Coding – uses <u>two non-zero voltage level</u> for represent. of two data levels - one positive & one negative

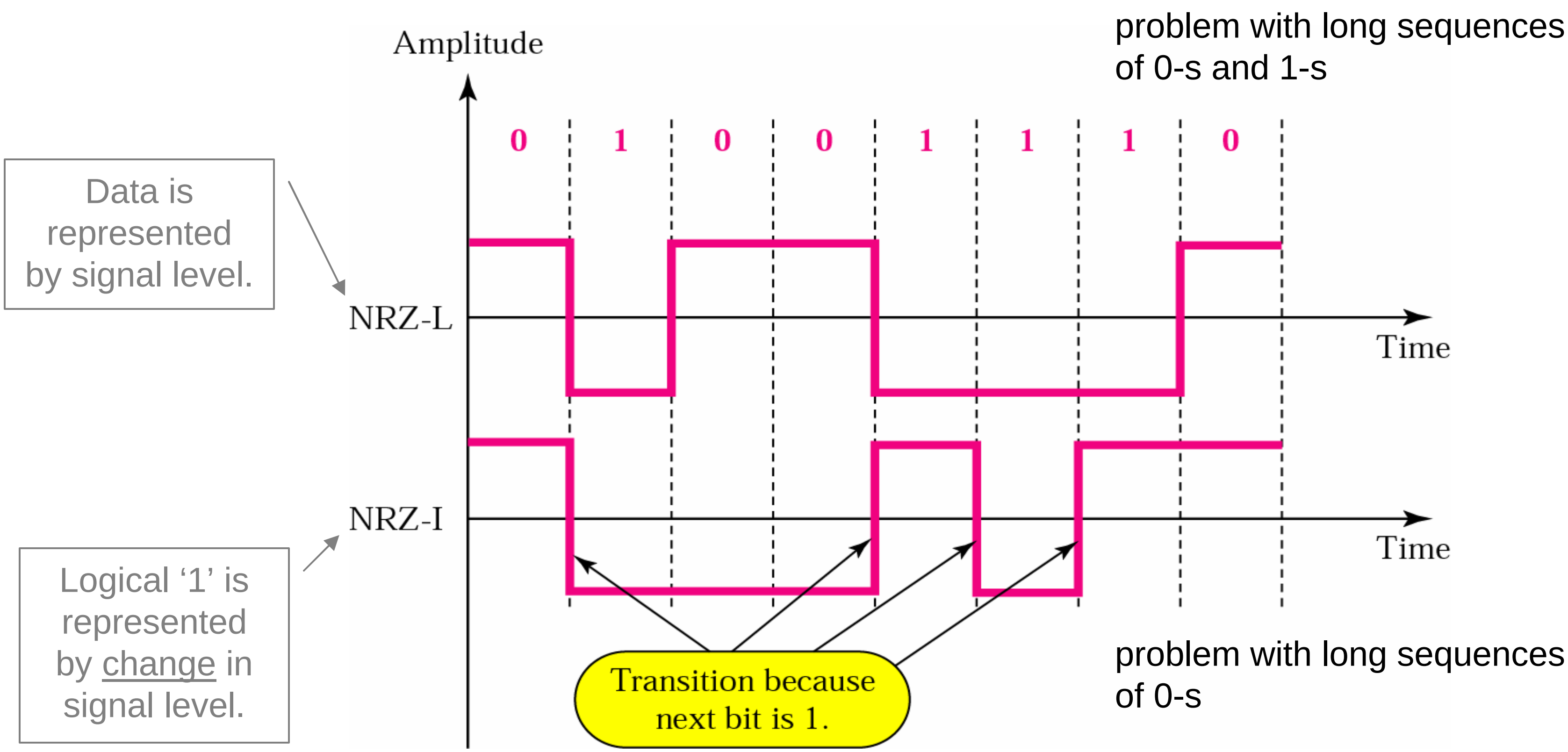
- "DC-problem" alleviated ☺
- 4 main types of polar coding:



(1) Nonreturn to Zero (NRZ)

- **NRZ-level**: signal level represents particular bit, (e.g.) 0 = positive volt. , 1 = negative volt.
  - poor synchronizat. for long series of 1-s & 0-s ☹
- **NRZ-invert**: inversion of voltage level = bit 1, no voltage = bit 0
  - 1s in data streams enable synchronization
  - long sequence of 0-s still a problem ☹

Line Coding: Polar (cont.)

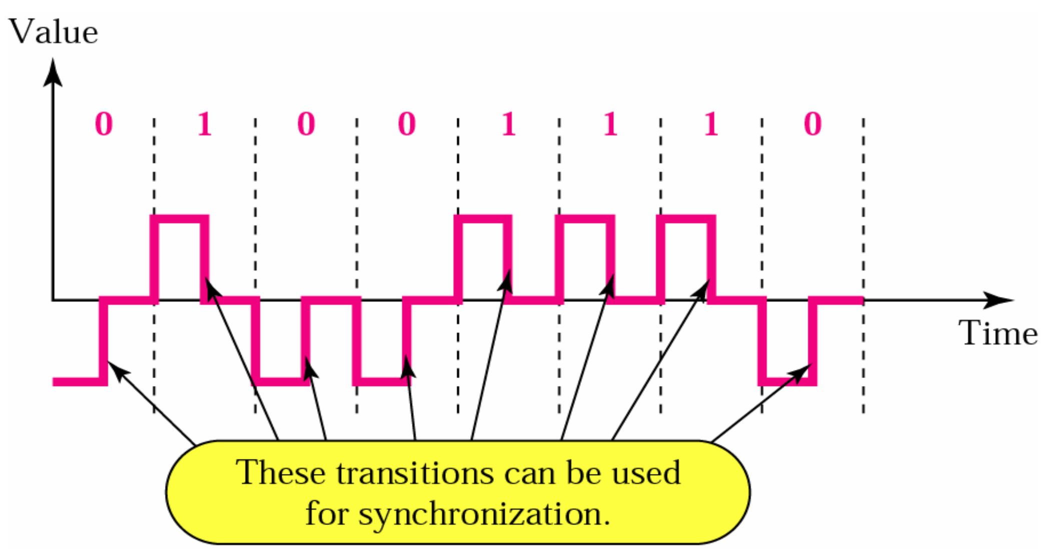


NRZ-I is better than NRZ-L, but it still does not provide complete synchronization.
To ensure complete synchronization, there must be a signal change for each bit.

Line Coding: Polar (cont.)

(2) Return to Zero (RZ) – **(e.g.) 0 = negative volt., 1 = positive volt., AND signal must return to zero halfway through each bit interval**

- **perfect synchronization** ☺
- **drawback – 2 signal changes to encode each bit ⇒ pulse rate is x2 rate of NRZ coding, i.e. more bandwidth is required, regardless of bit sequence** ☹

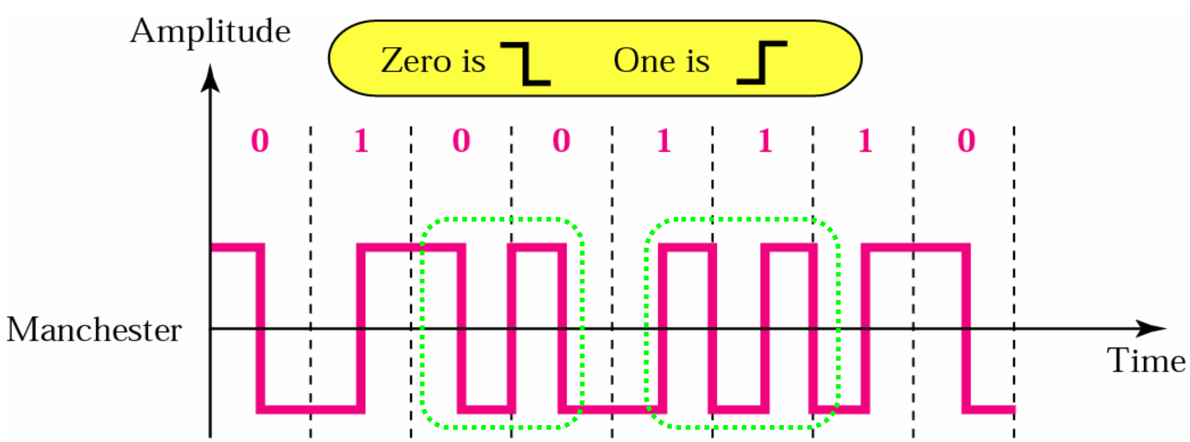


Non-zero level ⇒ beginning of a new bit.

Line Coding: Polar (cont.)

(3) Manchester – **inversion at the middle of each bit interval is used for both synchronization and bit representation**

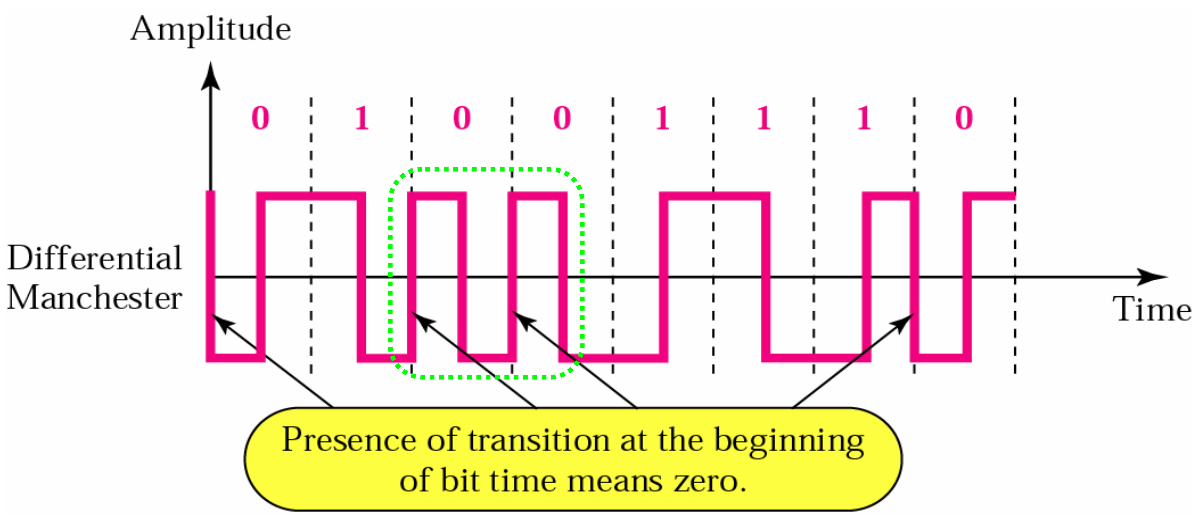
- **0 = pos-to-neg transition, 1 = neg-to-pos transition**
- **perfect synchronization ☺**
- **there is always transition at the middle of the bit, and maybe one transition at the end of each bit**
- **fine for alternating sequences of bits (10101), but wastes bandwidth for long runs of 1-s or 0-s ☹**
- **used by IEEE 802.3 (Ethernet)**



Line Coding: Polar (cont.)

(4) Differential Manchester – **inversion in the middle of bit interval is used for synchronization – presence or absence of additional transition at the beginning of next bit interval identifies the bit**

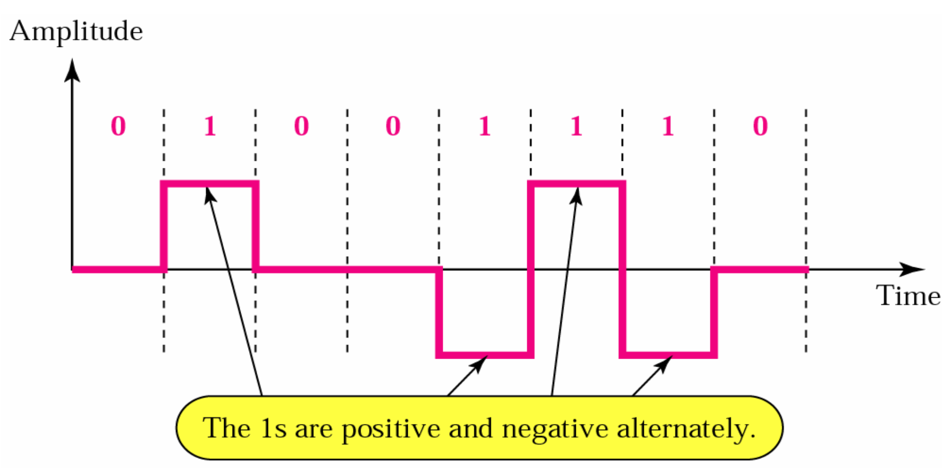
- **0 = transition, 1 = no transition**
- **perfect synchronization ☺**
- **fine for long runs of 1s, but wastes band-width for long runs of 0-s ☹**
- **used by IEEE 802.5 (Token Ring)**



Line Coding: Bipolar

Bipolar Line Coding – uses <u>two non-zero and zero voltage level</u> for representation of two data levels

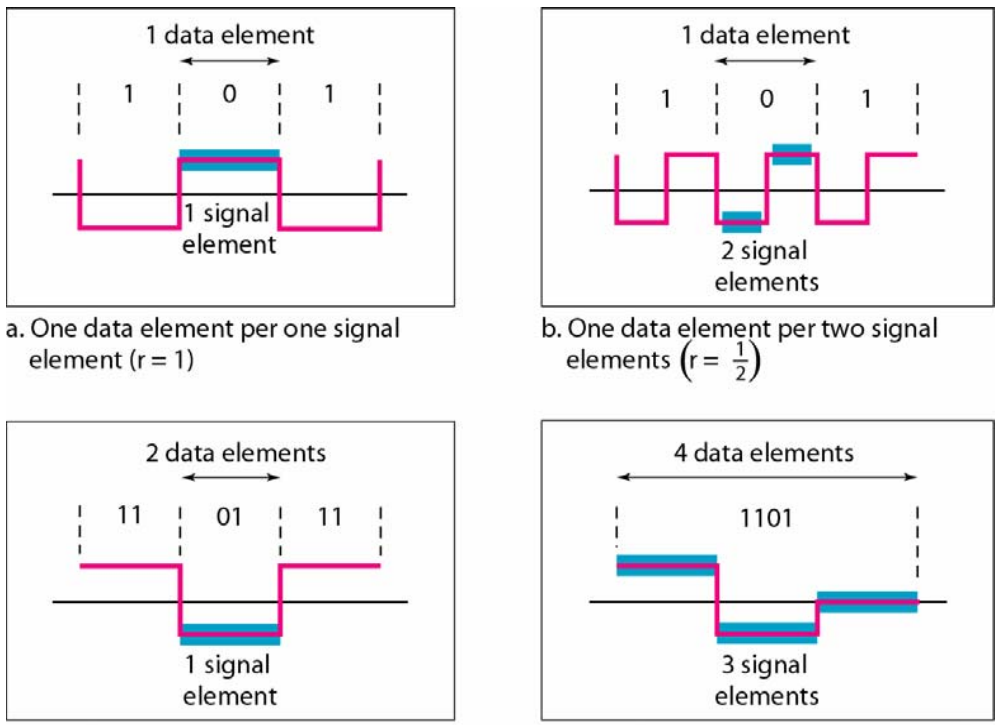
- **0 = zero level**; **1 = alternating pos and neg level**
- if 1st ‘bit 1’ is represented by positive amplitude, 2nd will be represented by negative amplitude, 3rd by positive, etc.
- less bandwidth required than with Manchester coding (for any sequence of bits) ☺
- loss of synchronization is possible for long runs of 0-s ☹



Data Rate vs. Baud Rate

Data Rate – # of data elements (bits) sent in 1 sec – unit: bps

Signal Rate – # of signal elements/pulses sent in 1 sec – unit: baud



a. One data element per one signal element ($r = 1$)

b. One data element per two signal elements $\left(r = \frac{1}{2}\right)$

c. Two data elements per one signal element ($r = 2$)

d. Four data elements per three signal elements $\left(r = \frac{4}{3}\right)$

One goal of data communications is to increase data rate (speed of transmission) **while decreasing signal rate** (bandwidth requirements).

Data Rate vs. Baud Rate (cont.)

r = data rate / signal rate – **ratio between data & signal rate**

Signal rate observed in case of a particular data-bit stream: – **depends on N [bps], 1/r [bit/pulse], and the actual data pattern**

- **signal rate for a pattern of all 1-s or all 0-s may be different from that for a patter of alternating 1-s and 0-s**

$$S = \underbrace{c}_{\text{case factor}} \cdot N \cdot \frac{1}{r} \quad \text{[pulses/sec]}$$

Data Rate vs. Baud Rate (cont.)

Example [data vs. signal rate]

A signal is carrying data in which one data element is encoded as one signal element (r=1).

If the bit rate is 100 kbps, what is the average value of the baud rate, assuming c is between 0 and 1?

Answer:

$C_{average}$ = 0.5

$$S = c \cdot N \cdot \frac{1}{r} = \frac{1}{2} \cdot 100{,}000 \cdot \frac{1}{1} = 50{,}000\ [\text{pulses/sec}] = 50\ [\text{kbaud}]$$

Digital Transmission Modes

How do we send bits / pulses over wire?

- **Serial Mode: 1 bit is sent with each clock tick**
  - **one communication channel / wire is needed**
- **Parallel Mode: multiple bits are sent with each clock tick**
  - **multiple channels / wires, bundled in one cable, are required**
  - **advantage: n-times faster than serial mode**
  - **disadvantage: cost = 8x wires (used only over short distances)**