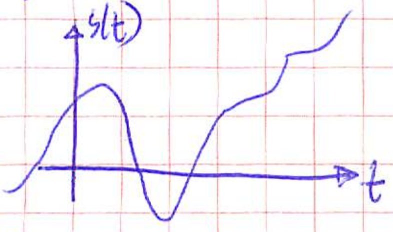
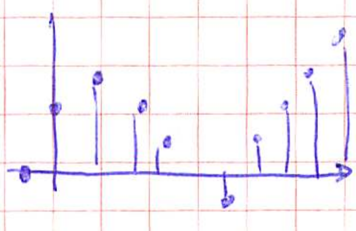
# Recitation 3

9-16-2011

Recitation 3

## (*) Analog vs. Digital



Analog: - continuous levels
- continuous time

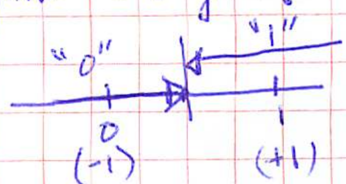
Digital: - discrete (quantized) levels
- discrete time

Nyquist - sample at 2x highest frequency (6.003)

Digital ~ approximate (represent) source by finite set of values at discrete times.

### Why Digital

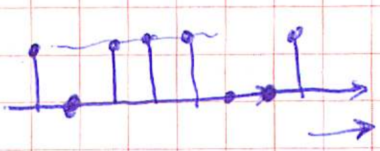
1. Easier to compress, transmit, preserve, reconstruct (nearly error-free)
   - ECC only possible in digital!
   - Can withstand reasonable noise!




2. - Viability of regeneration using buffers! (can reduce noise @ each stage)
3. - Digital (HW) flexible, "easier", cheaper!
4. - Multiplexing easier (several signals)
5. - Easier to learn & master!

### Representation

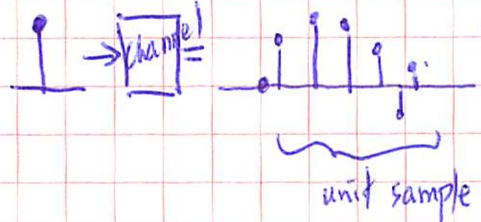
To send '10111001' —




(Not good! Why?)

## Impulse Response (Unit Sample Response) : →

Response to

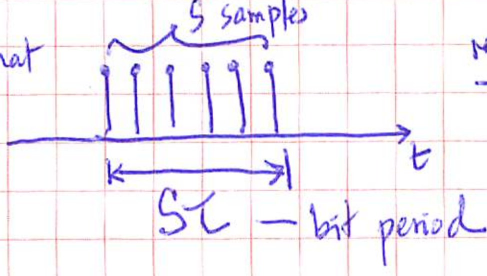



Receiver Input. rather than just

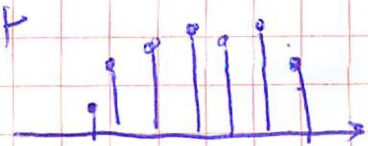
→ So need S samples for each bit !

i.e. S samples / bit

f ~ sampling frequency = $\frac{1}{\tau}$ So that




resulting output →



**Defns:**

$\tau$ = sampling period = $\frac{1}{f}$ (Hz) S = Num. samples per each bit (per bit)

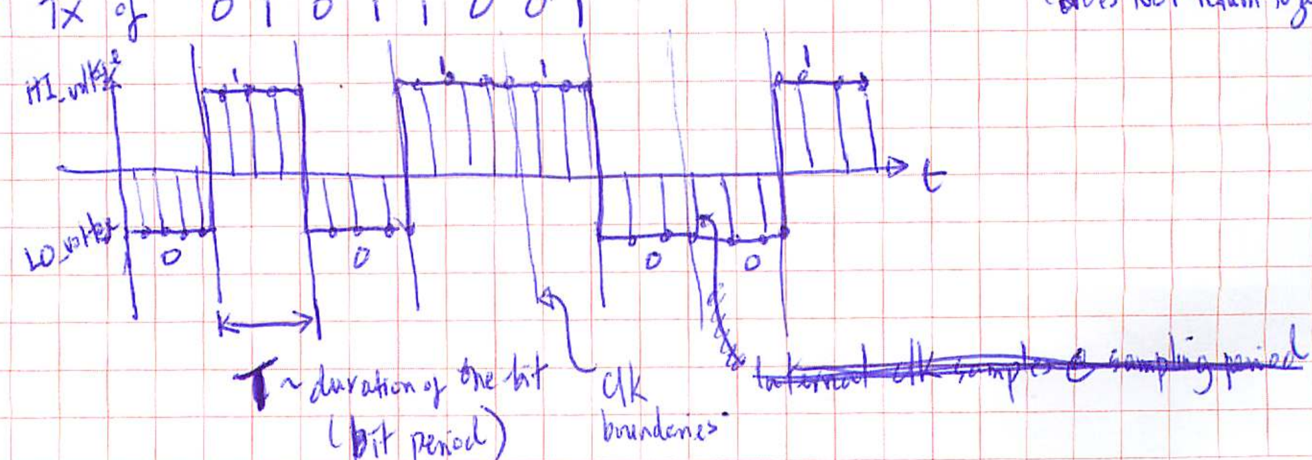
$T = S\tau$ = bit period (second)

Bit rate = $\frac{1}{S\tau}$ OR $\frac{f}{S}$ (bits/second)

---

**Clocking:** Simplest encoding scheme NRZ — voltage level stays constant over bit period. (does NOT return to zero)

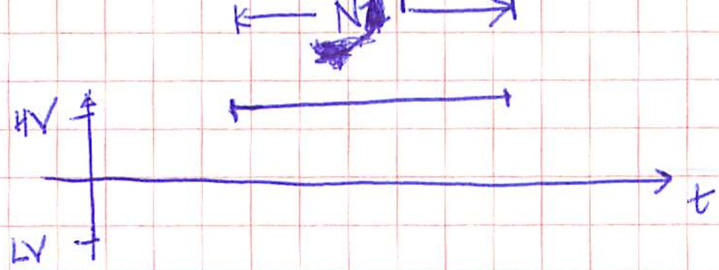
(*) Consider Tx of 0 1 0 1 1 0 0 1



(*) Now Consider,

(*) ~~Manchester Code~~ :

Consider




Q1: What "bit" is being transmitted? A: "1"

Q2: How many "bits" ✓ ✓ ? A: depends on 'bit period' $\tau$.

(*) If told Num Samples per bit S, then can find Num bits N.

Using 'ruler' to measure introduces errors — "variations" among users!
(Think TX and RX clocks).

(*) Define timing T (bit period) OR Num Samples ~~within~~ for a 'bit'.

~~Receiving~~ TX and Rx must agree on T (sec.)

→ If [Tx] - - - and [Rx] use 'same' clk to determine T, great!

→ How about your clocks? (In classroom, home...)

(*) Tx / Rx need mechanism to 'synchronize' timing!

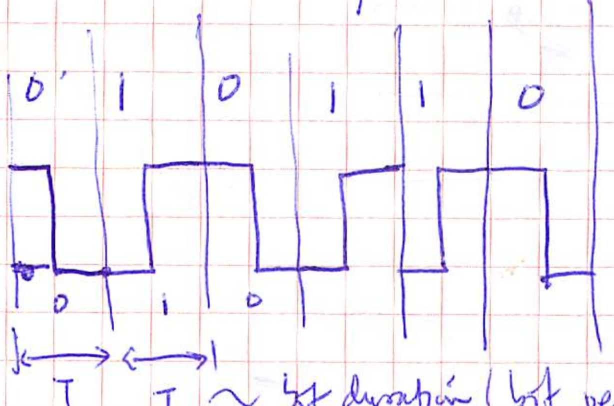
— With constant signal, no synchronization mechanism!
(A: sing 'doh' B: sing 'faa')

— If signal level changes, change can be used to synchronize!
(May need to force signal changes for this purpose)

- Manchester Code (aka - self-synchronizing code) — uses changes in signal level to synchronize Tx, Rx clks.

  ~ '0' — HI-to-LO voltage transition
  '1' — LO-to-HI voltage ✓

e.g.




T ~ bit duration (bit period).

(+) More transitions allow us to synchronize Tx / Rx clks better.

(−) Signal held for only T/2 (seconds). even for 1 1 —


~ 2X BW needed!
(signals must change twice as fast ~ needs 2f)

Bang Bang Approach: