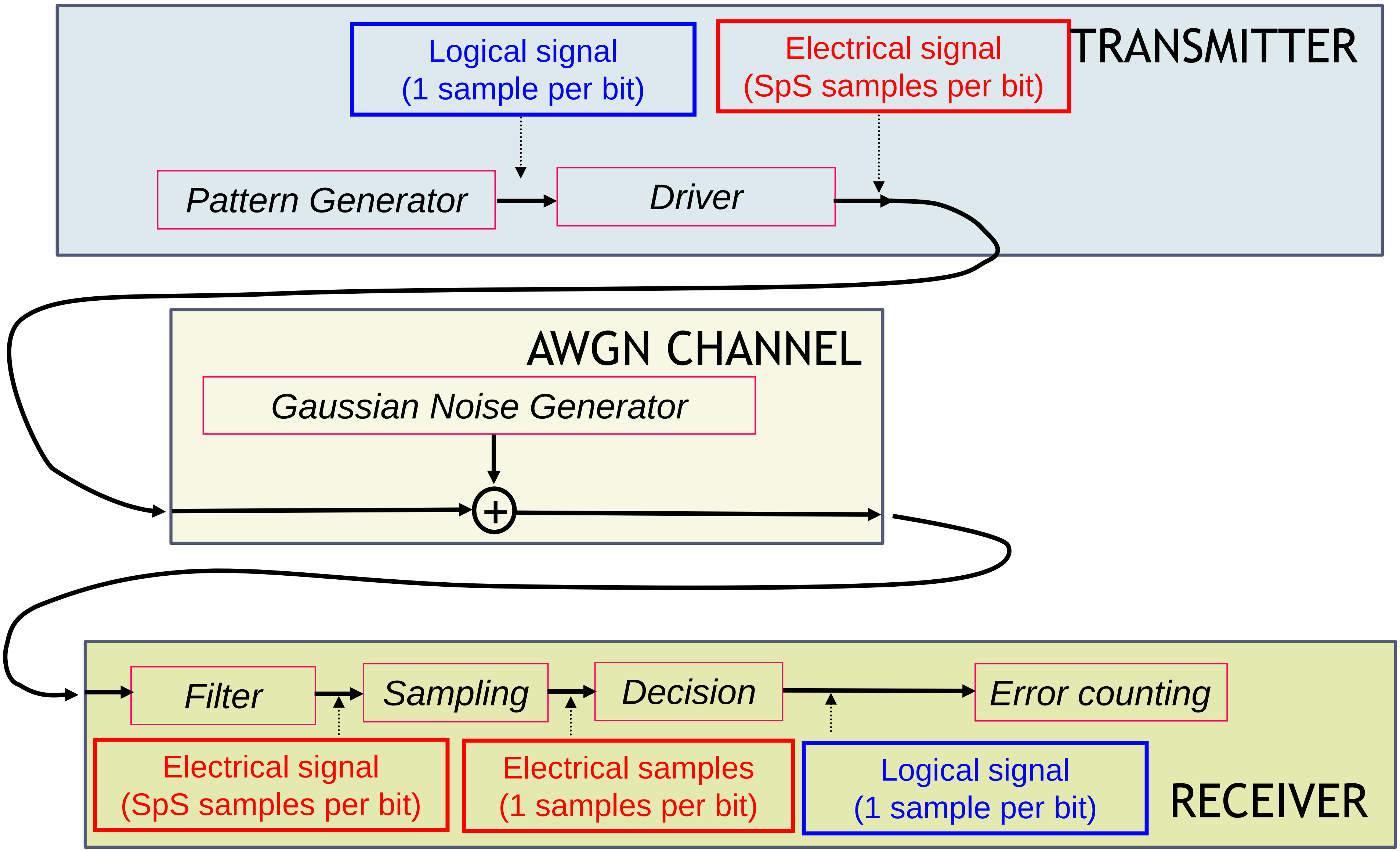
LAB #2

2-PAM system block diagram



TRANSMITTER - I

- Pattern generator: random bit generation using Matlab integer uniform distributed pseudo-random generator
  - BITS = RANDI([0 1],[Nbits,1])

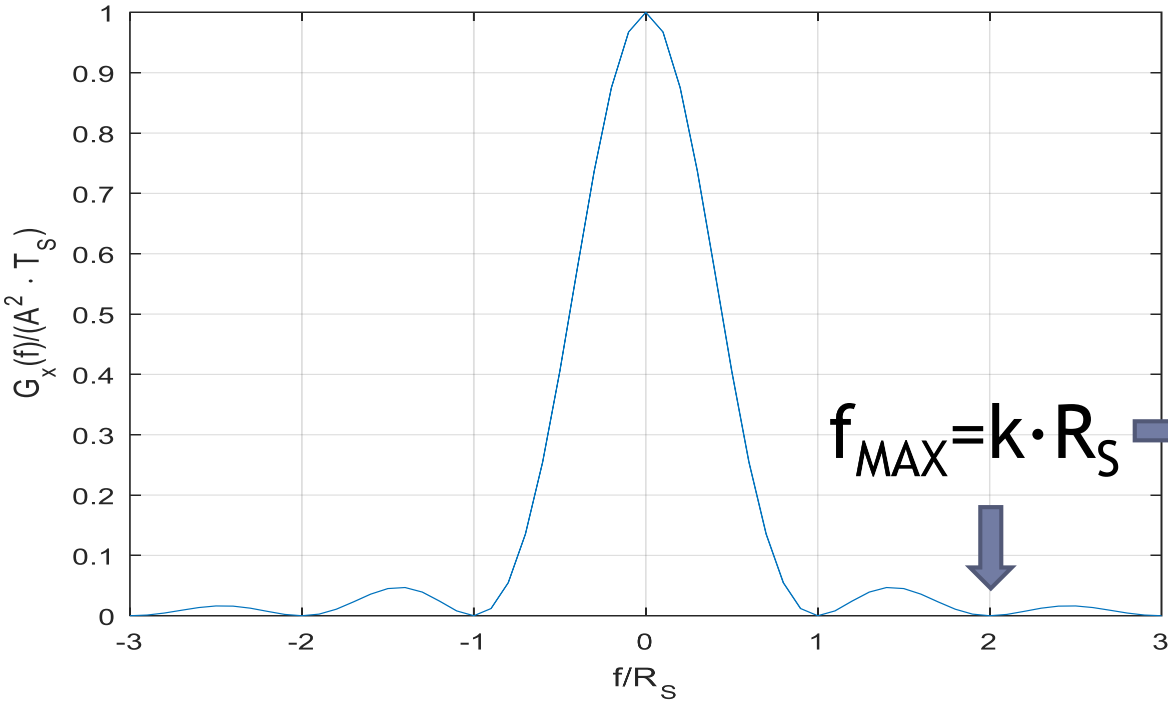
TRANSMITTER - II

- Driver
  - It maps the logical signal into the electrical signal
  - It depends on the signal s(t), i.e. the pulse
  - It depends on α_k, i.e. unipolar or antipodal

- The signal at the output of the driver it is an analog signal: in Matlab we have a sampled version of it
  - How do we define the SAMPLING TIME?
  - We must respect the sampling theorem

TRANSMITTER - III

- Signal bandwidth



$f_{MAX} = k \cdot R_S$

$f_c \geq 2 \cdot f_{MAX}$

$f_c \geq 2 \cdot k \cdot R_S$

- Define sampling frequency

TRANSMITTER - III

- From sampling frequency to sampling time

$$T_c = \frac{1}{f_c} \Rightarrow T_c = \frac{1}{2 \cdot k \cdot R_S}$$

- Simulation bandwidth $B_{SIM} = 2 \cdot k \cdot R_S$

- You can evaluate the number of samples per symbol

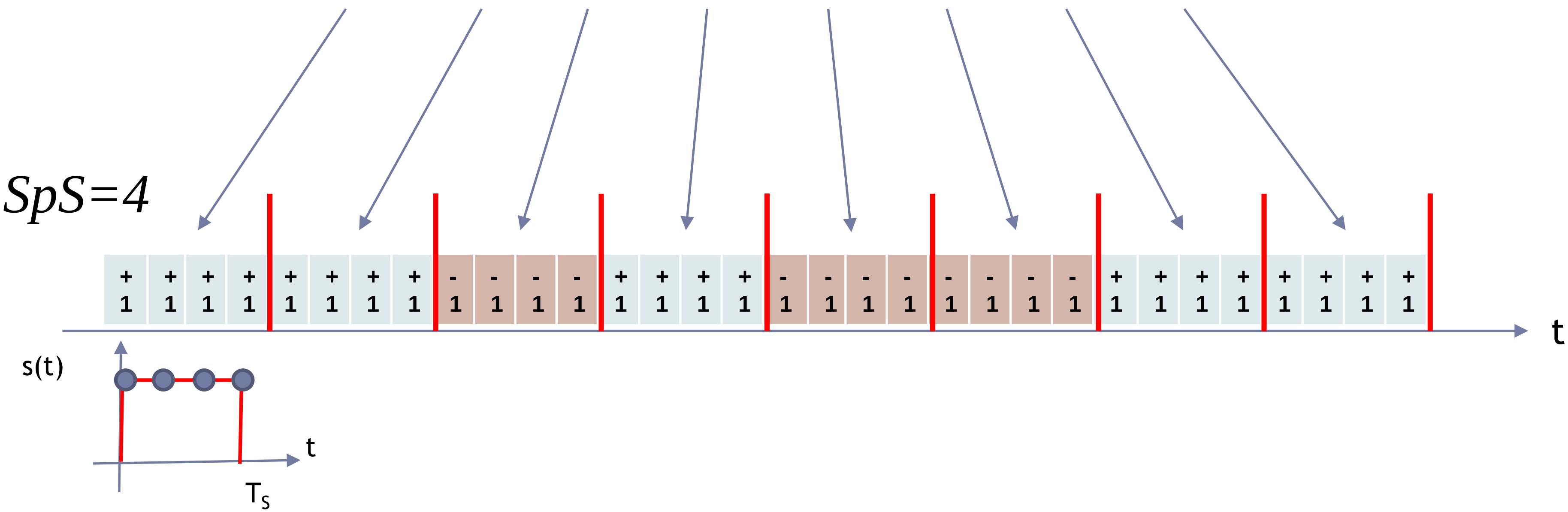
$$SpS = \frac{T_S}{T_c} = \frac{T_S}{\frac{1}{2 \cdot k \cdot R_S}} = 2 \cdot k \cdot R_S \cdot T_S = 2 \cdot k \Rightarrow B_{SIM} = SpS \cdot R_S$$

TRANSMITTER - III

$$v(t)=\sum_{k=-\infty}^{+\infty}\alpha_k s(t-kT_S)$$

BIT	1	1	0	1	0	0	1	1
Example: antipodal								
ALFA	+1	+1	-1	+1	-1	-1	+1	+1

SpS=4

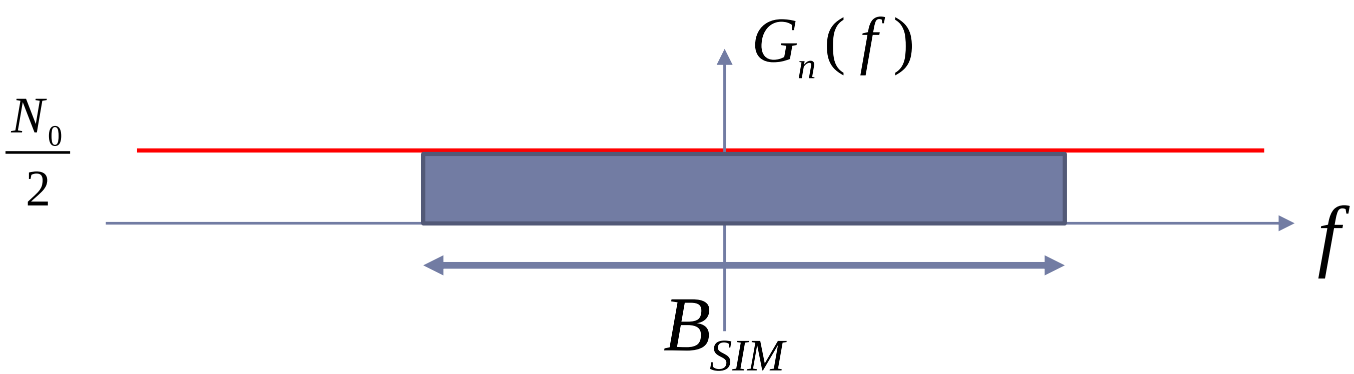


AWGN CHANNEL - I

- Noise generator: white and Gaussian
  - NOISE = RANDN(N,1)
  - Matlab generated noise has the variance equal to 1

- Which it is the noise level (density)?

$$\sigma_n^2 = \frac{N_0}{2} B_{SIM}$$



AWGN CHANNEL - II

- We MUST set a defined E_b/N_0

- How to evaluate N_0 and properly scale the noise variance?

$$\frac{E_b}{N_0} = \frac{P_{sig} \cdot T_b}{N_0} = \frac{P_{sig}}{N_0 \cdot R_b}$$

P_{sig} is the average power of the signal

Psig = mean(abs(signal).^2)

$$N_0 = \frac{P_{sig}}{\left(\frac{E_b}{N_0}\right) \cdot R_b}$$

$$\sigma_n^2 = \frac{N_0}{2} B_{SIM}$$

AWGN CHANNEL - III

- Let's evaluate the variance as a function of a given E_b/N_0

$$N_0 = \frac{P_{sig}}{\left(\frac{E_b}{N_0}\right) \cdot R_b} \qquad B_{SIM} = SpS \cdot R_S$$

$$\sigma_n^2 = \frac{N_0}{2} B_{SIM} = \frac{\frac{P_{sig}}{\left(\frac{E_b}{N_0}\right) \cdot R_b}}{2} \cdot SpS \cdot R_S = \frac{P_{sig} \cdot SpS}{2 \cdot \left(\frac{E_b}{N_0}\right)} \frac{R_S}{R_b} = \frac{P_{sig} \cdot SpS \cdot BpS}{2 \cdot \left(\frac{E_b}{N_0}\right)}$$

NOISE = sigma*RANDN(N,1)

RECEIVER FILTER - I

- Filter
  - Apply it in the frequency domain: transfer function filters are given

- Matched filter: $$H_{RX}(f) = k \cdot S^*(f) e^{-j2\pi f t_0}$$

- Single pole (RC) filter: $$H_{RX}(f) = \frac{1}{1 + j\frac{f}{f_p}}$$

RECEIVER FILTER - II

- Frequency domain
  - Apply fft to obtain signal spectrum X(f)
  - Filter using H(f)

$$Y(f) = X(f) \cdot H(f)$$

    - X(f) and H(f) must have the same length: we want to apply a sample by sample multiplication

  - Then, return in time domain using ifft
  - WARNING: be careful in using Matlab fft considering the proper positioning of frequency components in the output vector

MATCHED FILTER - I

$$H_{RX}(f) = k \cdot S^*(f) \cdot e^{-j2\pi f t_0}$$

- k is only a scaling factor: it does not affect performances
  - It multiply both signal and noise
- $e^{-j2\pi f t_0}$ is just a linear phase shift, a delay in time: it does not affect performances
  - It delays both signal and noise
- $S^*(f)$: what's that?

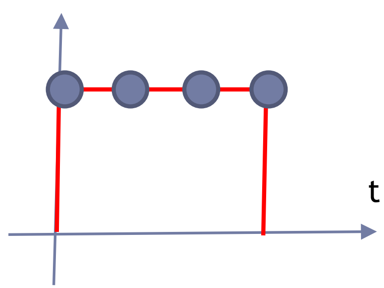
MATCHED FILTER - II

- $S(f)$ is the Fourier transform of $s(t)$, the pulse

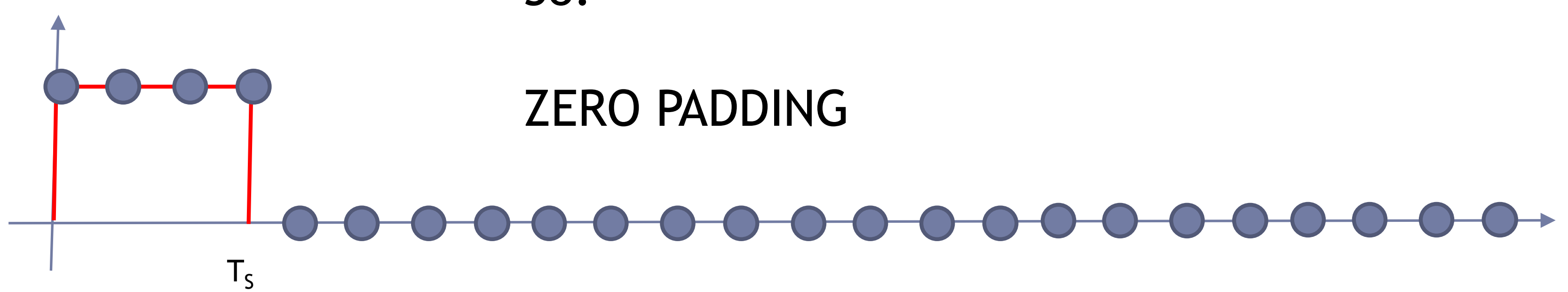
$$x_{TX}(t) = \sum_{k=-\infty}^{+\infty} \alpha_k s(t - kT_S)$$

- Example

NRZ
s(t)



Applying the fft to a single pulse described by SpS samples, we obtain SpS samples in frequency domain
So?



ZERO PADDING

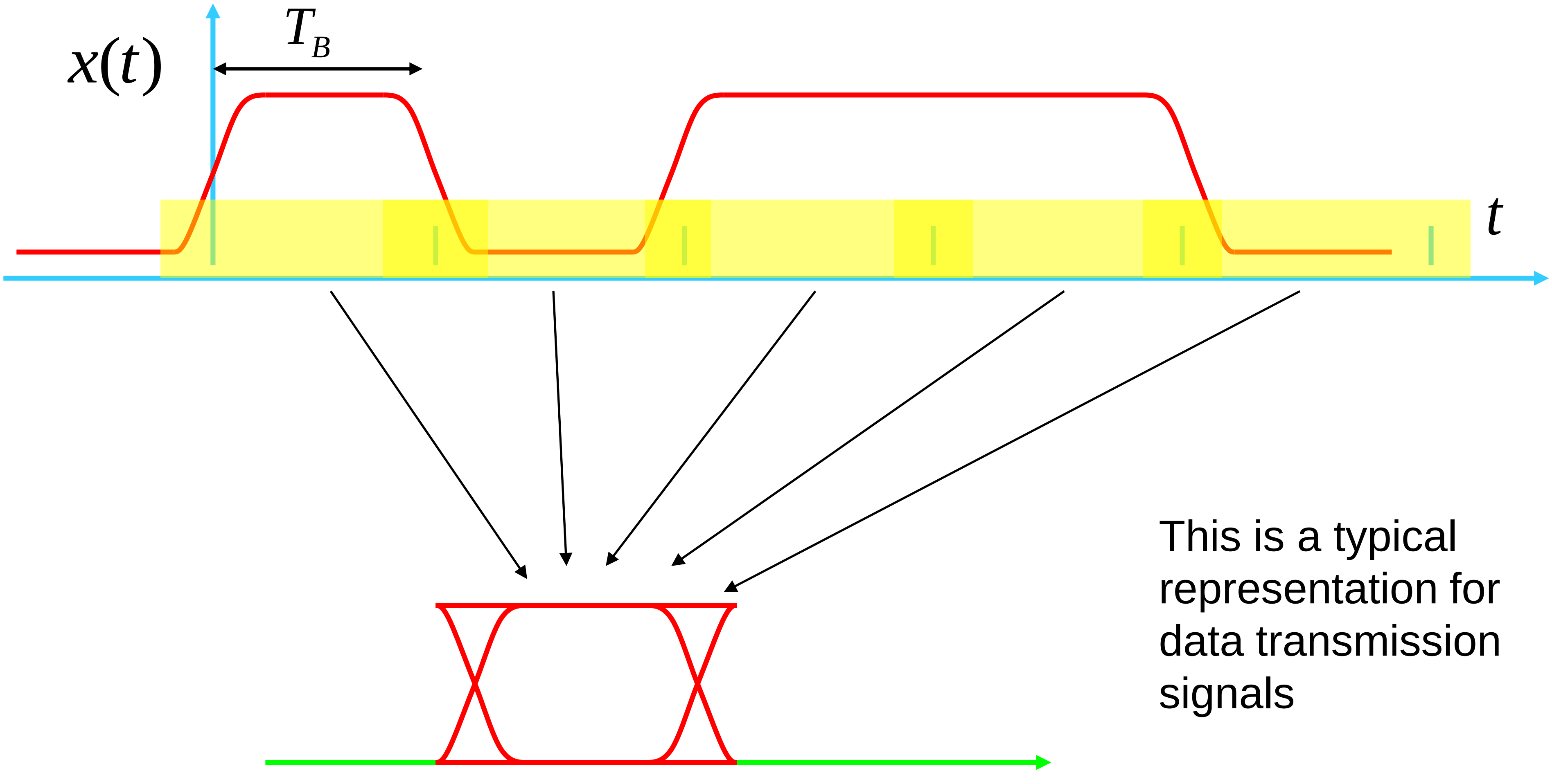
SAMPLING AND DECISION

- Sampling
  - Determine the optimum sampling instant
- Decision
  - 2-PAM it is a binary system: compare detected samples with a threshold
  - Which is the optimal threshold?
- Error counting
  - Compare TX and RX bit sequences, counting errors

SYNCHRONIZATION: COMMENT

- If the simulation algorithm introduces a delay, you must synchronize the received sequence with respect to the transmitted one
- Suggested frequency domain filtering approach does not introduce a delay

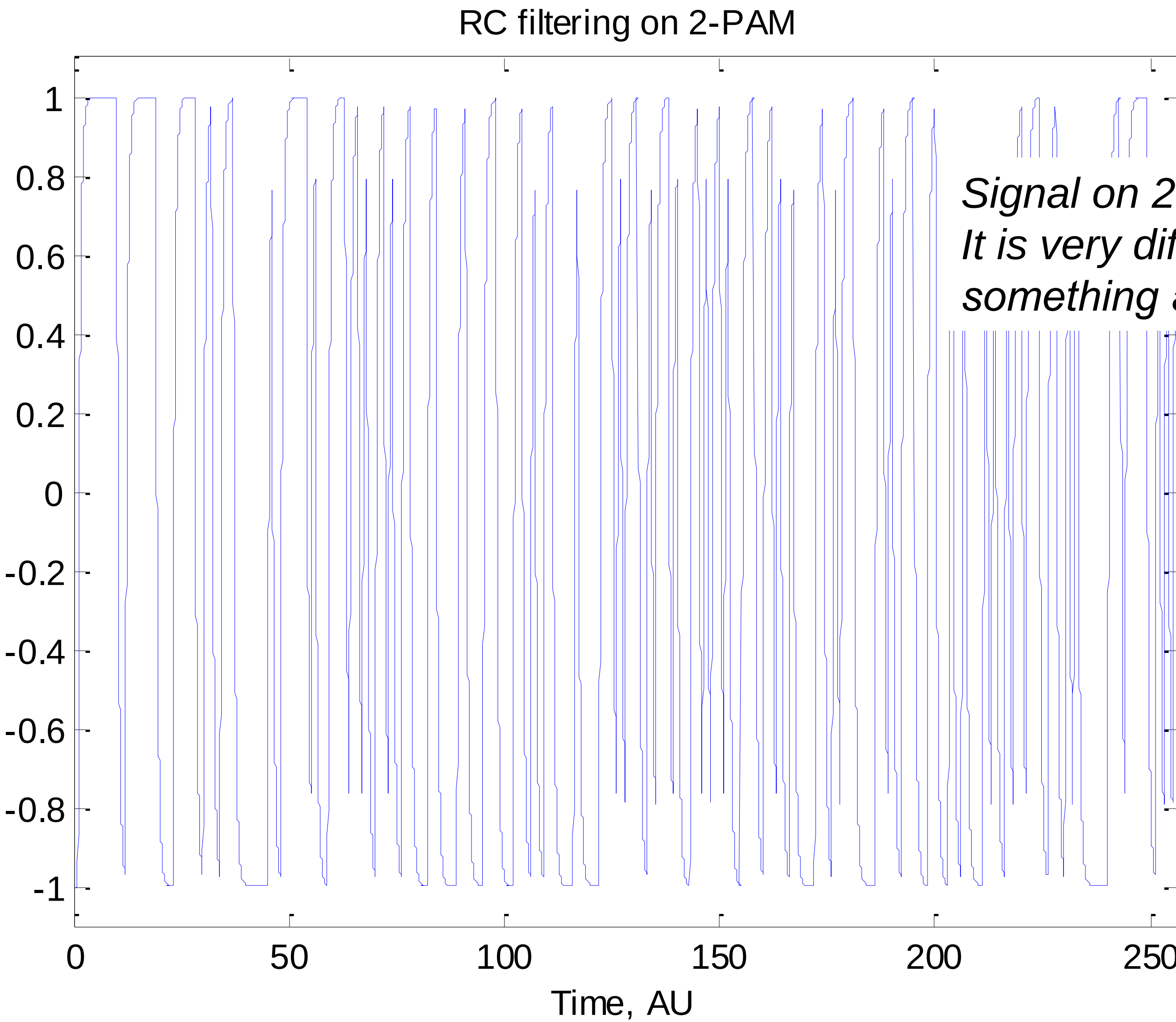
EYE DIAGRAM



Eye diagram

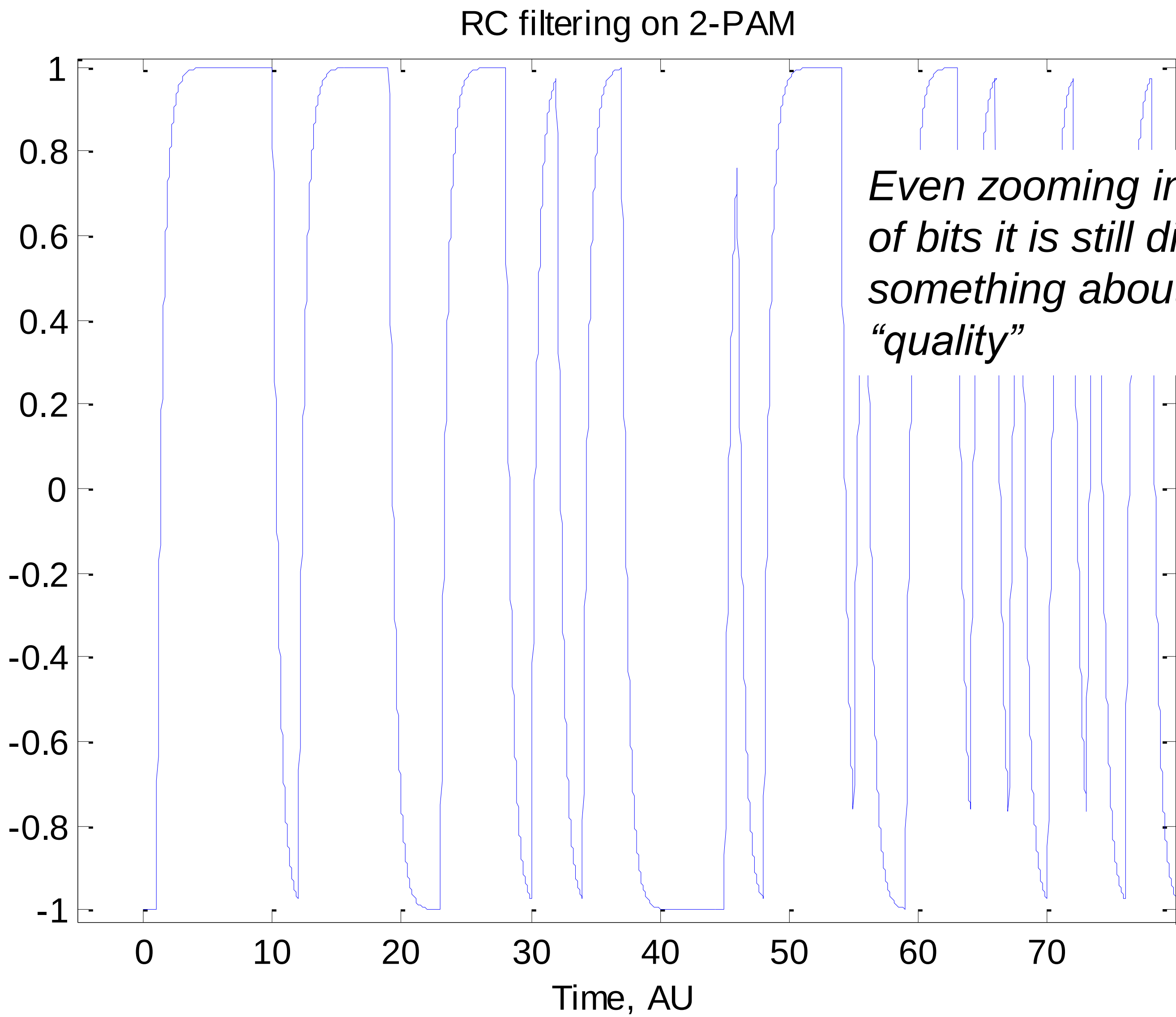
This is a typical representation for data transmission signals

RC-filtered 2-PAM: received signal, no noise



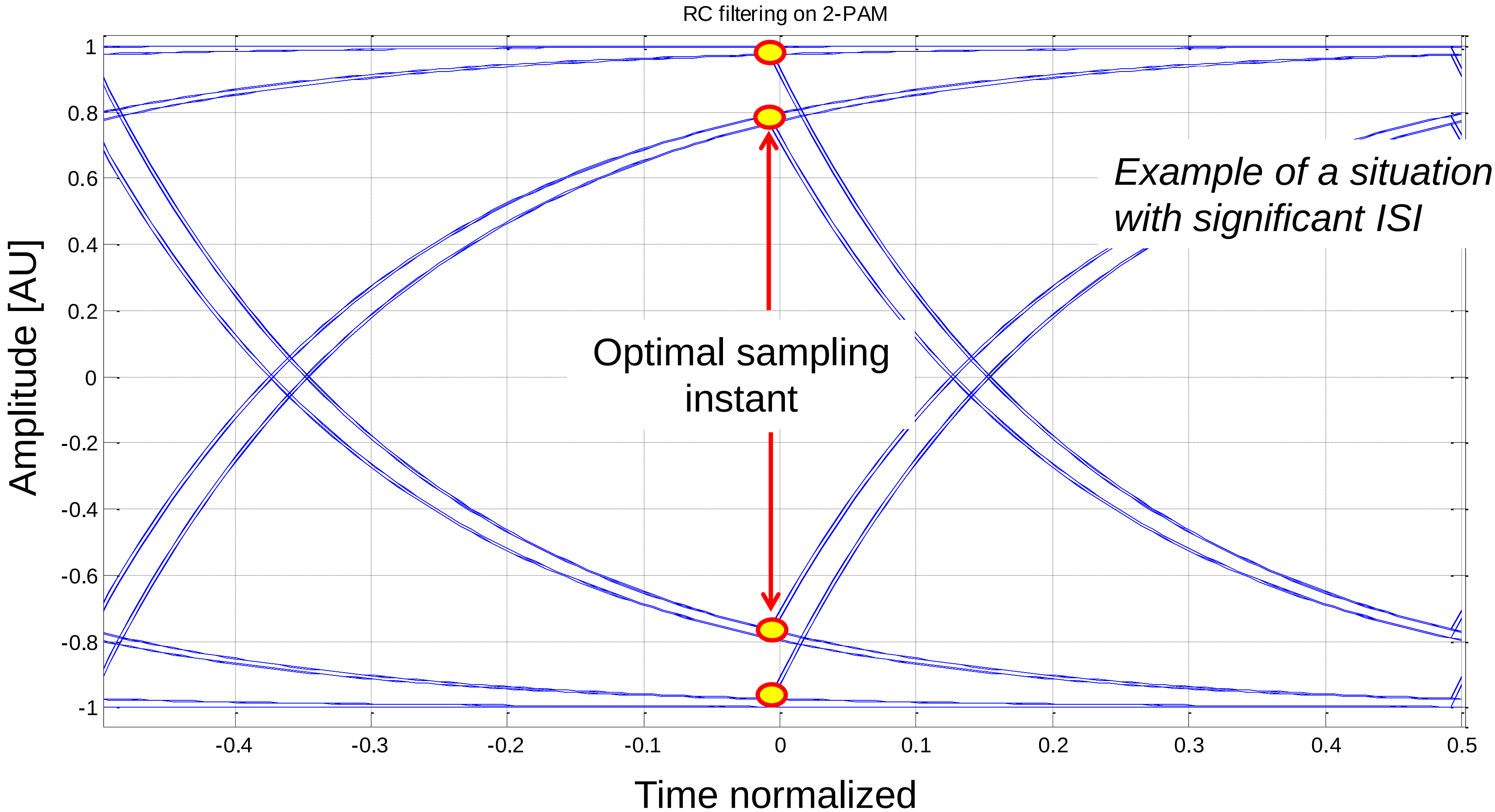
Signal on 256 bit
It is very difficult to say something about its "quality"

RC-filtered 2-PAM: received signal, no noise



Even zooming in on a few tens of bits it is still difficult to tell something about the signal "quality"

RC-filtered 2-PAM: eye diagram



EYE DIAGRAM - I

- How to get an eyediagram in Matlab?

EYEDIAGRAM(SIGNAL,2*SpS,2*SpS)

- Be careful!
  - Eyediagram must be launched over a signal with a maximum of about 10^4 samples otherwise your PC will get stuck

EYE DIAGRAM - II

- How to get an eyediagram in Matlab?

  EYEDIAGRAM(SIGNAL,2*SpS,2*SpS)

- Be careful!
  - Eyediagram must be launched over a signal with a maximum of about 10^4 samples otherwise your PC will get stuck