CAIRO UNIVERSITY
FACULTY OF ENGINEERING

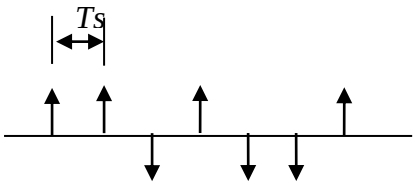
DEPT OF ELECTRONICS AND ELECTRICAL COMMUNICATIONS
3RD YEAR

Digital Communications

Project # 2 (5 points)
April, 2013

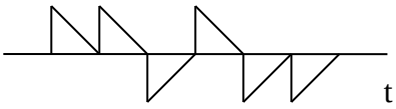
Project 2
Matched Filters, Correlators, ISI, and raised cosine filters

The PAM signals can be viewed as a result of convolution between impulses located every T_s and a pulse shaping function $p(t)$, where T_s is the symbol duration, the symbol rate is $R_s=1/T_s$.



Train of impulses representing the amplitudes of the transmitted pulses

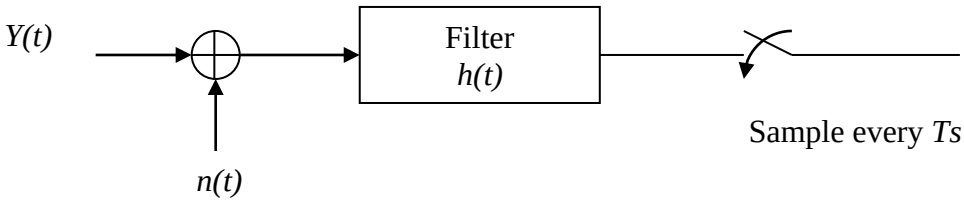
P(t): pulse shaping function



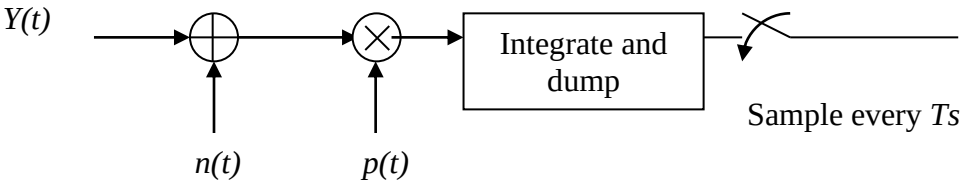
Y (t): Transmitted signal

For binary polar signaling, the impulses take one of two values : +1, -1

The resultant waveform is transmitted over an AWGN channel. Noise is added at the receiver front end, the PSD of the noise is No/2. At the receiver, it is required to use a filter that maximizes the SNR at the sampling instants. It can be proved that the matched filter $h(t)=p(T_s-t)$ is the optimum filter



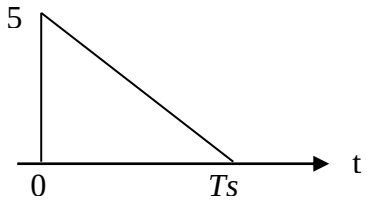
The correlator can also be used instead of the matched filter, the correlator block diagram is as follows:



Simulation procedure:

1- Matched filters and correlators in noise free environment

Consider a system using the following pulse shaping function:

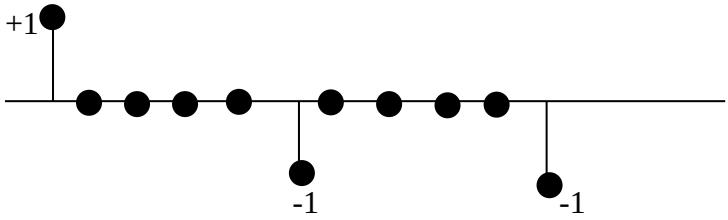


The system uses binary PAM signaling (+1,-1). The symbol duration *Ts= 1 sec*. To simulate the system, it is required to generate *10* random binary bits, convert the logic 1 to +1 and logic 0 to -1. To simulate the above systems, things should be discrete. It is required to represent the given pulse shaping function by 5 samples, equally spaced, so the difference between samples will be 1/5=200 *ms.*

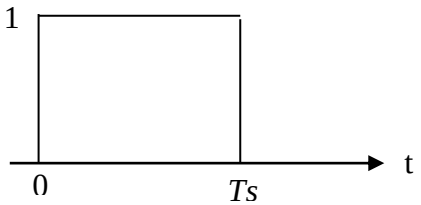
In Matlab, samples from the pulse are used. At the same time it is useful to use a normalized pulse, i.e., energy of the pulse should be unity. In Matlab, use the following pulse

```
p=[5 4 3 2 1]/sqrt(55);
```

a. Generate an array consisting of 10 bits
b. Convert the bit stream to +1's, -1's
c. Generate a signal consisting of impulses every *Ts*, the value of impulse equals to +1 or -1, depending on the corresponding bit. The signal should be samples every 200 *ms.* That means the number of zeros in the resulting array between non-zero samples is 4. In Matlab you may use the command **`upsample`**.



d. Convolve the above sequence with the discrete pulse shaping function to generate the signal at the output of the transmitter y[n]. Use the **`conv`** command in MATLAB
e. The sequence y[n] is to be filtered by one of the following filters:
   i. A filter matched to p[n]. (you can use the command **`fliplr`** in Matlab.)
   ii. The following filter (<u>after sampling and energy normalization</u>).



Requirement 1:

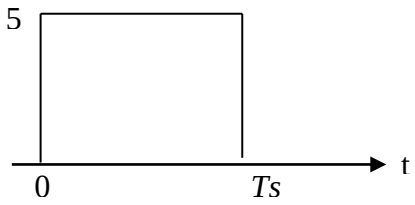
a) Draw the output of both filters (in (e) above) on two subplots in the same figure using two different colors, assuming a noise free system. Compare between the outputs of the filters at the sampling instants.
b) Draw the output of the matched filter and the output of a correlator to p[n] on the same plot with two different colors.

2- Noise analysis:

a- Repeat a, b, and c from 1 above but generate 10000 bits instead of 10 bits
b- Generate a unity variance, zero mean additive white Gaussian noise signal with the same size as y[n]. (in Matlab use the command **`randn`**)
c- Scale the noise sequence to have variance = N0/2 by multiplying the sequence by sqrt(N0/2).
d- Add the noise to the transmitted sequence y[n], v[n]=y[n]+n[n]
Filter v[n] using a filter matched to p[n]. Sample the result every *Ts (5 samples)*. The receiver will generate an array consisting of 10000 samples, calculate the probability of error.
e- It is required to change N0 "step 3" so that Eb/N0 changes from -2dB to 5 dB in 1dB steps, each time calculate the BER.

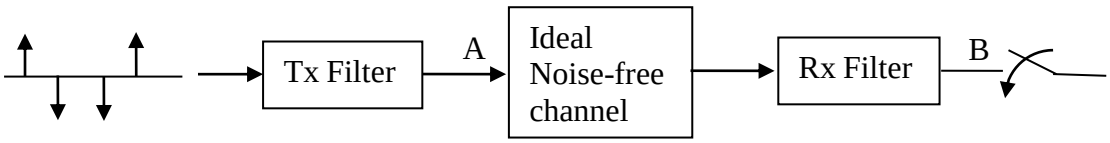
Requirement 2:

Plot the BER vs Eb/N0 in both cases of using a matched filter at the receiver and using the filter with the following response (after sampling). Plot both BER curves on the same graph along with the theoretical
BER=0.5 *erfc(sqrt(Eb/No))



3- ISI and raised cosine

A nice way to see the effect of ISI is to draw what is called the eye pattern. The eye pattern is explained in section 4.11 of the text book. Consider the following noise free system.



The transmit and receive filters are square root raised cosine filters (not raised cosine, but square root raised cosine). Ideally, the overall response of the filters at the transmitter and receiver will be a raised cosine filter. In Matlab use the command **`rcosine`** to generate the filter coefficients. As you know, the ideal square root raised cosine filter cannot be used in practice because it has an infinite impulse response. The parameter **`delay`** defines the length of the filter to be used practically while the parameter **`R`** is used to define the rolloff factor.

For the transmit and receive filters consider the following 4 cases

a- R = 0, delay=2
b- R = 0, delay=8
c- R = 1, delay=2
d- R = 1, delay=8

Requirement 3:

For the 4 cases mentioned above, plot the eye pattern for the data length of 100 bits at points A and B. In Matlab, eye pattern can be drawn using the command **`eyediagram.`** Comment on the relation between the sampling instant and the eye opening.