Lab session n°3+ 4
Wireless transmitter and receiver design

Objective:

This mini-project of 2 lab session deals with the design of a whole chain (transmitter, channel and receiver) for wireless communications.

Needed software before starting:

Before starting, make sure that you installed Matlab and the following toolbox:

- Communications
- Signal processing

Matlab will be used in this mini project and thus a report can be generated using Matlab publish tool. The use of variables of type structure is encouraged; it will allow you to represent blocks in your code, and thus have a better organization of your final program. Use Matlab help when necessary to have information on how to use the different functions mentioned below.

Part A: Transmitter design

The aim of this part is to start the realization of a digital communication system, starting with the transmitter (Figure 1) of a handheld unit. The system utilizes an M-ary amplitude modulation (M-ASK) with a symbol rate R_s=3.84 MHz, and a rectangular pulse shaping filter. The operating band of this transmitter is band I (1920 MHz-1980 MHz) defined by the 3GPP standard (UE transmit, Base station receive). The carrier frequency f_c must then be adjustable to take values inside this band. For the final project the same system will be implemented but with different modulation standard.

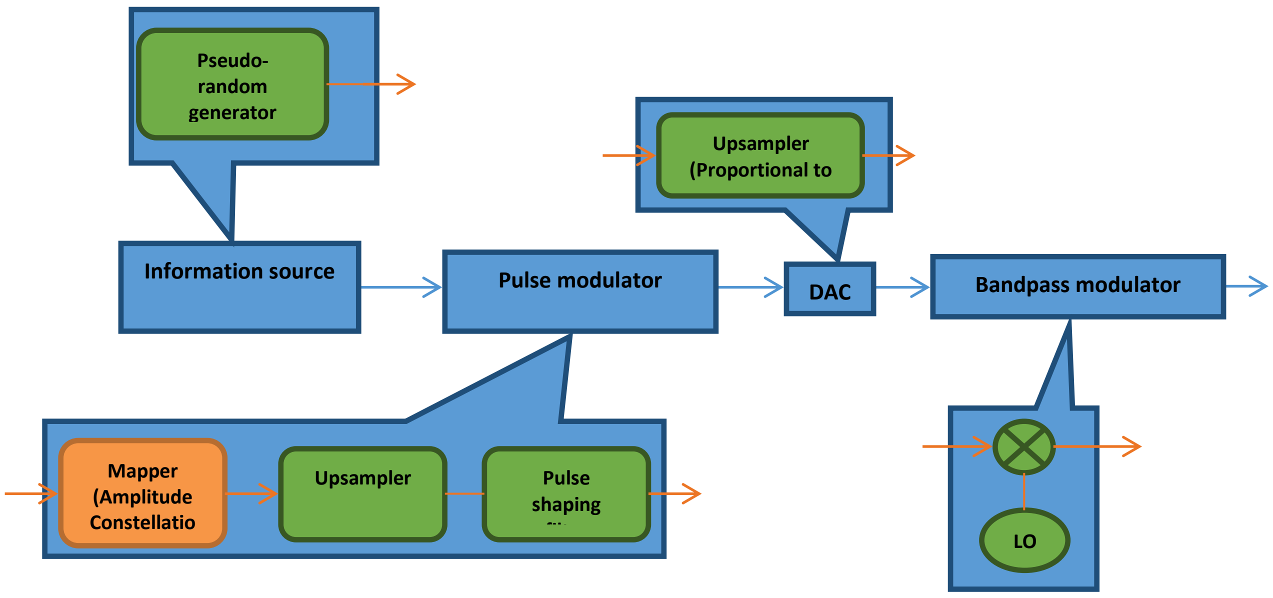

Figure 1. Transmitter

You should visualize the generated signals at the output of each block in time and frequency domain. For this, an appropriate time and frequency axis has to be defined. You can use any function in Matlab to compute the spectrum of a signal.

1. <u>Information source :</u>

To model the random information source, we will use a pseudo-random integer generator substituting the formatting block that may be included later. We want to implement an M-ary system, and thus there are M possible symbols $s_0, \ s_1, \cdots, s_{M-1}$. The integers generated (using the randi function of Matlab) represent symbol numbers (0,1,…, M-1). The randi() function can generate random integers in an arbitrary interval, with a uniform distribution. For our simulation, we want to generate a number, NbSymb, of symbols (e.g., NbSymb=500).

2. <u>Pulse modulator:</u>

As shown in figure 1, the pulse modulator is designed with three sub-blocks, the mapper, the upsampler and the pulse shaping filter.

a. *<u>Mapper :</u>*

The role of the mapper is to attribute predefined amplitude for each symbol. To determine the constellation of this modulation (possible amplitude values), we define a parameter V_d (units Volts) which is equal to the voltage difference between two consecutive symbol amplitudes. So we must first determine the constellation of the M-ASK modulation using this parameter. The output signal of the Mapper, is an impulse train with a period equal to symbol time T_s.

b. *<u>Upsampler:</u>*

The Upsampler increases the sampling frequency of this signal from R_s up to the sampling rate of the digital pulse shaping filter. This upsampler defines the number of samples per symbol, it is usually chosen as a power of 2 (2^n samples per symbol). To generate the output signal of this block we can use the ***upsample*** function of Matlab.

c. *<u>Pulse shaping filter:</u>*

The third sub-block is the pulse shaping filter, which must be a rectangular pulse. To design this filters, you can use the ***window*** function in Matlab and then add samples at zero to have a complete shape. You should define the number of samples needed for this filter given that the pulse has duration of T_s. To visualize the impulse and frequency responses of this filter, we can use the ***fvtool***.

In order to generate the output signal, we can use the ***conv*** function in Matlab, to compute the convolution between the input signal and the pulse.

Plot on the same figure, with the appropriate time axis for each signal, the output signals of all three blocks. Plot as well the power spectral density of these signals (we can use the psd class and the welch method to compute the power spectral density, create first a spectrum object using spectrum.welch).

3. <u>DAC :</u>

To model the digital to analog conversion after digital filtering, we will use an upsampler with a relatively very high frequency with respect to the sampling rate of the digital filter. The sampling frequency at the output of this block must thus be sufficiently high to correctly digitize the carrier. So we can choose an upsampling factor to have a new sampling frequency multiple of f_c. The function ***interp1*** can be used to upsample and interpolate at the same time.

4. Bandpass modulator:

To model this block, we need to generate a carrier (cosine or sine), with a frequency equal to f_c. This carrier is the output of the local oscillator (LO). Then as shown in figure 1, the passband signal can be generated by simply multiplying the output of the DAC with the carrier.
Plot on the same figure the output signal of this block, and the output signal of the DAC. Plot the PSD of the output signal of this block. Comment the results.

Part B: Receiver design

The aim of The aim of this practice session is to continue our design of a wireless digital communication system by implementing the receiver (Figure 2) of the base station. As defined in the part A, the system utilizes an M-ary amplitude modulation (M-ASK) with a symbol rate R_s=3.84 MHz. Now we consider that the pulse shaping filter is a raised-cosine pulse shaping filter with a roll-off factor $r = 0.22$. The operating band of this receiver is band I (1920 MHz-1980 MHz) defined by the 3GPP standard (UE transmit, Base station receive). The carrier frequency f_c must then be adjustable to take values inside this band.

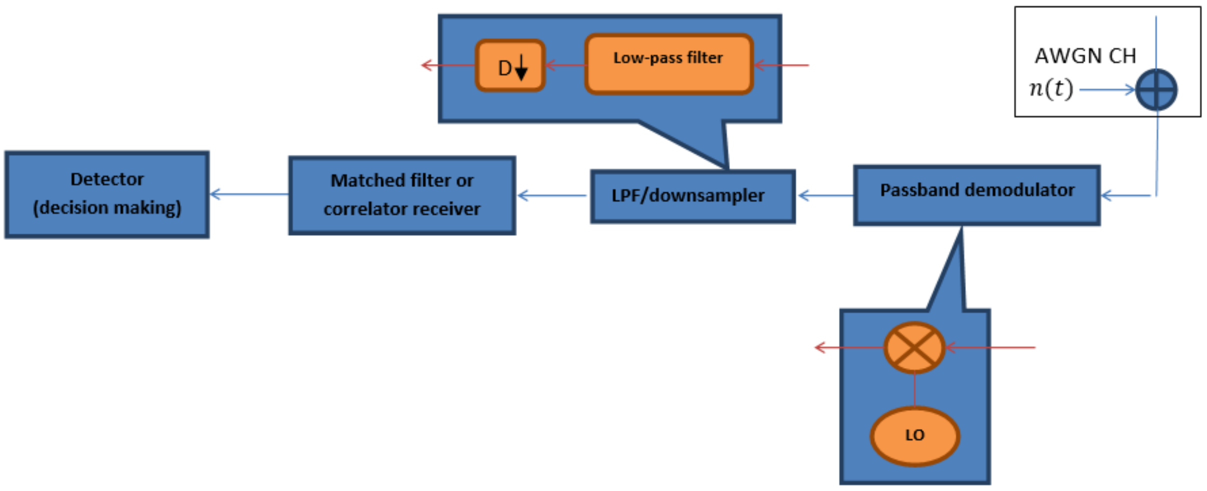

Figure 2. Receiver

1. AWGN channel

We will consider an Additive White Gaussian Noise channel for our analysis. In matlab, you must use the ***awgn***, or ***wgn*** function (see matlab help).

2. Passband demodulator:

The passband demodulator has exactly the same design as the passband modulator, i.e., a local oscillator and a mixer (see part A).

3. Low pass filter/dowsampler:

At the output of the mixer, we will have a signal with high and low frequency components. Give the theoretical explanation by developing the equations resulting from mixer operations on transmitter and receiver side. In order to get the useful signal, a low pass filter must be used. To design this filter, we will use the filter and design analysis tool in Matlab. Type in the command window, *fdatool*, and press enter. Find the lowpass filter parameters needed for our system and generate the filter.

After lowpass filtering we can reduce the sampling frequency to a much slower rate proportional to the bandwidth of the output signal.

4. Matched filter (optimal receiver):

Implement the receiving operation for this system.

5. Detector:

The detector distinguishes between the different possible symbols defined by the modulation scheme. You can compare the results with the ideal sequence on the transmitter side.