DIGITAL FM RECEIVER

SPECIFICATION AND SIMULATION OF DIGITAL SYSTEMS

CARLO CARAMIA
RICCARDO IENNACO
MARCO MANINO
SIMONE MARCHISIO
NICOLÒ MAUNERO

Summary

1. Introduction

The "FM receiver" project is a digital frequency modulated receiver, designed using VHDL. The target board is Zybo of which only the integrated FPGA and the analog to digital(XADC) converter was used.

During the implementation we tried to make the code a bit more flexible using generics to specify the width of each signal as well as other characteristics. In the case of Zybo both input and output signals are 12 bit wide and sampling frequency is 1MHz. The main component is the demodulator, which takes input samples from the ADC and it outputs a demodulated signal. Then the low-pass filter removes the noise generated by the digital elaboration. Eventually the output can be sent to a digital-to-analog converter and played with an actuator or else can be stored.

As described later, we evaluated the behavior of two different internal designs for the demodulator: the first tries to emulate an analog phase locked loop: it uses a quadrature of the input signal to obtain the phase; the second uses a different mechanism based on an exor and an integrator to demodulate.

However, this design alone cannot demodulate real-life FM radio signals due to the limited maximum clock frequency of the XADC that is 1 Mhz making it impossible to effectively sample a signal with a frequency greater than 500 kHz.

To overcome this limitation, it is possible to use an analog or digital mixer to shift the frequency of the signal to a (fixed) lower carrier frequency (Intermediate Frequency): this is the solution used in most of analog radio receivers.
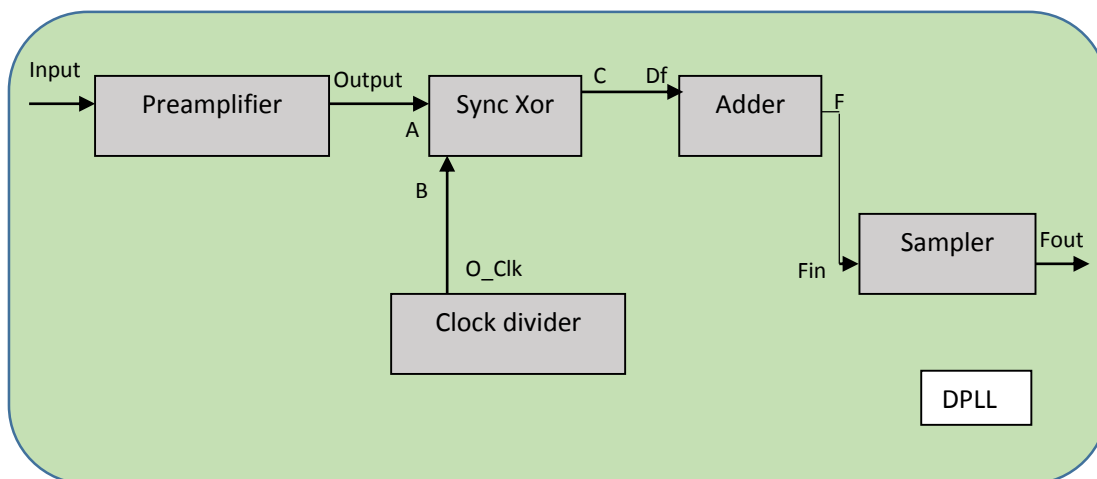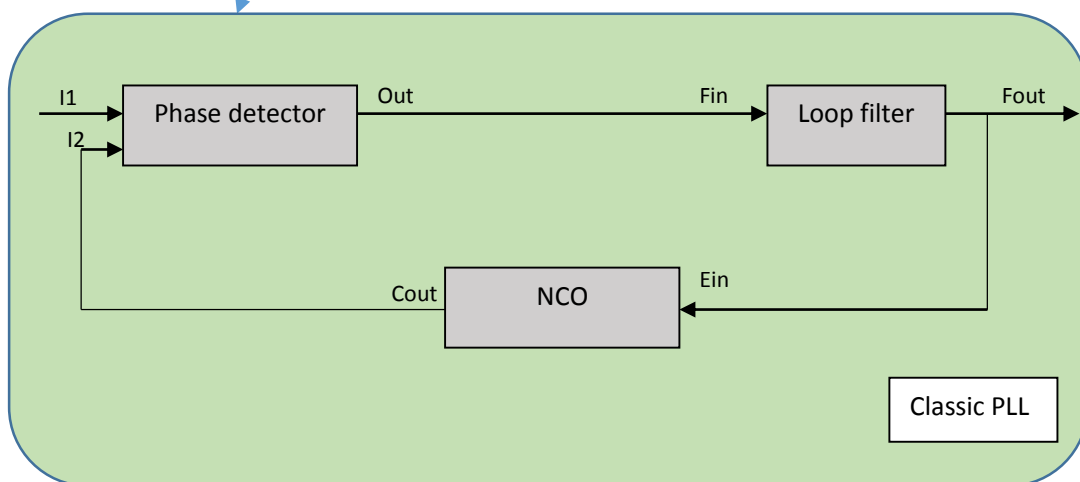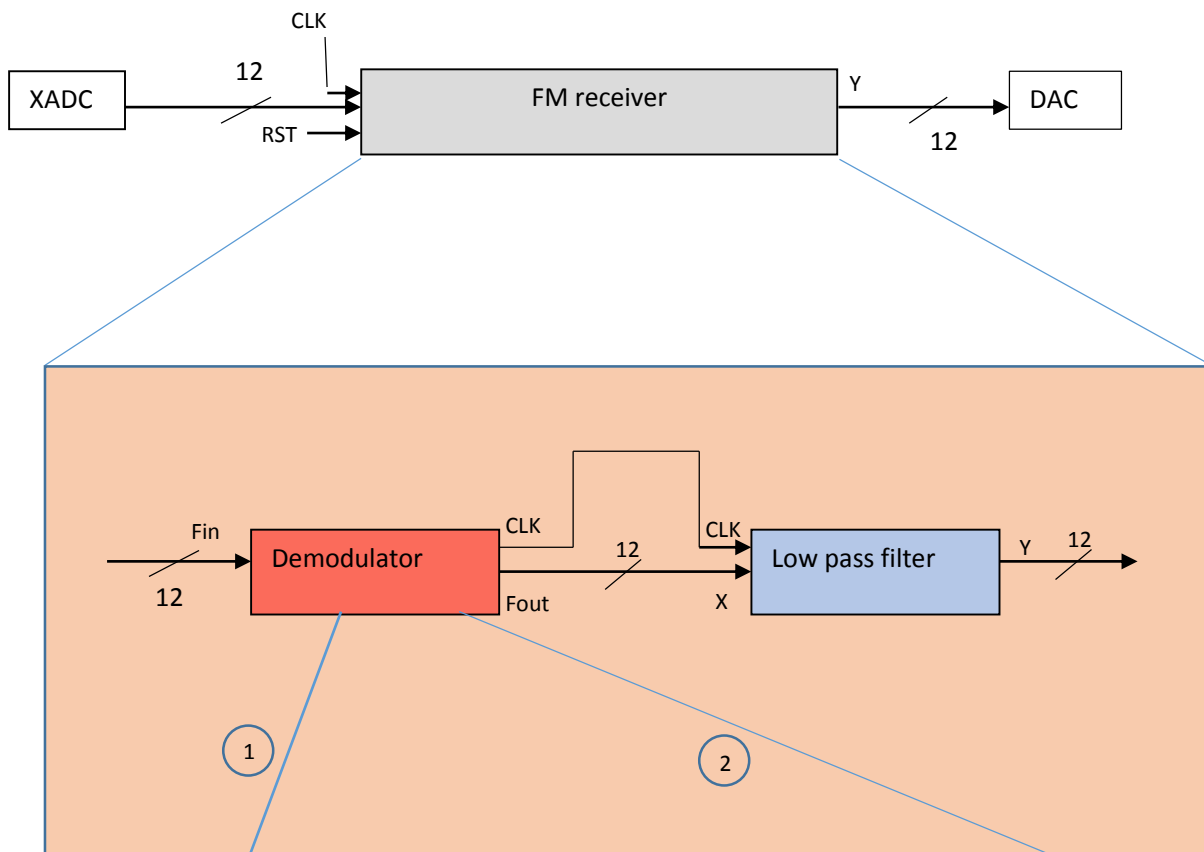
2. Demodulator

This entity, internally described using structural description, represents the phase locked loop (pll) used to demodulate the signal.
Because of the two different approaches we found to obtain the result, there are two different possible architectures: the first (Classic pll) uses feedback mechanism, while the other (the Dpll) doesn't need it.

3. Dpll (Digital PLL)

3.1 1-bit preamplifier with hysteresis

This component receives samples from the external adc, and every clock cycle simply outputs a bit, representing the sign of the sample. This can be considered as a really high gain saturated amplifier. Internally, the component uses hysteresis to partially eliminate the noise. The hysteresis threshold can be modified through generics.
The behavior is simple: each sample can be either positive or negative and the output can only be 0 or 1. So there are four possible cases, and four "if" conditions.
An asyncronous reset strategy was used, so if the reset signal is high, the output is forced to be 0 for every input value.

3.2 Clock divider

This component is used to generate a square wave, needed to demodulate the signal. Its simple implementation uses a counter to decide when to switch thus the frequency of the system clock is divided by this factor. It is possible to decide this value at compile time through generics as well as the width of the counter. In order to avoid misbehaviors, a sanity check is introduced to make shure that the width of the counter is high enough to contain the dividing factor.

3.3 Synchronous XOR

This is simply a xor gate syncronized with the clock.

3.4 Adder

The purpose of this component is to demodulate a Pulse Width Modulated (PWM) signal.
It takes in input the signal come from the XOR and outputs a demodulated signal to the Sampler.
The output is obtained by adding or subtracting a certain quantity based on the sign of the signal come from the XOR (i.e. integrating the signal).

3.5 Sampler

--da completare

4. Classic PLL

4.1 Phase detector

--da aggiungere

4.2 Loop filter

This component a low pass filter, takes in input a 12 bits signal from the Phase Detector and outputs a 12 bits filtered signal for the NCO.

Its pourpose is to remove the high frequency components, generated from the quadrature.

We decide to implement this component as a Moving Average Filter: it computes the average of N signals, and the value of N depends on what is the value of frequency we want to cut off. In our case N is equal to 4.
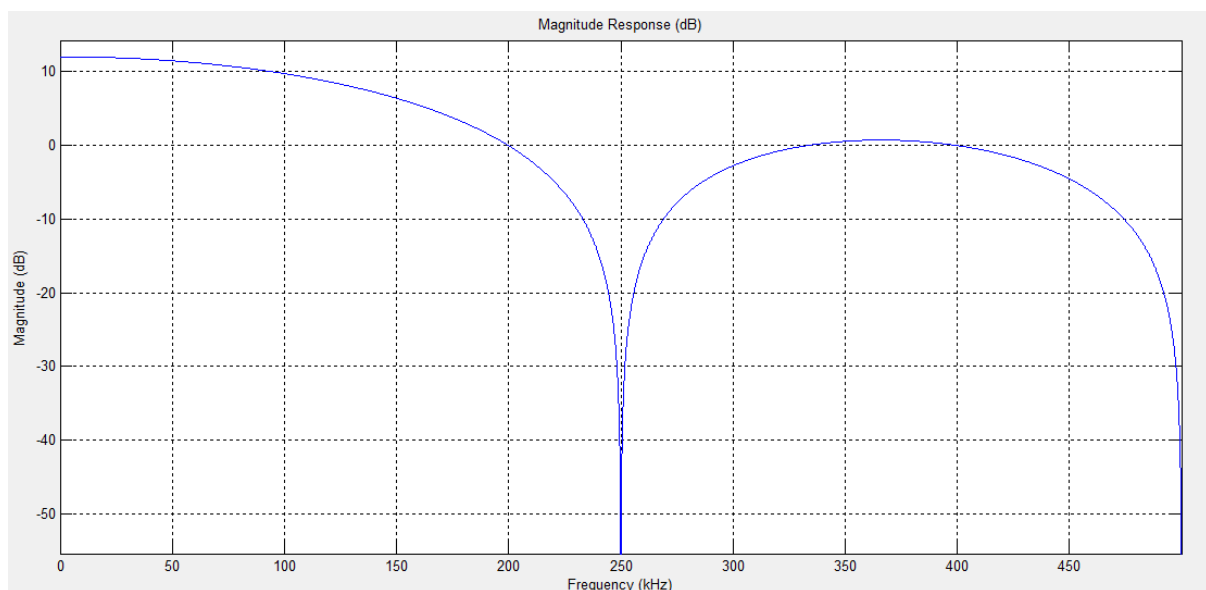
As we can see in the image above, obtained using Matlab as simulator, by computing the average of four signals we obtain a complete rejection of the carrier frequency. The graph clearly shows that in order to obtain a rejection of at least -20 dB, deviation should be less than 5.5 kHz.

4.3 Numerical controlled oscillator (NCO)

A Numerical Controlled Oscillator is a device that produces sample of a cosine wave at a given frequency (the carrier frequency) in order to keep the PLL in lock. For it to work a shift phase of 90 degrees is needed, so when the phase of the input signal changes, that change should be followed by the NCO, too. This i acheived through a negative feedback loop that tries to minimase the phase difference.

To avoid being too sensitive to noise (that can cause the loop to unlock and being instable), a scaling factor for the input phase (alpha) is defined in the range (0,1]. To avoid complex computations and simulations, a sperimental approach has been used to find the value used in the design. Besides, to avoid doing a slow and expensive operation like division, in the design this is implemented as a simple right arithmetic shift.

The entity implemented is basically a counter that increases the count of a certain amount each clock cycle: this amount can be specified through a generic and thus the carrier frequency can be decided; however this can be done at compile time only, making this design not user-tunable. To obtain such a feature, a mixer and a user-tunable oscilaltor can be added to shift the signal at an intermediate frequency.

Since computing the cosine, given a phase, involves complex computations, a rom is used and the value stored in the counter is used to index this memory. In each cell a sample is stored: a function is used to populate this rom without hardcoding any value in it and this way making it possible to expose generics in this entity, too.

5. Low pass filter

This component is a n-bits third order low pass filter and it is the last stage of the fm receiver. The filter takes as input the signal coming from the demodulator. The filter is designed using a Butterworth filter model whose coefficients were computed with Matlab using 15 KHz as cut-off frequency.
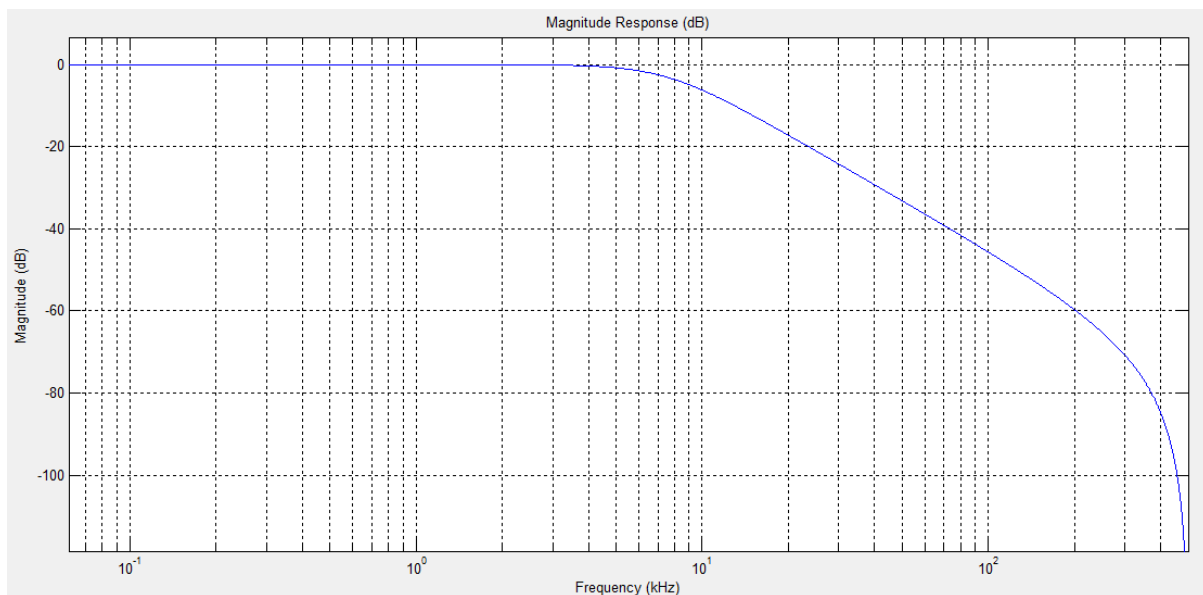
The filter is a "Direct Form II Transposed" implementation of the standard difference equation:

a(1)*y(n) = b(1)*x(n) + b(2)*x(n-1) + ... + b(nb+1)*x(n-nb) - a(2)*y(n-1) - ... - a(na+1)*y(n-na)

 b = 0.0005   0.0011   0.0005

 a = 1.0000   -1.9334   0.9355

a(1) is 1 because the filter was normalized and his gain is one and this way can be omitted.

6. FM receiver

This entity, internally described using structural description, represents the top level desing of the circuit, that takes input samples and outputs demodulated samples
The architecture is simple, because is just the combination of a demodulator, described above, and the final low pass filter.

7. Simulations

--da completare