**Matlab Funtions for Baseband Signals**

Jose Alejandro Galaviz Aguilar1

1 Tecnologico de Monterrey

galaviz@tec.mx

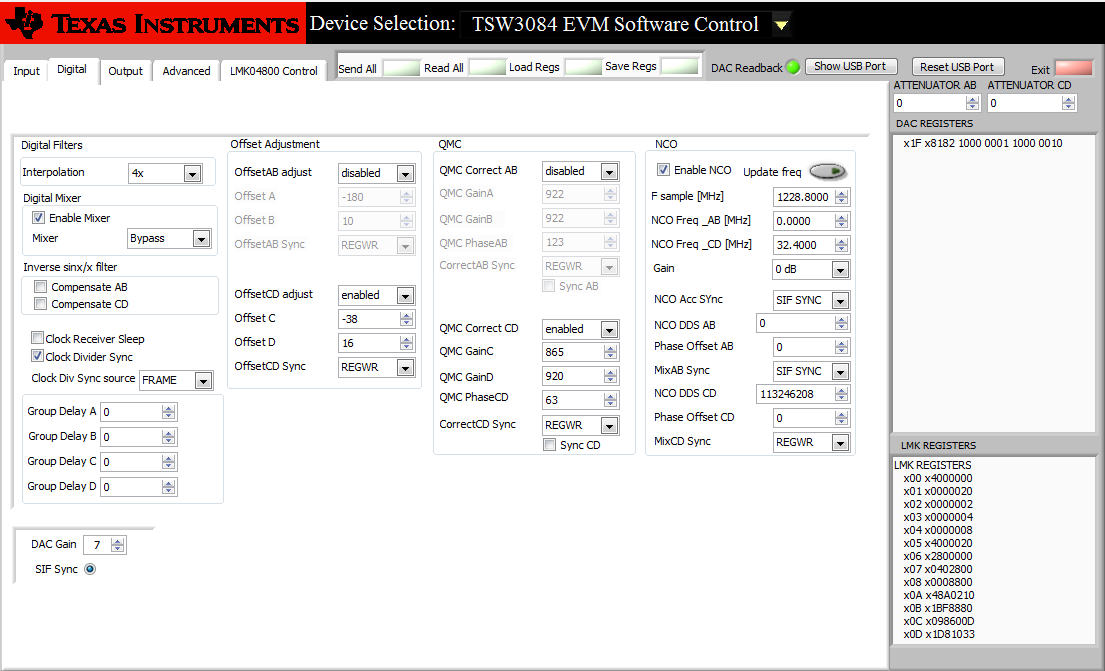
# **Data Signals and FPGA Board Characteristics**

The Figure I denote the experimental system used for performing the PA, characterization, measurements and data acquisition. The FPGA system works with a synchronous clock at 307 MHz, this clock also is distributed to IQ modulators at the transmitter and share a clock with the receiver.

# 

Figure I. Experimental testbed

To setup the system is necessary a calibration of the system. The signals using the GUI for setup the initialization and adjusting the values for a good IQ imbalance and power calibration for PA inputs. The sampling clock as show in the Figure 2 (red dashed rectangle) is set at GHz with the main clock according the mixing process is used a factor-4 filter interpolation to up-sampling the signal at  **GHz** in the up conversion, this is .



The data initially was generated with ADS LTE block set, so for the initially formatting the data from the output files generated from the ADS circuit is adequate to Matlab windowing data to place in FPGA using the following code, you may find the code at the Github link below with the name **ADStoMatlab.m.**

|  |
| --- |
| clear all;  close all;  clc;  I = importdata('S1Re'); %This data came from ADS Software  Q = importdata('S1Im'); %This data came from ADS Software  signalInput = complex( I , Q ) ;  %FIR arbitrary shape lowpass filter  f = [0 0.25 0.26 1] ; % frequency breakpoints  m = [1 1 0 0] ; % magnitude breakpoints  b = fir2(70,f,m) ; % Frequency sampling-based 70th-order FIR filter design  % Y = FILTER(B,A,X) filters  % the data in X with the filter described by vectors A and B to create the filtered data Y  % y = interp(x,4) Interpolate a signal by a factor of four  I = filter( b , 1 , interp( I , 16 ) ) ;  Q = filter( b , 1 , interp( Q , 16 ) ) ;  I = filter( b , 1 , I ) ;  Q = filter( b , 1 , Q ) ;  I = filter( b , 1 , I ) ;  Q = filter( b , 1 , Q ) ;  x = complex( I , Q ) ;  xup = resample( x , 5 , 4 ) ; %upsample 245.76\*5/4 = 307.2 % actually works well with 307 MHz  %Fs = 245.76;  Fs = 307.2 ;  freq = linspace( -Fs/2 , Fs/2 , length( x ) ) ;  frequp = linspace( -Fs/2 , Fs/2 , length( xup ) ) ;  figure (1) %after upsamlping  plot(frequp , 20\*log10(abs(fftshift(fft(xup)))))  signalInput=xup;  %save ('LTE1MHz307p2v1.mat', 'signalInput');  %figure (2) % Fs 245.76  %plot(freq, 20\*log10(abs(fftshift(fft(x))))) |

You may find all the codes at the following [link](https://github.com/Galaviz1/matlab_codes/tree/master/ADS_SignalFormat). The output data used in Matlab in this example is the save with the name 'LTE1MHz307p2v1.mat' as you can notice in the code above. The signal 'LTE1MHz307p2v1.mat is used and formatted with the code **SignalFormat.m.**