Advanced Engineering Forum *Online: 2013-06-27*
ISSN: 2234-991X, Vols. 8-9, pp 500-507
doi:10.4028/www.scientific.net/AEF.8-9.500
© 2013 The Author(s). Published by Trans Tech Publications Ltd, Switzerland.

Matlab Toolbox for RF Receiver Modeling

Botond Sandor KIREI[1, a], Marius Gheorghe NEAG[1,b]
and Marina Dana TOPA[1,c]

[1]Str. Memorandumului 17, Cluj Napoca, 400117, Romania

[a]botond.kirei@bel.utcluj.ro (corresponding author), [b]marius.neag@bel.utcluj.ro,
[c]marina.topa@bel.utcluj.ro

Keywords: direct conversion receiver, modeling of RF/analog blocks, thermal noise, nonlinearity, I/Q imbalance, phase noise, carrier frequency offset.

Abstract. In this paper a novel RF receiver modeling approach is presented. The novelty consists in the usage of the object oriented programming instead of the usual imperative programming. Classes were defined for entities as signal, non-ideality and RF/analog block, in order to achieve a basic RF receiver model. The main circuit non-idealities were identified, non-ideality parameters and their analytical models were encapsulated into classes. The effectiveness of the proposed modeling approach was demonstrated by developing in MATLAB a fairly complete model of a direct conversion receiver.

Introduction

The latest trend in wireless receiver development is the co-designing of the RF/analog front-end and the (usually digital) algorithms used to compensate the RF tuner non-idealities. Modern communication standards push ever higher both the carrier frequency and the channel bandwidth; as a result, the requirements imposed on the RF/analog blocks are more difficult to meet. In principle, this is (at least partially) mitigated by the usage of high-performance digital signal processing (DSP), including non-ideality compensation algorithms and error corrections. However, in order to assess the combined effect of RF/ analog block non-idealities and determine whether or not a given DSP is able to handle it, one needs very detailed modeling of both the RF & analog front-end and the DSP.

When modeling a modern communication standard transmission chain the imperative programming may not be the most suitable approach as the code may became unstructured, unreadable and hard to reuse. Following on this observation we propose the usage of object oriented programming (OOP) in order to achieve not only the accurate modeling of block-level non-idealities but also to enable a framework for transmission chain modeling.

This paper is organized as follows: the next section briefly introduces the direct conversion receiver (DCR) architecture, highlighting the main blocks within its structure; the third section comprises a brief description of the proposed toolbox classes; the fourth section describes the DCR model developed with this toolbox; finally, conclusions are drawn and a few possible directions for further developments are discussed.

The Direct Conversion Receiver

Let us first review the main blocks of the DCR architecture [[1]], with the aim of identifying the main parameters/functions/limitations of the RF/analog blocks within the signal path of the receiver.

The classic DCR receiver structure is depicted in Fig. 1. The antenna collects the RF signal, which is passed through a band pass filter (BPF) that selects the band used by the communication standard. The filtered RF signal is amplified by the low noise amplifier (LNA). The amplified signal is multiplied with a complex carrier generated by the local oscillator, based on a voltage controlled oscillator (VCO). This multiplication is performed by a quadrature mixer (MIX); thus, the RF signal is down-converted into a complex base-band signal, further processed by the in-phase

This article is an open access article under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0)

and in-quadrature (I/Q) signal paths: the low pass filtering (LPF) is needed to remove unwanted spectral components – in particular the neighboring channels. The automatic gain control (AGC) block amplifies the filtered signal, now comprising mainly the wanted channel, to the full input range of the next stage, the analog to digital converter (ADC).

The main non-idealities affecting such a receiver are: the non-linearity and noise of each block within the signal paths, the phase noise of the local oscillator, the I/Q imbalance due to phase and gain errors in the complex carrier and mismatches between the I and Q baseband paths, the aperture and clock jitter in the ADC [[2]].

Defined Classes

General Classes. Each class in the proposed toolbox was derived from the same parent class, called the BaseClass and having the diagram presented in Table 1. The aim of this class is to provide general functionalities that could be used in any derived class, for example the read and write of encapsulated properties.

The LNA, BPF, MIX, VCO and ADC are *RF/analog blocks* and these are communicating with the use of *signals*. Abstraction of the term *signal* leads to a class that includes properties (samples, sampling frequency, and center frequency) and methods (signal plot, arithmetic operations). The diagram of SignalClass is presented in Table 2.

Abstraction of *RF/analog blocks* leads to the definition of *BlockClass*. This class is derived from *BaseClass* too. The common *RF/analog block* parameters are the input impedance (Z_{in}) and output impedance (Z_{out}). *BlockClass* has a dedicated procedure, called *add*, to insert objects (class instances) into one of its field called *architecture*. Another important method is called *process*. This method can be used to implement the functionality of the blocks (amplification, filtering, mixing). The *BlockClass* is summarized in Table 3.

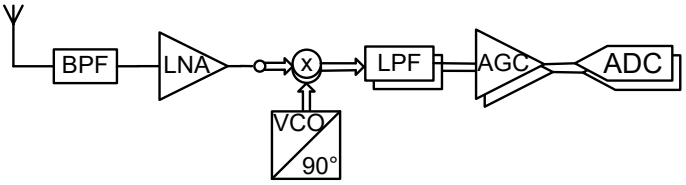


Fig. 1 – Basic DCR architecture.

Table 1

BaseClass diagram.

Field name		Description
Methods	**Write**(*property*,*value*)	writes *value* in field named *property*;
	value =**Read**(*property*)	returnes the value from field *property*

Table 2

SignalClass diagram.

SignalClass inherits all filed of *BaseClass*		
Field name		Description
Properties	**Samples**	samples of the signal, can be a scalar, vector or matrix
	Fs	scalar, sampling frequency
	Fc	scalar, center frequency
Methods	**object=Constructor (***Samples, Fs, Fc***)**	object constructor; the first argument contains the samples of the signal; the second and third argument stands for the sampling frequency and center frequency
	object= Constructor(*signal***)**	OOP allows polymorphism of functions; thus other constructor functions can be defined; this form takes as input argument an instantiation of *SignalClass*
	Plot	Plotting the contained values.

Table 3

BlockClass diagram.

BlockClass inherits all the fields of *BaseClass*		
Field name		Description
Properties	**architecture**	Field **architecture** is a structure that contains objects of *BlockClass*. With use of the procedure **add** (de below) an abject can be added to the **architecture** field
	signal	the return value is saved into field signal
	Zin	input impedance
	Zout	output impedance
	Constructor(*input*)	*input* is an object of *SignalClass*, mainly used to initialize the fields inherited from *SignalClass* (*Samples, Fs* and *Fc*)
Methods	**add**(*block*)	*block* is an object of *BlockClass* and it will be added to the **architecture** field
	result=**process**(*signal*)	signal is an object of *SignalClass*. The return value *result* is computed in the next way: the process function of objects inserted in field **architecture** are invited in the order they were added in the architecture passing the input signal from one to other.

Nonlinearity class. Widely known parameters of the non-linearity found in the literature are: power gain, power at saturation P_{SAT}, 1dB output compression point – *OP1*, third-order intercept point *IIP3* [[3]]. The effect of non-linearity can be described as:

$$y(t) = \sum_{k=1}^{K} a_k x^k(t), \tag{1}$$

where *x(t)* is the input signal, *y(t)* is the output signal and a_k are real coefficients. Usually the third order component is considered only, so relation (1) becomes:

$$y(t) = x(t)\left[a_1 + \frac{3a_3}{4}x^2(t)\right]. \tag{2}$$

Equation (2) is the starting point for the abstraction of nonlinearity, conducting to *NonliniarityClass* (Table 4.). The coefficients a_1 and a_3 are properties of the class. Equation (2) does not contain the saturation that can occur in circuits, thus further fields for the input and output amplitude limit AI_{sat} and AO_{sat}, are computed using:

$$AI_{sat} = \frac{2}{3}\sqrt{\frac{a_1}{3|a_3|}}, \text{ respectively } AO_{sat} = \frac{2}{3}AI_{sat}. \tag{3}$$

Due to the facilities provided by the OOP it is possible the implement the automatic update of a_1 and a_3 coefficients whenever the property values of AI_{sat} or AO_{sat} are accessed (modified). The processing described by equation (2) is implemented in the method *process*.

Table 4

NonlinearityClass diagram.

NonliniarityClass inherits all the fields of *BaseClass*		
Field name		Description
Properties	**a1**	gain of the first order term
	a3	gain of the third order term
	aisat	maximum input amplitude
	aosat	maximum output amplitude
Methods	*output*=**process(***input***)**	*input* and *output* are instances of *SignalClass*; this member executes the processing in equation (2)
	update*(object)*	this member updates the values **a1** and **a3** of when **aisat** or **aosat** is accessed

Table 5
TermalNoiseClass diagram.

TermalNoiseClass inherits all the fields of *BaseClass*		
Field name		Description
Properties	**K**	Bolzmann's constant $1.38*10^{-23}$ J/K
	R	Resistance
	B	Noise Band
	Temp	Temperature
	NF	Noise Figure
Methods	*output*=**process(***input***)**	*input* and *output* are instances of *SignalClass*; process executes the mathematical model in equation (5)
	update(*object***)**	the function updates the value of **Temp** when **NF** is accessed

Thermal noise class. The electron flow in a resistance *R* operating at the temperature *T* generates a noise voltage; its value, integrated over the frequency range *B*, is given by:

$$e^2 = 4RkTB, \tag{4}$$

where k=1.38*10^{-23} J/K (Boltzmann's constant). Thermal noise is present in each RF/analog block and its effect is described by:

$$y(t) = x(t) + e \cdot n(t), \tag{5}$$

where *n(t)* is a random process with unitary power and normal distribution. This non-ideality is encapsulated in *TermalNoiseClass* (inspect Table 5 for the description of the class' fields). The noise figure (NF) of an RF/analog block expresses the degradation of the input signal to noise ratio (SNR) to output SNR. Thus the equivalent thermal noise (in Kelvin) of an RF/analog block can be computed from its noise figure (in dB):

$$T_e = 290 \cdot \left(10^{NF/10} - 1\right). \tag{6}$$

Phase Noise. The signal generated by the VCOs is not purely sinusoidal in a silicon implementation. A frequency synthesizer comprises a reference oscillator – usually crystal-based - and a phase locked loop, PLL; the PLL is consists of a phase detector, a loop filter, a voltage controlled oscillator and a divider with a given division factor. Even when the PLL is locked at the desired frequency, the inherent noise results in a slight variation of the phase of its output signal compared to the reference signal.

Several models for phase noise can be found in literature; they all agree that the effect of phase noise can be modeled as a phase modulation of the baseband signal:

$$y(t) = x(t)e^{j\Phi(t)}, \tag{7}$$

where *Φ(t)* is the time domain description of the phase noise. Many interpretation exists for *Φ(t)* [[4]].

A special case of the phase noise is the carrier frequency offset. In this case phase noise *Φ(t)* is considered to be a linear function of time *Δft,* where *Δf* is the frequency offset. Thus equation (7) is:

$$y(t) = x(t)e^{j\Delta ft}. \tag{8}$$

Table 6

PhaseNoiseClass diagram.

PhaseNoiseClass inherits all the fields of *BaseClass*		
Field name		Description
Properties	**multiplier**	this field points to a function that computes $e^{\Phi(t)}$
Methods	*output*=**process(***input***)**	*input* and *output* are instances of *SignalClass*; process executes the mathematical model in equation (6)

PhaseNoiseClass (Table 6) provides a mean for the user to give its own function for $\Phi(t)$. This is implemented by property field dedicated for a function handler. Method process will invite the specified function to generate the $e^{j\Phi(t)}$ and multiply the returned value and the input signal.

I/Q Imbalance (Mismatch). The I/Q imbalance is a problem common for both direct conversion and low-IF receivers. The orthogonality between the I/Q paths is easily compromised by the amplitude and phase errors in the complex carrier generated by a quadrature oscillator. It was shown in [[5]] that the I/Q imbalance (in the case of a direct conversion receiver) will cause the superposition of the desired signals conjugate over the desired signal:

$$y(t) = k_1 x(t) + k_2 x^*(t). \tag{9}$$

where $x(t)$ is a complex valued, (representing the in-phase and quadrature signals), * stands for conjugation, and coefficients are given by:

$$k_1 = \frac{1 + ge^{-j\varphi}}{2},\ k_2 = \frac{1 - ge^{j\varphi}}{2} \tag{10}$$

where g is the amplitude and φ is the phase error. I/Q imbalance is measured using the image rejection ratio (IRR):

$$IRR = 20\log\frac{|k_2|}{|k_1|}. \tag{11}$$

Considering the wideband nature of the communications standards used nowadays I/Q imbalance should be considered frequency selective. Frequency selectivity is the result of the many processing stages in the RF receiver. The main contributors to frequency selective (FD) I/Q imbalance are the lowpass filters placed on the I/Q paths for desired band selection. Due to the physical implementation the LPFs' places on the in phase and quadrature paths are different, they have a deviation from their nominal values. Thus FD I/Q imbalance is modeled by:

$$y'(t) = h_i(t) * \mathrm{Re}\left\{k_1 x(t) + k_2 x^*(t)\right\} + h_q(t) * \mathrm{Im}\left\{k_1 x(t) + k_2 x^*(t)\right\}. \tag{12}$$

where h_i and h_q are the deviations from the LPFs nominal value. From (11) one can easily obtain:

$$y'(t) = \left(\frac{h_i + gh_q e^{-j\phi}}{2}\right) * x(t) + \left(\frac{h_i - gh_q e^{j\phi}}{2}\right) * x^*(t). \tag{13}$$

The diagram of the class created for the I/Q imbalance is presented in Table 7.

Table 7

IQImbalClass diagram.

I/QImbalClass inherits all the properties of *BaseClass*		
Field name		Description
Properties	**g**	amplitude error
	phi	phase error
	k1	imbalance parameter k_1
	k2	imbalance parameter k_2
	IRR	image rejection rate
Methods	*output*=**process(***input***)**	*input* and *output* are instances of *SignalClass*; process executes the mathematical model in equation (9)
	update	the update function for **k1**, **k2** and **IRR**; they will be updated whenever the values of **g** or **phi** are accessed

Table 8

RF/analog block parameters

Block	**gain (dB)**	**Noise Figure (dB)**	**Other**
BPF	*-3*	*0*	
LNA	*15*	*1.5*	
MIX	*15*	*7.8*	*IRR =29dB CFO = 1MHz*
LPF	*0*	*6*	$F_{pass}=0.4F_{sample}$
AGC	*50*	*20*	

Simulation Example

Let us consider the DCR architecture depicted in Fig. 2, with the parameters summarized in Table 8 [[6]]. For simplicity we considered all RF/analog block input and output impedances equal to 50 Ohm.One advantage of the OOP approach is a well-defined bounda-ry between the core model and support/add-on features such as graphical user interface. For example, it was an easy task to generate the tree view of the RF receiver architecture depicted in Fig. 2. The top level architecture contains a BPF, LNA, MIX, LPF and AGC objects. Each object is a container of selected non-idealities. E.g. the object MIX contains instances of non-ideality classes (*TermalNoise*, *Nonlinearity*, *IQImbalance* and *CFO*. Beside these non-idealities the MIX contains properties as: *Zin*, *Zout* and *signal*.

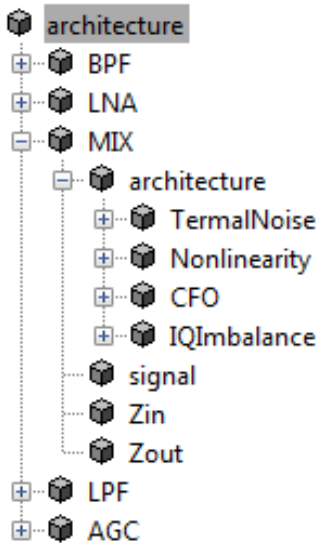
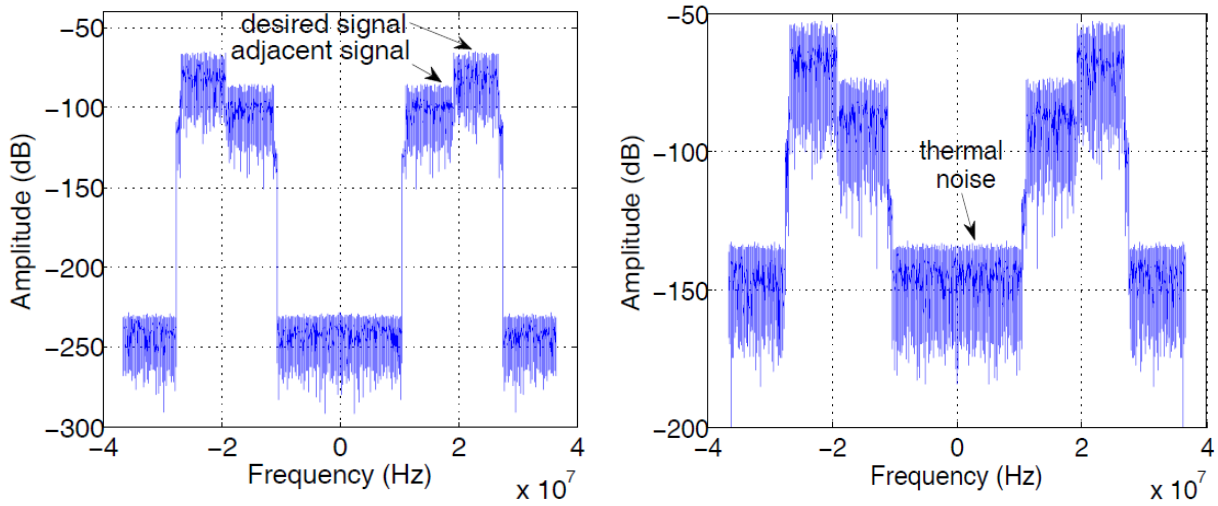


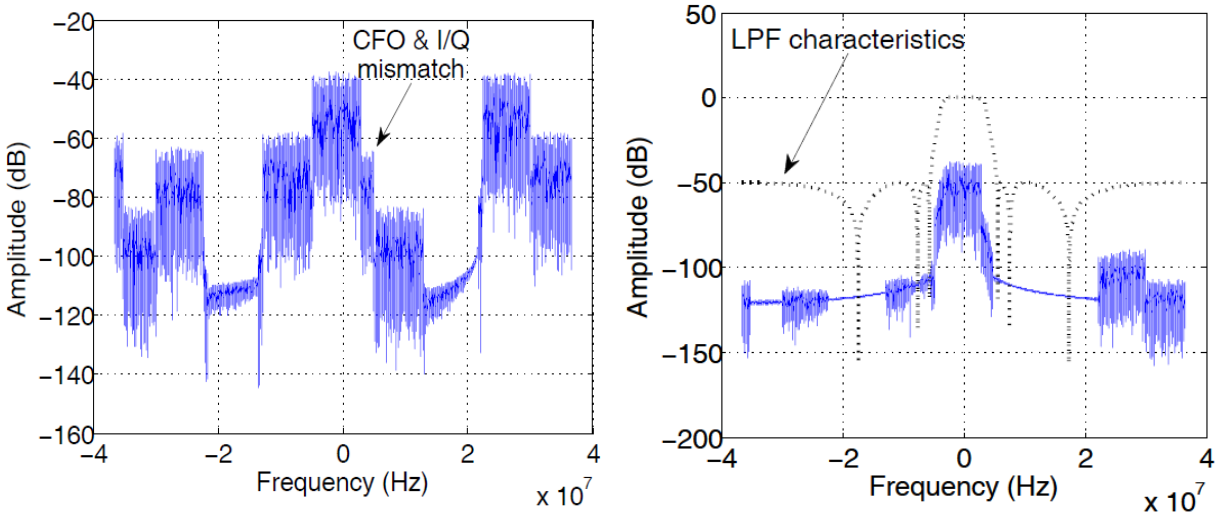
Fig. 2 – Tree view of the DCR container object properties

The input signal, depicted in Fig. 3a, consists of a desired and an adjacent channel, both an 8Mhz wide OFDM signal of N=1075 carriers. The desired and adjacent channel levels are -60dB and -70dB. The input signal undergoes a BPF filtering. This processing step does not modify substantially the spectrum of the desired signal, the gain of the BPF being -3 dB. Then the LNA amplifies the signal with 15dB (Fig 3b). The thermal noise introduced by the LNA is about -140dB.



a) b)

Fig. 3 – a) RF signal collected by the antenna; b) signal spectrum after LNA



a) b)

Fig. 4 – Signal spectrum after a) MIX b) LPF

The amplified signal enters into the down-conversion stage carried out by MIX. CFO and I/Q imbalance are modeled in the mixer. The CFO is set for an exaggerated 1MHz and the IRR for 29 dB, in order to make these two effects visible in Fig. 4a. As one can see on the right side of the desired signal there is a side lobe, which is the combined result of CFO and I/Q imbalance. The MIX contributes with 15dB gain in the overall receiver chain. Fig. 4b is the desired signal spectrum after LPF. The LPF characteristics (6^{th} order Chebyshev type II) are depicted too in the figure. Finally the AGC amplifies the received signal to the ADC input range.

Summary

A MATLAB toolbox for modeling the non-ideality of RF/analog blocks was developed. The novelty of this simulation toolbox is the OOP approach in the creation of the toolbox blocks. This approach makes it easy to encapsulate non-ideality parameters and their effects in one entity. The usage of the toolbox accelerates the creation of RF receiver modeling, thus potentially accelerating the verification of various algorithms employed in wireless communication, such as I/Q imbalance compensation, CFO mitigation, etc. Future work will comprise: completion of the toolbox with harmonic image distortion model and ADC specific non-ideality models (quantization noise, aperture and clock jitter, nonlinearity); the implementation of automated performance

characteristics plotting (signal to noise and distortion ration –SINAD–, difference with respect to reference –DwrtR– [[6]]); the development of a graphical user interface for the construction of RF receiver models.

Acknowledgments

This paper was supported by the project "Development and support of multidisciplinary postdoctoral programmes in major technical areas of national strategy of Research - Development - Innovation" 4D-POSTDOC, contract no. POSDRU/89/1.5/S/52603, project co-funded by the European Social Fund through Sectorial Operational Programme Human Resources Development 2007-2013.

References

[1] Q. Gu, "RF System Design of Transceivers for Wireless Communications", Springer Science + Business Media LLC, New York, 2005

[2] G. Fettweis, M. Lohning, D. Petrovic, M. Windish, P. Zillman and W. Rave, "Dirty RF: A New Paradigm", International Journal of Wireless Information Networks, Vol. 14, No. 2, June 2007

[3] W. F. Egan, "Practical RD System Design", A John Wiley & Sons, Inc., Hoboken, New Jersey, 2003

[4] F. Horlin, A. Bourdoux, "Digital Compensation for Analog Front-Ends", John Wiley & Sons Ltd, Chichester, England, 2008

[5] M. Mailand, R. Richter, And H.-J. Jentschel, "I/Q-Imbalance And Its Compensation For Non-Ideal Analog Receivers Comprising Frequency-Selective Blocks", Advances In Radio Science, Vol. 4, Pp. 189–195, 2006 (http://Www.Adv-Radio-Sci.Net/4/189/2006/Ars-4-189-2006.Pdf, Last Viewed 20.10.2010)

[6] R. Oneţ, V. Popescu, M. Neag, I. Sărăcuţ, M. Ţopa and S. McDonagh, "Matlab Modeling and Analysis of the Signal Path in Zero-IF DVB-T/H Radio Receivers" Proceeding of the IEEE International Symposium on Electronics and Telecommunications ISETC '10, Timisoara, Romania, 11-12 November 2010