Wireless Networks In-the-Loop: Emulating an RF front-end in GNU Radio

Sebastian Koslowski, Martin Braun, Jens P. Elsner, Friedrich K. Jondral
Karlsruhe Institute of Technology, Germany
{martin.braun, jens.elsner, friedrich.jondral}@kit.edu

***Abstract*—The objective of this work is to emulate the behavior of the Universal Software Radio Peripheral as an example of an RF front-end hardware for software radios. The model includes digital and analog signal processing. The emulator is implemented in GNU Radio and is intended to be used as part of a wireless network simulator.**

I. Introduction

When designing wireless networks simulations play an important part. This is especially true for the development of systems which need to operate alongside other systems in a heterogeneous radio frequency environment. In such an environment, radio front-end hardware design determines the performance to a great extent. To quantify the effects on various protocol layers, a simulation environment which accurately models the physical realities is needed. As part of such a simulation environment proposed in [1], this paper describes the steps necessary to model the digital and analog signal processing of a generic software radio front-end. The Universal Software Radio Peripheral (USRP) [2] and its RFX2400 daughterboard are modeled using the GNU Radio framework [3]. The considered effects include sample rate conversion, digital and analog filtering, IF mixing, as well as imperfections of the analog components such as I/Q-imbalance, non-linearities, phase noise and noise figure, both in the transmit and receive path.

The resulting USRP model has the same programming interface as the USRP and allows for seamless switching between simulation environment and real hardware. It is kept modular to allow emulation of other hardware by adjusting a set of parameters.

The remainder of this paper is structured as follows. In Section II the USRP and GNU Radio are introduced. Section III describes the emulation of the USRP motherboard. A description of the analog RF hardware model is given in Section IV. Test results are shown in Section V, while Section VI concludes.

II. GNU Radio and the USRP

A. GNU Radio

GNU Radio is a free software framework for developing software radio applications [3]. It provides signal processing functionality as well as interfaces to hardware. Software radio applications are designed as so-called flow graphs. Signal sources and sinks provide access to a number of hardware and software interfaces as well as to software synthesized signals. Signal processing operations are implemented in atomic blocks which are then interconnected to fulfill a certain processing task. GNU Radio is a hybrid C++/Python system. For optimal performance, signal processing blocks are written in C++. Flow graphs are built in Python. Memory management and scheduling is handled by the GNU Radio framework.

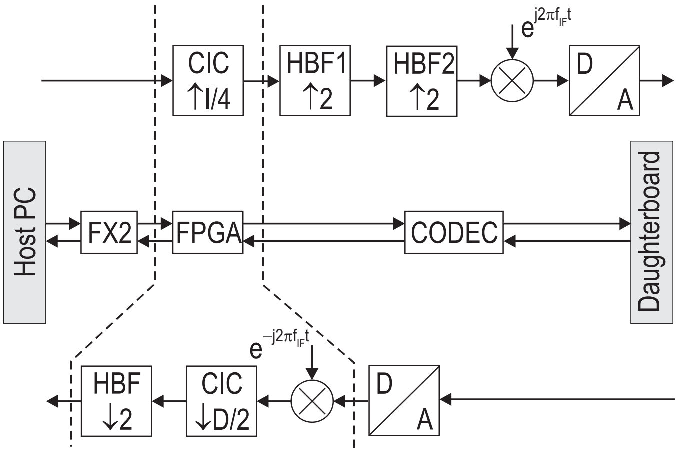

Fig. 1. Block diagram USRP motherboard

B. USRP

The USRP is a radio front-end developed by Ettus Research LLC [2]. It is well integrated into the GNU Radio framework. The USRP is composed of a motherboard and up to four daughterboards – two each for transmission (Tx) and reception (Rx) of radio signals. Data transport to the host PC is implemented over a Universal Serial Bus (USB 2.0). Figure 1 shows the basic structure of the motherboard. It houses a Cypress FX2 USB Controller which handles data transfer from and to the host PC running the software radio application and also distributes control messages throughout the USRP. Digital signal processing is implemented on an Altera Cyclone FPGA; this includes data routing, sample rate conversion by a user-defined factor and digital mixing to baseband in the receive path. The FPGA may contain multiple signal processing units – in the default configuration it can handle up to two complex signals for both Rx and Tx.

The motherboard also contains two AD9862 mixed-signal front-end processors (CODEC) which are connected to the FPGA. They perform analog to digital conversion (ADC) with a resolution of 14 bit at a sample rate of 128 MS/s and digital to analog conversion (DAC) with 12 bit at 64 MS/s. Prior to the DAC the baseband signal is interpolated by a fixed factor of 4 and shifted to an intermediate frequency (IF). Each CODEC chip connects with up to two extension cards (one Rx, one Tx).

The daughterboards contain the analog RF front-end hardware. A variety of daughterboards for different frequency ranges is available. In this work the RFX2400 transceiver board [2] is discussed and modeled as an example for any front-end. It integrates a complete transmitter and receiver and requires two extension card slots. A structural overview is shown in Figure 3. Direct conversion is used to shift the signal from and to the ISM-band at 2.4 GHz. Also included are low noise amplifiers (LNA), an analog bandpass and image rejection filters. The local oscillator signals are generated independently for the transmit and receive path.

III. Model of USRP motherboard

The significant signal processing components of the motherboard (cf. Figure 1) are modeled in GNU Radio blocks and implemented as described below. The simulation bandwidth (cf. [1]) is chosen to be 128 MS/s.

A. Transmit path

The transmit path of the USRP motherboard consists of a digital up converter (DUC) and a D/A converter for the transmission data. This functionality is spread over the FPGA and the CODEC chip (cf. Figure 1). The transport and buffering of the data from the host PC to the USRP induces a certain delay, which is modeled by a delay block. The delay can hence be set with the resolution of one sample.

Interpolation from the incoming user-defined sample rate to the ADC input rate of 32 MS/s is accomplished with a four stage Cascaded Integrator Comb (CIC) filter. The CIC filter is implemented as a custom GNU Radio block with 16 bit fixed-point arithmetic. The subsequent scaler compensating the word length growth is recreated as carried out in the FPGA.

Further upsampling to the DAC rate of 128 MS/s is provided by the CODEC chip using two half band filters. These filters are specified to have a maximum passband ripple of 0.1 dB and an out-of-band signal suppression of 60 dB or more. The passband edge is at 12 MHz using 39 taps in the first stage and 15 in the second stage [4]. The resulting transfer functions are shown in Figure 2. For the modeling of these filters the FIR-filter block included in the GNU Radio framework is used due to its optimized design utilizing x86 instruction set extensions.

Although all signal processing in the FPGA and CODEC chip is performed with fixed-point arithmetic, floating point operations are used (except for the CIC filters) for performance reasons in the model. To compensate for the higher range and precision of the floating point data format a rounding block was introduced. The block performs an intermediate conversion to fixed-point with a arbitrary word length. This block is also used to account for the quantization noise during AD/DA conversion.

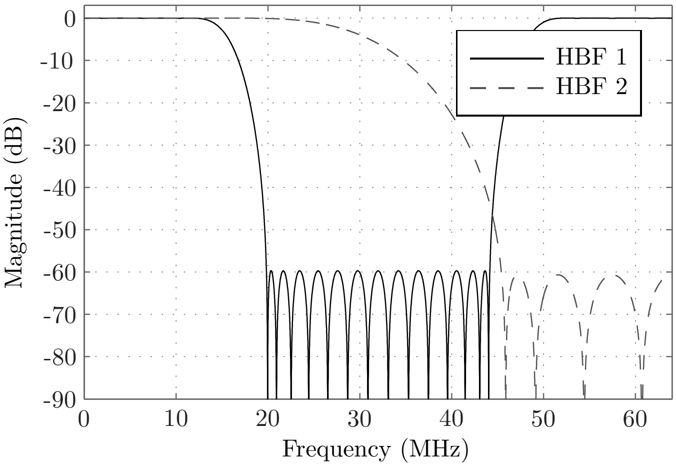

Fig. 2. Halfband filters in transmit path

The CODEC chip also conducts digital mixing of the signal. This is accomplished in two steps: first, a fine modulation is performed prior to upsampling to allow shifting of up to 1/4 of the sampling rate. Then, a coarse mixer shifts the signal by 1/4 or 1/8 of the DAC rate in either direction using a 2 stage CORDIC algorithm. This two step structure was replaced by a single operation in the model.

In a final step before forwarding the signal to the daughterboard model, the signal is rounded to the D/A conversion resolution of 14 bit and scaled to the appropriate output level.

B. Receive path

The receive path includes A/D conversion and a digital down conversion (DDC) of the two quadrature signals provided by the daughterboard. As the ADC rate is half of the simulation sample rate the signals need to be decimated by a factor of two. Anti aliasing filtering is done on the daughterboard as described in Section IV-A. The resolution of 12 bit is ensured by rounding.

The DDC is carried out completely on the FPGA. The Cordic-based IF mixer is replaced by a floating-point numerically controlled oscillator (NCO) block with a uniformly distributed initial phase. To introduce a frequency drift an additional noise signal can be added to the NCOs input.

The following reduction of the sample rate is realized in two steps. First, a CIC decimation filter performs half of the user-defined rate change factor. This filter is implemented in the same way as in the transmit path. Subsequently, a half band filter decimates further by a factor of two. The coefficients of this filter are the same as used in the FPGA.

The resulting samples are then provided to the software radio application.

IV. Model of the USRP daughterboard

In order to emulate the analog RF front-end on the daughterboard, its main functions up/down conversion, amplification and filtering have to be considered. All effects of the analog front-end are modeled in equivalent baseband.

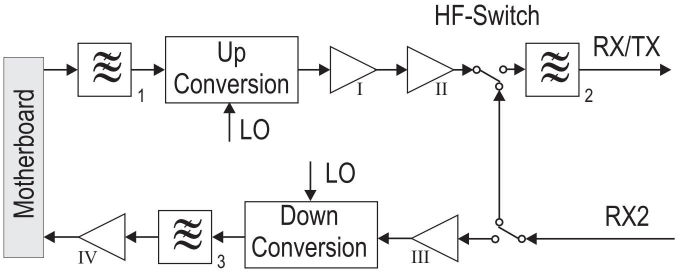

Fig. 3. Block diagram RFX2400 daughterboard

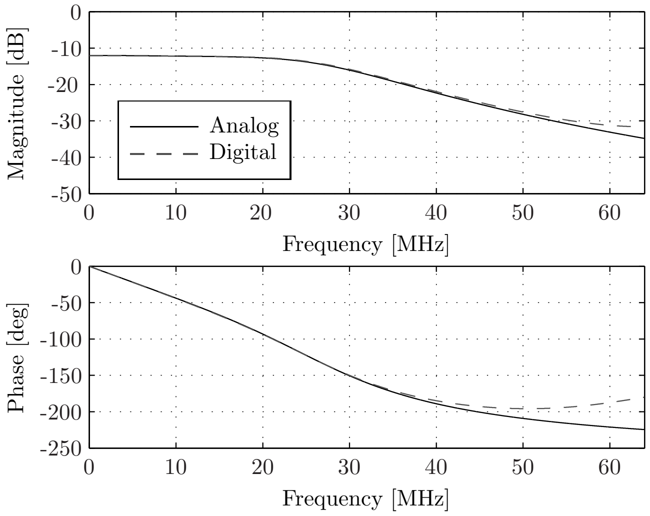

Fig. 4. Tx image-rejection filter

A. Analog filters

The transmit path includes an image-rejection lowpass filter to clean up the signal after the DAC before up conversion (cf. Figure 3 – 1). It is composed of a network of discrete components. Including the surrounding level and bias network the filter has a passband attenuation of 12 dB and a cutoff frequency of approx 28 MHz as shown in Figure 4. The filter is discretized by using the impulse invariance method and implemented in GNU Radio as a third order IIR filter. For simulation the filter gain was set to 0 dB as level control is centralized in the model.

The daughterboard is also equipped with a SAWTEC 855916 bandpass filter at the 2.4 GHz ISM band, placed in the transmit path (cf. Figure 3 – 2). For reception, the filter is bypassed by default using RF switches and a secondary port. According to its specifications [5] the filter has a typical passband ripple of approximately 1 dB and attenuates the signal by approximately 2 dB. The passband edge is within the guard zone near the maximum simulation frequency and the magnitude response in the simulation band is fairly flat. Furthermore, a phase response is not specified. Therefore, this filter was not included in the emulator.

The receive path contains an anti-aliasing filter composed of a network of discrete components (cf. Figure 3 – 3). It is a fifth order lowpass IIR filter with a passband edge at approximately 20 MHz. The attenuation in pass and stop band is 6 dB and 60 dB, respectively. For discretization the bilinear transform is used. Additional tweaking of the poles and zeros is necessary due to aliasing effects: two of the zeros are positioned to generate a notch at approximately 51 MHz, the remaining are moved to the band end. By scaling the poles by a factor of 0.92 the phase response deviation can be reduced. The resulting time discrete IIR filter response is shown in Figure 5. It approximates magnitude and phase within the passband fairly well.

B. Amplification and non-linearities

On the daughterboard, signal amplification is accomplished through multiple stages. Filters and RF switches reduce the signal level. The transmission signal is amplified in the up converter and the two subsequent low noise amplifiers (LNA) providing a maximum output power of 17 dBm [2]. The DAC has a built-in programmable gain amplifier (PGA) which allows gain reduction by up to 20 dB.

The receive path includes one LNA followed by the down converter. The overall maximum gain results in 75 dB. The built-in automatic gain control has a range of 70 dB. It is not included in the emulator for now. The PGA in the AD converter allows further amplification of the signal by up to 20 dB.

Non-linear effects occur in any active component, e.g., LNAs, mixers and baseband amplifiers. Additional frequency components are generated from which only those are of interest that fall into the simulation band. In particular, for devices operating with RF signals, these are third order intermodulation products. Assuming that signal levels are well below saturation, a third-order polynomial function is employed to model this non-linear behavior. In the baseband this gives

$$y(t) = \alpha_1 x(t) - \frac{3}{4}\alpha_3 x^2(t) x^*(t). \tag{1}$$

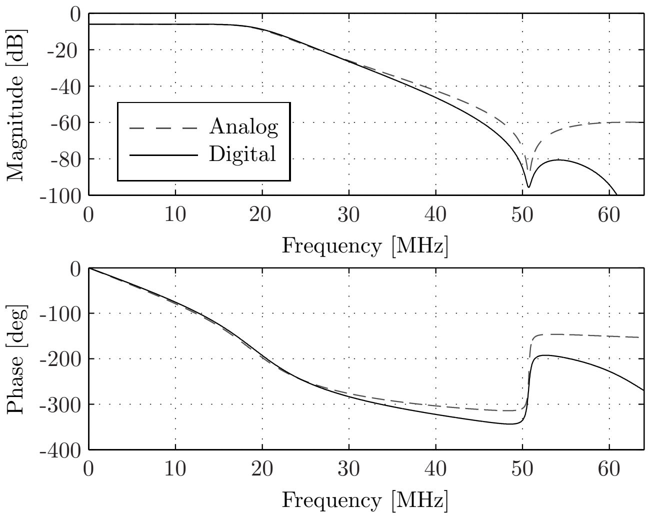

Fig. 5. Rx anti-aliasing filter

TABLE I
GAIN AND IIP3 OF RFX2400 TRANSMITTER COMPONENTS

Component	Gain [dB]	IIP3 [dBm]
Up converter	N/A	24.5
LNA I	12.5	18.5
LNA II	12.0	28
RF switch	-0.6	60

α_1 is set to 1 as gain is considered separately after applying the nonlinearities. α_3 can be calculated from the third order intercept point (IIP3 or TOI) [6].

$$\alpha_3 = 4/3\alpha_1 \cdot \text{IIP3}^{-2} \tag{2}$$

As there are a number of amplifying and mixing stages the non-linearities of each of these should be included into the model to allow for an exact recreation of all effects. This, however, means a great computing effort. Therefore, only one stage with non-linear behavior was included into each signal chain modeling the cumulative non-linear effect.

The overall third order intercept point IIP3_{total} can be obtained from the IIP3s and gain of the individual components in the Rx or Tx chain [6]:

$$\text{IIP3}_{\text{total}}^{-1} = \frac{1}{\text{IIP3}_1} + \frac{g_1}{\text{IIP3}_2} + \frac{g_1 g_2}{\text{IIP3}_3} + ... \tag{3}$$

Tables I and II outline the values specified in the components' data sheets [5], [7], [8], [9], [10], [11]. For the transmit and receive path this yields an IIP3_{total} of 16 dBm and -38...13 dBm, depending on the AGC level.

For the baseband amplification of the IF signals in the receive chain (cf. Figure 3 – IV) also even order intermodulation products are of interest.

$$y_{\text{Im/Re}}(t) = \beta_1 x_{\text{Im/Re}}(t) + \beta_3 x^2_{\text{Im/Re}}(t). \tag{4}$$

Again, β_1 is set to one, β_3 is determined by measurement.

C. Phase noise

Phase noise is an effect that occurs in every oscillator. An ideal oscillator signal consists of only one frequency component f_c. Real world oscillators have additional, undesired spectral components whose magnitude decreases proportionally to f^{-1}. This so-called phase noise or $1/f$-noise is measured by its power at an offset Δf from f_c in a unit bandwidth relative to the carrier power (dBc/Hz).

Discrete phase noise can be generated with a noise shaping filter with the frequency response [12]

$$H_{\text{PN}}(z) = \frac{1}{\sqrt{1 - z^{-1}}}. \tag{5}$$

This can be implemented with an IIR filter whose denominator coefficients result from a truncated power series expansion of (5). The more taps the filter has the better the approximation at frequencies near the oscillator frequency f_c. The resulting noise process $n_p(k)$ is modulated onto the signal.

$$y(k) = \mathrm{e}^{jn_p(k)} \approx (1 + jn_p(k)), \text{ for } n_p(k) \ll 1 \tag{6}$$

D. IQ-Imbalance

During up and down conversion the IF/RF signal is mixed with the local oscillator signal. To achieve quadrature mixing a phase shifted version of the LO signal is required. This phase shift has to be exactly 90°, preserving the amplitude of the signal. Any mismatches in either phase or amplitude result in additional spectral components.

This effect can be described by the following signal models [13]. The relative amplitude deviation shall be 2ε and the phase shift deviation 2ϕ. For the down conversion the resulting baseband signal can be obtained from

$$\begin{aligned} r'(t) = \; & (1-\varepsilon) r_{bp}(t) \cos(\omega_c t + \phi) - \\ & j(1+\varepsilon) r_{bp}(t) \sin(\omega_c t - \phi). \end{aligned} \tag{7}$$

The bandpass signal $r_{bp}(t)$ is replaced by its baseband signal $\text{Re}\{r(t)\exp(j\omega t))\}$ which results in

$$\begin{aligned} r'(t) = & \frac{1}{2}(\cos(\phi) - j\varepsilon \sin(\phi)) \cdot r(t) + \\ & \frac{1}{2}(\varepsilon \cos(\phi) + j \sin(\phi)) \cdot r^*(t). \end{aligned} \tag{8}$$

The up conversion yields similar results. Both signal models are incorporated into the signal chain. The parameters ε and ϕ can be time variant. For this simulation, however, they are assumed constant due to the unknown time dependency. Mean values for the up converter are $2\phi = 0.3°$ and $(1+2\varepsilon)$ = 0.1 dB [10] and for the down converter $2\phi = -0.8°$ and $(1 + 2\varepsilon)$ = 0.3 dB [9].

E. Noise Figure

The noise figure describes the loss in the signal-to-noise ratio (SNR) as the signal passes through the different receiver components, each of them increasing the noise level by a noise factor F.

$$\text{NF} = 10 \log F = \text{SNR}_{in,\text{dB}} - \text{SNR}_{out,\text{dB}} \tag{9}$$

To determine the overall noise figure of a cascade of components Friis' formula can be used [6]. A component n is characterized by its gain g_n and noise factor F_n.

$$F_{\text{total}} = 1 + (F_1 - 1) + \frac{F_2 - 1}{g_1} + \frac{F_3 - 1}{g_1 g_2} + ... \tag{10}$$

Table II shows the gain and noise figures of the components in the receive signal chain of the RFX2400 daughterboard as shown in Figure 3. The overall noise figure NF_{total} is 4.48 dB on the Rx2 port and 7.8 dB on the Tx/Rx port. Because not all of the components specify a noise factor in their datasheets those values have to be considered lower bounds for NF_{total}. To incorporate this into the model, Johnson thermal noise is added to the received signal. It follows a normal distribution with a variance of

$$\sigma^2 = 4Tk_B RB(F_{\text{total}} - 1). \tag{11}$$

where T is the temperature, k_B Boltzmann's constant, R the resistance and B the simulation bandwidth [14]. The down converter's noise figure is dependent on the AGC level [9].

TABLE II
GAIN, NF AND IIP3 OF RFX2400 RECEIVER COMPONENTS

Component	Gain [dB]	NF [dB]	IIP3 [dBm]
RF filter	-2.75	2.75	∞
HF switch	-0.6	0.6	60.0
LNA III	12.5	2.2	18.5
Down converter	39.5	11.75	32.2
IF filter	-6	6	∞
Baseband AMP	30	N/A	N/A

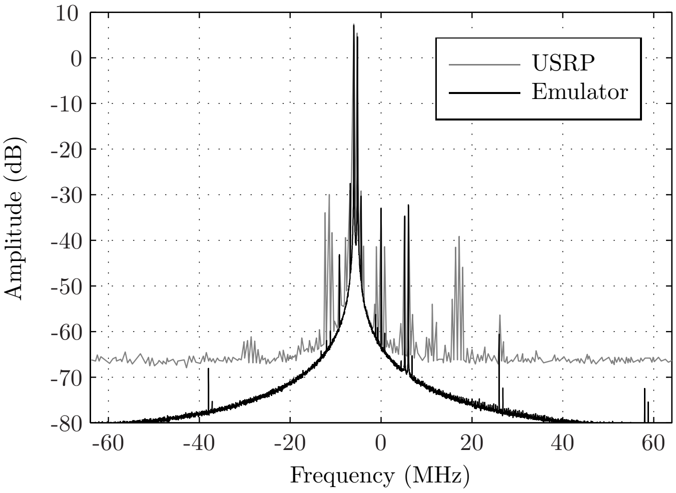

Fig. 6. Comparison of USRP and emulator Tx output

The value in Table II is only valid when the AGC is fully open. For lesser amplification by the AGC the overall noise figure can rise to well over 60 dB.

V. MEASUREMENTS

A. Transmission

For the evaluation of the emulator performance the following setup was used. A GNU Radio application synthesizes a signal containing two sinusoids with 800 kHz spacing, an often used setup for analyzing non-linear effects. For comparison, the USRP Tx/Rx port is connected to a spectrum analyzer. The interpolation is set to 32, the carrier frequency to 2.45 GHz. In this configuration the USRP driver chooses a LO frequency of 2.456 GHz. In order to achieve the desired carrier frequency the IF is set to -6 MHz.

Figure 6 shows a comparison between the USRP and the emulator output for this scenario. It is centered at the LO frequency. The third order intermodulation products, the carrier feedthrough, the sideband caused by the IQ-imbalance and the oscillator phase noise can be observed.

The USRP signal additionally contains fifth order intermodulation products as well as the thermal noise floor and a number of additional unidentified spectral components. These other tones appear to be modulated mixer spurs as outlined in the converter's datasheet [10]. The most significant ones are found at 2 and -4 times the IF. This mixer-specific behavior was not considered in the emulator and is therefore not observable in the emulator output.

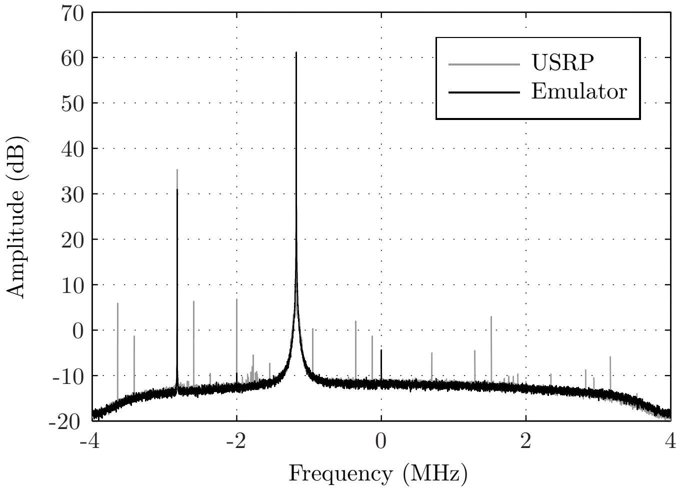

Fig. 7. Comparison of USRP and Emulator Rx Output

B. Reception

For measurement of the receive chain a single carrier tone at -30 dBm is used as input signal. To measure the receive spectrum of the USRP, a signal generator is connected to the RX2 port of the daughterboard. Its center frequency is set 1.2 MHz lower than the frequency the USRP and the emulator are tuned to. This allows observing some of the described effects without bypassing the IF mixer. For the emulator an equivalent baseband signal is synthesized with GNU Radio. The decimation is set to a factor of 8.

The AGC attenuation is set to 33 dB resulting in a theoretical noise figure of 28 dB. In measurement, the USRP's noise level was found to be about 2 dB higher and the noise figure was adjusted accordingly. Figure 7 shows the spectral output of the USRP and the emulator. The shape of the noise floor reflects the combined CIC decimation and halfband filter response. Due to the rather strong input signal, phase noise is also observable. In addition to the desired peak of the input signal additional peaks are caused by IQ-imbalance and non-linear baseband amplification. The USRP shows a number of additional spectral components. These appear to be spurious signals, probably introduced during the down conversion. Third order intermodulation products are not observable since only a single tone input is used.

VI. CONCLUSION

The digital signal processing chain on the USRP motherboard was modeled alongside with the analog signal processing on the RFX2400 daughterboard. Modeling included the most important non-idealities of the analog front-end such as analog filters, phase noise, D/A and A/D conversion. Together with a channel model the USRP emulation allows to evaluate software radio signal processing algorithms in a reproducible manner. Verification showed that most effects are captured very well. The limits of the model lie in the modeling of higher order non-linearities, as not all spurious signals seen in measurements were covered by the component-wise analysis used in this work.

REFERENCES

[1] J. Elsner, M. Braun, S. Nagel, K. Nagaraj, and F. Jondral, "Wireless Networks In-the-Loop: Software Radio as the Enabler," in *Software Defined Radio Forum Technical Conference*, 2009.

[2] M. Ettus, "Ettus Research LLC," http://ettus.com/.

[3] Free Software Foundation, "GNU Radio," http://gnuradio.org/.

[4] AnalogDevices, *AD9862 Mixed-Signal Front-End (MxFE) Processor*, 2002, http://www.analog.com/.

[5] SAWTEC, *855916 2441.8 MHz SAW Filter*, 2005, http://www.triquint.com/.

[6] T. J. Rouphael, *RF and digital signal processing for software-defined radio*. Newnes, 2009.

[7] H. M. Corporation, *HMC174MS8 GaAs MMIC T/R Switch*, 2009, http://www.hittite.com/.

[8] A. Technologies, *MGA-82563 Amplifier*, 2009, http://www.avagotech.com.

[9] AnalogDevices, *AD8347 Direct Conversion Quadrature Demodulator*, 2005, http://www.analog.com/.

[10] Analog-Devices, *AD8349 Quadrature Modulator*, 2004, http://www.analog.com/.

[11] R. M. Devices, *RF3315 Broadband High Linear Amplifier*, 2010, http://www.rfmd.com/.

[12] N. J. Kasdin, "Discrete simulation of colored noise and stochastic processes and $1/f^\alpha$ power law noise generation," *Proceedings of the IEEE*, vol. 83, no. 5, 1995.

[13] P. Rykaczewski, *Quadraturempfänger für Software Defined Radios : Kompensation von Gleichlauffehlern*. Institut für Nachrichtentechnik der Universität Karlsruhe (TH), 2006.

[14] U. Tietze and C. Schenk, *Electronic circuits: handbook for design and application*. Berlin: Springer, 2008.