Verification of Digital RF Processors: RF, Analog, Baseband, and Software

Khurram Muhammad, Thomas Murphy, and Robert Bogdan Staszewski, *Senior Member, IEEE*

***Abstract*—Single-chip RF SoCs are seeing widespread acceptance in wireless applications. In this paper we address the issue of design verification of single-chip RF SOCs in a framework that accepts RF input and analyzes receiver BER performance and transmitter output distortion and phase noise by processing several thousand packets of baseband information while compensation algorithms are simultaneously executed. No comprehensive methodology exists to date for designing such complex systems. This paper present a novel approach that allows building complex RF SoC systems based on VHDL modeling and simulation and opens up major avenues of model development for RF and analog circuits. This approach has been successfully applied to verify two generations of *digital RF processors* (DRP) in deep-submicron technologies.**

***Index Terms*—Analog, baseband, behavioral, bluetooth, cellular, deep submicron CMOS, design, digital RF processors (DRP), GSM, mobile phones, modeling, MTDSM, noise figure, phase noise, PLL, receiver, RF, SNR, SoC, transceiver, transmitter, validation, verification, VHDL, wireless.**

I. Introduction

LOW-COST wireless handsets are expected to encounter aggressive growth in Asia, South America, and Africa. Cost reduction has traditionally been achieved by developing higher integration and reducing the number of components used to build the solution. A typical wireless handset contains SAW filters, RF switches, power amplifiers, RF transceivers for several LAN and cellular standards, power management ICs, audio and video codecs, baseband processors, application processors, FLASH memory and several peripherals [1]. As higher integration is achieved, the complexity of verification of the complete solution grows many times. A simplified block diagram of an RF SoC for a single-chip GSM phone [2] is shown in Fig. 1. This IC integrates quad-band receiver and transmitter, memory, power management, dedicated ARM processor and RF built-in self test (RFBIST) in a single RF SoC for the second generation digital RF processor (DRP). The receiver uses direct RF sampling [2]–[5] of RF signal at Nyquist rate of the carrier. The transmitter uses an all-digital PLL (ADPLL) [5], [6] together with a digital pre-power amplifier (DPA) for generating the transmit output as well as for generating the LO for the receiver. The frequency reference is generated by an on-chip digitally controlled crystal oscillator (DCXO) [3]. The power supply of the analog and digital blocks is provided by an integrated power management (PM) system that comprises of several fully programmable

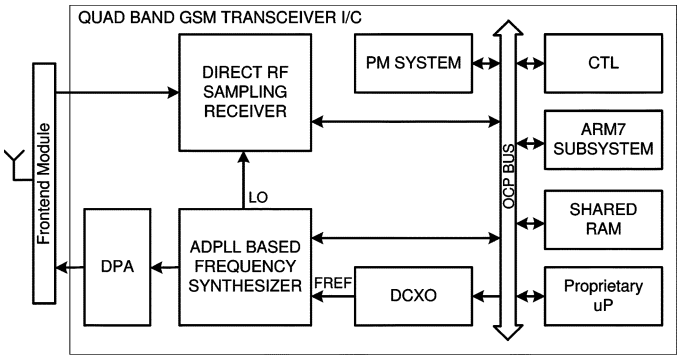


Fig. 1. The first single-chip GSM transceiver SoC [1].

low-dropout (LDO) voltage regulators. The data flow through the single-chip radio is accomplished using an Open Core Protocol (OCP) bus (see http://www.ocpip.org) that allows a proprietary microprocessor as well as an ARM7 sub-system (that has its own RAM and ROM) to control the radio functions. A small shared RAM is provided for the proprietary microprocessor. The control functions of the DRP are managed by the control (CTL) block that also incorporates system-wide JTAG control.

Verification of a complete RF SoC [7] such as the one shown in Fig. 1 requires passing several hundred to several thousand packets of GSM data through the transmitter and the receiver. The highest frequency amongst the four GSM bands is 1990 MHz which corresponds to a period of 502.5 ps. One packet of GSM spans approximately 577 μs and contains 156.25 bits. In order to obtain a 2% bit error rate (BER) estimate at specified sensitivity, several hundred to several thousand packets of data need to be received. Hence, the simulation needs to execute for approximately 100 ms to 1000 ms, thereby, creating an interesting computation problem. Similarly, the transmitter output must be generated for several hundred packets to resolve the modulation distortion, the -64 dBc/Hz phase noise at 400 kHz offset and -164 dBc/Hz noise at 20 MHz offset from the carrier frequency.

II. Transceiver Architecture

We will now briefly describe the transceiver architecture in order to appreciate the diverse nature of its building blocks. The analog receive chain shown in Fig. 2 uses direct sampling of RF at Nyquist rate of the carrier frequency. Following the sampling operation, we use discrete-time analog signal processing to down-sample, filter and analog-to-digital convert the received signal [2]–[5]. Following the low noise amplifier (LNA), the signal is converted to current using a transconductance amplifier (TA) stage and down-converted to a programmable

Manuscript received September 18, 2006; revised January 2, 2007.

The authors are with Texas Instruments Inc., Dallas, TX 75243 USA (e-mail: k-muhammad1@ti.com).

Digital Object Identifier 10.1109/JSSC.2007.894327

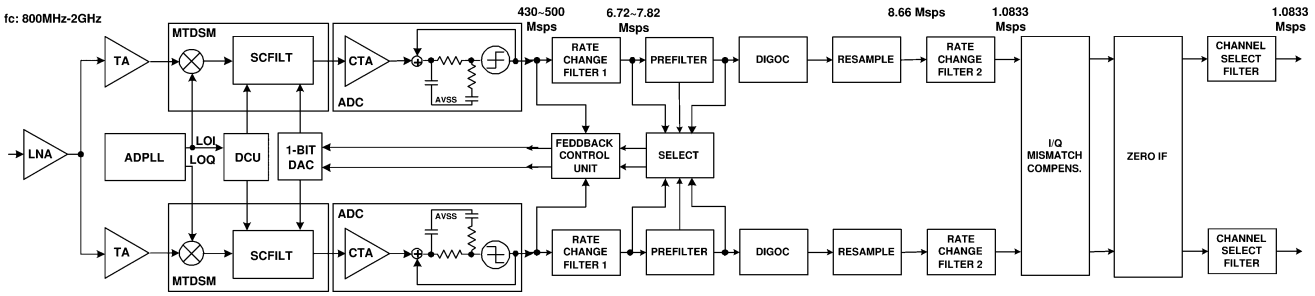


Fig. 2. Simplified block diagram of the receiver.

low-IF frequency by integrating this current on a sampling capacitor. The sampling is performed at the Nyquist rate of the RF carrier frequency. After initial decimation through a sinc filter response, a series of IIR filtering follows in order to reject close-in interferers. These signal processing operations are performed in the multi-tap direct sampling mixer (MTDSM) that receives its clocks from the digital control unit (DCU). The MTDSM is a mixer which embeds a switched-capacitor filter (SCFILT) for rejecting close-in interferers and blockers. A sigma-delta analog-to-digital converter (ADC) containing a front-end gain stage follows. A feedback control unit (FCU) provides a single-bit $\Sigma\Delta$ noise-shaped feedback to the MTDSM to establish the common mode voltage for the MTDSM while canceling out differential offsets. The output of the I/Q ADCs are passed on to digital receive (DRX) chain. The first rate change filter (RCF1) provides anti-aliasing and decimation filtering to reduce the clock rate by 16. Pre-filtering (PREF) is then performed to assist digital resampling (RES) operation. The residual DC offset that could not be corrected by the FCU is corrected by digital offset correction (DIGOC) block. The resampler follows and converts the sample rate from LO dependent clock rate to a fixed output rate of 8.66 Ms/s. Next, the sample rate is decimated by a second rate change filter to the following I/Q mismatch block. The IF frequency is then converted from the low-IF to DC by the ZERO IF block. The final filtering is performed using a fully programmable 64-tap FIR channel select filter (CSF).

The transmitter (shown in Fig. 3) is based on the all digital phase-locked loop (ADPLL) [6]. At the heart of the ADPLL lies a digitally controlled oscillator (DCO). The ADPLL operates in a digitally synchronous fixed-point phase domain as follows: The variable phase $R_V[i]$ is determined by counting the number of rising clock transitions of the DCO clock CKV, $R_V[i] = \sum_{l=1}^{i} 1$. The FREF-sampled variable phase $R_V[k]$, where k is the index of the FREF edge activity, is fixed-point concatenated with the normalized time-to-digital converter (TDC) output $\varepsilon[k]$. The TDC measures and quantizes the time differences between the FREF and DCO edges. The sampled differentiated variable phase is subtracted from the frequency control word (FCW) by a digital frequency detector. The frequency error samples $f_E[k]$, are accumulated to create the phase error samples $\phi_E[k] = \sum_{l=1}^{k} f_E[k]$, which are then filtered by a loop filter that is implemented as a fourth-order IIR filter scaled by a proportional loop attenuator α and a parallel feed with coefficient ρ. The parallel feed path, if enabled, adds an integrated term to create type-II loop characteristics that can be used to suppresses the DCO $1/\mathrm{f}$ noise. The DPA is used to control the output power level.

Good design of such digital and discrete-time intensive circuit techniques using high-speed sigma-delta engines driving analog components is extremely challenging. The hardest problem is to identify those operating conditions which would cause these circuits to violate the specification. An example is the determination of $\hat{K}_{\mathrm{DCO}}$ in Fig. 3 that is performed by a compensation algorithm that runs on the $\mu\mathrm{P}$. There is no known approach that can be adopted to design and verify compliance of the final design to the overall system specification. Hence, we concluded that we had to develop such strategy that is presented in the paper for the first time. The approach is quite simple—build the entire system and verify it for compliance with the specification.

III. Design Abstractions and Modeling

The single-chip RF SoC contains RF, analog and digital sections, each of which are designed and analyzed with different CAD tools. Generally co-simulation is widely advocated in commercial CAD tools available for RF system simulation. A variety of techniques such as time-domain methods, harmonic balance, mixed frequency-time, envelope, linear time-varying analysis, multi-time and periodic steady-state analysis can be used in a co-simulation environment [8]–[12]. However, as noted in [8] and [10], no comprehensive approach exists for the verification of a complete RFIC design. This is especially true when the RF SoC performance is tightly controlled with software [1] as is the case with the DRP technology. Performance verification also requires verifying correct selection and/or correct timing of analog controls during the simulation of the transceiver.

The circuits described earlier in this paper operate at diverse clock frequencies with the added complication that analog blocks are controlled by sigma-delta modulator engines as well as software [1]. With the nature of the transceiver being discrete-time and/or digital, analog simulation is very cumbersome and slow. Hence, each block needs to be modeled at a higher level of abstraction. Complete system verification requires a unified framework in which the critical details of the design are visible and behavioral models can be developed. This is achieved by matching the analog schematic at the block level and above to matching VHDL entities such that analog connectivity is exactly visible in VHDL. Next, behavioral models can be simply plugged in to provide the functionality to the block.

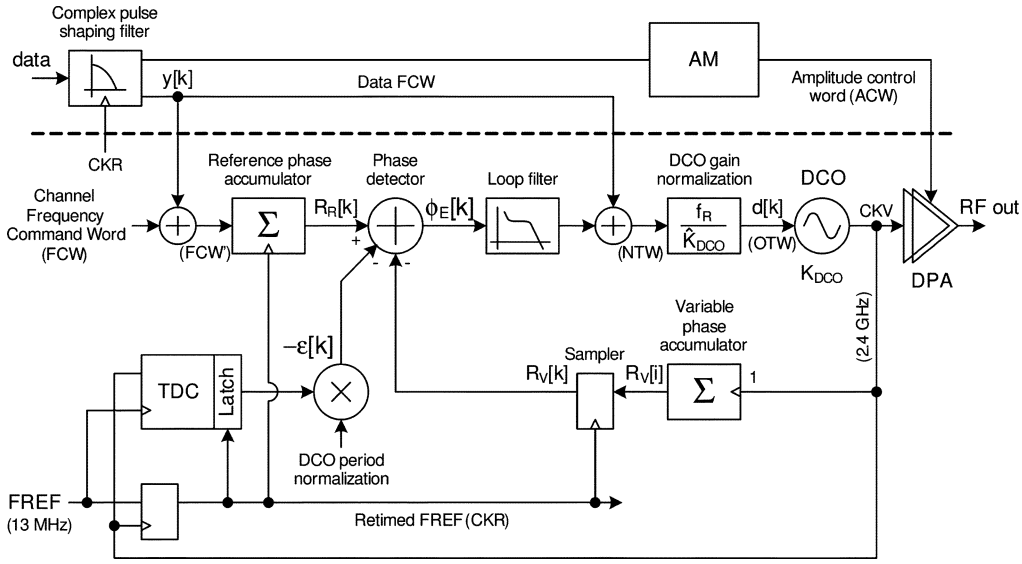


Fig. 3. Simplified block diagram of the transmitter.

The behavioral models are aimed at reduced complexity models of analog blocks that are efficient and accurate enough for verifying the system performance. This provides a fertile ground for novel model order reduction techniques of analog blocks for the purpose of system simulation. Judicious use of one or more techniques presented in [8]–[12] can augment the quality of results produced by the simulator and offers an area for future research. The digital design can be implemented initially from an entirely behavioral perspective. To ensure maximum re-use, the difference between the initial and final phase of the design process is the extent and accuracy of the analog models and the abstraction of the digital circuits. The initial analog models may only emulate coarse behavior similar to equations used in a spread-sheet for cascaded analysis while digital circuits may be expressed as real valued z-domain equations. The accuracy of the analog model is progressively improved while the digital circuits can be implemented to a higher level of detail. The key element is that an efficient flow requires that each subsequent step progressively improves the quality of results from the previous step.

The choice of VHDL is natural because it provides a means for developing the digital design and also provides event-driven simulation capability. The ADPLL generates an output that has stringent requirements on the phase noise that cannot be modeled using a simulator that is vector-based (and implies uniform rate sampling). The complexity of accurately modeling jitter in a modeling environment that has no notion of time makes most simulation environments very unattractive. The notion of time is captured by more digital oriented hardware simulation engines such as VHDL, Verilog and System-C. To maximize reuse of design efforts, a VHDL-based implementation is the optimal choice that allows maximum synergy between various team efforts and design phases.

A. Behavioral Models

The highest level of abstraction is the behavioral models of RF, analog and digital sections. The simplest amplifier models gain, noise, linearity and filtering response while comprehending the digital controls. These effects are easy to model in VHDL using a single process that is triggered by the sampling clock. During the course of time, as more modeling capabilities are developed, behavioral models are updated to improve accuracy. The digital sections may be modeled by real numbers or fixed-width-based simple implementations that are geared more towards obtaining the desired behavior rather than exact implementation. The ADPLL in its simplest form can also be studied behaviorally in order to assess the precise design requirements of its components. The DCO is simply modeled as an ideal clock with jitter and wander [13]. The simple behavioral models play an important role of being the baseline for comparing the final implementation.

1) TX Modeling: The transmitter shown in Fig. 3 is mostly digital with the exception of the DCO and the termination load of the DPA. For constant envelope modulation schemes, modeling the latter is simple. Determine the correct output power for the chosen load using an analog simulator and store the result in a look-up table. The DCO is an *LC* oscillator with "digitized" capacitor that is layed-out as four separate capacitor bank arrays with different unit sizes. As shown in Fig. 4, separate digital control words are needed for the three operational modes of the ADPLL: process-voltage-temperature (PVT), acquisition and tracking (both integer and fractional tracking banks are used in the tracking mode) [5], [13]. The control bits access the PVT, acquisition and tracking capacitor banks, respectively, as shown in Fig. 4 and Fig. 5.

A diagram illustrating a VHDL model of the DCO is shown in Fig. 5. Referring to Fig. 4, DCO_IN_P, DCO_IN_A, DCO_IN_TI, and DCO_IN_TF are the digital `std_logic_vector` inputs controlling the DCO oscillating frequency by controlling the *LC*-tank capacitance of the PVT, acquisition, tracking-integer, and tracking-fractional varactor banks, respectively. Signed-number integer representations of these inputs are multiplied by their respective unit time deviations of the "natural" period: DCO_QUANT_P,

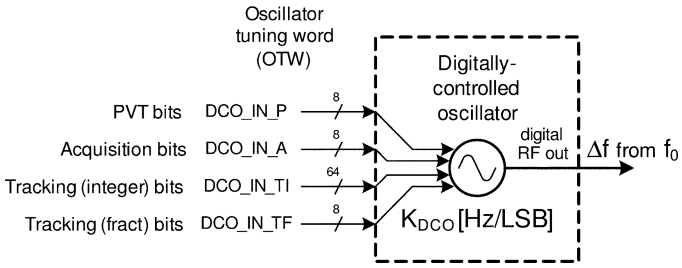


Fig. 4. Digitally controlled oscillator (DCO). Digital inputs select capacitor size that changes the period of an *LC* oscillator.

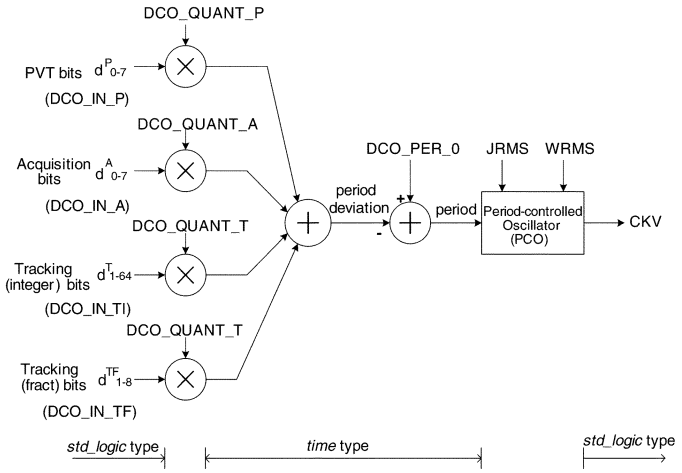


Fig. 5. DCO time-domain model in VHDL.

DCO_QUANT_A, and DCO_QUANT_T. They are VHDL generics rounded off to the closest femtosecond. Their outputs are then summed to create a composite period deviation signal of the VHDL `time`-type. This signal is then subtracted from the "natural" or center oscillating period DCO_PER_0 since the period deviation is of opposite sign to the frequency deviation. This `time`-type signal controls the instantaneous period of the DCO oscillation through a period-controlled oscillator (PCO), whose VHDL code fragment is listed below.

```
1  entity src_pco is
2    generic (
3      WANDER_RMS: time;
4      JITTER_RMS: time;
5      SEED: integer
6    );
7    port (
8      period0: in time;
9      en: in std_logic;
10     clk: out std_logic
11   );
12 end entity;
13
14 architecture behav of src_pco is
15   signal smp: bit := `0';
16 begin
17   process (smp) is
18     variable initial: boolean := true;
19     -instantaneous jitter value
20     variable jitter: time := 0 ns;
21     variable jitter_prev: time := 0 ns;
22     -instantaneous wander value
23     variable wander: time := 0 ns;
24     -the current clock period
25     variable period: time := 0 ns;
26     variable s1: integer := SEED;
27     variable randvar: real;
28   begin
29     if not initial then
30       -adjust the next period
31       period := period0;
32       -add Gaussian-distributed jitter
33       sub_randn(randvar);
34       jitter := randvar * JITTER_RMS;
35       period := period+jitter-jitter_prev;
36       jitter_prev := jitter;
37       -add Gaussian-distributed wander
38       sub_randn(randvar);
39       wander := randvar * WANDER_RMS;
40       period := period+wander;
41       -clock with 50% duty cycle
42       clk <= `1',`0' after period/2;
43       smp <= not smp after period;
44     else
45       period := period0;
46       clk <= `0';
47       -first transition
48       smp <= `1';
49       initial := false;
50     end if;
51   end process;
52 end architecture;
```

Input port "`period0`" of type `time` (line 8) controls the oscillator period during the next cycle. The "`smp`" internal signal is used to control event activity and to establish the next timestamp (line 43). It is also used to schedule rise and fall times of the "`clk`" clock (line 42). With each timestamp, cycle-to-cycle jitter and long-term wander values are added at lines 33–36 and 38–40, respectively. These are shown as function calls on lines 33 and 38 that use a random number generator with Gaussian distribution. Fig. 6 shows the simulated noise profile of the DCO that matches closely with the measured phase noise. The DCO modeling in VHDL is described in detail in [13].

2) RX Modeling: The RX chain shown in Fig. 2 offers several opportunities for reduced order modeling of analog blocks. The detailed models of receiver blocks is beyond the scope of this paper, however, for the purpose of illustration one model of an amplifier is shown below that is used at the highest level abstraction for system verification. While simple models are computationally efficient and reproduce almost all the effects needed for system verification, VHDL allows changing architectures to replace current models with more sophisticated models or vice versa. This allows scalable complexity of the simulation engine that is ideal for a new design.

```
1 entity AMP_MODEL is
2   generic (
3     AMP_IDE_YN : boolean := true;
4     );
5   port (
6     CTL_AMP: in std_logic_vector16;
7     AMP_INP: in real;-Positive input
8     AMP_INM: in real;-Negative input
9     IBIAS: in real;-Bias current
10    AMP_OUTP: out real;-Positive output
11    AMP_OUTM: out real-Negative output
12    );
13 end;
```

```
14
15 architecture behav of AMP_MODEL is
16   signal gainv, fc, noisev, c2, c3, beta,
17     thrsh1, thrsh2, v_o, v_o_prev : real;
18 begin
19   AMP_param_proc : process (CTL_AMP, T)
20   begin
21     --Process (P) and Temperature (T)
22     --are global signals of type real
23     get_AMP_param (CTL_AMP, IBIAS, P, T,
   gainv,
24       fc, noisev, c2, c3, beta, thrsh1,
   thrsh2);
25   end process AMP_param_proc;
26
27   main_model: process (AMP_INP, AMP_INM)
28     variable rv1, rv2, v_s : real;
29   begin
30    --Differential to single-ended conversion
31     v_s := AMP_INP-AMP_INM;
32     if AMP_IDE_YN=false then
33       --Generate total noise (input
   referred)
34       sub_randsn(rv1);--From N(0,1)
35       sub_1overf(fc, rv2);--1/f noise sample
36       --Nonlinearity added
37       v_s := gain * (v_s + c2*v_s**2 + c3*v_s**3);
38       --Noise model
39       v_s := v_s + noisev * rv1 + rv2;
40       --Unity gain RC filter model
41       v_o <= (1 - beta) * v_s + beta * v_o_prev;
42       v_o_prev <= v_o;--previous output
43       v_o <= clip (v_o, gain, c2, c3,
44         thrsh1, thrsh2);
45     else
46       --Ideal model
47       v_o <= gain * v_s;
48     end if;
49     --Single-ended to differential
   conversion
50     AMP_OUTP <= v_o/2.0;
51     AMP_OUTM<= -v_o/2.0;
52   end process main_model;
53 end;
```

The parameters to be used by the main model are obtained by calling a function get_AMP_param. This function uses the 16-bit control bus CTL_AMP and bias current in addition to the global inputs of process (P) and temperature (T) to return the noise and linearity related parameters to be used in the main model. This function is called by a process that uses CTL_AMP and T in its sensitivity list and can generate correct parameters whenever the control bits or temperature change.

The main model is shown between lines 31–51 and is executed whenever the input of the amplifier are changed. It first converts the differential input to single-ended. A polynomial equation (line 37) is used to model the small signal nonlinearity while a clipping function (line 43) models the large-signal nonlinearity. These functions can be as simple as hard-clipping or more sophisticated and use exponential clipping. Thermal and $1/\mathrm{f}$ noise are added in line 39 and passed through a single-pole *RC* filter with the corner specified by *beta* that was returned by the function get_AMP_param. By providing correct parameters to the main model accurate output of the amplifier can be obtained for any process and temperature. The mathematical details in the functions can be improved during the life of the project.

3) System Modeling and Design: A vast majority of the transceiver system is digital in nature whether implementing signal processing functions or control functions and procedures. Several transceiver functions are implemented with RF, analog, digital and software parts and need not only to be verified for correct functionality but also for compliance with the product specification. Some of such functions are listed below and can be designed to final detail in the VHDL-based framework.

1) Effect of the TDC resolution and nonlinearity on the close-in PLL phase noise performance and generated spurs.
2) Effect of the DCO phase noise on the PLL phase noise performance and generated spurs, especially when the PLL contains a higher-order digital loop filter and operates in fractional-N mode.
3) Effect of the DCO frequency resolution on the close-in phase noise of the PLL.
4) Effect of the $\Sigma\Delta$ DCO dithering on the far-out phase noise.
5) Effect of the DCO varactor mismatches on the modulated spectrum.

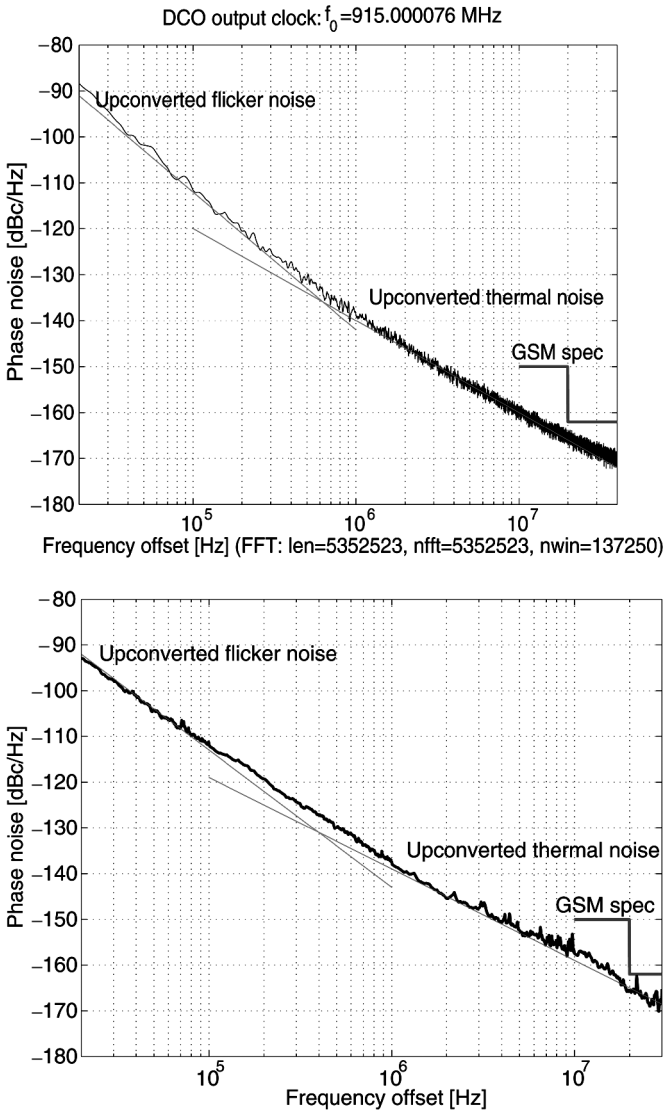


Fig. 6. DCO phase noise of the last channel of the GSM900 band: (top) simulated in VHDL; (bottom) measured using Aeroflex PN9000.

6) Effect of the DPA resolution and nonlinearity on the RF output spectrum.
7) Effect of the DCO phase noise on the degradation of the signal-to-noise ratio (SNR) in the direct RF sampling receiver.
8) Noise folding in the receiver.
9) Effect of the mixer capacitor mismatches on the receiver performance.
10) Operation of the common and differential mode feedback loops [2], [3], [14] in the receiver.
11) Automatic gain control.
12) Tradeoff between peak SNR and blocking performance in the receiver.
13) Software-based mixed-mode compensation loops in the transceiver.

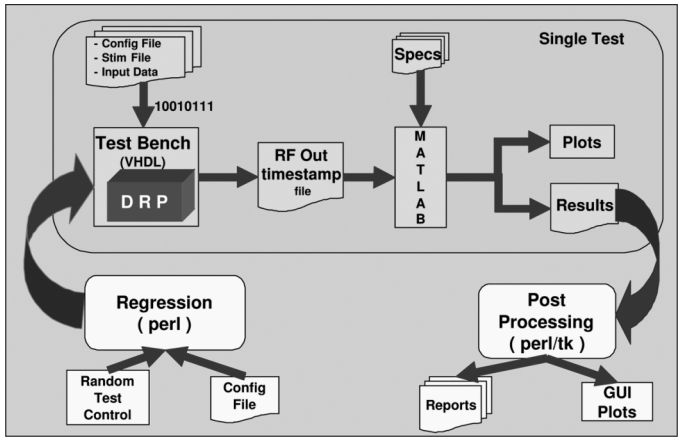


Fig. 7. Transmitter verification flow.

B. RTL Level Simulation

The next level of abstraction substitutes the behavioral digital sections with actual RTL implementations. These sections can be compared against the behavioral blocks and can be simulated together in the VHDL simulation environment while re-using the previous developments. As time progresses, the analog models keep on becoming more accurate as more details are plugged in. Similarly, compensation software such as KDCO estimation for the ADPLL and DC offset and gain compensation in the receiver starts to develop and be deployed. While the previous work is re-used, the next development expands the capability without discarding the previous work.

This is the critical phase of the design of digital sections and generally some surprises are encountered since every implementation detail cannot be comprehended in the behavioral modeling phase. The actual latencies through the blocks and the precise internal word lengths are actually finalized during this phase. Most of the test cases are also exercised and debugged in this phase of the design. The pass/fail criteria for each test is manually determined and programmed such that the simulation engine can report the test outcome. These tests must check that the noise added by digital implementations is not significant and the implemented digital signal processing does not create unwanted spurs due to poor implementation choices.

C. Gate Level Simulation

In this phase, synthesis tools generate the gate level netlist for the digital sections which is substituted for the RTL design. Verification of the system now enters the "regression" phase where verification is now conducted in an automated manner using the most recent version of analog behavioral models. From now onwards, all verification is automated and only debugging effort is needed. Bugs are resolved either as changes to the design or by developing software fixes.

During this phase, the simulation speed is considerably slower and a carefully selected subset of test cases are regressed. In the final phase of the design, the parasitics from the digital layout become available for conducting a detailed timing analysis. A few critical tests are conducted using the automated verification strategy to determine correct operation of the chip. These tests must verify that the final digital implementation in the silicon produces expected results and the noise floors, spur locations and spur levels stay where they are expected in the previous design phase. This verifies correct handling of digital finite precision arithmetic in the design as well as correct synthesis of digital design.

Some examples of the critical tests during this phase are sensitivity and error vector magnitude (EVM) performance. The simulation time during this phase for an RF SoC is dominated by digital design instead of RF or analog models since several thousand gates switch on every clock cycle.

IV. Verification Methodology

The verification methodology is based on the philosophy that all verification needs to be automated as early as possible with maximum reuse of development efforts. In a complex IC such as this, the most valuable tests are those that are exercised on a system level. Key specifications are translated to test cases that would be exercised in the final IC. Next, we need stimulus generators and software emulation of test equipment.

1) Transmitter Verification: For the transmitter, the stimulus is the data sequences that are transmitted and the test equipment is emulated by analysis routines that can analyze the output generated by the simulation, such as the EVM, phase noise at different offset frequencies and achievable SNR. This approach is shown in Fig. 7. The VHDL test-bench instantiates the DRP which accepts the programming instructions from a *Stim File* and configuration instructions from the *Config File* for a single test.

The input data is generated and applied to the DRP. The RF output is stored in a compressed form as zero-crossing time instants and post processed by analysis routines in Matlab. The analysis scripts compare the results against the specifications and generate PASS/FAIL flags and save data for generation of spectral plots. The results from the analysis scripts are post processed to generate desired reports as well as output plots. A web-based reporting shows a summary and status of verification in any part of the world and is ideally suited for a multi-site operation. The performance summary also compares the simulation results versus the target specification.

A key feature of the presented approach is progressive improvement that allows reuse of efforts across the teams as well as across the different design phases. The simulation can

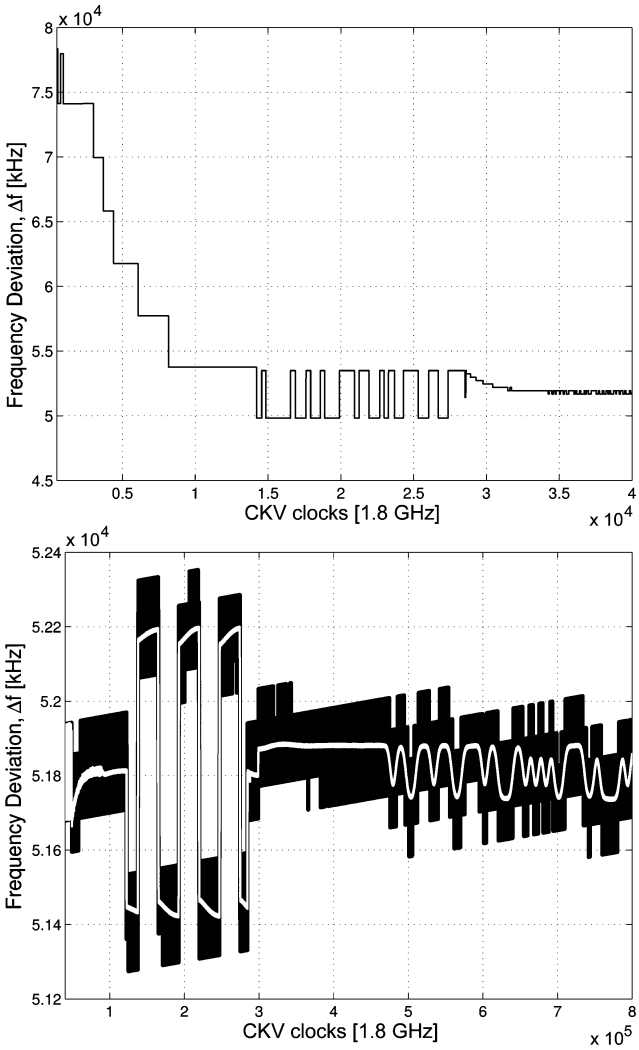


Fig. 8. DCO frequency deviation trajectory from: (Top) 500 to 40K CKV clock cycles. (Bottom) 40K to 800K CKV cycles. The slope shows that DCO drift is modeled as a function of time.

be regressed by changing the DRP configuration and generating random tests, thereby, automating the entire verification process. Dynamic reconfiguration of the transceiver is achieved by executing the software routines as control sequences as simulation time progresses. An example of a control sequence is the locking of the ADPLL. A desired FCW is presented to the ADPLL and the control sequence cycles the ADPLL through the PVT, acquisition and the tracking banks until the lock is achieved. Even before locking the ADPLL, the KDCO must be estimated using a control sequence that executes the KDCO estimation algorithm. Fig. 8 shows the start-up simulation of the transmitter in the single-chip Bluetooth radio where the ADPLL cycles through different phases of establishing a lock. The transient spectrum as well as the steady-state output spectrum can easily be analyzed in such a platform.

2) Receiver Verification: For the receiver, the baseband signal to be transmitted is generated by a stimulus generator in C or Matlab and passed on to the VHDL simulation environment that adds channel propagation impairments based on the performed test as shown in Fig. 9. Generating test data at baseband rate (4X oversampled symbol rate) ensures that relatively small files contain the data that is needed to be read into the VHDL simulation environment. The DRP is configured for operating conditions dictated by the test at hand through the *Config File*. These conditions may be the process corner or the temperature of the chip. Clearly, analog block models need to consider operating conditions for the tests to be of value, e.g., the noise should be modeled correctly and turned on in the model when conducting the sensitivity test.

Next, within the VHDL simulation environment the input data is up-converted to RF and interferers or blockers are added depending on the test. The total received RF signal at the antenna has now been synthesized—it is now passed on to the DRP. The software is executed through a baseband behavioral model that programs the DRP for desired operation and the test is conducted. An application processor interface (API) is developed and used during this phase of the verification. On completion of the test, the output of the DRX is analyzed by post-processing analysis scripts in C/Matlab for SNR, image rejection ratio (IRR), DC-offsets and group delay distortion. The receiver output is also passed to the baseband demodulator that computes the final BER and indicates the outcome of the test. The BER calculation is done in the C or Matlab framework that originally generated the transmitted signal. Similar to the transmitter case, a regression script can be executed that executes several hundred tests for various corner conditions.

The baseband model can generate stimulus for a variety of tests. These include various fading scenarios that require specified BER performance in the presence of receiver impairments. The baseband software also post-processes the output of the DRP to determine whether the BER is at acceptable level. For some tests, up to 4000 GSM bursts are needed. For such extensive simulations a pre-processor breaks the test into chunks of reasonable sizes and distributes the simulation on a cluster of workstations. Each workstation may run only up to 200 consecutive GSM bursts. The results from each workstation are regrouped to create a single output stream and fed to the baseband demodulator.

There are two versions of simulations that can be configured at compile time. The RF version considers LNA input that is 16 times oversampled at the RF carrier. This simulation allows modeling of RF impairments at RF frequencies and directly calculates the mixer output using the actual LO system. A lower complexity IF version simulation refers RF amplifiers to baseband and speeds up the simulation by using the ADC sampling rate as the reference clock for all amplifiers in the receive chain. The LO system is also referenced to baseband and all impairments mapped to baseband frequency. A vast majority of simulations are performed using this version of simulation as it is more than 64 times faster than the other version.

Fig. 10 demonstrates the potential of such simulations for the sensitivity test case. The output of each block is dumped to a file that is post-processed. At each point in the chain we can look at the output spectrum and the output metrics that measure the receiver quality. In this simulation, $1/f$ noise generation was turned off and the magnitude of complex output is plotted. The filtering offered by the MTDSM can be observed and the SNR of the input signal can be tracked along the chain. In this simulation, DC-offset is removed by the channel select filter

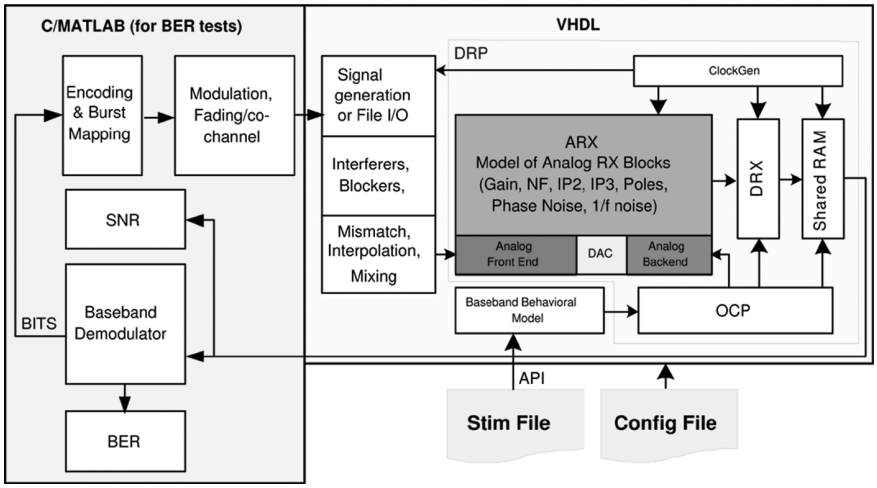


Fig. 9. Receiver verification flow.

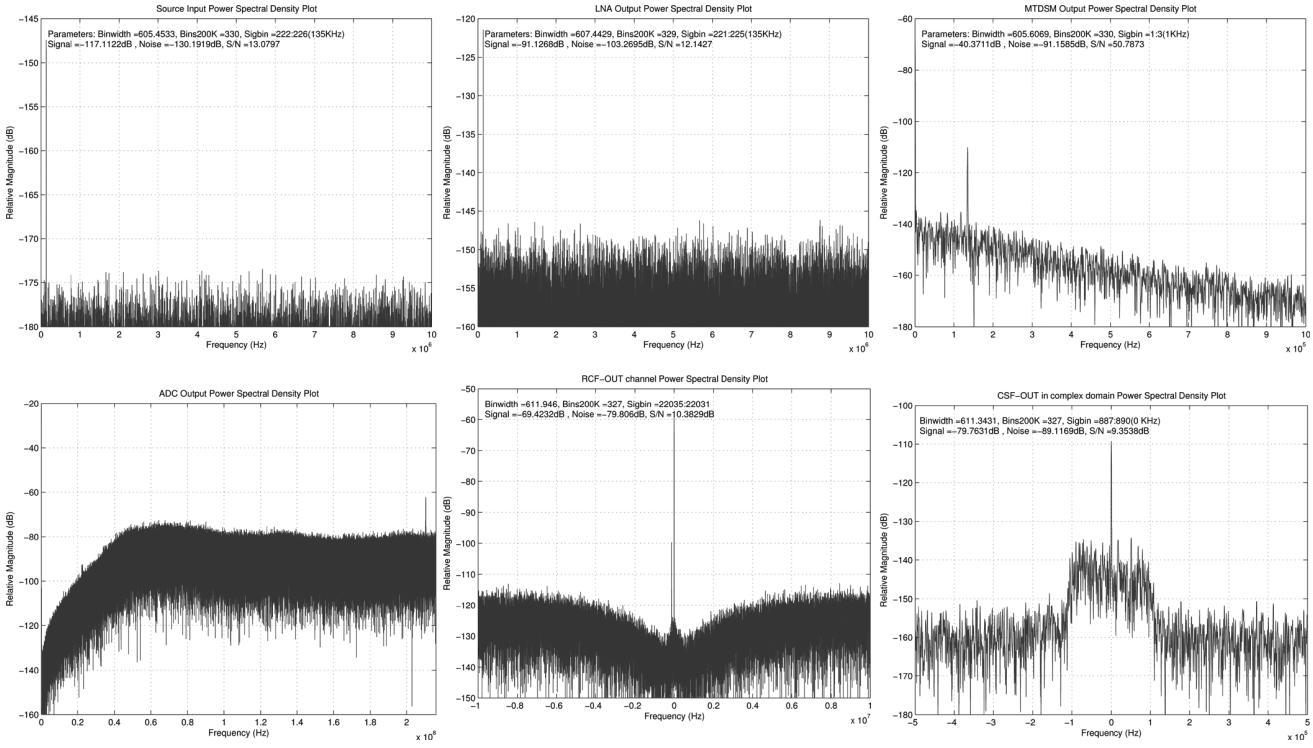


Fig. 10. Receiver simulation. Performance can be determined in the final design at every point in the receive chain.

(CSF) since after the second down-conversion it translates to a tone at IF frequency and falls in the stop-band of the CSF. However, the DC-offset correction performed by the feedback control unit (FCU) can easily be executed and verified for performance. If the chosen IF frequency is such that the DC-offset after the second down-conversion falls in the passband of the CSF, a different test case will be executed that would have the corresponding software controlling the receiver. Hence, this framework allows simulating the transceiver IC closest to how it would be operated in the laboratory and allows development as well as verification of the firmware.

A. Test-Bench

The test-bench instantiates the DRP together with the OCP control, input stimulus generation and baseband behavioral model. Additional test benches for sub-circuit validation improve productivity. Each sub-system shown in Fig. 1 has a dedicated test-bench that allows faster execution times for tests that are sub-system specific. This allows development of the module in isolation from the rest. After passing the sub-system tests, the verification moves to top-level test-bench.

B. Test Cases

One of the most critical aspect of the design is the specification of the test cases on individual block level, sub-system level and at top level. These tests need to be carefully designed and augmented with correct modeling of analog imperfections to allow verification. A library of tests is developed that can be re-used for future products.

As hinted earlier, test cases are carefully divided into categories that define their scope. Some are needed only during the early phases of the design while others may be used in all design phases. Examples of most used test cases are of those that determine the noise performance of the transceiver and include the receiver sensitivity and transmitter close-in and far-out phase noise.

C. Analysis Scripts

Analysis scripts are a substitute to the test equipment in the laboratory and must be carefully calibrated. Several analysis scripts are needed for certification of the quality of the data analyzed. These include SNR, IRR, phase noise, EVM, transient spectrum analysis, offsets and BER computation. A pass/fail criteria for each test must be identified such that the script can report whether or not the specification is met. This is a critical step in making verification fully automated. All efforts must be made to avoid human intervention to determine the outcome of a test in the final phases of design.

D. Regression

Once the automation is achieved, it is easy to regress an entire suite of test cases with different versions of configurations (operating conditions) or control software. It is now also possible to improve the quality of analog models and/or have different implementation of digital sections within the same simulation framework. Later, data obtained from the laboratory on measured analog performance can be used in conjunction with the rest of the system to study system behavior in order to debug silicon.

Post-silicon calibration of the system simulation is an important step to allow the simulator to reproduce the results seen in the lab. Undesired phenomena observed in the laboratory can be reproduced in the system simulation allowing evaluation of perceived fixes for the system-level bugs.

V. Compensation Algorithms

In deep-submicron design for low-cost, the approach is to simplify analog design and migrate the complexity to software [1]. Compensation algorithms now play a central role in determining the final system performance. Although VHDL is believed to be a digital functional verification tool, we deploy it as the main design and performance analysis tool. Hence, system verification is intertwined with the final design. One critical aspect of compensation algorithms is that these must meet the real-time needs imposed by the specification. Normally in a TDD system, a fixed amount of time is available between TX and RX operations and the operating timeline between packets of data need to be verified for proper operation with the baseband algorithms.

A typical timeline for the receiver operation must lock the ADPLL, set the control words of the analog and/or digital blocks according to the automatic gain control (AGC) setting, remove the DC-offset and wait for the receiver to settle. All these operations can easily be executed to verify that they do not need more time than what is allowed for any operating condition.

The proposed framework easily allows development and verification of compensation algorithms. Consider, for example, the amplifier model presented in Section III-A2. Although performance of the amplifier is determined by the chosen process and temperature, a desired performance is needed independent of the process and temperature. Let us take gain as an example. A gain compensation algorithm would choose a digital control word that would provide a constant analog gain despite the process and temperature. Assuming that the design provides measurable quantities that show correlation with process and temperature, a compensation algorithm can then be easily programmed to verify that the compensated performance meets the specification. In the case of our example, we can verify the variation of software compensated gain to be within the specification.

Similarly, in the transmitter, one of the key parameters that determines the TX output performance is the $\hat{K}_{\mathrm{DCO}}$ that must be compensated correctly. It is easy to set up a simulation in which temperature is swept and the compensation algorithm is executed simultaneously. The spectral output of the transmitter can now be easily verified to comply to the product specification with the actual design—no other framework allows verification to such detail!

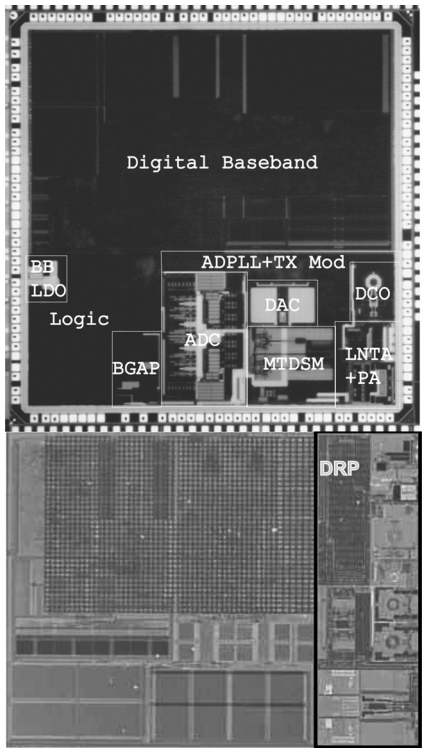


Fig. 11. Die micrographs of the commercial single-chip SoC's employing two generations of DRP: (top) first generation—Bluetooth; (bottom) second generation—GSM.

VI. Die Micrographs

Fig. 11 shows two chip micrographs representing the first and second generation of digital RF processor (DRP), respectively: commercial single-chip Bluetooth radio in 130-nm CMOS, and commercial single-chip GSM radio in 90-nm CMOS both built based on the presented methodology. The GSM SoC consists of the digital baseband with digital logic and SRAM memory on the left part, and the DRP that integrates memory, digital logic, analog and RF, on the right part. The 90-nm process is

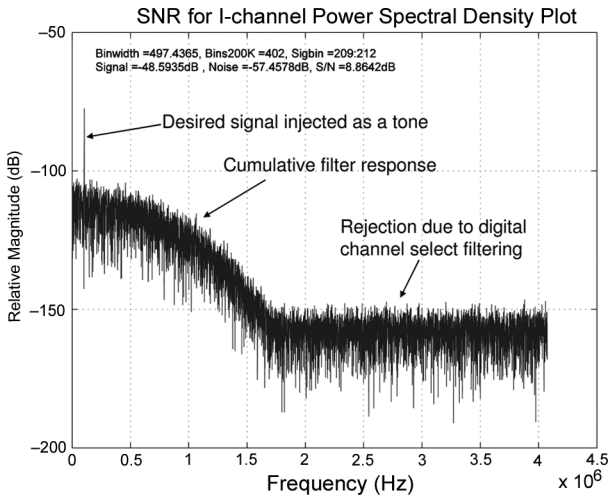


Fig. 12. Measured receiver output for GSM receiver in 90-nm CMOS.

characterized by the following parameters: 0.27-μm minimum metal pitch, five levels of copper metal, 1.2-V nominal transistor voltage, 2.6-nm gate oxide thickness, logic gate density of 250 kgates/mm^2, and SRAM cell density of 1.0 Mb/mm^2.

The number of pads in DRPs are minimized to drive the cost low, therefore, most analog points cannot be accurately measured. Therefore, most measurements are taken for the entire system. The measured RX sensitivity of -82 dBm for Bluetooth and -110 dBm for GSM, versus the respective specifications of -70 dBm and -102 dBm, is among the best in class. The overall GSM RX noise figure is only 2 dB. The measured output spectrum of the GSM receiver is shown in Fig. 12 for operation at a low IF of 100 kHz—the measured output closely matched the simulated results.

In our experience, the presented approach can accurately predict thermal noise contributions, sigma-delta noise contributions, systematic spurs from signal processing, image locations and magnitudes, quantization noise contributions and $1/f$ noise contribution in the complete transceiver solution. This is provided that the analog behavioral models are accurately modeling gain, noise and $1/f$ noise of the individual analog blocks. The "goodness" of the system prediction inherits the "goodness" of block models.

The presented approach, so far, does not comprehend impairments due to package. Package causes undesired cross-couplings between signals that can create undesired control loops. Impairments that could not be comprehended were the number of spurs in the receive band while transmitting and the degradation of receiver IP2 due to LO leakage [15].

VII. Conclusion

We have presented an approach that can be used for design verification of a RF SoC based on VHDL modeling and simulation. No methodology exists to date that can account for RF, analog, digital, and software from a system performance-centric perspective. The analog schematics are replicated in VHDL and block-level behavioral models are plugged in. The digital design progresses from behavioral model to RTL, gate and post-layout while system performance is evaluated in the same framework. This approach is not only used to verify the system performance but also to verify correct functionality of the digital implementation.

References

[1] K. Muhammad, R. B. Staszewski, and D. Leipold, "Digital RF processing: Toward low-cost reconfigurable radios," *IEEE Commun. Mag.*, vol. 43, no. 8, pp. 105–113, Aug. 2005.

[2] K. Muhammad *et al.*, "A discrete time quad-band GSM/GPRS receiver in a 90nm digital CMOS process," in *Proc. 2005 IEEE Custom Integrated Circuits Conf.*, Sep. 2005, pp. 804–807.

[3] K. Muhammad *et al.*, "The first fully integrated quad-band GSM/GPRS receiver in a 90-nm digital CMOS process," *IEEE J. Solid-State Circuits*, vol. 41, no. 8, pp. 1772–1783, Aug. 2006.

[4] K. Muhammad *et al.*, "A discrete-time Bluetooth receiver in a 0.13μm digital CMOS process," in *IEEE ISSCC 2004 Dig. Tech. Papers*, Feb. 2004, pp. 268–269.

[5] R. B. Staszewski *et al.*, "All-digital TX frequency synthesizer and discrete-time receiver for Bluetooth radio in 130-nm CMOS," *IEEE J. Solid-State Circuits*, vol. 39, no. 12, pp. 2278–2291, Dec. 2004.

[6] R. B. Staszewski *et al.*, "All-digital PLL and GSM/EDGE transmitter in 90nm CMOS," in *IEEE ISSCC 2005 Dig. Tech. Papers*, Feb. 2005, pp. 316–317.

[7] K. Muhammad, T. Murphy, and R. B. Staszewski, "Verification of RF SoCs: RF, analog, baseband and software," in *Proc. 2006 IEEE RFIC Symp.*, Jun. 2006, pp. 11–13.

[8] A. Mehrotra, A. Narayan, and R. Subramanian, "Next-generation silicon analysis tools for RF integrated-circuits," in *Proc. 2006 IEEE RFIC Symp.*, Jun. 2006, pp. 11–13.

[9] J. Craninckx and S. Donnay, "4G terminals: how are we going to design them?," in *Proc. 2003 IEEE Design Automation Conf.*, Jun. 2003, pp. 79–84.

[10] K. Mayaram, D. C. Lee, S. Moinian, D. A. Rich, and J. Roychowdhury, "Computer-aided circuit analysis tools for RFIC simulation: algorithms, features, and limitations," *IEEE Trans. Circuits Syst. II: Analog Digit. Signal Process.*, vol. 47, no. 4, pp. 274–286, Apr. 2000.

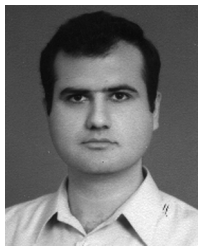
[11] A. Dunlop, A. Demir, P. Feldmann, S. Kapur, D. Long, R. Melville, and J. Roychowdhury, "Tools and methodology for RF IC design," in *Proc. 1998 IEEE Design Automation Conf.*, Jun. 1998, pp. 414–420.

[12] D. Leenaerts, G. Gielen, and R. A. Rutenbar, "CAD solutions and outstanding challenges for mixed-signal and RF IC design," in *Proc. 2001 ICCAD*, Nov. 2001, pp. 270–277.

[13] R. B. Staszewski, C. Fernando, and P. T. Balsara, "Event-driven simulation and modeling of phase noise of an RF oscillator," *IEEE Trans. Circuits Syst. I: Fundam. Theory Applicat.*, vol. 52, no. 4, pp. 723–733, Apr. 2005.

[14] K. Muhammad, R. B. Staszewski, and C. M. Hung, "Joint common mode voltage and differential offset voltage control scheme in a low-IF receiver," in *Proc. 2004 IEEE RFIC Symp.*, Jun. 2004, pp. 405–408.

[15] I. Elahi, K. Muhammad, and P. T. Balsara, "IP2 and DC offsets in the presence of leakage at LO frequency," *IEEE Trans. Circuits Syst. II: Expr. Briefs*, vol. 53, no. 8, pp. 647–651, Aug. 2006.



Khurram Muhammad received the B.Sc. degree from the University of Engineering and Technology, Lahore, Pakistan, in 1990, the M.Eng.Sc. degree from the University of Melbourne, Parkville, Victoria, Australia in 1993, and the Ph.D. degree from Purdue University, West Lafayette, IN, in 1999, all in electrical engineering.

From 1990 to 1991 and from 1993 to 1994, he worked in the Research and Development section of the Carrier Telephone Industries, Islamabad, Pakistan, where he developed board level designs for rural and urban telecommunication. He also worked for the Ghulam Ishaq Khan Institute of Engineering Science and Technology from 1994 to 1995 as a Research Associate in the Department of Electronic Engineering. In 1995, he developed fast simulation techniques for DS/CDMA systems in multipath fading environment at the Hong Kong University of Science and Technology, Kowloon, Hong Kong. Since 1999, he has been with Texas Instruments, Dallas, TX, earlier in the advance read channel development group, then in communications and control products, and currently in the wireless terminal business unit where he develops digital and mixed signal designs in fully integrated RF transceiver system-on-chip. He is currently the manager of the Digital Radio Processor Receive System Design Group and also leads the receiver

development. He is one of the original inventors of TI's DRP technology. He serves on the technical program committees of several IEEE conferences. His main research interests include CMOS mixed signal VLSI design, advance CAD techniques exploring digital design tradeoffs at circuit and system levels, low-complexity and low-power design.

Thomas Murphy received the B.S.E.E. degree from the University of Illinois at Urbana-Champaign in 1984.

He is with Texas Instruments, Inc., Dallas, TX, in the Digital RF Processor (DRP) group. He has worked with the DRP group since 2002. He has led DRP validation and verification and also led the DRP Reference Software development. He is a Member of the Technical Staff. Previously, he has worked in a variety of roles in industry and at research institutes. In 1990, he joined the staff at Center for Supercomputing Research and Development at the University of Illinois at Urbana-Champaign and contributed to the architecture and design of the Cedar supercomputer. He joined the International Supercomputing Technology Institute in Mulhouse, France, in 1994 where he was the Chief Architect for the AIDA supercomputer. From 1998 through 2002, he worked for Intrinsix Corporation in Glen Ellyn, IL, providing design services for ASIC and high-integration embedded systems.

Mr. Murphy is a Chick Evans Scholar Alumni.

Robert Bogdan Staszewski (M'94–SM'05) received the B.S.E.E. (*summa cum laude*), M.S.E.E. and Ph.D. degrees from the University of Texas at Dallas in 1991, 1992, and 2002, respectively.

From 1991 to 1995, he was with Alcatel Network Systems, Richardson, TX, working on Sonnet cross-connect systems for fiber optics communications. He joined Texas Instruments, Dallas, TX, in 1995 where he is currently a Distinguished Member of Technical Staff. Between 1995 and 1999, he was engaged in advanced CMOS read channel development for hard disk drives. In 1999, he co-started a Digital Radio Frequency Processor (DRP) group within Texas Instruments with a mission to invent new digitally intensive approaches to traditional RF functions for integrated radios in deep-submicron CMOS processes. He currently leads the DRP system and design development for transmitters and frequency synthesizers. He has authored or co-authored 40 journal and conference publications and holds 25 issued U.S. patents. His research interests include deep-submicron CMOS architectures and circuits for frequency synthesizers, transmitters, and receivers.