



저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

Master's Thesis

A Modeling Methodology for Verification of 5G NR-Band RF Transceiver

5G NR-밴드 무선 주파수 송수신기의
검증을 위한 모델링 방법

August 2021

Department of Electrical
and Computer Engineering
College of Engineering
Seoul National University

Chan Young Park

A Modeling Methodology for Verification of 5G NR-Band RF Transceiver

Thesis Advisor: Prof. Jaeha Kim

A thesis submitted to the department of
electrical and computer engineering

August 2021

Department of Electrical
and Computer Engineering
College of Engineering
Seoul National University

Chan Young Park

Confirming the master's thesis written by

Chan Young Park

August 2021

Chair	<u>Deog-Kyoon Jeong</u>
Vice Chair	<u>Jaeha Kim</u>
Examiner	<u>Taewhan Kim</u>

Abstract

In mobile RF transceiver systems, the large number of digital circuits employed to compensate or calibrate the non-idealities of the RF circuits call for models that can work within the digital verification platform, such as SystemVerilog. While baseband-equivalent real-number models (RNMs) are the current state-of-the-art for modeling RF transceivers in SystemVerilog, their simulation speeds and accuracy are not adequate predicting performance degradation. Since, its signals can only model the frequency components near the carrier frequency but not the DC offsets or high-order harmonic effects arising due to nonlinearities. Therefore, the growing impacts of nonlinearities call for nonlinear modeling of their key components to predict the overall system's performance.

This dissertation presents the models for a multi-standard, direct-conversion RF transceiver for evaluating its system-level performance and verifying its digital controllers. Also, this work demonstrates the Volterra series model for the nonlinear analysis of a low-noise amplifier circuit in SystemVerilog, leveraging the functional expression and event-driven simulation capability of XMODEL.

The simulation results indicate that the presented models, including the digital configuration/calibration logic for the 5G sub-6GHz-band and mmWave-band transceiver, can deliver 30-1800× higher speeds than the baseband-equivalent RNMs while estimating the quadrature amplitude modulation signal constellation and error vector magnitude in the presence of non-idealities such as nonlinearities, DC offsets, and I/Q imbalances. In addition, it implements functionality checkers and parameter coverage analysis to advance the completeness of system-level verification of the RF transceivers model.

Keyword : 5G and beyond 5G RF transceiver, System-level verification, Event-driven simulation, Digital and parameter coverage analysis, and Modeling methodology for memory effects

Student Number : 2019-23662

Table of Contents

Abstract	i
Table of Contents	iii
List of Figures	iv
Chapter 1. Introduction	1
1.1 Design and Verification Flow	
1.2 5G NR Band RF Transceiver IC	
1.3 Baseband–Equivalent and Passband Modeling	
1.4 Thesis Organization	
Chapter 2. Modeling and Simulation of RF Transceiver	11
2.1 Direct Conversion RF Transceiver	
2.2 Proposed Transceiver Models	
2.3 System and Simulation Performance	
Chapter 3. Nonlinear RF System Modeling	28
3.1 Volterra / Perturbation Method	
3.2 Low Noise Amplifier Example	
3.3 Nonlinearity Analysis	
Chapter 4. Coverage Analysis and Functional Verification	42
4.1 Model Parameter Coverage Analysis	
4.2 Self–Checking Testbench	

Chapter 5. Conclusion.....	54
Appendix	55
A.1 Trigonometric Equation for Non-Ideal Effects	
A.2 RNM Baseband Equivalent Modeling	
A.3 Parameter Coverage Analysis	
A.4 List of Models	
Bibliography	63
Abstract in Korean	65

List of Figures

Figure 1.1. Design and Verification Flow	3
Figure 1.2. Baseband and passband signals in RF transceiver	5
Figure 1.3. RF signals in UCM/DCM mixer	5
Figure 1.4. RF signals in frequency domain	7
Figure 1.5. Signal representation (a)RNM and (b)XMODEL	9
Figure 2.1. Overall test bench organization	12
Figure 2.2. Block diagram of the proposed transceiver model	12
Figure 2.3. Describing the functional model of a mixer with gain mismatch and DC offset calibration.....	14
Figure 2.4. Describing the functional model of an ABB circuit	16
Figure 2.5. Describing the functional model of a power amplifier with digital pre-distortion	18
Figure 2.6. Describing the functional model of a TX BB circuit that generates QAM or OFDM symbols (DAC).....	20
Figure 2.7. Describing the functional model of a RX BB circuit that recovers data from received symbols(ADC)	23
Figure 2.8. The simulated QAM signal constellations with various frequency bands, DC offsets, and I/Q imbalance conditions	24
Figure 2.9. Simulated waveforms and FFT analysis results	26
Figure 2.10. Simulation performances: run time and RMS error	27
Figure 3.1. Nonlinear system model design flow	29
Figure 3.2. Example of second-order nonlinear system	31
Figure 3.3. Nonlinear systems: complexity-performance	31
Figure 3.4. (a) Common-gate amplifier and (b) its equivalent circuit model	33
Figure 3.5. System diagrams of (a) the overall system and (b) the third-order nonlinearity	34
Figure 3.6. (a) CG amplifier with nonlinearity cancellation and (b) its equivalent circuit model	36
Figure 3.7. Filter model of the (a) conventional and (b) cancellation	

structure, and (c) system diagram of a third-order nonlinearity	37
Figure 3.8. Linearity improvement	37
Figure 3.9. Testbenches for the nonlinear RF amplifiers (a) two-tone test and (b) modulated-tone test	38
Figure 3.10. Time-domain simulation results with the event markers	38
Figure 3.11. The measured output spectrums	39
Figure 3.12. The OIP3 measurement results	41
Figure 4.1. Design verification problems related to digital-to-analog converter model	42
Figure 4.2. Design verification problems related to feedback loop amplifier model	43
Figure 4.3. Pseudocode of the module that measures the coverage results of weighted-sum models	44
Figure 4.4. Parameter coverage analysis example	45
Figure 4.5. Parameter coverage analysis process and results	46
Figure 4.6. Self-checking testbench organization for coverage analysis and functionality check	48
Figure 4.7. Simulation results of parameter coverage analysis (a) error-injection test, (b, c) coverage and detected errors	49
Figure 4.8. Testbench details and verification scenarios.....	51
Figure 4.9. Simulation results and error observability	52
Figure A.1. Describing the filter model in (a) s-domain and (b) z-domain	57
Figure A.2. Pseudocode for the proposed filter model.....	57
Figure A.3. Block diagram of the complex baseband equivalent channel	58
Figure A.4. Pseudocode of the module that measures the coverage results of filter model	61

Chapter 1. Introduction

1.1. Design and Verification Flow

Designing an IC chip is a series of highly arduous work, and it is common for painstakingly crafted chips to end up with chunks of silicon due to trivial design mistakes. Such failure may further lead to disastrous waste of development cost and development time. Therefore, for first-tape-out success, an efficient design flow and adequate verification method in each step are required. However, there is a difference in the design flow widely used when designing analog and digital ICs. First, digital designers design ICs as a top-down process from an RTL design to a physical geometric representation, whereas analog designers typically design ICs as a bottom-up process that constructs circuits using device characteristics. Inconsistencies in design flow often lead to embarrassing situations where analog and digital designers apply different specifications. In addition, analog SPICE and digital HDL models need to perform co-simulation, but it is challenging to perform top-level verification due to the slow simulation speed [1]. Therefore, a unified design flow that can efficiently perform design and verification, whether analog or digital, is required.

The design flow that effectively enables the design and verification of mixed-signal systems is the top-down of the two approaches. In the top-down flow, we first create a behavioral model to determine the architecture and specifications from the higher-level abstraction and complete the transistor-level design based on it. This design flow has a clear advantage over bottom-up flow: speed of simulation. In the bottom-up flow, simulations must be performed with circuit-level simulators such as SPICE, and verifying

the behavior of hundreds of millions of transistors takes a tremendous amount of time (a scalability issue). Consequently, it is efficient to build a behavioral model for the analog circuit and check the system-level functionality with the digital model on a single platform using the logic simulator in the early design cycle.

The top-down design flow for a mixed-signal system is as follows: First, the designer determines the system's behavior and creates an analog behavioral model (*real-number model* or *xmodel*) and a digital RTL model [2]. In this process, it is crucial to decide how to partition into subsystems. The designer must determine each partitioned block's specification to achieve the overall system's performance goals. After that, digital RTL is converted into gate-level netlist through synthesis (*Design-compiler*, *Prime-time*, *Formality*, ...) and verification (*VCS*, *Xcelium*, ...) process. At the same time, analog models are converted into physical libraries such as FRAM (*Milkyway*, ...) through schematic/layout generation (*Virtuoso*, ...), verification (*HSPICE*, *FineSim*, ...), and abstraction (*Abstract-Generator*, ...) process. A top-level Verilog-based pre-netlist is created by combining the netlists (*digital gate-level netlist*, *analog model netlist*, *FRAM*), standard cells, and pads generated through the previous process. Afterward, post-P&R top-level verification (*VCS/XA with SDF Annotate*, ...) is performed with the top-level post-netlist created through the auto place-and-route process (*IC compiler*, *Custom compiler*, ...), and the final .gds file for production is generated [3].

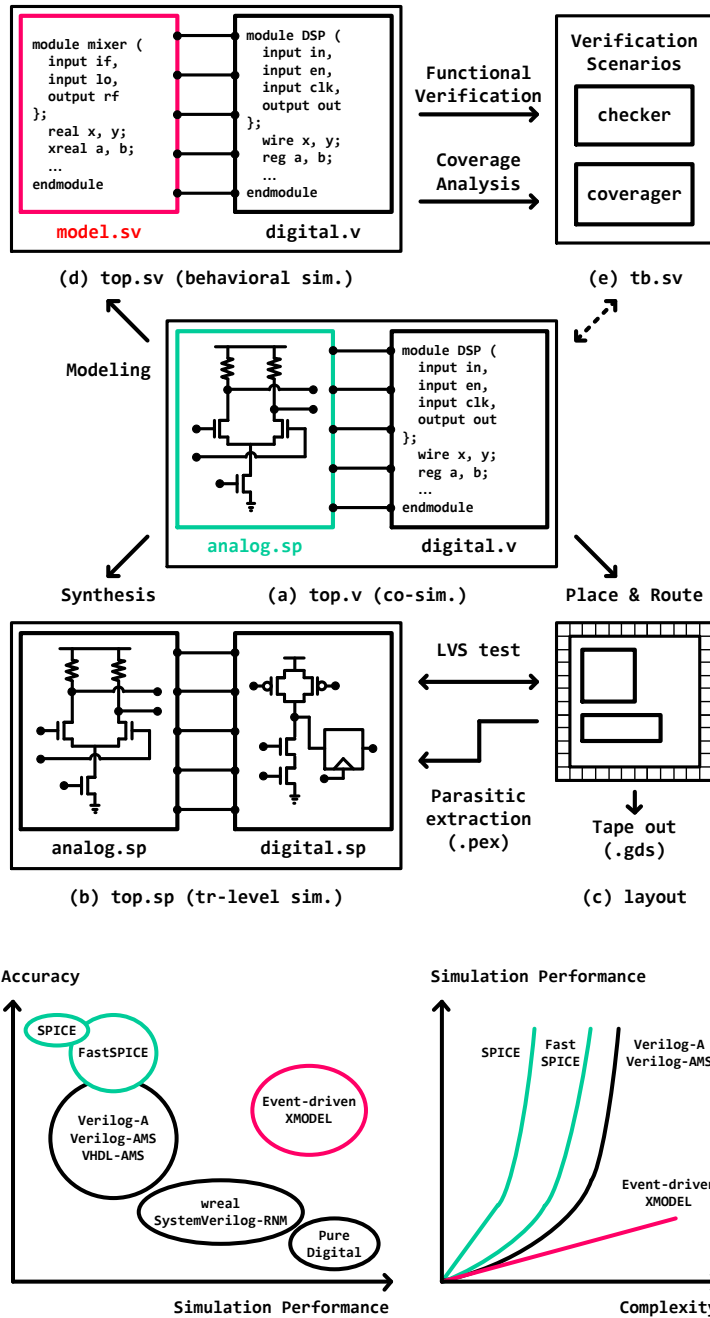


Figure 1.1. Design and verification flow.

1.2. 5G NR–band RF Transceiver IC

With the growing interaction between the RF analog front–end and the digital calibration/selection logic in 5G multi–standard RF transceivers (TRX), an efficient simulation solution that is entirely based on SystemVerilog is required to verify these transceivers’ functionality and to evaluate their performance. In RF transceivers that support various legacy bands and carrier aggregation, over 5,000 configuration bus bits are controlled by large complex digital logic to select a certain frequency band of operation, and various digital calibration and signal processing techniques are employed to improve their communication performance in the presence of non–idealities [4, 5].

To verify the functionality and evaluate performance metrics, such as error vector magnitude (EVM), fast time–domain simulations, including analog/RF and digital subsystems, are necessary. In general, the standard for analog block verification is a transistor–level simulation using SPICE, which is very accurate using the ODE solver. Nevertheless, the simulation is very slow, making it almost impossible for system–level verification [6,7]. Most critical errors encountered in practical designs are not immense, baffling errors inside the analog block but minor, mild errors such as pin connection errors, inverted polarity, incorrect bus order, or pins connected to the incorrect power domain [8]. It seems like a waste of time to do lengthy simulations with SPICE to find these cute errors. In this dissertation, to address the slow simulation speed of SPICE or SPICE–HDL co–simulation, multi–standard RF transceiver models are proposed that can run entirely within SystemVerilog and deliver 30–1800× faster speeds than those of the baseband–equivalent real–number models (RNMs) [9,10,11], leveraging the event–driven simulation of XMODEL [12].

1.2. Baseband–Equivalent and Passband Modeling

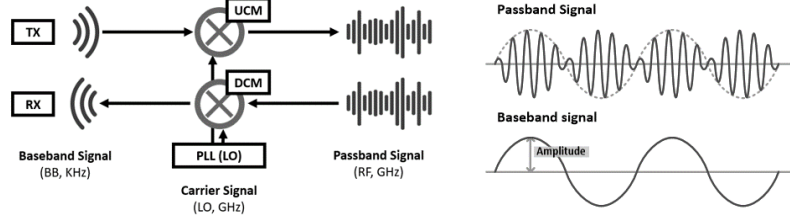


Figure 1.2. Baseband and passband signals in RF transceiver.

When a signal is transmitted through a channel, it is modulated in a high–frequency band in a wireless communication system. This technique allows long–distance transmission with a small antenna and increases signal bandwidth, making it more robust against noise and interference. In this case, the original signal to be transmitted in the low–frequency band is called a baseband signal. Furthermore, transmission efficiency can be increased by applying a quadrature amplitude modulation (QAM) method that sums two amplitude–modulated baseband signals having the same frequency but different phases of 90 degrees. In this case, when the in–phase component is $s_I(t)$, and the quadrature component is $s_Q(t)$, the baseband signal is expressed as $s_B(t) = s_I(t) + js_Q(t)$ in the form of a complex number and is referred to as a complex baseband equivalent signal [13,14].

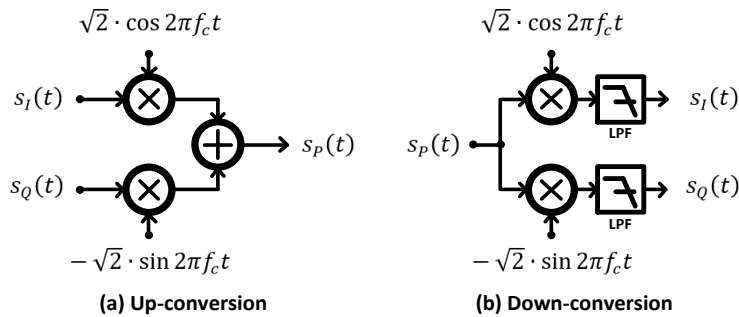


Figure 1.3. Baseband and passband signals in UCM/DCM mixer.

The baseband signal is mixed (upconverted) with the carrier signal at the transmitter, and any modulated RF (passband or bandpass) signal can be represented as:

$$\begin{aligned} s_p(t) &= \text{Re}[s_{BB}(t) \cdot (\sqrt{2} \cdot e^{j2\pi f_c t})] \\ &= \sqrt{2} \cos 2\pi f_c t \cdot s_I(t) - \sqrt{2} \sin 2\pi f_c t \cdot s_Q(t) \end{aligned} \quad (1.1)$$

where f_c is the carrier frequency. The amplitude of the carrier is multiplied by $\sqrt{2}$ to match the power of the transmitted/received signal. This signal is demodulated to a baseband signal at the receiver.

$$s_p(t) \cdot (\sqrt{2} \cdot \cos 2\pi f_c t) = s_I(t) + s_I(t) \cdot \cos 4\pi f_c t - s_Q(t) \cdot \sin 4\pi f_c t \quad (1.2)$$

Equation (1.1) describes the down-conversion process in the in-phase path. The passband signal is mixed with the carrier signal to produce the $s_I(t)$ signal and the $2f_c$ higher-order signal. Removing only the high-frequency signal component with a low-pass filter can be restored to an in-phase baseband signal.

Currently, there are mainly two approaches for modeling RF systems using RNMs: fast-but-inaccurate baseband-equivalent (BBEQ) modeling and accurate-but-slow passband modeling [4,9,13]. To express high-frequency RF signals that are modulated by low-frequency data, the baseband-equivalent models assume that the RF signals have a fixed-frequency carrier and only express its magnitude and phase information or, equivalently, the in-phase (I) and quadrature-phase (Q) information with a small number of events [14].

However, when these signals have to include the passband information, for example, to model frequency tones far from the carrier frequency such as the DC or high-order harmonic components, signals must be sampled with a sufficiently fine time-step to avoid aliasing, and the key benefits of the BBEQ modeling are lost. In these cases, the passband models that express the RF signals

may as well be used for their direct computations. However, the required large number of events typically slows down the simulations and limits the duration of the simulation to a few symbols, which is not sufficient to evaluate EVM and collect signal constellation, requiring at least 1,000 symbols [14].

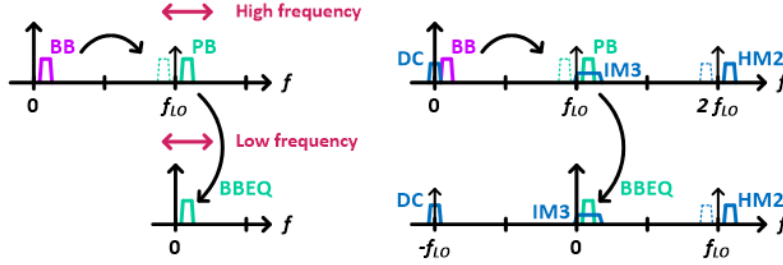


Figure 1.4. Baseband and passband signals in frequency domain.

In comparison, the event-driven signal representation and simulation algorithm used by XMODEL [15] can be an effective solution to address the aforementioned challenges. XMODEL is a plug-in extension to SystemVerilog developed by Scientific Analog [6,9,10] and can perform efficient event-driven simulations of analog models by using functional expressions for their analog signals, without incurring the overheads of AMS co-simulation. In other words, an event in XMODEL does not indicate a change in value, but a change in the functional expression, describing how the signal changes over time. It expresses the continuous-time waveform of an analog signal $x(t)$ using the following functional expression, which also has its counterpart $X(s)$ in the Laplace domain [15]:

$$x(t) = \sum_i c_i t^{m_i-1} e^{-a_i t} u(t) \rightarrow X(s) = \sum_i \frac{c_i}{(s + a_i)^{m_i}} \quad (1.3)$$

In other words, each event during the simulation updates the values of the coefficients c_i 's, m_i 's, and a_i 's, which collectively describe how the signal varies with time according to (1.3), thus enabling an efficient, event-driven simulation. The key difference between this approach and the RNM-based approaches is that the

former does not rely on a large number of events to express a time-varying RF/analog signal. As the expression in (1.3) can contain an arbitrary number of terms, it can include additional frequency components without triggering additional events. Furthermore, XMODEL offers a set of SystemVerilog primitives that can describe the diverse functionalities of analog circuits, and the resulting model is fully compatible with SystemVerilog. The XMODEL primitives, such as *sin_gen* and *multiply*, each of which performs the operation suggested by its name, make it easy to compose models simply by connecting them together [15], and suitable for verifying mixed-signal feedback loops including both the analog and digital models on a single platform of SystemVerilog.

Fig. 1.5 compares the RNM and XMODEL models for generating a sinusoidal signal of which amplitude is controlled by a digital code (*ctrl_amp*). The RNM model in Fig. 1.5(a) needs to evaluate a $\$sin()$ function at a constant time-step interval, which is set by the period of the clock (*clk*). This time-step interval will greatly determine the speed and accuracy of the simulation. On the other hand, the XMODEL model in Fig. 1.5(b) is described using a set of primitives and its simulation triggers events only when there is a change in the digital code (*ctrl_amp*). It is because (1.1) can express a sinusoidal function directly, without compromising accuracy.

```

module sin_gen #(
    parameter freq = 3.5e9,           // frequency
    parameter tsp = 10e-9             // time step period
)(
    input reg clk,                    // sampling clock
    input reg [11:0] ctrl_amp,        // control amplitude
    output real sin_out
);
real amp, t;
assign amp = 2 * (real'(ctrl_amp)) / 4096;

always @(posedge clk) begin
    t = $realtime;
    sin_out = amp * $sin(2*M_PI*freq*t*tsp);
end
endmodule

```

(a) Example of the conventional RNM model.

```

module sin_gen #(
    parameter freq = 3.5e9                // frequency
)(
    input reg [11:0] ctrl_amp,            // control amplitude
    output real sin_out
);
xreal amp, sin_unit;

dac    #(.num_bit(12), .min(0.0), .max(2.0))
      XP0 (.in(ctrl_amp), .out(amp));
sin_gen #(.freq(freq)) XP1 (sin_unit);
multiply XP2 (.in(amp, sin_unit), .out(sin_out));

endmodule

```

(b) Example of the proposed XMODEL model.

Figure 1.5. Signal representation (a) RNM and (b) XMODEL.

As the nonlinearities in mobile radio–frequency (RF) transceiver systems become prevalent, the SystemVerilog models that can reflect the resulting harmonic distortions and system dynamics without a significant loss in the simulation speed while accurately representing high–frequency signals are required. In real–number modeling (RNM), a sampling frequency of at least twice the maximum signal frequency is required according to the Nyquist theorem to accurately represent a high–frequency (\sim GHz) RF signal. It means the simulations must run for a long RF frame time (\sim ms) with a very small time–step, which can significantly impact the speed.

1.3. Thesis Organization

This dissertation showcases a multi-standard 5G RF transceiver model using XMODEL. First, chapter 2 addresses the key challenges in modeling the direct-conversion RF transceiver and describes the presented RF transceiver models. Furthermore, we discuss the verification and simulation results for the 5G sub-6 GHz and mmWave-band operations. Chapter 3 explains the Volterra-series modeling with the perturbation method and details the process of modeling the CG-LNA structure with and without the linearity-improvement circuits. Chapter 5 reviews parametric coverage analysis and explains how to achieve system-level verification with functional checkers and coverage analysis. And finally, chapter 5 concludes the paper.

Chapter 2. RF Transceiver Model

2.1. Direct–Conversion RF Transceiver

While the direct–conversion architecture has the lower implementation costs and simpler frequency plan that can cater to multiple standards when compared to its heterodyne counterparts [16,17], it may be susceptible to the degradations in the communication performance due to the DC offsets and I/Q imbalance issues. For instance, the LO signal that leaks into the mixer input can cause self–mixing and create a DC offset in the mixer output. On the other hand, the gain/phase mismatch can cause an imbalance between the I/Q phases, which can destroy the orthogonality of the quadrature amplitude modulation (QAM) symbols. Consequently, most modern transceivers employ digital calibration loops to compensate for such imbalances.

As mentioned previously, as the baseband–equivalent models are not suitable for expressing the DC offsets or high–order harmonics in the RF signals, various performance degradations in the direct–conversion RF transceivers are difficult to predict due to DC offsets or I/Q imbalances [16,17,18]. This also implies that the baseband–equivalent models are not suitable for verifying the digital calibration loops that address these non–idealities. Nonetheless, the passband models based on RNM cannot be used as an alternative solution because the number of events required to express high–frequency RF signals makes the simulation/verification impractical. The next chapter presents a passband model of a multi–standard, direct–conversion RF transceiver that utilizes the XMODEL event–driven algorithm. While delivering faster speeds, the proposed models can accurately model the DC offset and I/Q imbalance effects, as well as verify the operation of their digital calibration loops.

2.2. Proposed Transceiver Models

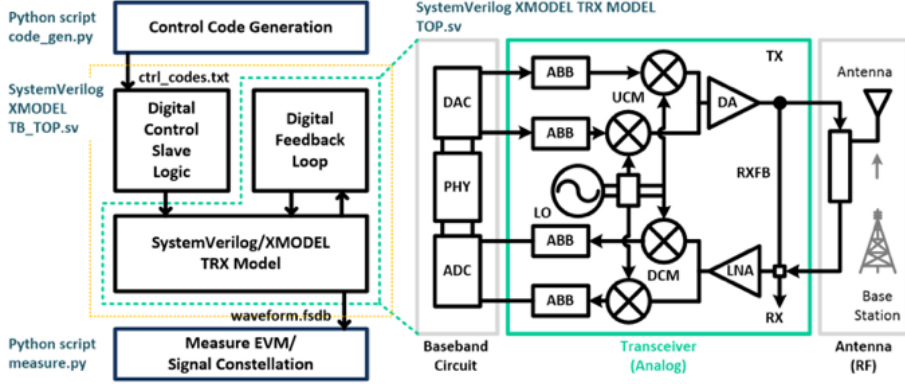


Figure 2.1. Overall test bench organization.

This chapter describes the proposed SystemVerilog models for the direct-conversion RF transceiver. Fig. 2.1 shows the overall testbench configuration for the system. The system comprises mainly an analog/digital transceiver model described in System Verilog and Python scripts that can generate control codes (*code_gen.py*) and compute system performance (*measure.py*) from the simulated results. The analog parts of the transceiver are modeled using XMODEL primitives, whereas the digital parts are described in pure Verilog, which can be synthesized into gate-level descriptions after their functionalities are verified.

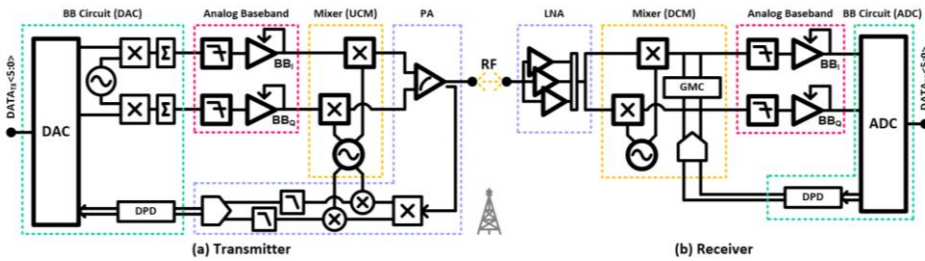
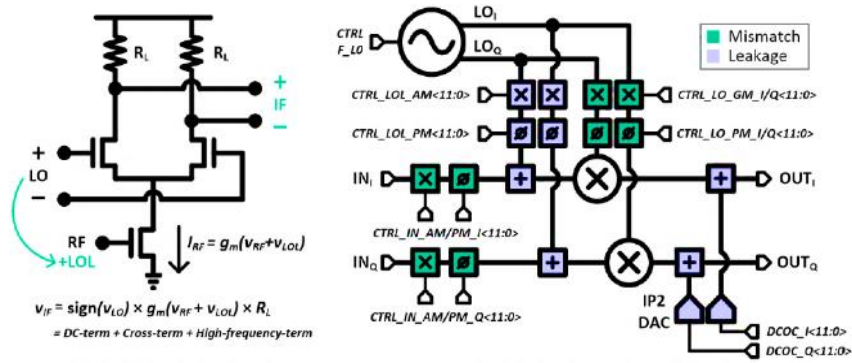


Figure 2.2. Block diagram of the proposed transceiver model.

The RF transceiver model is designed with a direct-conversion structure, as shown in Fig. 2.2. The transmitter (TX) and receiver (RX) models are composed of a baseband circuit block, analog baseband blocks, mixer blocks, and amplifier blocks. The performance of the system is predicted through the chain simulation of sending modulated symbols, such as the QAM or OFDM on the transmitter side and restoring the symbols to data on the receiver side.



(a) Single-balanced mixer (b) Block diagram of the mixer model.

```

module mixer (
    input reg [11:0] ctrl_IN_AM_I,           // IN_I gain mismatch
    input reg [11:0] ctrl_IN_PM_I,           // IN_I phase m.
    input reg [11:0] ctrl_IN_AM_Q,           // IN_Q gain m.
    input reg [11:0] ctrl_IN_PM_Q,           // IN_Q phase m.
    input reg [11:0] ctrl_LO_AM_I,           // LO_I amplitude
    input reg [11:0] ctrl_LO_PM_I,           // LO_I phase (cos)
    input reg [11:0] ctrl_LO_AM_Q,           // LO_Q amplitude
    input reg [11:0] ctrl_LO_PM_Q,           // LO_Q phase (sin)
    input reg [11:0] ctrl_LOL_AM_I,          // LO_I leakage mag.
    input reg [11:0] ctrl_LOL_PM_I,          // LO_I leakage phase
    input reg [11:0] ctrl_LOL_AM_Q,          // LO_Q leakage mag.
    input reg [11:0] ctrl_LOL_PM_Q,          // LO_Q leakage phase
    input reg [11:0] DCOC_I,                 // I-path DC offset
    input reg [11:0] DCOC_Q,                 // Q-path DC offset
    input xreal IN_I, IN_Q,                  // input RF signals
    input xreal LO_I, LO_Q,                  // input LO signals
    output xreal OUT_I, OUT_Q                 // output RF signals
);
// Input RF and LO signals
xreal g_IN_I, g_IN_Q, g_LO_I, g_LO_Q;      // gain
xreal s_IN_I, s_IN_Q, s_LO_I, s_LO_Q;      // scaled signal
xreal p_IN_I, p_IN_Q, p_LO_I, p_LO_Q;      // phase
xreal d_IN_I, d_IN_Q, d_LO_I, d_LO_Q;      // delayed signal

dac    #(.num_bit(12), .min(0.0), .max(2.0))
      XD0 (.in(ctrl_IN_AM_I), .out(g_IN_I));
multiply XM0 (.in(g_IN_I, IN_I), .out(s_IN_I));
dac    #(.num_bit(12), .min(0.0), .max(2.0))
      XD1 (.in(ctrl_IN_PM_I), .out(p_IN_I));
delay  XE1 (.delay(p_IN_I), .in(s_IN_I), out(d_IN_I));
...    // same for d_IN_Q, d_LO_I, d_LO_Q

```

```

// LO leakage signals added to input RF signals
xreal g_LOL_I, g_LOL_Q, s_LOL_I, s_LOL_Q; // scaled signal
xreal p_LOL_I, p_LOL_Q, d_LOL_I, d_LOL_Q; // delayed signal
xreal MIX_IN_I, MIX_IN_Q; // Mixer input signals

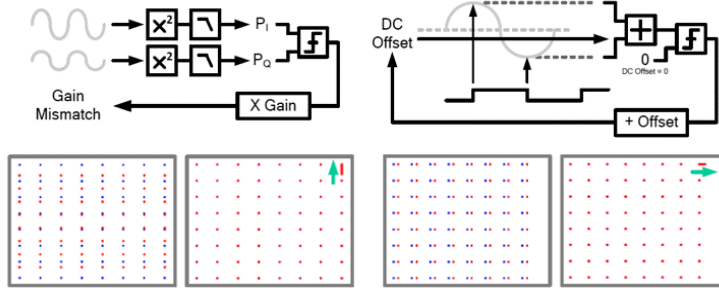
dac #(.num_bit(12), .min(0.0), .max(2.0))
    XD8 (.in(ctrl_LOL_AM_I), .out(g_LOL_I));
multiply XM4 (.in(g_LOL_I, LOL_I), .out(s_LOL_I));
dac #(.num_bit(12), .min(0.0), .max(2.0))
    XD9 (.in(ctrl_LOL_PM_I), .out(p_LOL_I));
delay XE3 (.delay(p_LOL_I), .in(s_LOL_I), out(d_LOL_I));
add XA0 (.in(d_LOL_I, d_LOL_I), .out(MIX_IN_I));
... // same for MIX_IN_Q

// Mixing operation
xreal MIX_OUT_I, MIX_OUT_Q; // Mixer output signals
multiply XM6 (.in(MIX_IN_I, d_LO_I), .out(MIX_OUT_I));
multiply XM7 (.in(MIX_IN_Q, d_LO_Q), .out(MIX_OUT_Q));

// DC offset calibration
xreal DC_offset_I, DC_offset_Q; // gain
dac #(.num_bit(12), .min(0.0), .max(2.0))
    XD12 (.in(DCOC_I), .out(DC_offset_I));
add XA3 (.in(MIX_OUT_I, DC_offset_I), .out(OUT_I));
... // same for OUT_Q
endmodule

```

(c) Proposed pseudocode for the mixer model



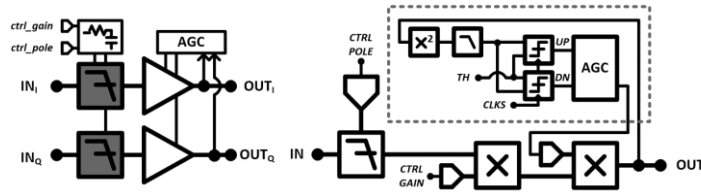
(d) Gain mismatch calibration (e) DC offset calibration

Figure 2.3. Describing the functional model of a mixer with gain mismatch and DC offset calibration.

The mixer block performs frequency conversion by multiplying the RF and carrier signals in time domain, and the modeling aims to reflect the nonlinear factors of the mixer that degrade the performance of the system. Any mismatch and leakage can cause a DC offset or gain/phase mismatch between the in-phase and quadrature-phase paths [19]. The proposed method can simulate the passband signal in a fully event-driven method; thus, the output signal can be calculated by multiplying the actual RF/carrier signals reflecting the amplitude/phase error and leakage in the time domain. Fig. 2.3 shows the pseudo code of the proposed mixer model and illustrations of the calibration methods, respectively. First, for the RF signals ($IN_{I/Q}$) and carrier signals ($LO_{I/Q}$) applied to the mixer model,

the gain and phase values are distorted by means of externally assigned control values, and any difference between these values may reflect the gain/phase mismatch between paths I and Q. The 12-bit gain/phase control codes are converted into an analog gain/phase value using the *dac* primitive, and the input signals are amplified by the gain value using the multiply primitive and delayed by the phase value using the *delay_var* primitive thereafter. Furthermore, some carrier signals (LO leakages) are leaked in the same manner and are added to the input signal using the add primitive. The input signals of the combined mixer are multiplied by the carrier signal, thus multiplying the actual mixing behavior as well.

Figs. 2.3 (b) and (c) show the gain mismatch calibration (GMC) method [20,21], DC offset calibration (DCOC) method [22], and signal constellation diagrams before and after the calibration. When the GMC calibration is turned on, the RX GMC block measures the power of each down-converted signal in paths I and Q, and then compares the magnitudes of the two signals. If the Q-path gain is greater than the I-path gain, it is lowered through the feedback loop, and vice versa. The DC offset value is calculated from the RX ADC's DCOC block and then assigned to the mixer through a feedback loop. These 12-bit offset values are converted to analog offset values via the *adc* primitive and then subtracted from each path using the add primitive. The DCOC block uses a set of four clock phases to sample the signals of the paths I and Q twice each, and searches for the offset values that makes the absolute difference between the two sample values equal to zero.



(a) Block diagram of the ABB model.


```

module abb_filter (
    input reg [5:0] ctrl_pole,           // control amplitude
    input reg [5:0] ctrl_gain,          // control phase
    input xreal IN,                     // input RF signal
    output xreal OUT                    // output RF signal
);
// Initial filter's parameters
int gain, polesN[10], zerosN[8]; // gain
int poles[10] = '{4002664.0, -11133886.0, 11798848.0,
-5631004.0, 11798848.0, 5631004.0, 4002664.0,
11133886.0, 100000000.0, 0.0 };
int zeros[8] = '{ -0.0, -25312276.0, -0.0, -61109241.0,
0.0, 61109241.0, 0.0, 25312276.0};
assign gain = 0.1 * (real'(ctrl_gain)/63 - 1);

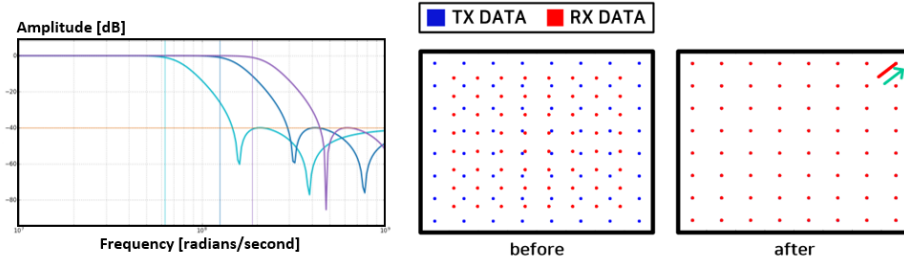
// Bandwidth variable filter's parameters
genvar i, j;
generate
    for(i=0; i<10; i=i+1) begin
        assign polesN[i] = real'(ctrl_pole) * poles[i];
    end
    for(j=0; j<8; j=j+1) begin
        assign zerosN[i] = real'(ctrl_pole) * zeros[i];
    end
endgenerate

// Filter model
filter_var #(.num_poles(10), .num_zeros(8))
XF0 (.poles(polesN), .zeros(zerosN), .in(in), .out(out));

endmodule

```

(b) Proposed pseudocode for the ABB model.



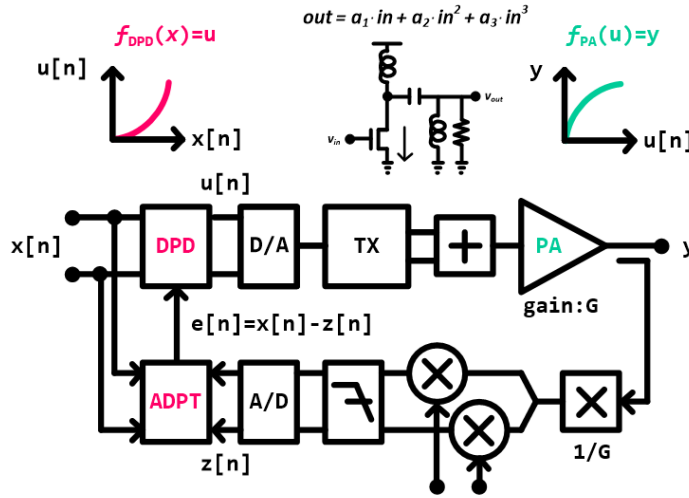
(c) Frequency response. (d) Signal constellation diagram.

Figure 2.4. Describing the functional model of an ABB circuit.

$$H(s/k) = \prod_{j=1}^{N_z} \left(1 + \frac{s}{k \cdot z_j}\right) / \prod_{i=1}^{N_p} \left(1 + \frac{s}{k \cdot p_i}\right) \quad (2.1)$$

The analog baseband model (ABB) filters the high-frequency components that are generated during the modulation process on the TX and RX sides and automatically controls the signal power to fit within a defined range. As shown in Fig. 2.4, the ABB model consists of a filter block and a variable-gain amplifier block. First, the filter model is designed with a Chebyshev type-2 structure using the channel bandwidth value as the cut-off frequency. As the characteristics of the filter are determined by the parameters (gain,

poles, and zeros) of the filter primitives, the values of these parameters must be changed according to the digital control code to design a variable-bandwidth filter. Furthermore, the variable-gain amplifier of the ABB maintains the amplitude of the output signal within a certain range through a feedback loop that automatically controls the amplifier gain. The input signals are amplified by the gain that is determined by the 12-bit external bus (*gain_dac*) and the gain generated through the AGC feedback loop (*gain_agc*). In the AGC loop, first, the power is calculated from the output signal (*out*) as in GMC, and this value is compared with a reference value (*th*) to ensure that the level of the output signal is within the reference level.



(a) Block diagram of the PA model.

```

module dpd (
    input reg clk,                // digital clock
    input reg [1:0] calib,        // calibration mode
    input xreal IN,               // PA output signal
    input xreal LO_I, LO_Q,       // input LO signal
    output reg [11:0] DPD7, DPD5 // control phase
);
xbit clkx;
xbit [11:0] DPD_Ix, DPD_Qx;
reg [11:0] DPD_I, DPD_Q, delta;
xreal ins, dcm_I, dcm_Q, fil_I, fil_Q, smp_I, smp_Q;

initial begin
    DPD7 = 12'b0000_0000_0000;
    DPD5 = 12'b0000_0000_0000;
    delta = 12'b0000_0000_0010;
end

bit_to_xbit XB0 (.in(clk), .out(clkx));
scale #(.scale(1.0/gain)) XP0 (.in(in), .out(ins));

```

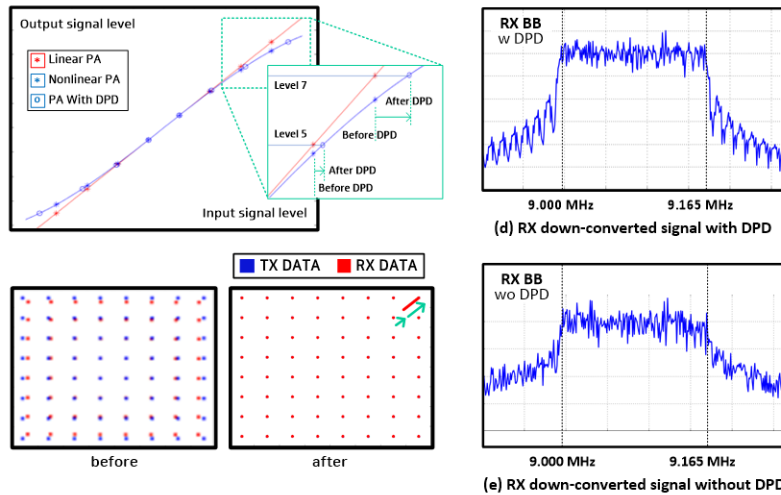
```

// I-path DPD feedback loop
multiply XM0 (.in(ins, LO_I), .out(dcm_I));
filter    #(.poles(pole)) XF0 (.in(ins, LO_I), .out(fil_I));
sample    XS0 (.in(fil_I), .trig(clkx), .out(smp_I));

adc       #(.num_bit(12), .min(9.0), .max(-9.0))
          XA0 (.in(smp_I), .out(DPD_Ix));
bit_to_xbit #(.width(12)) XB0 (.in(DPD_Ix), .out(DPD_I));
...       // same for Q-path
always @(posedge clk) begin
    if(calib == 2'b10) begin                // TX DPD (LV5)
        if(DPD_I > 12'b1100_0111_0001) begin
            if(DPD5 > 12'b0000_0000_0001) begin
                DPD5 <= DPD5 - delta;
            ...
        else if (DPD_I < 12'b1100_0111_0001) begin
            if(DPD5 < 12'b1111_1111_1110) begin
                DPD5 <= DPD5 + delta;
            ...
        else if(calib == 2'b11) begin        // TX DPD (LV7)
            if(DPD_I > 12'b1110_0011_1000) begin
                if(DPD7 > 12'b0000_0000_0001) begin
                    DPD7 <= DPD7 - delta;
                ...
            else if (DPD_I < 12'b1110_0011_1000) begin
                if(DPD7 < 12'b1111_1111_1110) begin
                    DPD7 <= DPD7 + delta;
                ...
            ...
        end
    endmodule

```

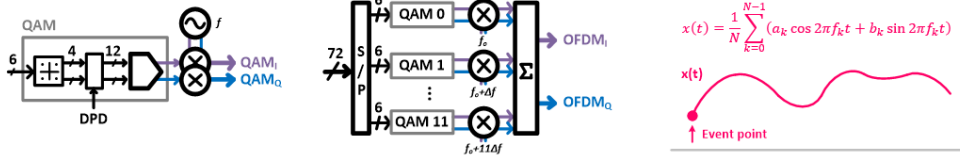
(b) Proposed pseudocode for the PA model.



(c) Nonlinear characteristic. (d) Signal constellation diagram.

Figure 2.5. Describing the functional model of a Power amplifier with digital predistortion.

The power amplifier (PA) block amplifies large power input signals outside of the linear region owing to the transconductance of the transistor. Consequently, the non-linearities should be compensated to avoid signal distortion. The gain-compression relationship between the input and output signal can be defined by



(b) Block diagram of the QAM/OFDM symbol generation model.

```

module qam_gen (
    input reg clk,                                // digital clock
    input reg [1:0] calib,                        // calibration mode
    input reg [11:0] DPD7, DPD5                  // control phase
    output real amp_I, amp_Q                      // input LO signal
);

reg [5:0] data;                                // transmitted data
reg [7:0] qam;                                 // 64-QAM informs.
reg [11:0] qam_i, qam_q;

always @(posedge clk) begin                    // data generation
    if(calib == 00) begin
        data[5:0] = prbs_data;
    else if(calib == 11) begin
        data[5:0] = 6'b001100;
    ...
end

always @(posedge clk) begin                    // encode data to
    case(data)                                QAM
        6'b000000 : qam <= 8'b1000_1000;
    ...
end

always @(posedge clk) begin                    // In-phase QAM
    case(qam[3:0])
        4'b0000 : qam_i <= 12'b0001_1100_0111 - DPD7;
        4'b0010 : qam_i <= 12'b0011_1000_1110 - DPD5;
    ...
        4'b1100 : qam_i <= 12'b1100_0111_0001 + DPD5;
        4'b1111 : qam_i <= 12'b1110_0011_1000 + DPD7;
    end
    ... // same for Quadrature-phase QAM

always @(posedge clk) begin                    // I/Q QAM level
    amp_I = real'(qam_i) * 18/4095 - 9;
    amp_Q = real'(qam_q) * 18/4095 - 9;
end

endmodule

```

(c) Proposed pseudocode for the QAM generation model.

Figure 2.6. Describing the functional model of a TX BB circuit that generates QAM or OFDM symbols (DAC).

OFDM (Orthogonal-Frequency-Division-Multiplexing) is a method of multiplexing a transmission signal by modulating it with multiple orthogonal subcarriers. As shown in Figure 1, the OFDM modulator divides a data stream with a high data rate into a plurality of data streams with a low data rate, modulates and multiplexes each

subcarrier, and transmits the subcarriers in parallel.

When $d_k = a_k + jb_k$ is a complex QAM symbol in the k -th subcarrier, the N -point IFFT OFDM symbol $x(n)$ may be expressed as a discrete signal as in (2.2). The resulting OFDM signal is in digital form:

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} d_k e^{\frac{j2\pi kn}{N}} \quad (2.2)$$

In general, RF symbols are digitally generated, as shown in the preceding equation; however, these signals can be equivalently denoted as analog signals as shown in (2.3) and can be implemented only as one event in XMODEL.

$$x(t) = \frac{1}{N} \sum_{k=0}^{N-1} (a_k \cos 2\pi f_k t + b_k \sin 2\pi f_k t) \quad (2.3)$$

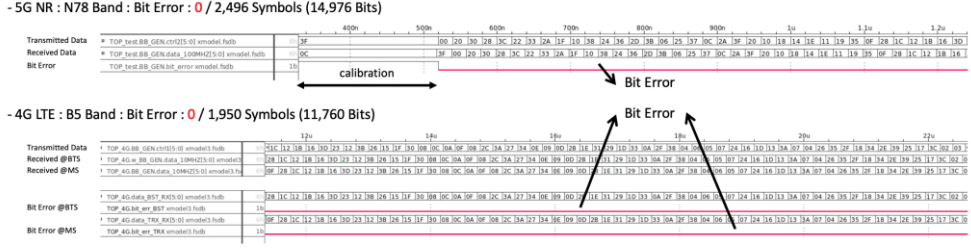
The baseband circuit (BB) block models the function used in the transceiver modem to process signals on both the TX and RX sides. The TX side performs a DAC function that generates pseudo-random binary sequence data and converts it to a QAM/OFDM symbol, and the RX side performs the reverse function of the transmitter to restore the data, that is, the ADC function.

Fig. 2.6 shows the block diagrams of the QAM and OFDM symbol generation models. When the “*qam_gen*” block converts the 6-bit data into I/Q QAM level which reflects the DPD information, the “cosine” and “sine” signals, each consisting of a sub-carrier frequency, are multiplied and sent to the TX side. In addition, the OFDM generation block models the behavior of the inverse fast Fourier transform, which then multiplies the signals generated from 12 QAM generation blocks by each sub-carrier signal and combines all these signals [14].

(a) Block diagram of the ADC and ADC unit model.

(b) Block diagram of the ADC and ADC unit model.

(c) Proposed pseudocode for the ADC model.



(d) Bit-error detection.

Figure 2.7. Describing the functional model of a RX BB circuit
That recovers data from received symbols (ADC).

Fig. 2.7 describes the ADC model, which samples the output signal of the RX ABB and restores this information to the original data sequences. The QAM and DC offset information can be obtained by sampling these two signals with a 4-phase clock. First, the “clk” clock signal with period “time_sym” is delayed by $T/4$, $T/2$, and $3T/4$ to generate $clkq$, $clkb$, and $clks$, respectively. When the clock ($clkq$ or $clkb$) is triggered, each input signal ($BB_{I/Q}$) is sampled by the sample primitive and converted into 4-bit QAM level information in the “adc_unit” model. Finally, an 8-bit QAM signal (4 bits in each I/Q path) is demapped to a 6-bit original data sequence.

2.3. System and Simulation Performance

The EVM metric measures the distance of the QAM constellation points from the reference locations and is computed as a root-mean-squared magnitude of these error vectors normalized to the ideal signal level. Fig. 2.8 shows the signal constellation diagrams and EVM values, obtained by post-processing the simulated results in Python. For each constellation diagram, the blue and red dots represent the I/Q reference and I/Q received symbols, respectively.

$$EVM = \frac{\sqrt{\frac{1}{N} \cdot \sum_{n=0}^{N-1} I_{err}^2(n) + Q_{err}^2(n)}}{EVM \text{ Normalization Ref.}} \times 100\% \quad (2.4)$$

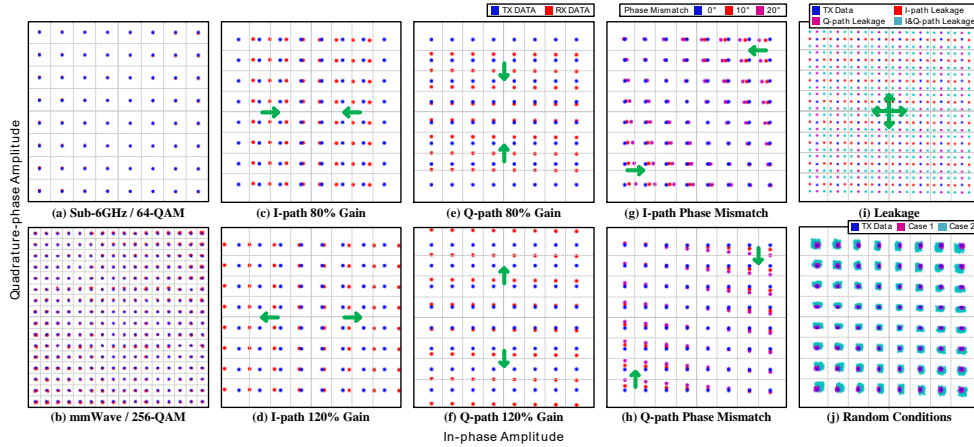


Figure 2.8. The simulated QAM signal constellations with various frequency bands, DC offsets, and I/Q imbalance conditions.

Fig. 2.8(a) shows the 64-QAM signal constellation when the transceiver model is operating at the 5G NR n78 band, with a 3.5-GHz carrier frequency and a 300-Mbps data rate. Since the up-link and down-link uses the same carrier frequency, the testbench checks the error by running a 200- μ s long simulation and comparing the signals received by the receiver feedback path (RXFB) with the transmitted signals. The simulated EVM was 0.78%. On the other

hand, Fig. 2.8(b) shows the 256-QAM signal constellation when the transceiver model is operating at the 5G NR n257 band, which is one of the next-generation mmWave bands of 3GPP specification [14]. It supports a data rate of 1.6Gbps at a 28-GHz carrier frequency. The simulated EVM after a 100- μ s long simulation was 1.23%.

Fig. 2.8(c)–(j) show the simulated signal constellations for the transceiver operating at the 5G NR n78 band with various non-ideality conditions assumed. For instance, Fig. 2.8(c)–(f) show the results with various gain mismatches between the I- and Q-paths, leading to the distortion in the constellation diagram in the vertical or horizontal direction. Fig. 2.8(g)–(h) show the results with the phase mismatch between the I- and Q-paths, where the constellation points are first rotated but then shaped into a rhombus by the AGC loop.

Fig. 2.8(i) shows the results with various LO leakage conditions causing DC offsets. The resulting DC offset shifts the constellation points either horizontally or vertically. The dots color-coded in red, blue, and turquoise in Fig. 2.8(i) indicate the simulated shifts in the 64-QAM constellations due to a $\pm 1\Delta$ -amplitude leakage into the I-path input, Q-path input, and both, respectively.

The proposed model and testbench can also randomize these non-ideality conditions and Fig. 2.8(j) shows the results for two randomly-selected cases, giving the EVM of 1.81% and 4.56%, respectively. Hence, it is possible to run a set of simulations each with different non-ideality conditions and verify whether the offset/gain calibration loops properly settle and whether the transceiver yields an EVM within the desired specification.

Figs. 2.9(a) to (d) show the simulated waveforms assuming a non-ideal case, where the DC offset of the I-path is -0.5 , and the gain of the Q-path is 20% greater than that of the I-path. Figs. 2.9(e) and (f) show the FFT analysis results of OFDM symbols with 15 kHz

spacing from 9 to 9.165 MHz and 3.5 GHz up/down-link: the OFDM symbol, the up-converted signal, and RX signals with and without DPD calibration in the PA. Without DPD, a spectral regrowth due to the intermodulation components deteriorates the RX sensitivity.

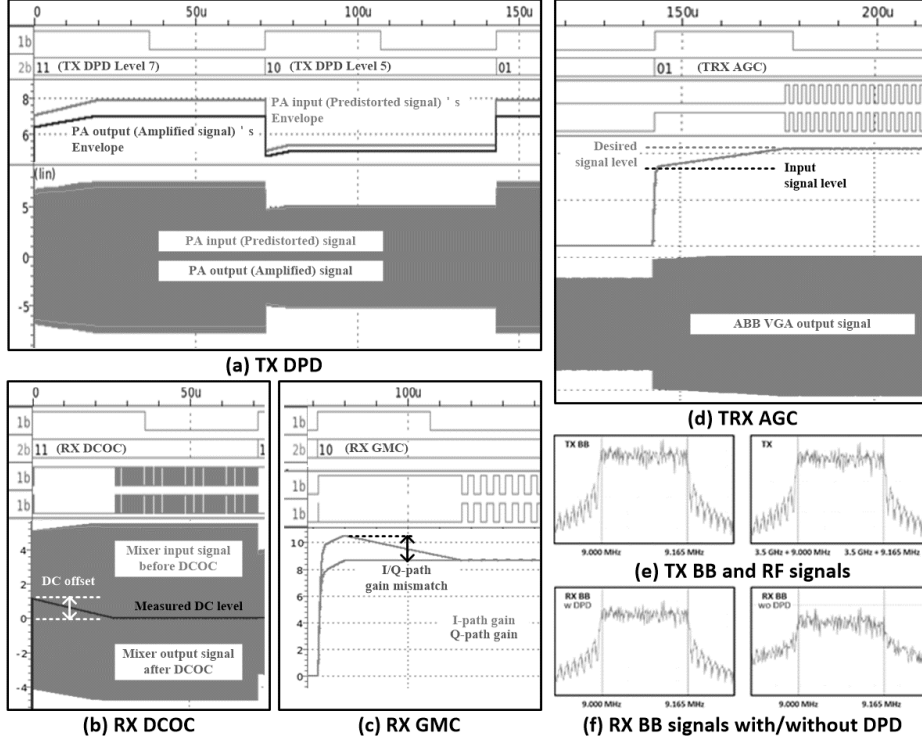


Figure 2.9. (a–d) Simulated waveforms and (e–f) FFT analysis results.

To evaluate the speed of simulation of the proposed model, we created a baseband-equivalent RNM that could perform the same operation. In RNM, only the frequency components of the baseband-equivalent signal near the carrier frequency are modeled, and the high-frequency harmonics and DC components are ignored. The non-idealities and non-linearities, caused due to the mismatch between the gain and phase, are equally reflected in the models. In addition, the filters are modeled in the z -domain through bilinear transformation. Both the RNM and XMODEL models share the same pure Verilog digital control/calibration loop, thus creating a model that performs exactly the equivalent operations [Appendix 2].

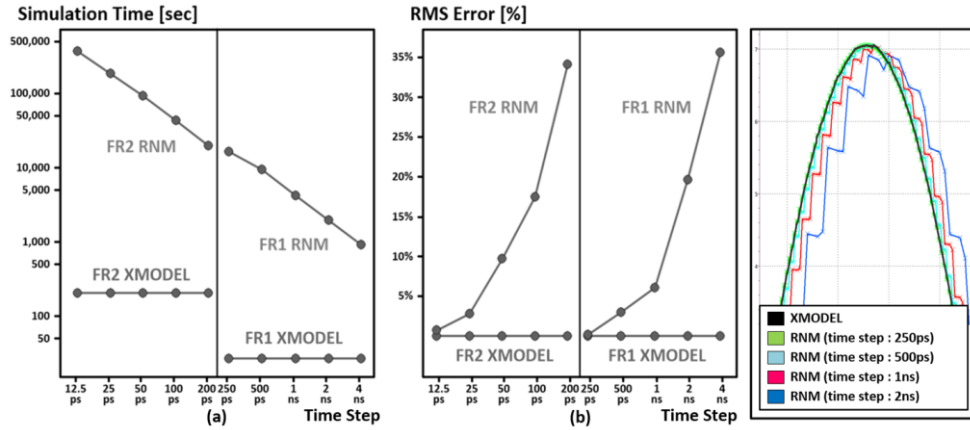


Figure 2.10. Simulation performances:
(a) run time and (b) RMS error.

The RNM witnesses a trade-off between the simulation performance and accuracy over time-step. The simulation speed and accuracy of the RNM models are dependent on the simulation time-step of SystemVerilog, which determines the time spacing between the adjacent points on the RF signal waveforms. Figs. 2.10(a) and (b) show the speed and accuracy of the simulation results when the simulation is run for 1 s under various time-step conditions. For the FR1 band (3.5 GHz carrier and 10 MHz bandwidth), the runtime of the RNM decreases from 16,367 to 910 s when the simulation is performed while sweeping time-steps from 250 ps to 4 ns, whereas the RMS percentage error increases from 0.2 to 35.69%. Moreover, for the FR2 band (28 GHz carrier and 200 MHz bandwidth), the runtime decreases from 366,660 to 19,587 s, whereas the RMS error increases from 0.7 to 34.25 with sweeping time-steps from 12.5 ps to 200 ps. However, the proposed XMODEL-based model delivers a constant runtime and accuracy regardless of the time-steps (27 s in the FR1 band and 203 s in the FR2 band). For both FR1 and FR2 bands, the XMODEL-based model runs 30–1,800 times faster while transmitting high-frequency passband signals without sacrificing the accuracy, as shown in Fig. 2.10(b).

Chapter 3. Nonlinear Model

3.1. Volterra / Perturbation Method

The Volterra series expansion is well known for modeling the memory effects of weakly-nonlinear circuits such as the spectral regrowth in LNAs [26,27]. The Volterra series is a comprehensive method to model the nonlinear dynamics of the circuits with high accuracy, where the output response $y(t)$ to an excitation $s(t)$ can be expressed as a sum of multidimensional convolution integrals as follows:

$$\begin{aligned}
 y(t) &= \sum_{n=1}^{\infty} y_n(t) = (h_1 * s)(t) + (h_2 * s * s)(t) + \dots \\
 &= \sum_{n=1}^{\infty} \int_{-\infty}^{\infty} \dots \int_{-\infty}^{\infty} h_n(\tau_1, \tau_2, \dots, \tau_n) \prod_{i=1}^n s(t - \tau_i) d\tau_i \\
 y_n(t) &= \int_{-\infty}^{\infty} \dots \int_{-\infty}^{\infty} h_n(\tau_1, \dots, \tau_n) \cdot s(t - \tau_1) \dots s(t - \tau_n) \cdot d\tau_1 \dots d\tau_n
 \end{aligned} \tag{3.1}$$

where $y_n(t)$, the n^{th} -order output of the system, is computed as n -times repeated convolution with the n th-order Volterra-kernel $h_n(\tau_1, \tau_2, \dots, \tau_n)$. This kernel information and output response can be Fourier transformed and expressed in the Laplace domain as follows:

$$\begin{aligned}
 H_n(\omega_1, \omega_2, \dots, \omega_n) \\
 = \int_{-\infty}^{\infty} \dots \int_{-\infty}^{\infty} h_n(\tau_1, \dots, \tau_n) \cdot \exp(-j(\omega_1\tau_1 + \dots + \omega_n\tau_n)) \cdot d\tau_1 \dots d\tau_n
 \end{aligned} \tag{3.2}$$

$$Y(j\omega) = H_1(j\omega) \circ S + H_2(j\omega_1, j\omega_2) \circ S^2 + H_3(j\omega_1, j\omega_2, j\omega_3) \circ S^3 + \dots$$

where, “ \circ ” means multiplication of the amplitude of each frequency component, and shift of the phase with the appropriate value [ref].

However, it is difficult to obtain the kernel information, so the perturbation method is used to convert the Volterra series expansion into a collection of multiple linear sub-systems [28,29]. This chapter explains a method to model the Volterra-series system obtained via perturbation method by combining the XMODEL functional primitives and discusses their simulation results. For example, the models of the common-gate (CG) amplifier presented in [29] demonstrate that the technique proposed in [29] indeed improves the linearity of the LNA.

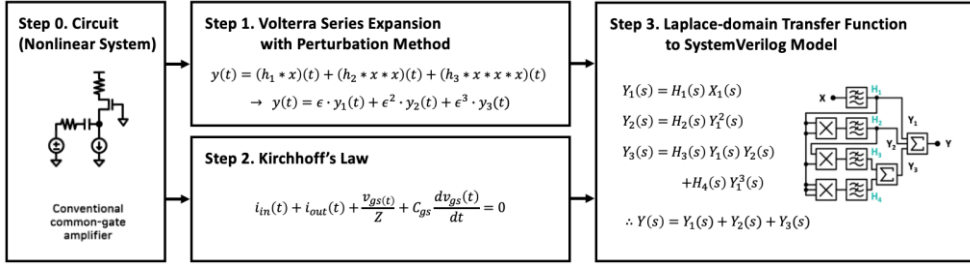


Figure 3.1. Nonlinear system model design flow.

To model nonlinear circuits in SystemVerilog, we apply the perturbation method to the Volterra series expansion and model it as a composition of multiple linear subsystems over three steps, as illustrated in Fig. 3.1. This chapter describes in detail the process of deriving the Laplace-domain transfer functions from the Volterra series model and expressing its system model using the XMODEL primitives in SystemVerilog.

First, we assume a weakly-nonlinear system where the input is x and the output is y , and this output can be expressed up to the third order in the Volterra series expansion, as shown in Eq. (3.3).

$$y(j\omega) = A_1(j\omega) \circ x(j\omega) + A_2(j\omega_1, j\omega_2) \circ x^2(j\omega) + A_3(j\omega_1, j\omega_2, j\omega_3) \circ x^3(j\omega) \quad (3.3)$$

Then, we can convert this Volterra series expansions into perturbation-series expansions by applying a small perturbation to

the input around the DC operating point, i.e., $x(t) = x_0 + \epsilon \Delta x(t)$, where ϵ is an arbitrarily small scalar value [30]. The n th-order perturbation-series expansion ($n=1,2,3$) is obtained by Eq. (3.4).

$$y(t) = \epsilon \cdot y_1(t) + \epsilon^2 \cdot y_2(t) + \epsilon^3 \cdot y_3(t) \quad (3.4)$$

Next, the nonlinear system is derived from the circuit equations based on Kirchhoff's laws and expressed in differential equations, where $q(\cdot)$, $g(\cdot)$ and $u(\cdot)$ are nonlinear functions of the resistive, dynamic, and input, respectively.

$$\frac{d}{dt}q(y(t)) + g(y(t)) = u(x(t)) \quad (3.5)$$

By substituting the perturbation input and the output of Eq. (3.4) into Eq. (3.5), the responses of the system are classified according to their orders, and each response is Laplace-transformed into an s -domain transfer functions as follows, where the operator “ \otimes ” denotes the time-domain multiplication in the Laplace domain [4].

$$\begin{aligned} \epsilon^1 - \text{term} : Y_1(s) &= H_1(s) \cdot X(s) \\ \epsilon^2 - \text{term} : Y_2(s) &= H_2(s) \cdot [Y_1 \otimes Y_1](s) \\ \epsilon^3 - \text{term} : Y_3(s) &= H_3(s) \cdot [Y_1 \otimes Y_2](s) + H_4(s) \cdot [Y_1 \otimes Y_1 \otimes Y_1](s) \end{aligned} \quad (3.6)$$

The resulting equations can be easily implemented using the XMODEL's functional primitives, i.e., by combining the filter and multiply primitives. First, the filter primitive describes a linear filter of which transfer function is in the form of Eq. (3.7) with three design parameters (DC gain, poles, and zeros). Therefore, the four transfer functions (H_1 – H_4) obtained in Eq. (3.6) can be modeled with four filter primitives, each of which expresses the equivalent transfer function in the following form:

$$H(s) = \text{Gain} \times \prod_{j=1}^{N_z} \left(1 + \frac{s}{z_j}\right) / \prod_{i=1}^{N_p} \left(1 + \frac{s}{p_i}\right) \quad (3.7)$$

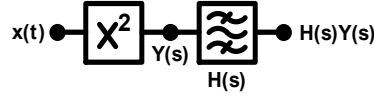


Figure 3.2. Example of second-order nonlinear system.

Second, the time-domain multiplications in the Laplace domain (“ \otimes ”) expressing the high-order terms can be modeled with the multiply primitives. For example, in a nonlinear system, if the output response with respect to $x^2(t)$ has a transfer function $H(s)$ (obtained through the perturbation method), then it can be modeled using a multiply primitive followed by a filter primitive, as shown in Fig. 3. In the first stage, $x(t)$ is squared in the Laplace-domain, and the output is $Y(s) = [x(t) \otimes x(t)]$. In this way, the response for each order of the perturbation series can be obtained, and the entire nonlinear system can be implemented through their linear combination. These equations represent nonlinearities using a collection of linear systems, so each system’s transfer function can be modeled with the XMODEL’s functional primitives in SystemVerilog [31].

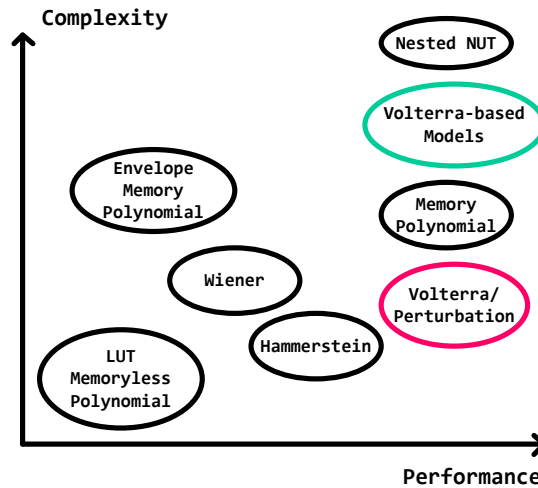


Figure 3.3. Nonlinear system model design flow.

3.2. Low-noise Amplifier Example

This chapter describes the application of the method introduced in chapter 3.1 to create a nonlinear model of the conventional and nonlinearity-canceled common-gate low-noise amplifier (CG-LNA) introduced in [29]. In both types of LNAs, the output of the system is a small-signal drain current (i_{out}) and gate-to-source voltage (v_{gs}), which have the nonlinear characteristics owing to the nonlinear transconductance of the MOSFET. They can be expressed as a Volterra series, as in step 1, and converted to Eq. (3.8) by applying the perturbation method.

$$\begin{aligned}
 v_{gs}(j\omega) &= A_1(j\omega)^\circ i_{in} + A_2(j\omega_1, j\omega_2)^\circ i_{in}^2 + A_3(j\omega_1, j\omega_2, j\omega_3)^\circ i_{in}^3 \\
 &\rightarrow v_{gs}(t) = \epsilon \cdot v_{gs1}(t) + \epsilon^2 \cdot v_{gs2}(t) + \epsilon^3 \cdot v_{gs3}(t) \\
 i_{out}(j\omega) &= B_1(j\omega)^\circ i_{in} + B_2(j\omega_1, j\omega_2)^\circ i_{in}^2 + B_3(j\omega_1, j\omega_2, j\omega_3)^\circ i_{in}^3 \\
 &\rightarrow i_{out}(t) = \epsilon \cdot i_{out1}(t) + \epsilon^2 \cdot i_{out2}(t) + \epsilon^3 \cdot i_{out3}(t)
 \end{aligned} \tag{3.8}$$

Furthermore, since their relationship is expressed by the Taylor series expansion, i.e., $i_{out} = g_{m1}v_{gs1} + g_{m2}v_{gs2}^2 + g_{m3}v_{gs3}^3$, substituting Eq. (3.8) into this relationship gives the solution for each order as follows.

$$\begin{aligned}
 \epsilon^1 - term : i_{out1}(t) &= g_{m1}v_{gs1}(t) \\
 \epsilon^2 - term : i_{out2}(t) &= g_{m1}v_{gs2}(t) + g_{m2}v_{gs1}^2(t) \\
 \epsilon^3 - term : i_{out3}(t) &= g_{m1}v_{gs3}(t) + g_{m2}v_{gs1}(t)v_{gs2}(t) + g_{m3}v_{gs1}^3(t)
 \end{aligned} \tag{3.9}$$

where g_{m1} is the transistor's small-signal transconductance, and the higher-order coefficients (g_{m2} , g_{m3}) indicate the corresponding nonlinearities:

$$\begin{aligned}
 g_{m1} &= \frac{\partial i_{out}}{\partial v_{gs}} = g_m & g_{m2} &= \frac{1}{2} \frac{\partial^2 i_{out}}{\partial v_{gs}^2} = \frac{1}{2} g'_m \\
 g_{m3} &= \frac{1}{6} \frac{\partial^3 i_{out}}{\partial v_{gs}^3} = \frac{1}{6} g''_m
 \end{aligned} \tag{3.10}$$

A. Conventional CG Amplifier Model

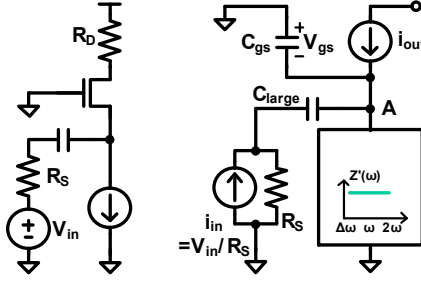


Figure 3.4. (a) CG amplifier and (b) its equivalent circuit model.

From the equivalent circuit of the conventional CG amplifier shown in Fig. 3.4, we can derive the equation below by applying the Kirchhoff's current law at node A:

$$i_{in}(t) + i_{out}(t) + \frac{v_{gs}(t)}{Z} + C_{gs} \frac{dv_{gs}(t)}{dt} = 0 \quad (3.11)$$

Substituting Eq. (3.9) into Eq. (3.11), we can obtain a set of three differential equations by separately equating each coefficient of power of ϵ to zero.

$$\begin{aligned} \epsilon^1 - term : i_{in}(t) + i_{out1}(t) + \frac{v_{gs1}(t)}{Z} + \frac{d}{dt} (C_{gs} v_{gs1}(t)) &= 0 \\ \epsilon^2 - term : i_{out2}(t) + \frac{v_{gs2}(t)}{Z} + \frac{d}{dt} (C_{gs} v_{gs2}(t)) &= 0 \\ \epsilon^3 - term : i_{out3}(t) + \frac{v_{gs3}(t)}{Z} + \frac{d}{dt} (C_{gs} v_{gs3}(t)) &= 0 \end{aligned} \quad (3.12)$$

The following partial responses can be obtained by transforming the equations in Eq. (3.12) to the Laplace domain.

$$\begin{aligned} \epsilon^1 - term : V_{gs1}(s) &= H_1(s) \cdot I_{in}(s) \\ \epsilon^2 - term : V_{gs2}(s) &= g_{m2} \cdot H_1(s) \cdot [v_{gs1} \otimes v_{gs1}](s) \\ \epsilon^3 - term : V_{gs3}(s) &= g_{m2} \cdot H_1(s) \cdot [v_{gs1} \otimes v_{gs2}](s) \\ &\quad + g_{m3} \cdot H_1(s) \cdot [v_{gs1} \otimes v_{gs1} \otimes v_{gs1}](s) \end{aligned} \quad (3.13)$$

$$\begin{aligned}
\epsilon^1 - \text{term} : I_{out1}(s) &= H_1(s) \cdot I_{in}(s) \\
\epsilon^2 - \text{term} : I_{out2}(s) &= g_{m2}/g_{m1}^2 \cdot H_2(s) \cdot [i_{out1} \otimes i_{out1}](s) \\
\epsilon^3 - \text{term} : I_{out3}(s) &= g_{m2}/g_{m1}^2 \cdot H_2(s) \cdot [i_{out1} \otimes i_{out2}](s) \\
&\quad + \left(\frac{g_{m2}^2}{g_{m1}^4} - \frac{g_{m3}}{g_{m1}^3} \right) \cdot H_2(s) \cdot [i_{out1} \otimes i_{out1} \otimes i_{out1}](s)
\end{aligned} \tag{3.14}$$

where,

$$H_1(s) = -\frac{1}{g_{m1} + 1/Z + sC_{gs}} \quad H_2(s) = \frac{1/Z + sC_{gs}}{g_{m1} + 1/Z + sC_{gs}}. \tag{3.15}$$

Now we can express the LNA output signals, the gate-to-source voltage (V_{gs}), and output current (I_{out}), by summing the partial responses in the perturbation series:

$$\begin{aligned}
\therefore V_{gs}(s) &= V_{gs1}(s) + V_{gs2}(s) + V_{gs3}(s) \\
\therefore I_{out}(s) &= I_{out1}(s) + I_{out2}(s) + I_{out3}(s)
\end{aligned} \tag{3.16}$$

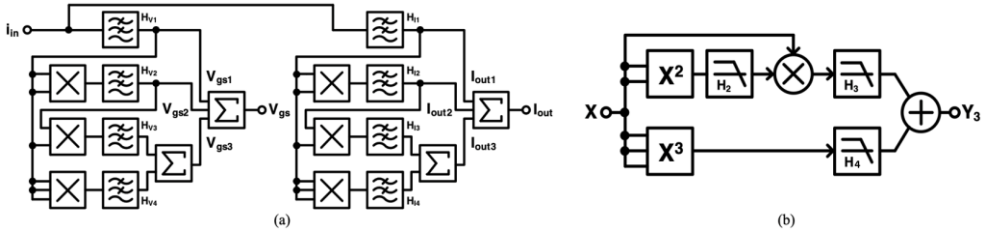


Figure 3.5. System diagrams of (a) the overall system and (b) the third-order nonlinearity.

TABLE I
FILTER'S PARAMETER FOR CONVENTIONAL CG-AMPLIFIER

Filter	DC Gain	Pole	Zero
H_{V1}	$-zA$	f_p	-
H_{V2}	$-g_{m2}A$	f_p	-
H_{V3}	$-g_{m2}A$	f_p	-
H_{V4}	$-g_{m3}A$	f_p	-
H_{I1}	$-g_{m1}A$	f_p	-
H_{I2}	$(g_{m2}/g_{m1}^2)A$	f_p	f_z
H_{I3}	$(g_{m2}/g_{m1}^2)A$	f_p	f_z
H_{I4}	$(g_{m1}g_{m3}-g_{m2}^2)A/g_{m1}^4$	f_p	f_z

TABLE II
FILTER'S PARAMETER FOR NONLINEARITY CANCELLATION CG-AMPLIFIER

Filter	DC Gain	Pole	Zero
H_{V1}	1	$f_{p1,2}$	$1/2\pi L$
H_{V2}	g_{m2}	$f_{p1,2}$	$1/2\pi L$
H_{V3}	g_{m2}	$f_{p1,2}$	$1/2\pi L$
H_{V4}	g_{m3}	$f_{p1,2}$	$1/2\pi L$
H_{I1}	g_{m1}	$f_{p1,2}$	$1/2\pi g_{m1}L$
H_{I2}	g_{m2}/g_{m1}	$f_{p1,2}$	$\pm i/(2\pi\sqrt{LC})$
H_{I3}	g_{m2}/g_{m1}^2	$f_{p1,2}$	$\pm i/(2\pi\sqrt{LC})$
H_{I4}	$(g_{m1}g_{m3}-g_{m2}^2)/g_{m1}^4$	$f_{p1,2}$	$\pm i/(2\pi\sqrt{LC})$

where, $A = 1/(1 + g_{m1}Z)$, $f_p = (1 + g_{m1}Z)/2\pi C_{gs}Z$, $f_z = 1/2\pi C_{gs}Z$

In Eq. (3.14), each term in the partial responses is basically a response of a linear filter to a product between the lower-order

partial responses or input. Therefore, each term can be modeled using a pair of multiply and filter primitives, and the parameters used for each filter (gain, poles, and zeros) are summarized in Table 1.

With the nonlinear system models for the CG-LNAs described so far, as shown in Fig. 3.5(b), we can analyze for the key factors affecting the nonlinear output response, by dividing it into two pathways: one is for the effect of the third-order intermodulation (IM3) and the other is for effects of the second-order harmonic term. We can see that the IM3 is directly influenced by the third-order nonlinear term via the g_{m3} value in $H_4(s)$ and suppressing g_{m3} is the key to improve linearity. Also, the second-order effect creates unwanted harmonics and intermodulation components, which are converted in the multiplier to produce IM3 components and cause long-term memory effect. This effect is modeled in $H_2(s)$ and $H_3(s)$.

B. CG Amplifier Model with the Nonlinearity–Cancellation Technique

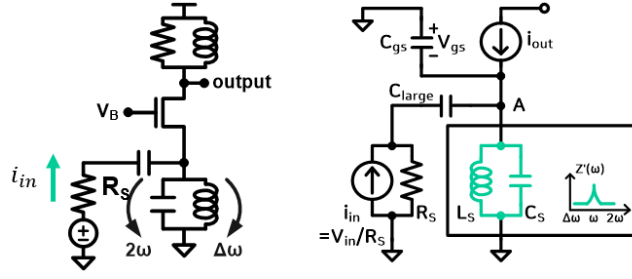


Figure 3.6. (a) CG amplifier with nonlinearity cancellation and (b) its equivalent circuit model

The CG amplifier with the nonlinearity–cancellation technique introduced in [29] uses an RF current source with a capacitor and inductor in parallel to maximize the impedance at the resonance frequency (ω) and minimize the impedance at $\Delta\omega$ and 2ω , as shown in Fig. 3.6. This technique can reduce the overall nonlinearity arising from the second–order distortion effect. The differential equation below describes the relationship between the input i_{in} and the output i_{out} in node B:

$$i_{in}(t) + i_{out}(t) + (C_{gs} + C_s) \frac{dv_{gs}(t)}{dt} + \frac{1}{L_s} \int_0^t v_{gs}(\tau) d\tau = 0 \quad (3.15)$$

Again, by substituting Eq. (9) derived from the Volterra/perturbation–series expansion method into Eq. (15), the partial responses of V_{gs} and I_{out} of the circuit can be computed, where the Volterra/perturbation–series expansion is obtained in the same manner as in section A. The only difference is with the transfer functions describing each filter. Therefore, the diagram for the resulting system model is almost identical to that for the conventional CG LNA as shown in Fig. 3.4, and the parameter values of the filters are summarized in Table 2, where $f_{p1,2} = 1/\pi(g_{m1}L \pm \sqrt{(g_{m1}L)^2 - 4LC})$.

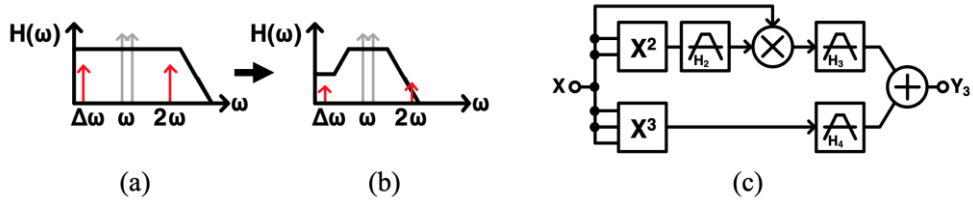


Figure 3.7. Filter model of the (a) conventional and (b) cancellation structure, and (c) system diagram of a third-order nonlinearity.

In this LNA with the nonlinearity-cancellation technique, the employed RF current sources create additional poles and zeros in the transfer function, which serves the roles of suppressing the second-order nonlinearity term. Fig. 3.7 illustrates how it works. In the conventional CG-LNA, the single-pole low-pass filter cannot sufficiently suppress the second-order nonlinearity term and the resulting intermodulation component. On the other hand, in the CG-LNA with the nonlinearity-cancellation technique, the corresponding filter has a band-pass characteristic due to the additional poles and zeros and can suppress the intermodulation components. This reduction in the second-order nonlinear terms leads to the lower intermodulation distortion (IMD), lower interference, and better linearity.

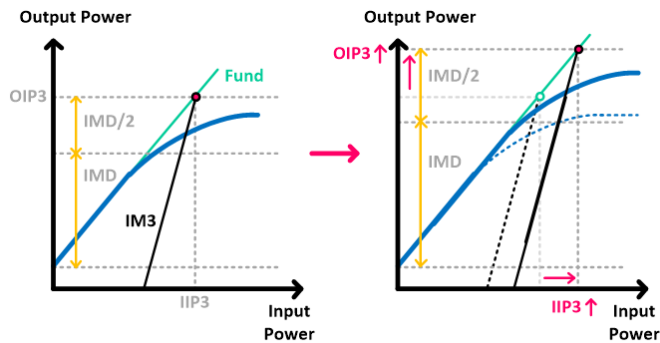


Figure 3.8. Linearity improvement.

3.3. Nonlinearity Analysis

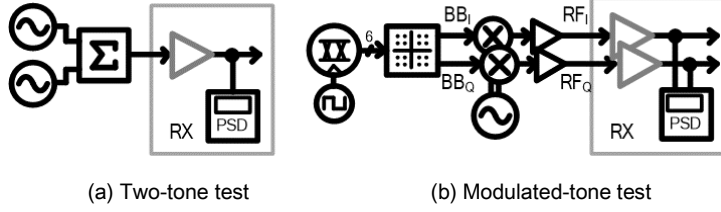


Figure 3.9. Testbenches for the nonlinear RF amplifiers.

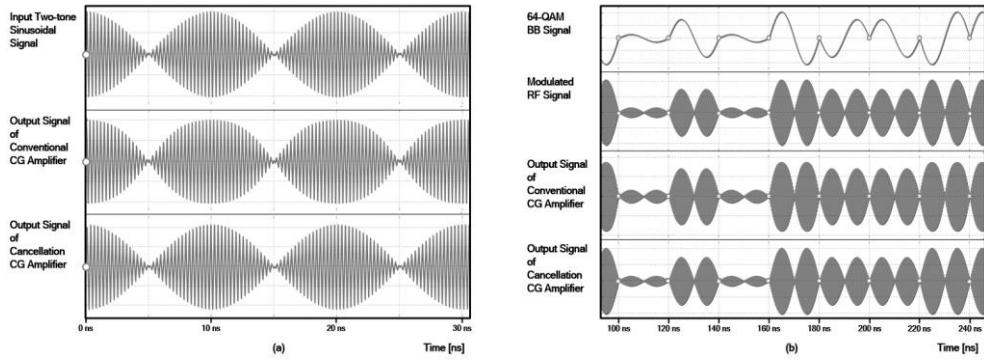


Figure 3.10. Time-domain simulation results with event markers.

Fig. 3.9(a) shows a testbench that feeds a two-tone signal to the two CG-LNA models. The two sinusoidal signals (3.45 and 3.55 GHz) with the same power of 16 dB are added and fed into the amplifiers. This two-tone test is a widely-used method to assess the nonlinearity of the circuits by measuring the power of the second and third intermodulation components generated in response to the two-tone input. For both amplifiers, the circuit parameters C_{gs} , L , and C are assumed to be 153-fF , 8.2-pF , and 0.25-nH , respectively. Fig. 3.9(b) shows a testbench that feeds the RF signals that are up-converted from 64-QAM baseband signals. First, the quadrature amplitude modulated signals centered at DC (BB_I , BB_Q) are generated with 50 MHz symbol rates, and they are up-converted to a carrier frequency of 3.5 GHz. The modulated RF signal (RF_I , RF_Q) can be expressed as:

$$\begin{aligned} \text{RF}_I &= A_I \cdot \cos(\omega_{50\text{MHz}}t) \cdot \cos(\omega_{3.5\text{GHz}}t) \\ \text{RF}_Q &= A_Q \cdot \sin(\omega_{50\text{MHz}}t) \cdot \sin(\omega_{3.5\text{GHz}}t). \end{aligned} \quad (3.16)$$

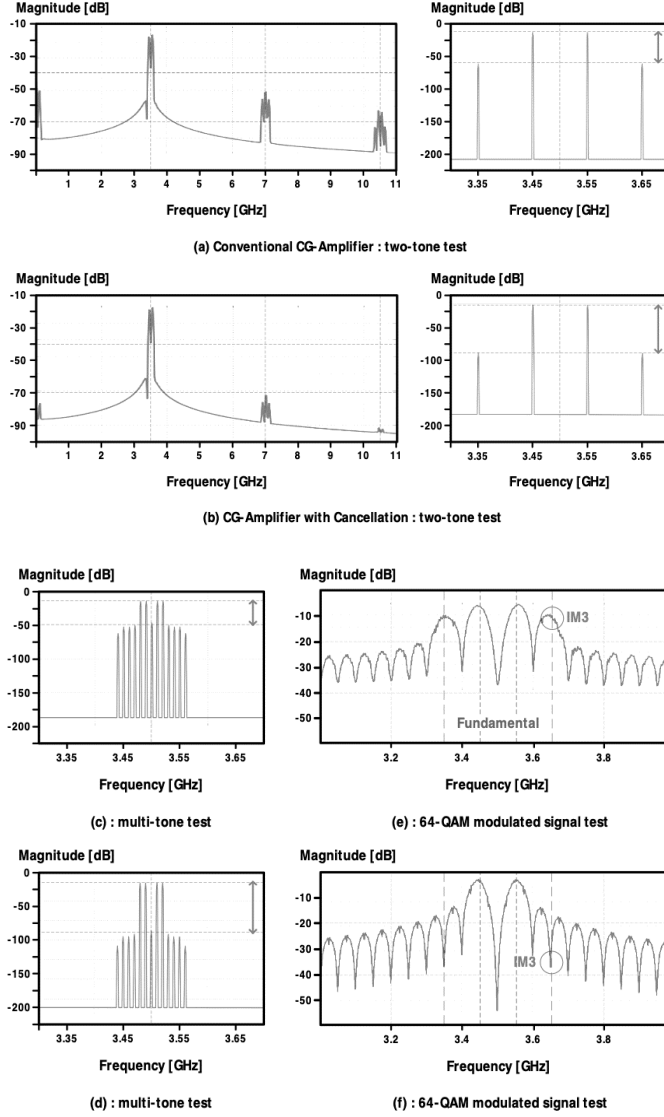


Figure 3.11. The measured output spectrums.

Fig. 3.10 shows the simulated waveforms with the two-tone input, along with the event markers that indicate where the actual computations were performed. The output waveforms have events only when the input waveforms have events, demonstrating the event-driven simulation discussed earlier. Yet, the intermodulation

components embedded in the signals can be analyzed accurately, thanks to the functional expressions used by XMODEL. For this particular simulation, the input and output signals in fact have only one event, performing a 100- μ s simulation in only 0.40 seconds. Even with the modulation RF signals, the 100- μ s simulation transmitting 5,000 symbols takes only 8.55 seconds.

Another way to analyze the simulated results is to plot the frequency-domain spectrum of the signal via FFT. Figs. 3.11(a) and (b) plot the second and third harmonics and intermodulation components in a two-tone test, respectively. Comparing the output spectrum of the nonlinearity-cancellation amplifier with the conventional amplifier, the second-order distortion components are reduced by 20 dB, and consequently, the third-order distortion components are greatly reduced. The magnitude of the IMD (the difference between the fundamental power and IM3 power) is improved from -45 dBc to -70 dBc with the nonlinearity cancellation technique.

The spectral regrowth analysis is also possible by applying a multi-tone signal. Figs. 3.11(c) and (d) shows the output spectrum when four sinusoidal inputs with the same amplitude and equal frequency spacing are applied to each amplifier. When these multiple-frequency tones are processed at once in a channel, the resulting IMD terms form a band, which can degrade the performance. After the nonlinearity cancellation, the power of the spectral regrowth is reduced by 30 dB. The power of the fundamental signal (the purple line, $\omega_{3.5\text{GHz}} \pm \omega_{50\text{MHz}}$) and IM3 signal (the red line) for the QAM-modulated signal are shown in Figs. 3.11(e) and (f). When comparing the IM3 components in the output spectrums of the conventional and nonlinearity-cancellation amplifiers, the amount of the IM3 component suppressed by the nonlinearity-cancellation technique is significant.

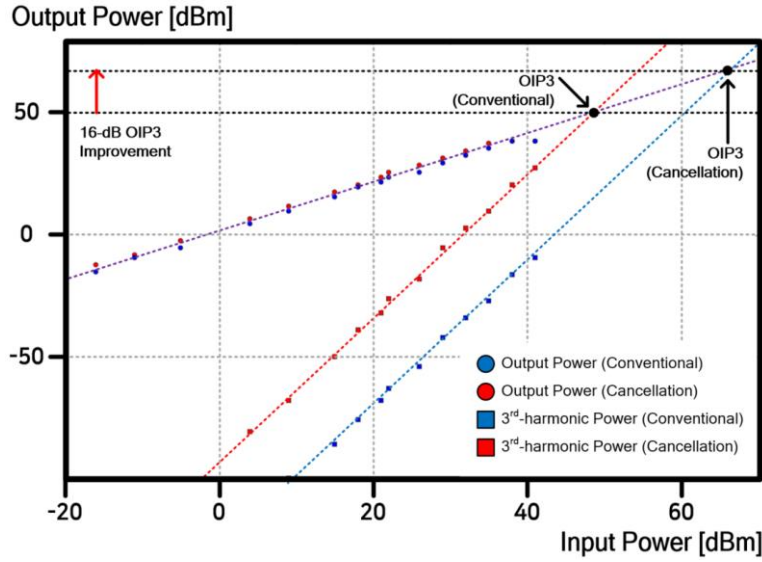


Figure 3.12. The OIP3 measurement results.

Fig. 3.12 plots the powers of the fundamental and IM3 signals measured while increasing the input power from -20 dBm to 50 dBm. At the point where the input power is 35 dBm, the output signal starts getting saturated. The third-order output intercept point (OIP3) was measured in both amplifiers and the nonlinearity-cancellation technique achieves a 16 dB improvement.

The proposed modeling method has several advantages in the design and verification of nonlinear RF systems. First, fast simulations are possible without compromising accuracy as both the low-frequency baseband signals and high-frequency RF and harmonic signals can be expressed in equations. An event is triggered only when the coefficients of these equations change. Second, the presented models can run on a digital simulation platform (SystemVerilog) suitable for verifying RF systems enclosed by various digital feedback loops and digital control logic. Finally, the process of breaking a nonlinear system into a set of linear sub-systems can identify factors that degrade the performance of the system and gain design insights.

Chapter 4. Coverage Analysis and Functional Verification

The previous chapters described the behavioral modeling method of RF transceivers to speed up performance estimation and functional verification. This chapter introduces a testbench using a parameter coverage analysis technique and a functionality checker for complete verification.

4.1. Model Parameter Coverage Analysis

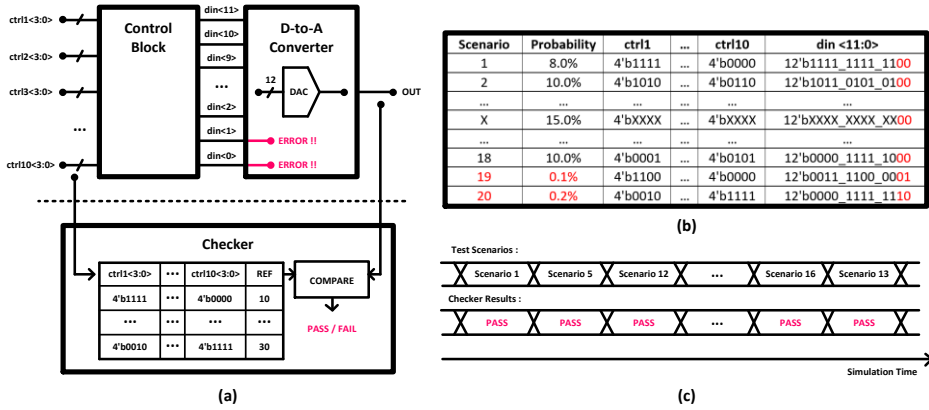


Figure 4.1. Design verification problem related to digital-to-analog converter (DAC) model.

The example in Fig. 4.1 shows the limitations of checker-only verification. Even if there are errors in the LSB 2-bit of the DAC model, the control codes ($din[11:0]$) corresponding to these bits are rare, and the checker may not find the error. These errors are

detected only by running the simulation “long enough” for the checker to find them. However, since it is difficult to figure out how long the verification designer must simulate to ensure that the parameters are error-free, a technique must quantify whether the simulation has sufficiently verified the parameters.

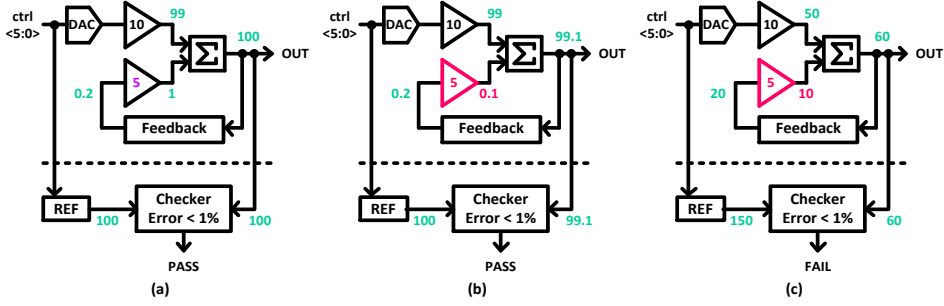


Figure 4.2. Design verification problem related to feedback loop amplifier model.

Figure 4.2 shows that if the checker is not sensitive to small signals, it may not detect errors. When the gain of the feedback loop amplifier has a value of ‘0.5’ instead of ‘5’ as an error and the value generated through the feedback loop is small enough, the checker will not be able to find any errors. Therefore, to find the parameter error of these amplifiers, it is necessary to “sufficiently” simulate various scenarios and check the case where the value generated in the feedback loop is large enough, as shown in Fig. 4.2 (c). Again, a verification method is needed to quantify whether the simulation has sufficiently checked the parameters during simulation.

Therefore, this chapter introduces parameter coverage analysis [32] and functionality checker, which are methods to check whether test scenarios are suitable for verifying that the TRX model has the appropriate design parameters such as gain, bandwidth, etc. First, the testbench's coverage measurement (*meas_cov*) module calculates the sensitivity and threshold. Sensitivity is a value that indicates how sensitive the output to the input is when the parameter value changes and the threshold is a value that determines whether the sensitivity is enough or not. This value is chosen as the optimal value based on

the information theory when defining the stochastic input signal space [Ref, Appendix]. Next, the *meas_cov* module compares the sensitivity to the threshold and, if the sensitivity is greater than the threshold, concludes that the parameter is covered.

```

module meas_cov #(
    `parameter_integer(num_in, 1),           // number of inputs
    `parameter_real(scale[num_in-1:0], '{num_in{1.0}}) // input scale factors
)(
    `input_xreal out,                         // output
    `input_xreal [num_in-1:0] in             // input
);
    xreal [num_bit-1:0] sen;                 // sensitivity
    xreal thr;                               // threshold
    xbit [num_bit-1:0] cov; bit [num_in-1:0] coverage; // coverage
    // (1) Calculate Optimum Threshold
    calc_thr #(.num_in(num_bit), .scale(scale)) thr_calc (thr);
    genvar i_gen;
    generate
        for (i_gen=0; i_gen<num_bit; i_gen=i_gen+1) begin: genblock
            // (2) Calculate Signal Sensitivity
            calc_cov calc(.in_ref(out), .in_pert(in[i_gen]), .sen(sen[i_gen]));
            // (3) Decide Parameter Coverage
            slice cov_result (.in(sen[i_gen]), .in_ref(thr), .out(cov[i_gen]));
        end
    endgenerate
    xbit_to_bit #(.width(num_bit)) xtb_cov (.in(cov), .out(coverage));
    covergroup cg;
        // (4) Report Coverage Result
        cover_point : coverpoint coverage;
    endgroup
    cg cg_inst = new();
    always @(posedge clk) begin
        cg_inst.sample();
    end
endmodule

```

```

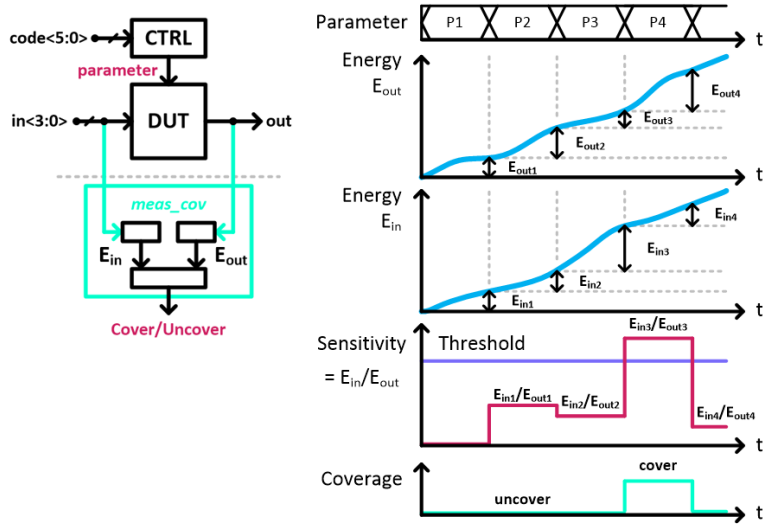
module calc_cov (
    `input_xreal in_ref, in_pert,
    `output_xreal sen
);
    xreal e_ref, e_pert;
    real e_rreal, e_preale, sensitivity;
    calc_ene #(.scaler(scaler)) icalc1(.in(in_ref), .out(e_ref));
    calc_ene #(.scaler(scaler)) icalc2(.in(in_pert), .out(e_pert));
    xreal_to_real xtr_er (.in(e_ref), .out(e_rreal));
    xreal_to_real xtr_ep (.in(e_pert), .out(e_preale));
    assign sensitivity = e_preale/e_rreal;
    real_to_xreal rtx (.in(sensitivity), .out(sen));
endmodule

module calc_thr #(
    `parameter_integer(num_in, 2),           // number of inputs
    `parameter_real(scale[num_in-1:0], '{num_in{1.0}}) // input scale factors
)(
    `output_xreal thr
);
    integer sum, i;
    real threshold;
    initial begin
        sum = $rtoi(scale[0]);
        for (i=1; i<num_in; i=i+1) begin
            sum = sum + $rtoi(scale[i]);
        end
    end
    assign threshold = 1.0 / (sum * sum);
    real_to_xreal rtx (.in(threshold), .out(thr));
endmodule

```

Figure 4.3. Pseudocode for the module that measures the coverage results of weighted-sum models.

Fig. 4.3 introduces the pseudo-code of the *meas_cov* module that measures the coverage results of weighted-sum ($y = \sum_{i=1}^N w_i x_i$) blocks such as ADD and DAC. Adder blocks can combine multiple paths in the RF transceiver to support multi-band/multi-standard operations, such as phased array structure, beamforming MIMO, or carrier aggregation. Also, the digital-to-analog conversion block can convert externally applied control codes into the model's parameter or codes generated in a digital loop into the calibration value.



$$S_k(\mathbf{w}_k, x) = w_k^2 \cdot \frac{\int_0^{T_{period}} x^2(t) dt}{\int_0^{T_{period}} y^2(t) dt} = \frac{E_{in}}{E_{out}}$$

Figure 4.4. Parameter coverage analysis example.

The *meas_cov* module performs 4 main functions: threshold value calculation (*calc_thr*), sensitivity value calculation (*calc_cov*), coverage determination and coverage report. The *calc_cov* module uses the *calc_ene* module to measure the energy of the input signal (*in_pert*) and the output signal (*in_ref*) and then calculates the sensitivity value as a ratio of these values. After that, the slice primitive compares the sensitivity value to the threshold value, outputting 1'b1 if the sensitivity value is high and 1'b0 if the threshold value is high. When testbench specifies this value as a coverage point,

VCS measures the coverage and reports it to the dashboard using features such as coverage metrics, assertions, and coverage groups. Similarly, the *meas_fcov* module can measure coverage results for other analog blocks such as filter, slicer, comparator, and scaler [Appendix 3].

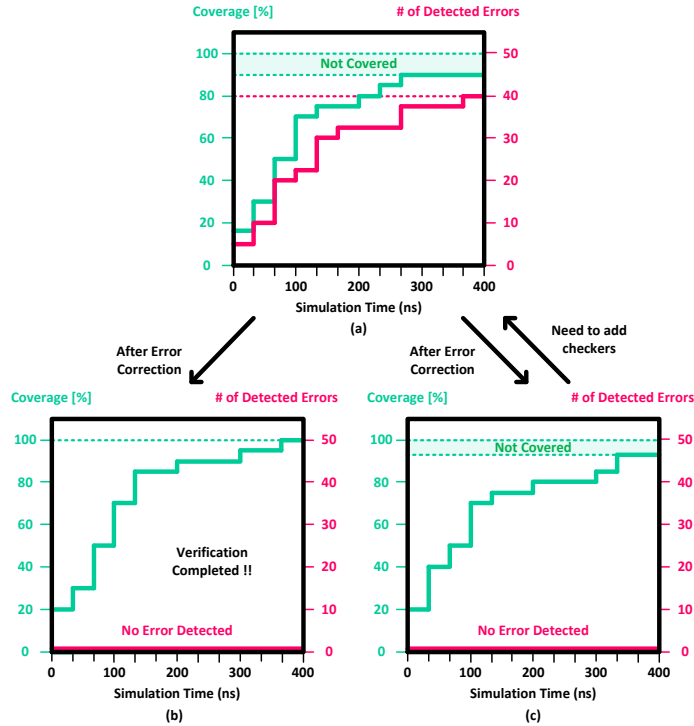


Figure 4.5. Parameter coverage analysis process and results.

Fig. 4.5 (a) shows the results of the parameter coverage analysis for each verification process, where the red line represents the coverage results, and the green line represents the number of detected errors. As the simulation runs longer, both values gradually increase and then saturate to a certain value when the checker no longer finds any errors. Suppose all found errors are corrected, and the verification engineer performs the same verification process. In that case, the coverage reaches 100%, as shown in Fig. 4.5 (b), which means that all parameters have been sufficiently verified without errors.

However, if the checker is not sufficient to verify the model's functionality, the coverage will not reach 100% even if no errors are detected, as shown in Fig. 4.5 (c). In such a case of insufficient coverage, the verification engineer must add explicit checkers to the testbench and repeat the error finding process until the coverage reaches 100%.

4.2. Self-checking Testbench

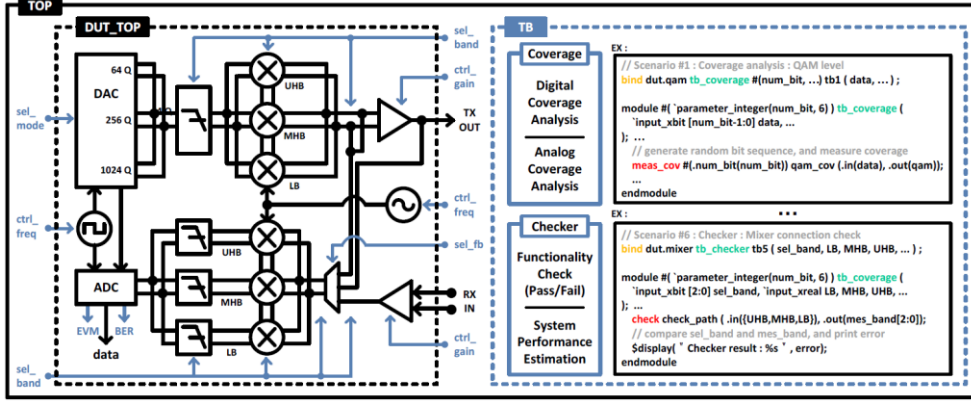


Figure 4.6. Self-checking testbench configuration for coverage analysis and functionality check

The self-checking testbench "*top*" is composed of the RF transceiver model "*dut_top*" introduced in chapter 2 and the sub-test benches "*TB*" for verifying test scenarios [7]. The *dut_top* performs TX-to-RX chain simulation under various test conditions, and the values of each sub-block parameter set determine these test conditions. Testbench top randomly generates these parameters under certain constraints. Both TX and RX have UHB, MHB, and LB operating paths depending on the carrier frequency range and can work in 64/256/1024-QAM modulation mode. The *TB* consists of checkers that perform functional verifications and checkers that perform digital/analog parameter coverage analysis. The test benches in *TB* are instantiated into *dut_top* using SystemVerilog's *bind* construct, allowing access to the signal inside *dut_top* without modifying the source code of the RFIC top model [33]. The *TB* consists of test benches that can check the functionality of each block, the connection between blocks, and system performance.

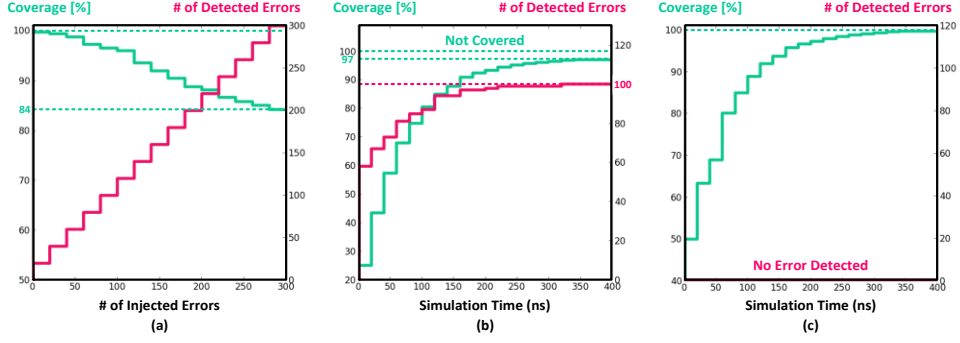


Figure 4.7. Simulation results of parameter coverage analysis
 (a) error-injection test, (b, c) coverage and detected errors.

Fig. 4.7 shows the coverage results (red line) and the number of detected errors (green line) in the RFIC model with 4000 parameters. First, Fig. 4.7 (a) illustrates the simulation result after injecting an arbitrary number of errors into the model. When the error was not injected, the parameter coverage result strikes 100%. Yet, as the number of injected errors increased, the coverage gradually declined as the number of non-covered parameters increased.

Figure 4.7(b) shows the number of errors found in the checker and the parameter coverage results as the simulation time increases when verifying the model with 100 errors injected. Although the checker quickly found dozens of errors at the beginning of the simulation, long simulations are needed to find the last 1–5 errors afterward. By correcting all the detected errors, the coverage reaches 100% without errors, as shown in Fig. 4.4 (c)

Fig. 4.8 illustrates the testbench configuration for the six key sub-blocks of the RFIC model: LO, DAC, filter, mixer, amplifier, and ADC.

TB 1: Verification scenarios of LO model.

- (1) Compare the frequency of the I/Q-carrier signals with the reference frequency.
- (2) Compare the frequency band of the LO output signals with the frequency band determined by the band selection code.
- (3) Measure digital and parameter coverage in 12-bit control code, control code-to-frequency model, and band selection model.

TB 2: Verification scenarios of DAC model.

- (1) Compare the frequency of the clock with the reference frequency (symbol rate).
- (2) Compare the QAM mode/level with the reference mode/level.
- (3) Measure digital and parameter coverage in 12-bit control code, control code-to-QAM level model, and QAM mode selection model.

TB 3: Verification scenarios of filter model.

- (1) Compare the pole value with the reference bandwidth.
- (2) Calculate the gain from the input and output signal power and check that the value is similar to the filter gain (default DC gain = “1” and gain for each frequency is previously stored in the LUT).
- (3) Measure digital and parameter coverage in 12-bit control code and filter model.

TB 4: Verification scenarios of mixer model.

- (1) Check that the input I/Q signals are multiplied by the I/Q signals of the appropriate bands, respectively.
- (2) Measure digital and parameter coverage in selection code and path connection model.

TB 5: Verification scenarios of amplifier model.

- (1) Compare the gain of the amplifiers with the reference gain.
- (2) Measure digital and parameter coverage in selection code and path connection model.

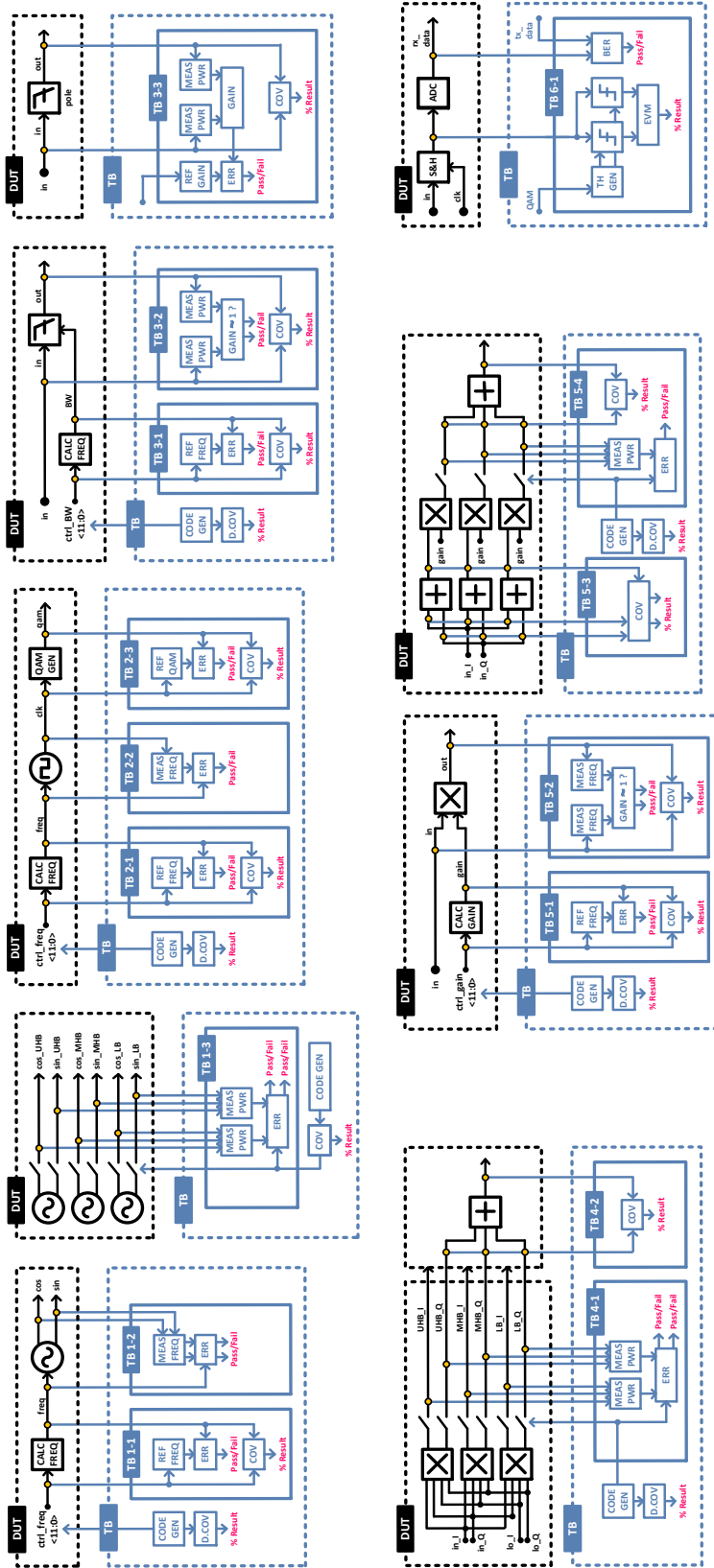


Figure 4.8. Testbench details and verification scenarios.

TB 6: Verification scenarios of ADC model.

- (1) Check for bit errors, and calculate EVM value. (After sampling the peak value, compare the value to the threshold values (th_{up}/th_{dn}) of QAM level. The peak value should be less than th_{up} and greater than th_{dn} .)
- (2) Measure digital and parameter coverage in ADC output data and compare model.

After the simulation, the pass/fail results for each verification scenario are displayed on the monitor, allowing verification engineers to identify which blocks have functional errors or have not achieved sufficient coverage. Fig. 4.9 shows the simulation results of the functionality checker and coverage analysis when design errors are intentionally injected into the model.

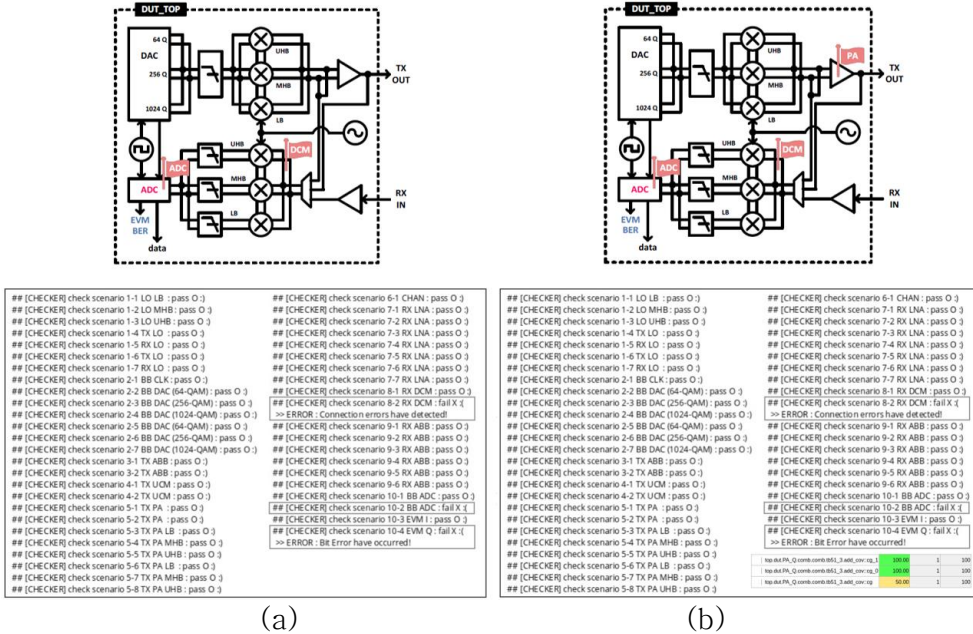


Figure 4.9. Simulation results and error observability.

First, in case 1, the checker results in Fig. 4.8 (a) can confirm that errors occur in subsequent blocks due to a connection error in the Q-path of the down-conversion mixer block (RX-DCM). For

case 2, the checker reported the same error as case 1, but the coverage results show that the error has already occurred in the PA block before the DCM block. These errors would have been challenging to find before tape-out, and even if seen, it would have been challenging to determine where the actual error occurred. Therefore, functional checkers and parametric coverage analysis can improve the completeness and observability of verification.

Chapter 5. Conclusion

In this dissertation, the SystemVerilog models for a multi-standard, direct-conversion RF transceiver enables efficient event-driven simulation. The models can estimate the performance metrics of the systems in the presence of various non-ideality conditions, such as DC offsets and I/Q imbalances, and verify the operation of the digital configuration/calibration controllers. The proposed models can serve the roles of the high-level MATLAB models for performance evaluation and digital RTL models for digital verification, and SPICE netlists for analog simulation while delivering fast speed entirely within SystemVerilog. The proposed models can be used as a simulation platform for exploring various RF transceiver architectures before IC design and as a verification testbed for checking the digital configuration/ calibration controllers before sign-off. In addition, modeling methodology using the XMODEL functional primitives based on the Volterra/ perturbation-series expansion is presented and explained with the RF low-noise amplifier examples. The simulation results show the adjacent channel interference and spectral regrowth generated due to the LNAs' nonlinearity and show that the OIP3 is improved by 16 dB after applying the nonlinearity cancellation technique.

Appendix

Appendix 1. Trigonometric Equation for Non-Ideal Effects

A. I/Q gain mismatch in mixer model [14].

$$\begin{aligned}
 BB'_I &= K_I \cdot RF \times LO_I \\
 &= K_I \cdot \{A_I \cdot \cos \omega_{BB}t \times \cos \omega_{LO}t - A_Q \cdot \sin \omega_{BB}t \times \sin \omega_{LO}t\} \times \cos \omega_{LO}t \\
 &= \frac{A_I K_I}{2} \cos \omega_{BB}t \{1 + \cos 2\omega_{LO}t\} - \frac{A_Q K_I}{2} \cdot \sin \omega_{BB}t \cdot \{\sin 2\omega_{LO}t + \sin 0\} \\
 &= \frac{A_I K_I}{2} \{\cos \omega_{BB}t\} \\
 \therefore BB_I &= BB'_I \times (-2) = -A_I K_I \cdot \cos \omega_{BB}t \rightarrow \therefore BB_I(\omega_{BB}t = 180^\circ) = A_I K_I
 \end{aligned} \tag{A.1}$$

$$\begin{aligned}
 BB'_Q &= K_Q \cdot RF \times LO_Q \\
 &= K_Q \cdot \{A_I \cdot \cos \omega_{BB}t \times \cos \omega_{LO}t - A_Q \cdot \sin \omega_{BB}t \times \sin \omega_{LO}t\} \times \sin \omega_{LO}t \\
 &= \frac{A_I K_Q}{2} \cdot \cos \omega_{BB}t \cdot \{\sin 2\omega_{LO}t - \sin 0\} - \frac{A_Q K_Q}{2} \cdot \sin \omega_{BB}t \cdot \{1 - \cos 2\omega_{LO}t\} \\
 &= -\frac{A_Q K_Q}{2} \{\sin \omega_{BB}t\} \\
 \therefore BB_Q &= BB'_Q \times (-2) = A_Q K_Q \cdot \sin \omega_{BB}t \rightarrow \therefore BB_Q(\omega_{BB}t = 90^\circ) = A_Q K_Q
 \end{aligned} \tag{A.2}$$

B. I/Q phase mismatch in mixer model [14].

$$\begin{aligned}
 LO_I &= \cos(\omega_{LO}t \pm \varphi) = \cos \omega_{LO}t \cdot \cos \varphi \mp \sin \omega_{LO}t \cdot \sin \varphi \\
 LO_Q &= \sin(\omega_{LO}t \pm \varphi) = \sin \omega_{LO}t \cdot \cos \varphi \pm \cos \omega_{LO}t \cdot \sin \varphi
 \end{aligned} \tag{A.3}$$

$$\begin{aligned}
 BB'_I &= RF \times LO_I \\
 &= \{A_I \cdot \cos \omega_{LO}t - A_Q \cdot \sin \omega_{LO}t\} \times \{\cos \omega_{LO}t \cdot \cos \varphi \mp \sin \omega_{LO}t \cdot \sin \varphi\} \\
 &= \frac{A_I}{2} \cos \varphi \pm \frac{A_Q}{2} \sin \varphi \rightarrow \therefore BB_I = BB'_I \times 2 = A_I \cos \varphi \pm A_Q \sin \varphi
 \end{aligned} \tag{A.4}$$

$$\begin{aligned}
 BB'_Q &= RF \times LO_Q \\
 &= \{A_I \cdot \cos \omega_{LO}t - A_Q \cdot \sin \omega_{LO}t\} \times \{\sin \omega_{LO}t \cdot \cos \varphi \pm \cos \omega_{LO}t \cdot \sin \varphi\} \\
 &= \pm \frac{A_I}{2} \sin \varphi - \frac{A_Q}{2} \cos \varphi \rightarrow \therefore BB_Q = BB'_Q \times 2 = \pm A_I \sin \varphi - A_Q \cos \varphi
 \end{aligned} \tag{A.5}$$

$$\therefore |BB| = \sqrt{A_I^2 + A_Q^2} \quad \therefore \angle BB = \angle(A_Q, A_I) \mp \varphi \tag{A.6}$$

C. LO leakage in mixer model [14].

$$\begin{aligned}
BB'_I &= (RF \pm K \cdot LO_I) \times LO_I \\
&= \{A_I \cos \omega_{BB}t \times \cos \omega_{LO}t - A_Q \sin \omega_{BB}t \times \sin \omega_{LO}t \pm K \cos \omega_{LO}t\} \times \cos \omega_{LO}t \\
&= \frac{A_I}{2} \cdot \{\cos \omega_{BB}t\} \pm \frac{K}{2} \\
\therefore BB_I &= BB'_I \times (-2) = -A_I \cdot \cos \omega_{BB}t \mp K \rightarrow \therefore BB_I(\omega_{BB}t = 180^\circ) = A_I \mp K
\end{aligned} \tag{A.7}$$

$$\begin{aligned}
BB'_Q &= (RF \pm K \cdot LO_Q) \times LO_Q \\
&= \{A_I \cos \omega_{BB}t \times \cos \omega_{LO}t - A_Q \sin \omega_{BB}t \times \sin \omega_{LO}t \pm K \sin \omega_{LO}t\} \times \sin \omega_{LO}t \\
&= -\frac{A_Q}{2} \cdot \{\sin \omega_{BB}t\} \pm \frac{K}{2} \\
\therefore BB_Q &= BB'_Q \times (-2) = A_Q \cdot \sin \omega_{BB}t \mp K \rightarrow \therefore BB_Q(\omega_{BB}t = 90^\circ) = A_Q \mp K
\end{aligned} \tag{A.8}$$

D. Even-order distortion in mixer model.

$$\begin{aligned}
x(t) &= A_1 \cdot \cos \omega_1 t + A_2 \cdot \cos \omega_2 t, \text{ where } \omega_1 \approx \omega_2 \\
y(t) &= \alpha_1 \cdot x(t) + \alpha_2 \cdot x^2(t) \\
&= \alpha_1 \cdot A_1 \cdot \cos \omega_1 t + \alpha_1 \cdot A_2 \cdot \cos \omega_2 t + \alpha_2 \cdot A_1^2 \cdot \cos^2 \omega_1 t \\
&\quad + \alpha_2 \cdot A_2^2 \cdot \cos^2 \omega_2 t + \alpha_2 \cdot A_1 \cdot A_2 \cdot \cos(\omega_1 + \omega_2)t \\
&\quad + \alpha_2 \cdot A_1 \cdot A_2 \cdot \cos(\omega_1 - \omega_2)t \rightarrow \text{almost DC component}
\end{aligned} \tag{A.9}$$

$$\begin{aligned}
x(t) &= (A + \varepsilon \cdot \cos \omega_m t) \cdot (a \cdot \cos \omega_c t + b \cdot \sin \omega_c t) \\
x^2(t) &= (A + \varepsilon \cdot \cos \omega_m t)^2 \cdot (a \cdot \cos \omega_c t + b \cdot \sin \omega_c t)^2 \\
&= 2 \cdot (a^2 + b^2) \cdot A \varepsilon \cdot \cos \omega_m t + \dots \rightarrow \text{Baseband signal}
\end{aligned} \tag{A.10}$$

E. Nonlinearity in amplifier model [11].

$$\begin{aligned}
y(t) &= a_1 x(t) + a_2 x^2(t) + a_3 x^3(t) \\
a_1 &= 10^{\frac{G_{dB}}{20}}, \quad a_2 = \frac{a_1}{\sqrt{2 \cdot 50 \cdot 10^{\frac{iip2}{10}}}}, \quad a_3 = a_1^3 \cdot \frac{10^{P1dB/20} - 10^{(P1dB+1)/20}}{10^{3(P1dB+1)/20}}
\end{aligned} \tag{A.11}$$

G_{dB} is the gain (V_{out}/V_{in} , [dB]), $iip2$ is the second-order intercept point, and $P1dB$ is the 1dB compression point of the linear function.

Appendix 2. RNM Baseband Equivalent Model

A. Z-domain filter model.

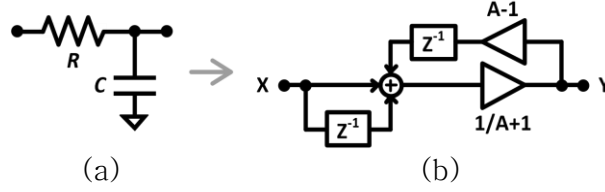


Figure A.1. Describing the filter model in (a) s-domain and (b) z-domain.

```

module abb_filter #(
    parameter gain = 1,
    parameter fmax = 1.0e9,
    parameter fmin = 1.0e6
)()
    input [11:0] ctrl_pole,
    input real in,
    output real out
);
    real period, pole, tau, Ax, Ay, XD, YD;

    // discrete-time filter: Y = X + Ax*Z-1 + Ax*Ay*Z-1
    assign period = 1.0/50.0e6;
    assign pole = real'(ctrl_pole) * (fmax-fmin)/4095.0;
    assign tau = 1.0/(2.0*3.141592*pole);
    assign Ax = 1.0/(1.0+(2.0*tau/period));
    assign Ay = (2.0*tau/period)-1;

    always @(posedge clk) begin
        XD = in_I;
        YD = out_I*Ay;
    end
    assign out = gain*Ax*(in_I+XD+YD);
endmodule

```

Figure A.2. Pseudocode for the proposed filter model.

Bilinear transformation can transform the s-domain filter characteristics (XMODEL) into the z-domain (RNM) with the same frequency characteristics. The first-order low-pass filter in Fig. A.1 (a) has a transfer function $H_a(s)$. It is converted to $H_d(z)$ through the following derivation process.

$$\begin{aligned}
 H_a(s) &= \frac{1}{sC} = \frac{1}{1 + RCs} = \frac{1}{1 + \tau s}, \quad \tau = \frac{1}{\omega_c} = \frac{1}{2\pi f_p} \\
 H_D(z) &= H_a\left(\frac{2}{T} \cdot \frac{z-1}{z+1}\right) = \frac{1+z^{-1}}{\left(1+\frac{2\tau}{T}\right) + \left(1-\frac{2\tau}{T}\right)z^{-1}} = \frac{Y}{X}
 \end{aligned} \tag{A.12}$$

$$\begin{aligned}
Y &= \frac{X}{1 + \frac{2\tau}{T}} + \frac{X \cdot z^{-1}}{1 + \frac{2\tau}{T}} + \frac{\left(\frac{2\tau}{T} - 1\right) \cdot Y \cdot z^{-1}}{1 + \frac{2\tau}{T}} \\
&= A_x \cdot \{X + X \cdot z^{-1} + A_Y \cdot Y \cdot z^{-1}\} \\
\text{where, } A_x &= \frac{1}{1 + \frac{2\tau}{T}}, \quad A_Y = \frac{2\tau}{T} - 1
\end{aligned} \tag{A.13}$$

Eq. (A.13) can be drawn as a block diagram shown in Fig A.1 (b) and implemented in SystemVerilog as shown in Fig A.2.

B. Complex-baseband equivalent channel.

[34] describes how to represent a complex baseband equivalent channel model. The following is a brief introduction to how to do that.

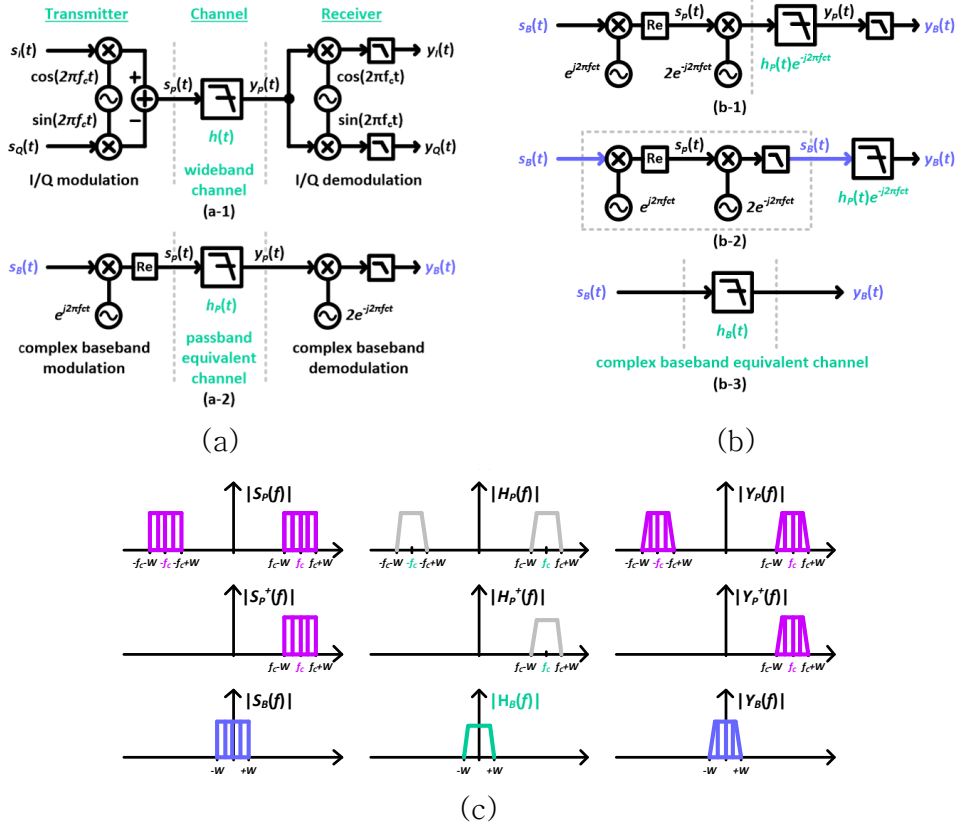


Figure A.3. Block diagram of the complex baseband equivalent channel.

$s_P(t)$ and $y_B(t)$ are passband signals, and $s_B(t)$ and $y_B(t)$ are baseband equivalent baseband signals. Since filtering $s_P(t)$ with the wideband channel response $h(t)$ is equivalent to filtering $s_P(t)$ with the passband equivalent channel response $h_P(t)$, $h(t)$ can replace with $h_P(t)$ as shown in Fig. A.3 (a). By deriving the output response of the passband filter as follows, we can redraw the block diagram as shown in Fig. A.3 (b-1) [34].

$$\begin{aligned}
y_P(t) &= [s_P(t) * h_P(t)] \cdot 2e^{-j2\pi f_c t} = \int s(\tau) \cdot h_P(t - \tau) d\tau \cdot 2e^{-j2\pi f_c t} \\
&= \int s(\tau) \cdot 2e^{-j2\pi f_c \tau} \cdot h_P(t - \tau) \cdot 2e^{-j2\pi f_c (t - \tau)} d\tau \\
&= [s(t) \cdot 2e^{-j2\pi f_c t}] * [h_P(t) \cdot e^{-j2\pi f_c t}]
\end{aligned} \tag{A.14}$$

As LPF and $h_P(t) \cdot e^{-j2\pi f_c t}$ are both LTI systems, the order of the two blocks can be reversed, as shown in Fig. A.3 (b-2). In this figure, the block of the squared part (modulate the baseband signal and then demodulate it again) receives $s_B(t)$ and produces $s_B(t)$, so we can remove them, as shown in Fig. A.3. (b-3). The remaining block, $h_P(t) \cdot e^{-j2\pi f_c t}$, is the baseband equivalent channel model [34].

$$h_B(t) = h_P(t) \cdot e^{-j2\pi f_c t} \tag{A.15}$$

Appendix 3. Parameter Coverage Analysis

The threshold is a constant determined when the model is defined. In contrast, the *meas_cov* module needs to calculate the sensitivity by comparing the relationship between the input and output signals during the simulation run. For reference, sensitivity is a function of the model parameter p_k and the input signal $x(t)$. Suppose N model parameters $\mathbf{p} = [p_1, p_2, \dots, p_k, \dots, p_N]$ produce output $y(t|\mathbf{p})$ for input $x(t)$. In that case, if the k th parameter is changed Δ (parameter is \mathbf{p}' and output is $y(t|\mathbf{p}')$), the sensitivity can be expressed as the difference between the two output signals, as shown in the following equation [32].

$$\begin{aligned}
 S_k(p, x(t)) &\equiv \lim_{\Delta \rightarrow 0} \frac{D(y(t|\mathbf{p}), y(t|\mathbf{p}'))}{0.5 \left(\frac{\Delta}{p_k}\right)^2} = p_k^2 \cdot \frac{\left\| \frac{\partial y(t|\mathbf{p})}{\partial p_k} \right\|_2^2}{\|y(t|\mathbf{p})\|_2^2} \\
 &= p_k^2 \cdot \frac{\int_0^{T_{sym}} \left(\frac{\partial y(t|\mathbf{p})}{\partial p_k} \right)^2 dt}{\int_0^{T_{sym}} (y(t|\mathbf{p}))^2 dt}
 \end{aligned} \tag{A.16}$$

The analog adder outputs weighted-sum values for multiple inputs, and the N -bit DAC outputs weighted-sum values for N digital input codes. In both cases, it produces results $y = \sum_{i=1}^N w_i x_i$ for the input $\mathbf{x}(t) = [x_i(t)]$ and weight $\mathbf{w}(t) = [w_i(t)]$ parameters. The following equations define sensitivity and threshold, respectively, and the physical meaning of the sensitivity function is the relative energy of the input signal/code [32].

$$S_k(p, x(t)) = w_k^2 \cdot \frac{\int_0^{T_{sym}} (x_k(t))^2 dt}{\int_0^{T_{sym}} (y(t))^2 dt}, \quad \theta_k = \frac{w_k^2}{\sum_{i=1}^N w_i^2} \tag{A.17}$$

The filter model can model the high frequencies generated by modulation in the TX/RX stage, the band characteristics of the RFIC, and the channel characteristics. These analog filters can be expressed as a transfer function $H(s)$ in the s domain with poles $\mathbf{p} = [p_i]$ and zeros $\mathbf{z} = [z_j]$ as parameters [10]. The following equations

define sensitivity and threshold, respectively, and the physical meaning of the sensitivity function is the spectral distribution of the input power over the maximum power [32].

$$H(s) = \prod_{j=1}^{N_z} \left(1 + \frac{s}{z_j}\right) / \prod_{i=1}^{N_p} \left(1 + \frac{s}{p_i}\right) \quad (\text{A.18})$$

$$S_k(p, z, x(t)) = p_k^2 \cdot \frac{\int_0^{T_{\text{sym}}} \left(\frac{\partial(x(t) * h(t))}{\partial p_k} \right)^2 dt}{\int_0^{T_{\text{sym}}} (x(t) * h(t))^2 dt} \quad (\text{A.19})$$

$$\theta_p = p^2 \cdot \frac{\int_0^{T_{\text{sym}}} \left(\frac{\partial h(t)}{\partial p} \right)^2 dt}{\int_0^{T_{\text{sym}}} (h(t))^2 dt}$$

```

module meas_fcov (
    `input_xreal out,           // output
    `input_xreal in,           // input
    `input_real p               // pole (bandwidth)
);
    xreal sen, thr;             // sensitivity, threshold
    xbit cov; bit coverage;     // coverage

    real g;
    always @(p) g = 1.0/(6.24*p);

    // (1) Calculate Signal Sensitivity
    filter #(.gain(g), .poles(`{p, 0, p, 0}), .zeros(`{0.0}))
    ipert_1 (.in(in), .out(out_pert_neg));
    filter #(.gain(g), .pole(`{p, 0}), .zeros(`{0.0}))
    ipert_2 (.in(in), .out(out_pert_pos));
    add #(.num_in(2), .scale(`{-1.0, 1.0}))
    ipert_3 (.in({out_pert_neg, out_pert_pos}), .out(out_pert));

    // (2) Calculate Optimum Threshold
    assign thr = 0.5 / (6.28*6.28*p*p);

    // (3) Decide Parameter Coverage
    cmpare cov_result (.in(sen), .in_ref(thr), .out(cov), .trig(clk));
    xbit_to_bit #(.width(num_bit)) xtb_cov (.in(cov), .out(coverage));

    // (4) Report Coverage Result
    covergroup cg;
        cover_point : coverpoint coverage;
    endgroup
    cg cg_inst = new();
    always @(posedge clk) cg_inst.sample();
endmodule

```

Figure A.3. Pseudocode for the module that measures the coverage results of filter model

Appendix 4. List of Models

A. RF Transceiver Models

```
// RF Transceiver Model (XMODEL):  
TOP.sv, transmitter.sv, receiver.sv, scs_gen.sv, qam_gen.sv  
ABB.sv, channel.sv, GMC.sv mixer_cal.sv, PA.sv, receiver.sv,  
ADC.sv, DCOC.sv, mixer.sv, ref_gen.sv, LNA.sv, sw.sv, mux.sv,  
Route.sv, BB.sv, add_cov.sv, dac_cov.sv, meas_cov.sv, VGA.sv  
  
// RF Transceiver Model (BBEQ-RNM):  
(...) zfilter.sv  
  
// Low-Noise Amplifier Model:  
LNA_wCC.sv, LNA_woCC.sv
```

B. Testbench and Post-processing Scripts

```
// RF Transceiver Model (XMODEL/BBEQ-RNM):  
TB_IDEAL.sv, TB_woDCOC.sv, TB_woGMC.sv, TB_woDPD.sv,  
TB_woAGC.sv, TB_256QAM.sv, TB_64QAM.sv TB_FR1.sv  
TB_FR2.sv, TB_TOP.sv, TB_FR1_2ns/1ns/500ps/250ps.sv,  
TB_FR2_100ps/50ps/25ps/12_5ps.sv  
  
// Low-Noise Amplifier Model  
TB_640QAM.sv, TB_640QAM_two_LNA.sv, TB_Fund_IM.sv,  
TB_LNAs_meas.sv, TB_multi_tone.sv, TB_QAM.sv, TB_TOP.sv,  
TB_two_tone.sv, TB_LNA_wc.sv, TB_LNA_woC.sv  
  
// Parameter Coverage Analysis  
add_cov.sv, dac_uniform.sv, common.sv, slice_cov.sv, dac_cov.sv,  
compare_cov.sv, file1_1st_static.sv, scale_cov.sv  
  
// Performance Estimating Script  
coverage.py, calculate.py, cal_pole.py, imp_cal.py, chebyshev.sv,  
cmd_ofdm.py, plot_constellation.py, calc_evm.py
```

Bibliography

- [1] J. Kim, et al., “A Model–First Design and Verification Flow for Analog–Digital Convergence Systems: A High–Speed Receiver Example in Digital TBs,” Int’ Symp. Circuits and Sys. (ISCAS), May 2012.
- [2] E. Shera, “Buck Converter Modeling in SystemVerilog for Verification and Virtual Test Applications,” Int’ Mixed–Signal Test Workshop (IMSTW), Jun. 2015.
- [3] B. J. Lamere, Introduction to Logic Circuits & Logic Design with Verilog, 2nd ed. Switzerland: Springer, 2019.
- [4] C. Y. Park, et al., “Event–Driven Modeling and Simulation of 5G NR–Band RF Transceiver in SystemVerilog,” in IEEE Intl’ conf. on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD), Jul. 2021.
- [5] J. Lee, et al., “21.6 A Sub–6GHz 5G New Radio RF Transceiver Supporting EN–DC with 3.15Gb/s DL and 1.27Gb/s UL in 14nm FinFET CMOS,” in IEEE ISSCC Dig. Tech. Papers, Feb. 2019.
- [6] W. Dunham, “RFIC Design Methodology: Functional Verification,” in IEEE Region 5 Tech. Conf., Apr. 2007.
- [7] H. Chang, et al., “Verification of Complex Analog and RF IC Designs,” Proc. Of the IEEE, Mar. 2007.
- [8] K. Karnane, et al., “Solutions for Mixed–Signal SoC Verification,” Cadence Design Systems, 2009.
- [9] J. E. Chen, “A Modeling Methodology for Verifying Functionality of a Wireless Chip,” in IEEE Behavioral Modeling and Simulation Workshop (BMAS), Sep. 2009.
- [10] J. He, et al., “System–Level Time–Domain Behavioral Modeling for a Mobile WiMax Transceiver,” in IEEE Behavioral Modeling and Simulation Workshop (BMAS), Sep. 2006.
- [11] J. B. David, “Radio Receiver Mixer Model of Event–Driven Simulators to support Functional Verification of RF–SOC Wireless Links,” in IEEE Behavioral Modeling and Simulation Workshop (BMAS), Sep. 2010.

- [12] J. Jang, et al., “True Event-Driven Simulation of Analog/Mixed-Signal Behaviors in SystemVerilog: A Decision-Feedback Equalizing (DFE) Receiver Example,” in IEEE Custom Integr. Circuits Conf., Sep. 2012.
- [13] C. Beyerstedt, et al., “Baseband Equivalent Modeling Approach for Analog Linear Transfer Functions in Event-driven Simulations,” in IEEE Intl’ conf. on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD), Jul. 2019.
- [14] J. E. Chen, “Modeling RF Systems,” The Designer’s Guide Community, accessed Jul. 29, 2021, <http://designers-guide.org/modeling/modeling-rf-systems.pdf>
- [15] XMODEL Reference Manual, Release 2020.05, Scientific Analog, Inc., 2020.
- [16] Mak, et al., “Transceiver Architecture Selection: Review, State-of-the-art Survey and Case Study,” in IEEE Circuits Syst. Mag., Sep. 2007.
- [17] T. Lee, The Design of CMOS Radio-Frequency Integrated Circuits, 2nd ed. Cambridge, UK: Cambridge University Press, 2004.
- [18] J. W. Jeong, et al., “Built-In Self-Test and Digital Calibration of Zero-IF RF Transceiver,” Trans. Very Large Scale Integ. Syst. (TVLSI), Jan. 2016.
- [19] H. Ye., et al., “A Digital IQ imbalance Self-Calibration in FDD Transceiver,” Int’ Sym. VLSI Design, Auto. and Design (VLSI-DAT), Apr. 2017.
- [20] L. Anttila., et al., “Circularity-Based I/Q Imbalance Compensation in Wideband Direct-Conversion Receivers,” Trans. Vehicular Tech. (TVT), Jul. 2008.
- [21] L. Yu., “A Novel Adaptive Mismatch Cancellation System for Quadrature IF Radio Receivers,” IEEE Trans. Circuits Syst. II, Jun. 1999.
- [22] I. Elahi, “IIP2 Calibration by Injecting DC Offset at the Mixer in a Wireless Receiver,” IEEE Trans. Circuits Syst. II, Exp. Briefs, Dec. 2007.
- [23] Y. Liu, et al., “A General Digital Predistortion Architecture Using Constrained Feedback Bandwidth for Wideband Power Amplifiers,” IEEE Trans. Microw. Theory Techn., May 2015.

- [24] A. Zaidi, et al., 5G Physical Layer Principles, Models and Technology Components, Academic Press, 1st ed. UK: Academic Press, 2018.
- [25] S. A. Maas, Nonlinear Microwave and RF Circuits, 2nd ed. Boston, USA: Artech, 2003.
- [26] C. Eun, et al., “A new Volterra predistorter based on the indirect learning architecture,” IEEE Trans. Signal Proc., Jan. 1997.
- [27] C. Y. Park, “A Volterra–Series Model in SystemVerilog/XMODEL for Nonlinear RF Low–Noise Amplifiers,” Design and Verification Conf. (DVcon), Mar. 2021.
- [28] J.–E. Jang, et al., “Event–Driven Simulation of Volterra Series Models in SystemVerilog,” in IEEE Custom Integr. Circuits Conf., Sep. 2013.
- [29] T. W. Kim, “A Common–Gate Amplifier with Trans–conductance Nonlinearity Cancellation and Its High–Frequency Analysis Using the Volterra Series,” IEEE Trans. Microw. Theory Tech., Jun. 2009.
- [30] J. Roychowdhury, “Reduced–Order Modeling of Time–Varying Systems,” IEEE Trans. Circuits Syst. II, Exp. Briefs, Oct. 1999.
- [31] S. Li, et al., “An E–Band High–Linearity Antenna–LNA Front–End with 4.8dB NF and 2.2dBm IIP3 Exploiting Multi–Feed on–Antenna Noise–Canceling and Gm–Boosting,” in IEEE ISSCC Dig. Tech. Papers, Feb. 2020.
- [32] J. Lee, “Parameter Coverage Analysis on Simulation of Analog Functional Models,” Ph.D Dissertation, Aug. 2019.
- [33] D. Rich, “The Missing Link: The Testbench to DUT Connection,” Design and Verification Conf. (DVcon), Mar. 2012.
- [34] P. Schniter, et al., “Introduction to Analog and Digital Communications,” accessed May 17, 2021, <http://cnx.org/content/col10968/1.2/>.

Abstract

도래한 초연결시대에서는 스마트폰뿐만 아니라 다양한 사물 인터넷 디바이스들이 5세대 이동통신 시스템을 활용하면서, 늘어난 데이터량과 트래픽을 감당하기 위해 밀리미터파 대역의 사용이 필수적일 것이다. 시스템이 보다 대용량화 그리고 광대역화 됨에 따라, 통신 규약을 만족시키기 위해, 점차 거대한 디지털 캘리브레이션 및 신호처리 로직이, 무선 통신 전단부 칩에 함께 집적되고 있다. 따라서 멀티-도메인의 신호(아날로그/디지털/무선통신 신호)가 복잡하게 혼성된 무선통신 집적회로 칩을, 짧은 개발 기간 동안 충분히 검증하기엔 어려움이 따른다. 일반적으로 혼성 신호 시스템을 검증하기 위해서는, 하위 시스템을 모두 포함해서 시간 도메인의 시뮬레이션을 수행해야 하는데, 이를 위한 스파이스와 스파이스-하드웨어 기술 언어의 co-시뮬레이션은 지나치게 느리다는 한계가 있기 때문이다. 따라서, 멀티-도메인의 신호를 빠르고 정확하게 시뮬레이션 가능하게 하는 모델링 방법과, 다양한 시나리오의 검증 완성도를 향상시켜줄 있는 검증 기술이 모두 요구된다.

혼성 시스템을 검증하기 위해서는, 아날로그와 무선 통신 블록들을 시스템 베릴로그 상에서 구현된 함수적 모델로 대체하고, 디지털 블록들과 함께 하나의 디지털 플랫폼에서 시뮬레이션하는 것이 효과적이다. 실제 설계할 때, 문제가 되는 대부분의 에러들은, 연결 오류, 부호 오류, 신호 순서 오류, 혹은 잘못된 파워 도메인과의 연결과 같이 사소한 오류들이다. 이러한 오류를 찾기 위해, 오래 걸리는 트랜지스터-레벨의 시뮬레이션을 수행하기보다는, 아날로그 스파이스 모델들을 시스템 베릴로그 모델들로 대체하고, 보다 다양한 시나리오를 빠르게 검증하는 방법이 검증 완성도를 향상시키는데 적합하다. 그럼에도, 지나치게 단순한 선형 모델이나, 중요한 회로 특성이 빠진 모델로는 원하는 수준의 검증이 불가능할 수 있다. 예를 들어, 직접 변조 구조의 무선통신 송수신기에서 발생하는 비이상 효과, 저전력 동작을 하면서 발생하는 비선형 효과, 그리고 흔히 메모리 효과는 모델에 효과를 충분히 반영해 주어야만, 주파수 도메인에서의 검증, 성능 예측 등의 검증을 의미 있게 수행할 수 있다. 문제는 비선형 시스템은 훨씬 복잡한 식으로 표현되며, 시뮬레이션 시 연산량도 크게 늘어나기 때문에, 비선형 모델을 만들고 시뮬레이션

하기가 쉽지 않다는 것이다. 따라서 모델이 비이상성들을 충분히 반영하면서도 효과적인 검증을 가능하게 하는 모델링/시뮬레이션 방법 역시 요구된다.

본 학위 논문에서는, 무선통신 송수신기 집적회로 전체의 모사 모델을 제안한다. 모델은 누설 신호와 신호 간 불일치에 의한 비-이상적인 효과를 엑스모델의 알고리즘을 활용해 반영하였고, 비선형성과 메모리 효과를 볼테라-섭동법을 활용해 반영하였다. 제안하는 모델은 다양한 주파수 대역과 동작 모드를 검증하는데, 기존 등가 베이스밴드 모델보다 30~1800배 빠르게 시뮬레이션 할 수 있었고, 비이상 효과에 대해, 통신 성능들(심볼의 오류 백터의 크기, 인접 채널의 파워 그리고 비트 에러)을 평가 가능했다. 나아가, 아날로그 검사기를 활용한 기능 검증법과 모델 파라미터 커버리지 분석법을 적용하여, 시스템-레벨 검증의 완성도를 향상시켰다. 무선통신 집적회로 모델에 다양한 디자인/파라미터 오류를 주입하고, 시뮬레이션 동안 검사기가 찾은 에러의 개수와 커버리지 결과를 실험적으로 보였다.