Modeling RF Systems

Jesse E. Chen

Version 1, 6 March 2005

This paper discusses behavioral models of RF blocks. The RF blocks amplify, filter, modulate, and demodulate RF carriers. The paper classifies behavioral RF models as either passband or baseband. Like device-level models, passband models explicitly simulate every cycle of the carrier. However, baseband models suppress the carrier to trade some accuracy for a dramatic increase in execution speed. The first part of this paper explains one way in which behavioral models of RF blocks fit into the design flow. The next two sections describe passband and baseband models. The paper concludes with an example of how behavioral models can help diagnose anomalous lab observations of an OFDM (orthogonal frequency division multiplexing) system.

Last updated on November 10, 2011. You can find the most recent version at www.designers-guide.org. Contact the author via e-mail at wb6mcc@aol.com.

1 Behavioral RF Models

This paper discusses behavioral models of RF blocks. RF blocks amplify, filter, modulate, demodulate, and synthesize RF carriers. There are two basic kinds of behavioral RF models, passband and baseband. Like device-level models, passband models explicitly simulate every cycle of the carrier whereas baseband models suppress the carrier to trade some accuracy for a dramatic increase in execution speed. The first part of this paper explains how behavioral models of RF blocks fit into the design flow. The next two sections describe passband and baseband models. The paper concludes with an example of how behavioral models can help diagnose anomalous lab observations of an OFDM (orthogonal frequency division multiplexing) system.

2 Behavioral RF Models and Top-Down Design

To use a set of models effectively, it helps to know how the set fits into the overall design flow. Behavioral RF models can play a key role in top-down design flow. Here, “top” refers to a system specification, a document specifying system performance. A system specification describes what the system should do, not how to build it. A specification is the highest level of abstraction. Models at the top level are specified, not extracted. The bottom of the design flow is a detailed design. A detailed design is a set of schematics, a device library, and a layout. A detailed design describes exactly how to build the system. Everything is known about a detailed design. The detailed design is the lowest level of abstraction. Models at the bottom are extracted. Top-down design refers to the transformation of system specifications into detailed designs. In contrast, bottom-up design refers to the process of designing a system around an existing component. In practice today, schedule and cost constraints force a mix of top-down and bottom-up design flows.

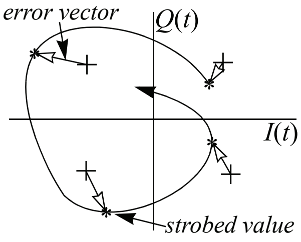
The behavioral models discussed in this paper help explore the design space near the top of the design flow. Near the top, only those parameters that affect the decisions at hand are modeled; all other parameters are ignored because they needlessly and geometrically expand the design space. The first step in the top-down design of an RF system is to choose the architecture. The architecture is a diagram of interconnected RF function blocks showing the sequence of operations the RF system performs to transform inputs into outputs. The relevant features of the RF blocks are specified by standard RF metrics such as gain, bandwidth, noise figure, and third order intercept point. Standard RF metrics evolved as measurements that quickly determine whether a block or architecture is appropriate for the application at hand. The most common way of designing the architecture is to first choose a candidate architecture then balance the block specifications to achieve end-to-end performance. Conventional RF system design methods rely on spreadsheets [1] and closed form expressions relating block-level RF specifications to end-to-end RF specifications. Conventional spreadsheet methods will not be replaced in the foreseeable future because they are highly interactive and well understood. However, as communications systems become more digital, conventional design methods must be augmented. DSP engineers do not work directly with the deterministic standard RF metrics mentioned above. DSP engineers work with statistical metrics. One way to relate RF and DSP metrics is through Monte Carlo simulation [2]. Simulation requires three elements that form a design environment: a simulator to perform the numerical computations, a user interface to make data entry / visualization and simulator control efficient, and a set of models to translate the physical problem into a mathematical problem. By themselves, the individual elements of a design environment are of limited use. Efficiently integrated, they make a powerful design tool. However, a gap separates DSP and RF design environments. The gap is due to the disparate nature of the two design disciplines. Properly managed, behavioral models can bridge that gap.

To bridge the gap between DSP and RF design environments, the models must run in both environments and share parameters. Let's start with the DSP design environment. Many DSP environments use simple baseband equivalent models of the RF blocks to keep run times reasonable. The key DSP metric is always bit error rate (BER). To reliably simulate a BER, you should count at least 100 errors. To simulate a marginal BER of say 10^{-4}, you would need to simulate at least one million bits. Even at a bit rate of 10Mb/s, you should simulate 100ms of action. To simulate nonlinearity without introducing simulation errors due to aliasing, the sample rate must be at least three times higher than any frequency of interest. If adjacent channel interferers are present, even the simplest nonlinearity will noticeably slow data flow simulation. To make matters worse, DSP engineers never look at just one BER data point. Typically, they repeatedly simulate families of BER curves as they test, diagnose, and refine their algorithms. Baseband equivalent models of the RF blocks are not as accurate as passband models but the increase in execution speed far outweighs the loss in accuracy when performing architectural trade-offs in terms of BER.

On the analog side, baseband equivalent models of the RF blocks let the analog baseband designers estimate how their designs affect system performance. Some receivers have a significant amount of baseband analog circuitry. For example, only a small fraction of the transistors in a direct conversion receiver operate at RF, most operate at baseband. BER depends on gain and the automatic gain control (AGC) loop sets the gain. Communications standards such as 802.11a/b send data in bursts. When signals arrive in bursts, the receiver must set the gain quickly, sometimes based on only two or three measurements of signal level. In steady state, the baseband analog circuitry can be modeled in the DSP design environment as a linear time invariant filter. However, during AGC transients, the baseband analog circuits can saturate or slew rate limit. Slew rate limiting is not easily modeled in a DSP environment but is modeled in VerilogAMS with just one line. Simple saturation within active filters can also be hard to model in a DSP design environment. The non-ideal filter would require an elaborate model to trick the data flow simulator into mimicking an analog simulator. Elaborate models invite human error. Any decent analog simulation user interface reduces human errors by letting the user simply draw the schematic. The amplifier model could be implemented in VerilogAMS with just a handful of lines. Since the analog baseband circuitry has no RF carrier, it could even be simulated at the transistor level over night. The AGC algorithm itself is usually simple enough to model at the behavioral or RTL level in VerilogAMS. However, to simulate the end-to-end system on anything less than geological time scales, the RF circuitry must be modeled with baseband equivalent models.

BER takes too long to simulate in the analog design environment but there are statistical alternatives. One alternative is the error vector magnitude (EVM). A general quadrature amplitude modulated RF carrier can be described as

$$V_{\mathrm{rf}}(t) = I(t)\cos(\omega t) - Q(t)\sin(\omega t) \tag{1}$$



where ω is the carrier frequency in radians per second. $I(t)$ and $Q(t)$ represent the information riding on the carrier. Plotting $I(t)$ against $Q(t)$ gives a trajectory in a 2-dimensional plane. With no impairments, if we strobe the trajectory at the symbol rate we see a set of clearly defined points that form the symbol constellation. Impairments smear the points into clouds. The error vector is the vector between the transmitted and received constellation point. EVM is often the normalized RMS value of the magnitude of all the error vectors. Normalization depends on the standard (802.11a, 802.11b, etc.). Simulating a repeatable EVM measurement can easily require hundreds of symbols. Each symbol usually spans thousands of carrier cycles. Explicitly simulating a 100,000 carrier cycles with passband models is a tough job for any analog simulator. However, baseband equivalent models can simulate hundreds of symbols in just a few minutes regardless of carrier frequency. The price of such speed is accuracy; but despite their limitations, baseband models are still a computational bargain. If a candidate design does not work with simple-but-fast models, there is little reason to invest in more accurate models because the resulting design would not be very robust. Baseband equivalent models quickly narrow the design space to a size that can be explored in more detail with bottom-up models.

An RF architectural model implemented in RF and DSP design environments can serve as an executable specification. An executable specification succeeds where conventional specifications fail because with the proper tests, it creates a safety net for catching misunderstandings and unforeseen problems. For example, the usual RF specifications do not address AC coupling. AC coupling [3] simplifies the bias networks and minimizes steady state DC offset. A DSP team may not consider the effects of AC coupling. Not being analog circuit designers, they may not be aware of the biasing and DC offset difficulties. In a direct conversion receiver, AC coupling capacitors in the baseband legs introduce long transients that can interfere with the AGC and carrier frequency offset correction (CFOC) algorithms. It is extremely trivial for the analog team to insert the AC coupling capacitor into their version of the executable specification. If the DSP team included AGC and CFOC tests with the executable specification, the analog team would immediately see the system level problems associated with AC coupling. Without the executable specification, system problems caused by a simple AC coupling capacitor might not be detected until silicon arrived. The schedule hit can be more costly in the end than the extra silicon spin.

The rest of this paper describes the most basic building blocks of RF transmitters and receivers: amplifiers, mixers, and filters. Passband models are discussed first because they are less abstract and therefore better suited for introducing the standard RF metrics. Mixers are covered as *IQ* modulators and demodulators. *IQ* modulators/demodulators are composed pairs of mixers. Pairs of mixers are slightly more challenging to model than single mixers because mixer pairs can introduce crucial impairments associated with asymmetry. Passband filter models are not covered since they require no special modeling techniques. The baseband sections explain how to suppress the RF carrier. The baseband equivalent amplifier model introduces an impairment that is hard if not impossible to model at passband. The baseband section concludes with a section on modeling passive reactive elements. Baseband equivalent models of reactive elements are most useful in constructing ideal filter models that are compatible with analog simulators such as SPICE and Spectre. The paper concludes with some examples.

3 Passband Models

Two kinds of passband models exist, behavioral and device-level. Device-level models of RF blocks are inherently passband models. Behavioral passband models still simulate every carrier cycle but run faster because they are mathematically simpler than their device-level counterparts. Besides being useful in validating baseband equivalent models, passband behavioral models are useful in creating efficient test benches for device-level model of RF blocks. For example, suppose you are responsible only for the final down converting mixers in a receiver and would like to know how your design affects two consecutive symbols in the short training sequence (STS) of an 802.11a packet.[1] Two symbols cover only a few thousand carrier cycles and can be quickly simulated with transient-envelope methods [4]. Since you are interested only in distortion caused by your circuitry, simulating the rest of the RF circuitry with behavioral models ensures that the only impairments you see are your fault.

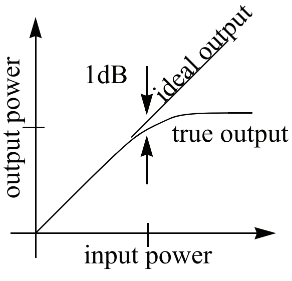
This section describes nonlinear passband behavioral models of amplifiers, modulators, and demodulators. Near the top of the design flow, these blocks are all treated as static blocks. Static blocks have no memory; the present output depends only on the present input. In contrast, a filter is a dynamic block because it has memory; the present output depends on the history of the input through a convolution integral. Dynamic models depend on frequency, static models do not. Nonlinear dynamic passband models are most often based on the Volterra series [5]. However, near the top of a design flow, there is rarely enough information to build a Volterra Series model and the added accuracy does not affect many architectural decisions. Volterra Series models are more useful for bottom-up verification [6].

The most basic RF block is the amplifier. Nonlinear and noise behavior of all the other blocks are derived from the amplifier model.

3.1 Amplifiers

3.1.1 Nonlinear Behavior

A practical behavioral model is parameterized in terms of standard metrics. Half the challenge in modeling RF blocks is in translating standard RF metrics into modeling parameters. Let's start by reviewing the metrics for nonlinear behavior. An ideal amplifier simply scales the input signal. Real amplifiers can not scale an arbitrarily large input signal because the supply voltage limits the maximum output voltage the amplifier can produce. As the input increases, the output saturates at some voltage. In RF literature, saturation is often referred to as AM/AM conversion (AM stands for amplitude modulation). AM/AM conversion in amplifiers can depend on carrier frequency but over the bandwidth of the signal, it is nearly static. The one dB compression point [7] is a common specification for the onset of saturation. At the one dB compression point, the true output falls short of the ideal output



[1] The IQ trajectory of an STS symbol can qualitatively reveal several impairments such as quadrature error, IQ gain imbalance, filter effects, frequency pulling, and frequency offset.

by 1dB. The one dB compression point can be measured along input or output axes. The one dB compression point is most often used to specify nonlinearity of the power amplifier (PA) within a transmitter.[2] The simplest way to control nonlinear distortion in the PA is to keep the average power far enough below the one dB compression point that the largest input power excursion does not drive the amplifier past the compression point. The required "back off" from the one dB compression point can be estimated by pumping up-converted worst case baseband signals through a behavioral model of the nonlinear power amplifier and checking for EVM and spectral mask violations.

Standard RF metrics for receiver nonlinearity are slightly less direct than the one dB compression point because they address a different problem. Receivers must often process signals from distant transmitters while nearby transmitters operate in adjacent channels. The nearby transmitter problem is called adjacent channel interference (ACI). The receiver metrics of nonlinearity (or linearity) are called intercept points [8]. Intercept points lead nicely into a discussion of static nonlinear models because, unlike the one dB compression point, intercept points are defined in terms of a model. The model is a Taylor series expansion of the static nonlinearity. Let the input and output voltages of the amplifier (or receiver) be v_i and v_o. The nonlinear relationship between input and output voltage can be approximated by:

$$v_\mathrm{o}(v_\mathrm{i}) = v_\mathrm{DCO} + av_\mathrm{i} - bv_\mathrm{i}^2 - cv_\mathrm{i}^3 - \text{higher order terms}\,. \tag{2}$$

v_DCO is a constant offset voltage and usually not a concern since RF circuits are usually AC coupled[3]. a, b, and c are constants that constitute the modeling parameters. The second term in (2) is the desired term. The desired term is the linear gain times the input. All other terms distort the output and are therefore undesired. The second order intercept point is related to b, the third order intercept point is related to c, and so on. The higher order terms are usually ignored, especially at the architectural design phase. Let the input voltage be composed of signals of identical power from two transmitters operating at different frequencies. The input signal is $v_\mathrm{i}(t) = A\cos(\omega_1 t) + A\cos(\omega_2 t)$. Let the DCO term and terms of higher order than three be negligible. Substituting into (2) gives

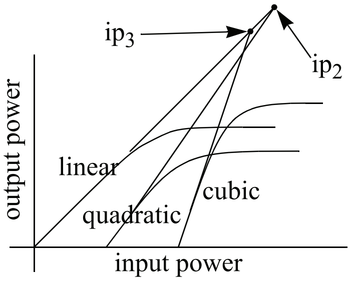
$$\begin{aligned} v_\mathrm{o} = {} & aA(\cos(\omega_1 t) + \cos(\omega_2 t)) - bA^2(\cos(\omega_1 t) + \cos(\omega_2 t))^2 - \\ & cA^3(\cos(\omega_1 t) + \cos(\omega_2 t))^3\,. \end{aligned} \tag{3}$$

Expanding the trigonometric terms and applying some trigonometric identities [8] gives:

[2] As mentioned earlier, one problem with digital communication systems is that the system metrics are not easily related to RF metrics. Nonlinearity in modern transmitters is specified in terms of spectral masks, adjacent channel power ratio, and/or EVM (after equalization). The most straightforward way to translate transmitter RF metrics into system metrics is through simulation and that requires behavioral models.

[3] AC-coupling capacitors are common in RF circuitry. However, AC-coupling in the baseband legs can cause long transients.

$$\begin{aligned} v_{\mathrm{o}} = {} & aA(\cos(\omega_1 t) + \cos(\omega_2 t)) - \\ & bA^2 + b\frac{A^2}{2}(\cos(2\omega_1 t) + \cos(2\omega_2 t)) - \\ & bA^2(\cos(\omega_1 t + \omega_2 t) + \cos(\omega_1 t - \omega_2 t)) - \\ & c\frac{9}{4}A^3(\cos(\omega_1 t) + \cos(\omega_2 t)) - \\ & c\frac{1}{4}A^3(\cos(3\omega_1 t) + \cos(3\omega_2 t)) - \\ & c\frac{3}{4}A^3(\cos(2\omega_1 t + \omega_2 t) + \cos(2\omega_2 t + \omega_1 t)) - \\ & c\frac{3}{4}A^3(\cos(2\omega_1 t - \omega_2 t) + \cos(2\omega_2 t - \omega_1 t)) \,. \end{aligned} \tag{4}$$



The terms involving $2\omega_i - \omega_j$ are of particular interest because they can propagate through the downstream filters. Note that as the input power increases, the desired output power increases linearly. However, power in the intermodulation terms, terms involving linear combinations of the two frequencies, increases at a faster rate because they involve A^2 and A^3. As the input increases further, more power slips into higher harmonics and the lower order coefficients approach limits. Intercept points are defined by extrapolating from the small-signal conditions. The advantage of intercept points is that one can apply simple geometric arguments to derive expressions for dynamic range [8]. (Here, we are only concerned with parameter definitions and how to parameterize the model in terms of a standard RF specification, not how to estimate dynamic range.) The extrapolated power level at which the power in one of the input sinusoids equals the power in the $\omega_1 - \omega_2$ intermodulation tone is called the second order intercept point, ip2, and can be referred to the input or output. Similarly, the extrapolated power at which the power in one of the input sinusoids equals the power in the $2\omega_1 - \omega_2$ intermodulation tone is called the third order intercept point, ip_3, and can also be referred to either the input or output.

Now let's relate the input referred intercept points, iip_2 and iip_3, to the modeling parameters b and c[4], starting with b. Intercept points (iip_2 and iip_3) are usually specified in terms of dBm. Power in dBm equals ten times the base ten logarithm of the power in milliWatts. We need to relate dBm to peak sinusoidal Volts since A represents peak sinusoidal Volts. To relate Volts to Watts we need a reference resistance, commonly 50 Ohms. Thus, if we let the value of A associated with iip_2 be A_{iip2},

$$\mathrm{iip}_2 = 10\log\left(A_{\mathrm{iip2}}^2 \frac{1000}{2(50)}\right) \tag{5}$$

and

[4] Linear gain and DC offset are easily related to their modeling parameters since they are specified directly, i.e. not indirectly through an intercept point.

$$A_{\text{iip2}} = \sqrt{\left(2\frac{(50)}{1000}(10)^{\frac{\text{iip}_2}{10}}\right)}. \tag{6}$$

Now, using the definition of the second order intercept point,

$$aA_{\text{iip2}} = bA_{\text{iip2}}^2 \tag{7}$$

and

$$b = \frac{a}{A_{\text{iip2}}}. \tag{8}$$

Similarly for the third order intercept point, we have

$$A_{\text{iip3}} = \sqrt{\left(2\left(\frac{50}{1000}\right)10^{\frac{\text{iip}_3}{10}}\right)}. \tag{9}$$

From the definition of the third order intercept point, we have

$$aA_{\text{iip3}} = \frac{3}{4}cA_{\text{iip3}}^3 \tag{10}$$

and finally,

$$c = \frac{4}{3}\frac{a}{A_{\text{iip3}}^2}. \tag{11}$$

The one dB compression point can also determine *c*. In modeling power amplifiers, the second order term is often neglected because it has no power near the carrier fundamental. The DC and second harmonics of the carrier are filtered out either by bandpass filters or antenna matching networks. Thus, the goal is to relate the one dB compression point to *c*. Note that an attenuation of one dB corresponds to a scale factor of 0.8913. Let $A_{1\text{dbcp}}$ be the amplitude of the sinusoidal input at the one dB compression point and let cp_1 be the input referred one dB compression point in dBm. From the definition of dBm,

$$A_{1\text{dbcp}} = \sqrt{\left(2\left(\frac{50}{1000}\right)10^{\frac{\text{cp}_1}{10}}\right)} \tag{12}$$

From the definition of the one dB compression point,

$$aA_{1\text{dbcp}} - \frac{3}{4}cA_{1\text{dbcp}}^3 = 0.8913aA_{1\text{dbcp}}, \tag{13}$$

and finally,

$$c = 0.145\frac{a}{A_{1\text{dbcp}}^2} \tag{14}$$

The amplifier model presented here is a polynomial. The trouble with polynomials is that they blow up as the input approaches infinity. This can cause convergence errors and is also just not realistic. It is therefore a good idea to clamp the polynomial at the critical points, points where the slope of the transfer curve goes to zero. Dropping the higher order terms in (2) and setting the derivative with respect to v_i to zero gives the critical points for clamping as

$$v_{\text{critical}} = \frac{-b \mp \sqrt{b^2 + 3ac}}{3c} . \tag{15}$$

Note that if b is non-zero, the critical points are not symmetric about zero. The clamp at negative inputs is greater in magnitude than the clamp at positive inputs. iip_2 causes asymmetry about zero input because second order distortion is an even function; the second order distortion term is the same sign regardless of the sign of the input.

3.1.2 Noise

Real amplifiers also add noise to the output. Noise figure [7] characterizes the amount of noise the amplifier adds. Noise figure, NF, is defined as

$$NF = 10\log 10(F) \tag{16}$$

where F is the noise factor. Noise factor is the ratio of two ratios: the input signal to noise ratio (SNR), divided by the output SNR. A high SNR means the signal is much stronger than the noise, which is usually desirable. A low SNR means the signal will be harder to detect. Since a receiver always adds noise, the output SNR will always be less than the input SNR and the noise figure will never be smaller than 0 dB.

When modeling noise, if we refer receiver-generated noise to the amplifier input and add it there, we do not have to scale the noise generator with amplifier gain. Referred to the input, the noise consists of an equivalent voltage source and an equivalent current source [7]. However, since the decomposition into current and voltage components is rarely known at the architectural design phase, it is convenient to choose one or the other. Voltage sources are a convenient choice for integrated circuits because receiver RF amplifiers are specified as voltage devices, as opposed to available power devices.[5] With noise voltage sources, instead of noise current sources, an ideal voltage source driving an amplifier does not short out the amplifier's noise source. The price of modeling noise as a voltage source is an extra node in the model.

As before, the immediate challenge is to relate a common RF specification to the appropriate modeling parameter. In this case, since the noise source will be a white noise source, we need to express the noise power spectral density (PSD) in terms of noise factor, *F*. Let the input signal be S_i, the input noise be N_i, and the voltage gain be *G*. Also, let the total output noise be $G^2N_i+N_o$ where N_o is the output noise generated within the amplifier. From the definition of the noise factor,

[5] When RF circuits were built from discrete devices, each stage could be tested separately and RF power is easier to measure than RF voltage. With integrated circuits, interstage measurements are rarely possible, so measurements are simulated. In simulation, ideal voltage measurements are trivial.

$$F = \frac{SNR_{\text{in}}}{SNR_{\text{out}}} = \frac{\dfrac{S_{\text{i}}}{N_{\text{i}}}}{\dfrac{G^2 S_{\text{i}}}{(G^2 N_{\text{i}} + N_{\text{o}})}} = \frac{G^2 N_i + N_{\text{o}}}{G^2 N_{\text{i}}} \tag{17}$$

Solving for the input referred noise PSD, N_{o}/G^2, gives

$$\frac{No}{G^2} = (F - 1)4kTR \tag{18}$$

where N_{i} has been replaced by $4kTR$ and k is Boltzmann's constant, T is the temperature in Kelvin, and R is the source resistance driving the amplifier when the noise figure was measured (usually 50 Ω).

3.1.3 Implementation

Listing 1 shows one possible implementation of a low noise passband amplifier. We call this one a low noise amplifier because iip_3 specifies the nonlinearity instead of the one dB compression point. Listing 1 uses differential terminals but baseband listing in the next section uses single ended terminals. The difference is explained in the next section. For now, think of "*I*" and "*Q*" as "*P*" and "*N*" respectively.

3.2 IQ Modulators

IQ modulators mix baseband signals, $I(t)$ and $Q(t)$, up to RF frequencies. Figure 1 shows a block diagram of a linear *IQ* modulator. Despite the existence of multipliers, we still say the *IQ* modulator shown in Figure 1 is linear because the distortion of the baseband signal is linear. In a typical top-down design flow, a single iip_3 or one dB compression point specifies *IQ* modulator nonlinearity. In that case, nonlinearity can be added by following the *IQ* modulator with a nonlinear amplifier model already discussed. Noise can also be added to the inputs as was done for the amplifier.

Linear distortion introduced by the *IQ* modulator comes from asymmetry between the mixers. In the analog world, distinct devices are never identical. Some asymmetry always exists. Asymmetry between the mixers can introduce gain imbalance and quadrature error. Quadrature error refers the phase difference between the two local oscillator signals, aside from the inherent 90 degree difference between sine and cosine operators. Parameters ε and Δϕ characterize asymmetry between the mixers. ω is the RF carrier frequency in radians per second. *a* is a scaling factor related to the gain of the mixer. θ represents the phase difference between an arbitrary phase reference and the phase of the local oscillator. θ accounts for phase noise and/or frequency offset. θ is more useful in the baseband equivalent model because passband simulations take too long to cover the time scales over which phase noise and frequency offset matter.

According to the block diagram, the RF signal is

$$V_{\text{rf}}(t) = aI_{\text{in}}\left(1 + \frac{\varepsilon}{2}\right)\cos\left(\omega t + \frac{\Delta\phi}{2} + \theta\right) - aQ_{\text{in}}\left(1 - \frac{\varepsilon}{2}\right)\sin\left(\omega t - \frac{\Delta\phi}{2} + \theta\right). \tag{19}$$

As mentioned earlier, half the challenge in building behavioral models of RF circuits is to relate modeling parameters to standard specifications. In this case, we need to relate

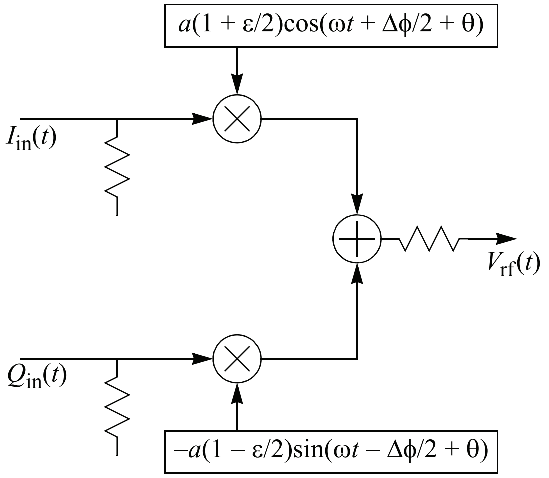
LISTING 1 *Passband amplifier model (a low noise amplifier).*

```
// Differential passband view of a low noise amplifier.
`include "constants.h"
`include "discipline.h"

module LNA_PB(I_in, I_out, Q_in, Q_out);
inout I_in, Q_in, I_out, Q_out;
electrical I_in, Q_in, I_out, Q_out, I_inn;
parameter real gain = 30;                   // Gain in dB.
parameter real ip3 = –20;                   // 3rd order intercept point in dBm.
parameter real rin = 50 from (0:inf);       // Input resistance in Ohms.
parameter real rout = 50 from (0:inf);      // Output resistance in Ohms.
parameter real nf = 10 from [0:inf];        // Noise figure in dB.
real a, c, ip, rho, rhooutmax, rhoinmax, rhoout, noise_voltage_squared, rnf;

analog begin
    // Convert the input parameters from engineering units to implementation units.
    @(initial_step) begin
        a = pow(10,gain/20);
        rnf = pow(10,nf/10);
        c = a∗(4/3)/(pow(10,ip3/10)∗2∗50∗0.001); // 50 Ohm ref. assumed.

        // Compute the critical point.
        rhoinmax = sqrt(a/(3∗c));
        rhooutmax = (2∗a/3)∗rhoinmax;
        noise_voltage_squared = 4∗(rnf–1)∗‘P_K∗$temperature∗50;
    end

    // Assign the input voltage to the variable, rho
    rho = V(I_inn, Q_in);

    // Apply the third order non–linearity. Clamp the output for extreme inputs.
    if (abs(rho) < rhoinmax ) rhoout = (a – c∗rho∗rho)∗rho;
    else if (rho >0) rhoout = rhooutmax;
    else rhoout = –rhooutmax;

    // Load the input with the input resistance, rin.
    I(I_inn, Q_in) <+ V(I_inn, Q_in)/rin;

    // Apply the output signal to the output terminal through the output resistance, rout.
    I(I_out, Q_out) <+ (–rhoout + V(I_out, Q_out))/rout;

    // Apply the noise source. The noise source lies between the input resistance and
    // rho definition to make the noise independent of the source resistance.
    V(I_in, I_inn) <+ white_noise(noise_voltage_squared, "LNA_PB");
end
endmodule
```

the scaling factor, a, to modulator gain G. Confusion arises in defining G because the modulator has two inputs and one output and because the output rides on an RF carrier while the input does not. Do we define gain from one input to the output or from two inputs to the output? Do we use peak or RMS voltage? Does it matter if we use DC or sinusoidal signals to define G?

Suppose we define G as the ratio of peak output volts to peak “input” volts. Let’s explore four ways to define the input:

FIGURE 1 *Passband IQ modulator model.*


$a(1 + \varepsilon/2)\cos(\omega t + \Delta\phi/2 + \theta)$
$I_{\text{in}}(t)$
$V_{\text{rf}}(t)$
$Q_{\text{in}}(t)$
$-a(1 - \varepsilon/2)\sin(\omega t - \Delta\phi/2 + \theta)$

1. Let the *I*-input be *A* volts DC and let the *Q*-input be grounded. The peak input voltage equals *A* volts. Assuming there is no *IQ* mismatch for now, the peak output voltage equals *aA*. In this case, *a* equals *G*.
2. Let *I*-input be a sinusoid with peak voltage *A* while the *Q*-input remains grounded. Since the peak input and output voltages are *A* and *aA*, *a* still equals *G*.
3. Let the *Q*-input also be a sinusoid, 90 degrees out of phase with the *I*-sinusoid. What do we mean by peak input volts when neither input is grounded? Thinking of the input signals as components of a two dimensional vector in polar coordinates, we could define peak volts to be the maximum radius of the input vector. In this case, the magnitude is a constant *A* volts. The RF output is a single RF tone, displaced from the local oscillator frequency by an amount equal to the modulation frequency. The peak output voltage is still *aA* volts, which means once again that *a* equals *G*.
4. Let the input be the same is the last option but define *G* as the ratio of the peak output voltage to the peak voltage on only one input. In this case, *a* still equals *G*. Peak volts are convenient because *a* equals *G* regardless of which option we choose.

Let's revisit the four options in terms of RMS volts instead of peak volts:

1. The *I*-input is *A* volts DC while the *Q*-input is grounded. The RMS input voltage is *A* volts but the output RMS voltage is $aA/\sqrt{2}$ volts. In this case, *a* equals $\sqrt{2}G$.
2. The *I*-input is a sinusoid while the *Q*-input is grounded. The RMS input voltage equals $A/\sqrt{2}$. The output voltage equals *aA*/2. Once again, *a* equals $\sqrt{2}G$.
3. The inputs are sinusoids in quadrature. The RMS value of the input radius equals *A* volts. The RMS output voltage is $aA/\sqrt{2}$ volts. Again, *a* equals $\sqrt{2}G$.
4. The input is the same as in the last option but G is defined as the ratio of RMS output voltage to RMS input voltage on only one input. The RMS voltage on one input is $A/\sqrt{2}$ .The RMS output is still $aA/\sqrt{2}$ volts. Now *a* equals *G*, which is what we get using peak units.

The fourth approach is less prone to errors. Say the output referred compression point of an amplifier stage before the mixer is 10dBm. If we considered both *I* and *Q* signals, the compression point would be 13dBm. The later definition would be foreign to most circuit designers. To refer the 13 dBm compression point to the modulator output, you would use the third RMS definition of modulator gain and pay attention to whether you are using RMS or peak units. To translate the 10dBm number to the modulator output you would use the fourth option regardless of whether the gain was defined in terms of RMS or peak volts.

3.2.1 Implementation

Listing 2 shows one possible implementation of a passband *IQ* modulator. The average gain of the two mixers comprising the modulator is defined in terms of only one input. The module includes a phase error pin through which phase noise and/or frequency offset can be introduced. The gain mismatch is defined as the difference between the *I*-gain and *Q*-gain, divided by the average gain, times 100%. The quadrature error is defined as the total phase difference between the *I* and *Q* local oscillators, not including the 90 degree phase difference that ideally exists between them. This implementation allows for *I* and *Q* signal paths to have separate output referred one dB compression points.

3.3 IQ Demodulators

IQ demodulators, cascaded with low pass filters, extract the *I*(*t*) and *Q*(*t*) baseband signals from a modulated RF carrier. Despite the existence of multipliers, we still say the *IQ* demodulator shown in Figure 2 is linear because the effect on the baseband signal is linear. In a typical top-down design flow, a single iip_3 or one dB compression point specifies *IQ* demodulator nonlinearity. In that case, nonlinearity can be added by preceding the *IQ* modulator with a nonlinear amplifier model already discussed. ε and $\Delta\phi$ specify the same kind of impairments described in the *IQ* modulator section.

The relationship between the demodulator gain, *G,* and *a* can be confusing but there is no need to wade through the options. The various options for defining *IQ* demodulator gain are duals of the *IQ* modulator options. Instead, we jump right to the gain least likely to cause misunderstandings among members of a design team. Let the input RF signal be as shown below in (20):

$$V_{\text{rf}} = A\cos((\omega + \omega_{\text{m}})t) = A\cos(\omega t)\cos(\omega_{\text{m}} t) - A\sin(\omega t)\sin(\omega_{\text{m}} t) \tag{20}$$

ω_{m} is the frequency of a baseband complex tone. Consider the *I*-output, assuming no *IQ* mismatch exists and that θ is zero:

$$I_{\text{out}} = aA\cos^2(\omega t)\cos(\omega_{\text{m}} t) - aA\sin(\omega t)\cos(\omega t)\sin(\omega_{\text{m}} t)\,. \tag{21}$$

Expanding the trigonometric terms gives:

$$I_{\text{out}} = aA\left(\frac{1}{2} + \frac{\cos(2\omega t)}{2}\right)\cos(\omega_{\text{m}}) - aA\frac{\sin(2\omega t)}{2}\sin(\omega_{\text{m}} t)\,. \tag{22}$$

Low pass filtering after the mixer removes the 2ω terms, leaving

$$I_{\text{out}} = \frac{aA}{2}\cos(\omega_{\text{m}} t)\,. \tag{23}$$

From (20), the peak input voltage is *A* and the peak RMS input voltage is $\text{A}/\sqrt{2}$. The relevant peak *I*-output voltage equals *aA*/2 and the RMS value is $aA/(2\sqrt{2})$. G is the same in terms of peak or RMS voltages, making *a* equal 2*G* in either case.

LISTING 2 *Passband IQ modulator model.*

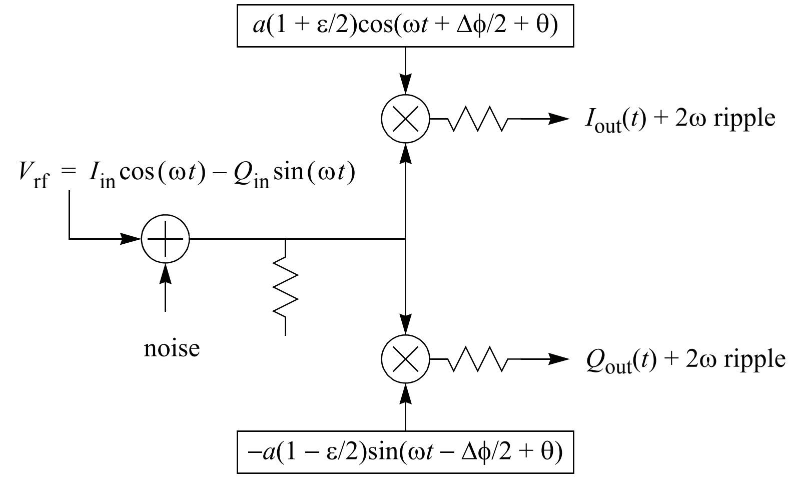
```
//Differential passband view of an IQ modulator.
`include "constants.h"
`include "discipline.h"

module IQ_mod_BB(I_in, I_out, Q_in, Q_out, phase_err);
inout I_in, Q_in, phase_err;
electrical I_in, Q_in, phase_err;
inout I_out, Q_out;
electrical I_out, Q_out;
parameter real nf = 2;           // Noise figure in dB.
parameter real Q_cpdb = 100;     // Output ref. Q-path one dB compression pt. in dBm.
parameter real I_cpdb = 100;     // Output ref I-path one dB compression pt. in dBm.
parameter real rout = 50;        // Output resistance in Ohms.
parameter real rin = 50;         // Input resistance in Ohms.
parameter real gain = 100;       // Average gain in dB.
parameter real flo = 1e9;        // Local oscillator frequency in Hz.
parameter real epsilon = 0;      // Gain mismatch in percent.
parameter real phi = 0;          // Quadrature error in degrees.
real quad_error, I_a, Q_a, I_c, Q_c, I_cmp, Q_cmp,I _rho, Q_rho, I_rhooutmax;
real Q_rhooutmax, I_rhoinmax, Q_rhoinmax, I_rhoout, Q_rhoout, I_theta;
real Q_theta, tmp, I_cp, Q_cp, noise_current_sqrd, rnf;

analog begin
    // Convert input parameters from engineering units to implementation units.
    @(initial_step) begin
        quad_error = `M_PI*phi/180;
        I_a = pow(10,gain/20)*(1+epsilon/200);
        I_cmp = sqrt(pow(10,(I_cpdb+1)/10)*2*rout*0.001)/I_a;
        I_c = I_a*(0.145)/(I_cmp*I_cmp);
        I_rhoinmax = sqrt(I_a/(3*I_c));
        I_rhooutmax = (2*I_a/3)*I_rhoinmax;
        Q_a = pow(10,gain/20)*(1–epsilon/200);
        Q_cmp = sqrt(pow(10,(Q_cpdb+1)/10)*2*rout*0.001)/Q_a;
        Q_c = Q_a*(0.145)/(Q_cmp*Q_cmp);
        Q_rhoinmax = sqrt(Q_a/(3*Q_c));
        Q_rhooutmax = (2*Q_a/3)*Q_rhoinmax;
        rnf = pow(10,nf/10);
        noise_current_sqrd = 4*(rnf–1)*`P_K*$temperature/50;
    end
    I_rho = V(I_in);
    Q_rho = V(Q_in);

    // Apply the third order non–linearity. Clamp the output for extreme inputs.
    if ( abs(I_rho) < I_rhoinmax ) I_rhoout = (I_a – I_c*I_rho*I_rho)*I_rho;
    else if (I_rho < 0) I_rhoout = –I_rhooutmax;
    else I_rhoout = I_rhooutmax;
    if ( abs(Q_rho) < Q_rhoinmax ) Q_rhoout = (Q_a – Q_c*Q_rho*Q_rho)*Q_rho;
    else if (Q_rho < 0) Q_rhoout = –Q_rhooutmax;
    else Q_rhoout = Q_rhooutmax;
```

Continued on the next page.

LISTING 2 *Passband IQ modulator model.*

```
    // Calculate I and Q phase
    bound_step(0.05/flo);
    tmp = 2∗`M_PI∗flo∗($realtime);
    I_theta = tmp+V(phase_err)+quad_error/2;
    Q_theta = tmp+V(phase_err)–quad_error/2;
    I(phase_err) <+ V(phase_err)/50;

    I(I_in) <+ V(I_in)/rin;
    I(Q_in) <+ V(Q_in)/rin;

    // Calculate output
    I(I_out, Q_out) <+ (–(I_rhoout∗cos(I_theta)–Q_rhoout∗sin(Q_theta));
    I(I_out, Q_out) <+ V(I_out,Q_out))/rout;
    I(I_in) <+ white_noise(noise_current_sqrd, "IQ_modulator_i");
    I(Q_in) <+ white_noise(noise_current_sqrd, "IQ_modulator_q");
end
endmodule
```

FIGURE 2 *Passband IQ demodulator model.*


$a(1 + \varepsilon/2)\cos(\omega t + \Delta\phi/2 + \theta)$
$I_{\text{out}}(t) + 2\omega$ ripple
$V_{\text{rf}} = I_{\text{in}}\cos(\omega t) - Q_{\text{in}}\sin(\omega t)$
noise
$Q_{\text{out}}(t) + 2\omega$ ripple
$-a(1 - \varepsilon/2)\sin(\omega t - \Delta\phi/2 + \theta)$

Noise is added to the *IQ* demodulator input in exactly the same manner as for the amplifier. In particular, noise figure (factor) specifies the noise and it is injected at the input. Should we worry about how to specify and model noise when there are two outputs and only one input? Let's consider the two options. First, define noise factor to be the SNR at the *I*-output divided by the input SNR. Denote the total noise at the *I*-output to be N_{ot}, which equals $N_{\text{i}}G^2 + N_{\text{o}}$ in (17). Substituting into (17) gives

$$F = \frac{N_{\text{ot}}}{G^2 N_{\text{i}}}. \qquad (24)$$

Now define output SNR in terms of both inputs. G increases by $\sqrt{2}$ to account for the additional output and so does N_{ot}. The output SNR does not depend on whether we use one or two outputs. Therefore, noise factor does not depend on the number of outputs. Does it matter that some of the noise at the two outputs comes from common circuitry upstream of the mixers? Common circuitry might introduce some correlation between the noise observed at the *I*- and *Q*-outputs. However, the definition of SNR does not distinguish between correlated and uncorrelated *I*- and *Q*-noise. Furthermore, near the top of a top-down design flow, correlation between *I*- and *Q*-noise does not affect any decisions.

3.3.1 Implementation

Listing 3 shows one possible implementation of a passband *IQ* modulator. The average gain of the two mixers comprising the modulator is defined in terms of only one output. The module includes a phase error pin through which phase noise and/or frequency offset can be introduced. The gain mismatch is defined as the difference between the *I*-gain and *Q*-gain, divided by the average gain, times 100%. The quadrature error is defined as the total phase difference between the *I* and *Q* local oscillators, not including the 90 degree phase difference that ideally exists between them. This implementation allows for *I* and *Q* signal paths to have separate input referred third order intercept points.

4 Baseband Equivalent models

Baseband models run thousands of times faster than passband models because they suppress the RF/IF carrier. Baseband modeling does for RF transceivers what state space averaged models do for switch-mode power supplies, what phase domain models do for PLLs, and what rotating reference frames do for rotating machines. All of these modeling methods dramatically reduce run times by eliminating high frequency signals. Another feature these methods share is that they all trade accuracy for speed. It is important to know the implicit assumptions you make when using baseband equivalent models.

The basic assumption behind baseband equivalent models is that you only care about distortion of the information riding on the carrier; you do not care about harmonic distortion of the carrier. One immediate but not-so-obvious consequence of baseband modeling is that real voltages and currents in a passband model become complex numbers in a baseband equivalent model. Recall the expression for a general modulated carrier, (1). (1) describes a scalar. However, the information has two scalar components, *I*(*t*) and *Q*(*t*), defined as the in-phase and quadrature components respectively. We define the complex baseband equivalent signal, *B*(*t*), as

$$B(t) = I(t) + jQ(t) \tag{25}$$

where $j = \sqrt{-1}$. *B(t)* is nothing more than a slowly varying phasor [9].

It is sometimes more convenient to work with the analytic version of the modulated carrier. The modulated carrier is the real part of an analytic signal, *Z(t)*, defined as

$$Z(t) = B(t)e^{j\omega t} \tag{26}$$

such that

$$V_{\mathrm{rf}}(t) = \mathcal{R}\{Z(t)\} \tag{27}$$

where ω is the carrier frequency in radians per second. Although passband analysis only deals with the real part of *Z*(t), we will later need to use the imaginary part as well. For

LISTING 3 *Passband IQ demodulator.*

```
//Differential passband view a passband IQ demodulator.
`include "constants.h"
`include "discipline.h"
module IQ_demod_BB(I_in, I_out, Q_in, Q_out, phase_err);
inout I_in, Q_in, I_out, Q_out, phase_err;
electrical I_in, I_out, Q_in, Q_out, phase_err;
electrical I_inn;

parameter real gain = 0; // Gain from input to one output, in dB.
parameter real epsilon = 0; // Gain mismatch in percent.
parameter real phi = 0; // Quadrature error in degrees.
parameter real I_ip3 = 100; // Input referred 3rd order intercept for I-output.
parameter real Q_ip3 = 100; // Input referred 3rd order intercept for Q-output.
parameter real rin = 50 from (0:inf); // Input resistance in Ohms.
parameter real rout = 50 from (0:inf); // Output resistance in Ohms.
parameter real nf = 2 from [0:inf]; // Noise figure in dB.
parameter real flo = 1G; // Local oscillator frequency in Hz.
real quad_error, I_a, Q_a, I_b, Q_b, I_ip, Q_ip, rho, I_rhooutmax;
real Q_rhooutmax, I_rhoinmax, Q_rhoinmax, I_rhoout, Q_rhoout;
real theta, I_theta, Q_theta, tmp, noise_voltage_sqrd, rnf;

analog begin
    // Convert the input parameters from engineering units to implementation units.
    @(initial_step) begin
        quad_error = `M_PI∗phi/180;
        I_a = pow(10,gain/20)∗(1+epsilon/200);
        I_ip = sqrt(pow(10,I_ip3/10)∗2∗50∗0.001); // 50 Ohm reference.
        I_b = I_a/(I_ip∗I_ip)∗4.0/3.0;
        I_rhoinmax = sqrt(I_a/(3.0∗I_b));
        I_rhooutmax = (2.0∗I_a/3.0)∗I_rhoinmax;
        Q_a = pow(10,gain/20)∗(1–epsilon/200);
        Q_ip = sqrt(pow(10,Q_ip3/10)∗2∗50∗0.001); // 50 Ohm reference.
        Q_b = Q_a/(Q_ip∗Q_ip)∗4.0/3.0;
        Q_rhoinmax = sqrt(Q_a/(3.0∗Q_b));
        Q_rhooutmax = (2∗Q_a/3)∗Q_rhoinmax;
        Q_theta = 0;
        rnf = pow(10,nf/10.0);
        noise_voltage_sqrd = 4.0∗(rnf–1)∗`P_K∗$temperature∗50;
    end

    rho = V(I_inn, Q_in);

    // Apply the third order nonlinearity. Clamp the output for extreme inputs.
    if ( abs(rho) < I_rhoinmax ) I_rhoout = (I_a – I_b∗rho∗rho)∗rho;
    else if (rho < 0) I_rhoout = –I_rhooutmax;
    else I_rhoout = I_rhooutmax;
    if ( abs(rho) < Q_rhoinmax ) Q_rhoout = (Q_a – Q_b∗rho∗rho)∗rho;
    else if (rho < 0) Q_rhoout = –Q_rhooutmax;
    else Q_rhoout = Q_rhooutmax;

    bound_step(0.05/flo);
```

Continued on the next page.

LISTING 3 *Passband IQ demodulator.*

Continued from previous page.

```
    // Adjust the angles.
    I_theta = 2*`M_PI*flo*($abstime)+V(phase_err);
    I(phase_err) <+ V(phase_err)/50.0;
    I(I_inn, Q_in) <+ V(I_inn, Q_in)/rin;
    I(I_out) <+ (–2*I_rhoout*cos(I_theta+quad_error/2) + V(I_out))/rout;
    I(Q_out) <+ (2*Q_rhoout*sin(I_theta–quad_error/2) + V(Q_out))/rout;

    V(I_in, I_inn) <+ white_noise(noise_voltage_sqrd, "IQ_demodulator");
end
endmodule
```

those interested in digging into the literature on baseband signal processing, the imaginary part of *Z*(t) is the Hilbert transform [2,10] of the modulated carrier.

Since many nonlinear effects depend only on the amplitude of the modulated carrier, the polar representation of the modulated carrier is also useful. To convert to polar coordinates we let

$$\begin{aligned} rf(t) &= A(t)\cos(\omega t + \theta(t)) \\ &= A(t)\cos(\omega t)\cos(\theta(t)) - A(t)\sin(\omega t)\sin(\theta(t)) \,. \end{aligned} \tag{28}$$

Equating the in-phase and quadrature signals to those in [1], gives:

$$I(t) = A(t)\cos(\theta(t)) \tag{29}$$

and

$$Q(t) = A(t)\sin(\theta(t)) \,, \tag{30}$$

which can be rearranged to express A and θ in terms of *I* and *Q*:

$$A(t) = \sqrt{I(t)^2 + Q(t)^2} \tag{31}$$

and

$$\theta(t) = \operatorname{atan}\left(\frac{Q(t)}{I(t)}\right) . \tag{32}$$

Signal Power. When using baseband models, one must take care to use the correct signal power [11]. Let the baseband signal be

$$B(t) = \rho e^{j\omega_{\mathrm{m}} t} . \tag{33}$$

The associated real passband RF signal is

$$rf(t) = R\{\rho e^{j\omega_{\mathrm{m}} t} e^{j\omega t}\} = \rho\cos((\omega_{\mathrm{m}} + \omega)t) . \tag{34}$$

Since the time average of a sinusoid is 1/2, the average power in the RF signal is

$$P_{\mathrm{PB}} = \frac{\rho^2}{2} \text{ Watts,} \tag{35}$$

if we assume for notational simplicity that reference resistance is one Ohm. The total average power in the baseband signal is

$$P_{\mathrm{BB}} = \rho^2 \text{ Watts.} \tag{36}$$

Baseband and passband powers differ by a factor of two. The baseband model simulates the information in units of peak carrier volts. Power, in terms of peak sinusoidal volts, requires an additional factor of two. Some textbooks define the carriers as $\sqrt{2}\cos(\omega t)$ and $\sqrt{2}\sin(\omega t)$ instead of $\cos(\omega t)$ and $\sin(\omega t)$ so that baseband and passband powers can be equated.

4.1 Amplifiers

4.1.1 Nonlinear behavior

The baseband equivalent model of the amplifier is essentially a describing function found in control theory literature [12]. Describing functions are nonlinear transfer functions derived by driving the nonlinear device with a sinusoid and computing the fundamental Fourier component of the output. A describing function is the fundamental Fourier component of the output, normalized with respect to input amplitude. In general, a describing function is a complex nonlinear function of input amplitude. In control theory, describing functions help adapt classical frequency domain stability analysis to feedback systems that include a single static nonlinear block. We will use describing function here to suppress the carrier.

In our application, it is not necessary to use normalized describing functions but as we shall see later, normalization has some advantages. We start with the polar form of an input modulated carrier: $v_{\mathrm{i}}(t) = A(t)\cos(\omega t + \theta(t))$. Substituting into (2), ignoring the DC term and ignoring the higher order terms, gives

$$v_{\mathrm{o}} = aA(t)\cos(\omega t + \theta(t)) - bA^2(t)\cos^2(\omega t + \theta(t)) - cA^3(t)\cos^3(\omega t + \theta(t)) \tag{37}$$

Now expand the cosine terms:

$$\cos^2(\omega t + \theta(t)) = \left(\frac{1}{2}\right)(1 + \cos(2\omega t + 2\theta(t))) \tag{38}$$

$$\cos^3(\omega t + \theta(t)) = \frac{3}{4}\cos(\omega t + \theta(t)) + \frac{1}{4}\cos(3\omega t + 3\theta(t)). \tag{39}$$

The basic assumption lets us ignore the second order terms because they produce nothing at the carrier frequency, ω. Second order nonlinearities must combine with another impairment to be of concern [7]. Second order harmonics of the carrier are assumed to be filtered out before they reach the next nonlinear block. It is possible to model the composite impairment of two cascaded nonlinear blocks but that is beyond the scope of this discussion. Thus, the only contribution from the nonlinear terms we care about is $0.75\cos(\omega t + \theta(t))$.

Applying the basic assumption to (37), the passband output becomes:

$$v_{\mathrm{o}} = \left(aA(t) - \frac{3}{4}cA^3(t)\right)\cos(\omega t + \theta(t)). \tag{40}$$

Expanding (40) back into Cartesian coordinates explicitly shows the in-phase and quadrature components,

$$v_o = \left(aA(t) - \frac{3}{4}cA^3(t)\right)\cos(\theta(t))\cos(\omega t) - \left(aA(t) - \frac{3}{4}cA^3(t)\right)\sin(\theta(t))\sin(\omega t) \quad (41)$$

The baseband equivalent signal of the amplifier output is

$$B_{out}(A, \theta) = \left(aA(t) - \frac{3}{4}cA^3(t)\right)\cos(\theta(t)) + j\left(aA(t) - \frac{3}{4}cA^3(t)\right)\sin(\theta(t)). \quad (42)$$

(42) can also be written in polar form as

$$B_{out}(A, \theta) = R(A)e^{j\theta(t)}, \quad (43)$$

where

$$R(A) = \left(aA(t) - \frac{3}{4}cA^3(t)\right) \quad (44)$$

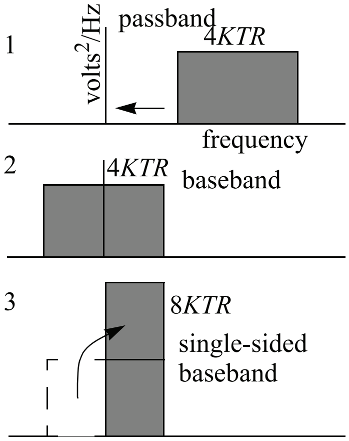
In terms of *R(A)*, the analytic output signal is

$$Z_{out}(t) = R(A(t))e^{j\theta(t)}e^{j\omega t}. \quad (45)$$

(We could have jumped directly from (40) to (45) but going through the intermediate steps paves the way for modeling more complicated devices.)

4.1.2 Noise

Two factors of two complicate baseband modeling of passband noise. The first factor of two is associated with bandwidth. Most simulators work with single-sided power spectral densities. The single sided noise PSD at the amplifier input is $N_o = 4kTR$. Noise exists on either side of the RF carrier with a single-sided density of N_o. When we suppress the carrier, the noise PSD slides down to baseband where it straddles zero Hz. To convert to a single sided density, you must fold the negative frequency noise over to positive frequencies, thereby doubling the single-sided density to $2N_o$.



The second factor of two results from the noise being wide-sense stationary. Wide-sense stationary means the statistics describing the noise do not depend on time. The derivation below, which follows one in [13], shows why we must add the full noise to *both I* and *Q* inputs, apparently doubling the noise power yet again. Invoking the basic assumption of baseband equivalent modeling, we only care about noise in a small band straddling the carrier. Any narrow band signal centered about the carrier can be expressed as

$$w(t) = x(t)\cos(\omega t) - y(t)\sin(\omega t). \quad (46)$$

Let *w*(*t*) be the passband noise. Then *x*(*t*) + *jy*(*t*) is its baseband equivalent. If the variance of *x* and *y* each equals the variance of *w*, then we must add the full noise power to both *I* and *Q* inputs. We obtain the variance from the autocorrelation. The autocorrelation of *w*(*t*) is the expectation of

$$w(t+\tau)w(t) = [x(t+\tau)\cos(\omega(t+\tau)) - y(t+\tau)\sin(\omega(t+\tau))] \times \\ [x(t)\cos(\omega t) - y(t)\sin(\omega t)]\,. \tag{47}$$

Let *E* denote the expectation operator. The relevant autocorrelations are:

$$R_{\mathrm{xx}}(\tau) = E[x(t+\tau)x(t)] \tag{48}$$

$$R_{\mathrm{yy}}(\tau) = E[y(t+\tau)y(t)] \tag{49}$$

$$R_{\mathrm{xy}}(\tau) = E[x(t+\tau)y(t)] \tag{50}$$

$$R_{\mathrm{yx}}(\tau) = E[y(t+\tau)y(t)] \tag{51}$$

$$R_{\mathrm{ww}}(\tau) = E[w(t+\tau)w(t)]\,. \tag{52}$$

Multiplying, applying some trigonometric identities, and taking the expectation of both sides of (47) gives:

$$2R_{\mathrm{ww}}(\tau) = [R_{\mathrm{xx}}(\tau) + R_{\mathrm{yy}}(\tau)]\cos(\omega\tau) - [R_{\mathrm{xy}}(\tau) + R_{\mathrm{yx}}(\tau)]\sin(\omega\tau) + \\ [R_{\mathrm{xx}}(\tau) - R_{\mathrm{yy}}(\tau)]\cos(\omega(2t+\tau)) - [R_{\mathrm{xy}}(\tau) + R_{\mathrm{yx}}(\tau)]\sin(\omega(2t+\tau))\,. \tag{53}$$

Note that the last two terms depend on time. To comply with our wide-sense stationary assumption the coefficients of the time-dependent terms must be zero. Thus,

$$R_{\mathrm{xx}}(\tau) = R_{\mathrm{yy}}(\tau) \tag{54}$$

$$R_{\mathrm{xy}}(\tau) = -R_{\mathrm{yx}}(\tau)\,. \tag{55}$$

Consequently,

$$R_{\mathrm{ww}}(\tau) = R_{\mathrm{xx}}(\tau)\cos(\omega\tau) + R_{\mathrm{yx}}\sin(\omega\tau)\,. \tag{56}$$

Since variance is the autocorrelation evaluated at a τ of zero, the variance of *w* equals the variance of *x* (and also *y,* by (54)). The full noise power must therefore be added to both inputs of the baseband model.

That last factor of two is perhaps not very intuitive but it is consistent with the factor of two between passband and equivalent baseband signal power levels. Since both the signal and noise in the baseband equivalent model have factors of two, the signal-to-noise-ratios are the same in baseband and passband models.

Despite physically common noise sources, the *I*- and *Q*-noise sources are assumed to be uncorrelated. The degree of correlation depends on how white the noise is over the bandwidth of interest but is not worth modeling. Correlation does not drive any architectural decisions.

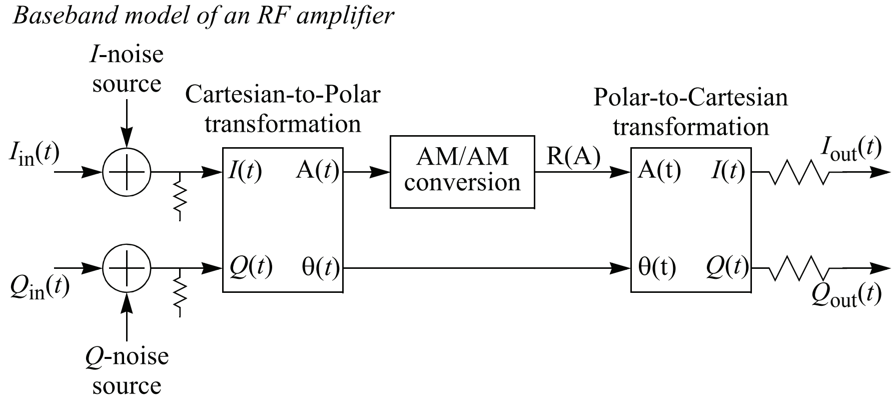
4.1.3 Terminal Resistances

Terminal resistances are easily added to both models. In the passband model, just add a resistor from the input terminal to ground and a series resistance between the unloaded output voltage and output node. The same is true for the baseband model but the *I*- and *Q*-terminals must have their own resistors. Why? Start with (1). The passband current is just the passband voltage divided by the resistance. In deriving the baseband model, the resistance carries over directly to the complex baseband signal because it is a simple static linear scaling factor. Thus, in-phase and quadrature components of the baseband current are just the in-phase and quadrature components of the baseband voltage divided by the resistance. As we shall see later, baseband equivalent models of passband reactive terminal impedances are more involved.

4.1.4 Implementation

Figure 1 shows a block diagram of the baseband amplifier model. The associated listing that follows uses single ended terminals. As mentioned earlier, passband voltages split into two baseband components, each requiring its own node. By using differential passband nodes and single-ended baseband nodes, the modules can be swapped without changing the number of nodes.[6]

FIGURE 3 *Low noise amplifier block diagram.*



The implementation in Figure 3 works well for transient analysis but requires help with DC operating point analysis. The nonlinearity depends on signal amplitude. Computing the signal amplitude from the baseband signal requires computing the square root of (x^2+y^2). Most simulators choke on the pure square root operation at the origin because the gradient of square root of (x^2+y^2) is not defined at the origin. Some simulators get around the problem by adding a small positive number to the argument. With such a trick, the gradient at the origin is well behaved but inaccurate. The error does not affect transient analyses. The transient analysis converges to the correct point as long as the gradient points in the right direction. However, AC analyses will use the wrong incremental gain for the square root block because the operating point analysis will be incorrect. One work around for AC analysis is to bias the input away from the origin. You can control the amount of bias needed by introducing a scaling factor, K. Replace square root of (x^2+y^2) with square root of $(K^2(x^2+y^2))/K$. The larger you make K, the less bias you have to use. This can be important when modeling the low noise amplifier in a receiver because the input signals can be very small. Fortunately, there is a more elegant solution to the singularity at the origin.

[6] The idea of using differential passband nodes and single-ended baseband nodes was suggested by Fredrik Ivarsson of Cadence Design Systems.

A Cartesian version of the baseband model has no singularity at the origin. We can convert the model to Cartesian coordinates by returning to the true describing function, one normalized with respect to the complex input. In (42), factor out $A(t)$:

$$B_{\text{out}}(A, \theta) = \left(a - \frac{3}{4}cA^2\right)A\cos\theta + j\left(a - \frac{3}{4}cA^2\right)A\sin\theta \tag{57}$$

From (29) and (30), we can write (57) as

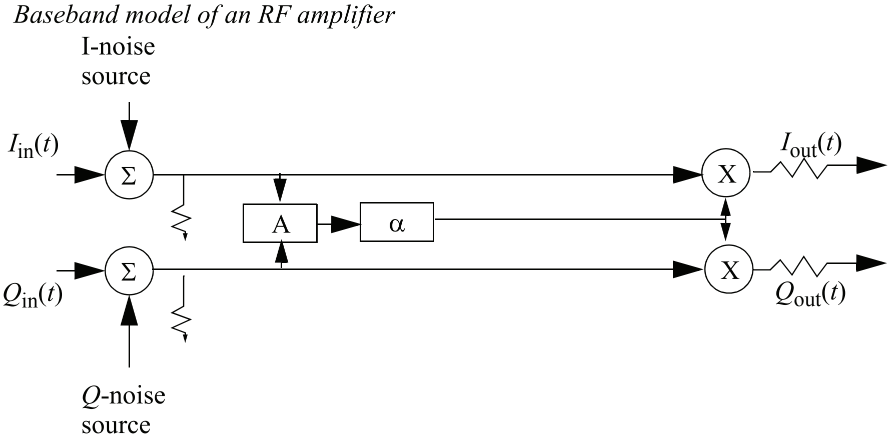
$$B_{\text{out}}(A, \theta) = \alpha(A)I(t) + j\alpha(A)Q(t) \tag{58}$$

where

$$\alpha(A) = a - \frac{3}{4}cA^2 . \tag{59}$$

$\alpha(A)$ is a describing function. A is still not quite correct near the origin but since it goes to zero near the origin, α is correct. Figure 4 shows a block diagram of the Cartesian implementation of the baseband equivalent model of an amplifier.

FIGURE 4 *Cartesian version of the low noise amplifier.*



As with the passband model, it is a good idea to clamp the output past the critical points. However, to clamp the output of the baseband model for large inputs, the describing function must drop off as the inverse of the amplitude of the input. It is more convenient to switch back to polar coordinates to clamp the output. Since the critical point is usually far form the origin, the inaccuracy of the square root operator is not a problem there. Listing 4 contains the code for the model.

The amplifier model discussed so far is sufficient for most low noise amplifiers in a receiver but lacks an important impairment that afflicts some transmitter power amplifiers. Nonlinear capacitors can introduce a phase shift in the carrier that depends on amplitude [2,7,14,15,16]. The amplitude dependent phase shift is commonly referred to as AM/PM conversion and is fairly easy to include in a baseband equivalent model. A describing function is fully capable of modeling AM/PM conversion. However, unlike AM/AM conversion, there is no standard metric for AM/PM conversion. AM/PM conversion is not as well understood as AM/AM conversion but it is easily modeled from

LISTING 4 *Baseband low noise amplifier.*

```
// Baseband equivalent model of a low noise amplifier.
`define K 1000
`include "constants.h"
`include "discipline.h"

module LNA_BB(I_in, I_out, Q_in, Q_out);
inout I_in, Q_in, I_out, Q_out;
electrical I_in, Q_in, I_out, Q_out, I_inn, Q_inn;
parameter real gain = 30 from [0:inf);      // voltage gain (dB)
parameter real ip3 = −20;                   // input referenced IP3 (dBm)
parameter real rin = 50 from (0:inf);       // noise figure (dB)
parameter real rout = 50 from (0:inf);      // input resistance (Ohms)
parameter real nf = 10 from [0:inf];        // output resistance (Ohms)
real a, cbb, alpha, rho, rhooutmax, rhoinmax, noise, rnf, iout, qout, theta;

analog begin
    // Convert the input parameters from engineering units to implementation units.
    @(initial_step) begin
        a = pow(10,gain/20);
        cbb = a/(pow(10,ip3/10.0)∗2∗50∗0.001);
        rnf = pow(10,nf/10.0);
        //Compute input and output critical points.
        rhoinmax = sqrt(a/(3.0∗cbb));
        rhooutmax = (2∗a/3)∗rhoinmax;
        noise = 8.0∗(rnf−1)∗‘P_K∗$temperature∗50;
        // Replace (rnf-1) by rnf to include noise from the source of 4kTR, R=50.
    end

    // Compute the input angle and radius.
    if (V(I_inn) !=0) theta = atan2(V(Q_inn),V(I_inn));
    else if (V(Q_inn) > 0) theta = `M_PI/2;
    else theta = –`M_PI/2;
    rho = hypot(`K∗V(I_inn),`K∗V(Q_inn))/`K;

    // Apply the third order non-linearity. Clamp the output for extreme inputs.
    if (rho < rhoinmax ) begin
        alpha = (a – cbb∗rho∗rho);
        iout = V(I_inn)∗alpha;
        qout = V(Q_inn)∗alpha;
    end else begin
        iout = rhooutmax∗cos(theta);
        qout = rhooutmax∗sin(theta);
    end

    I(I_inn) <+ V(I_inn)/rin;
    I(Q_inn) <+ V(Q_inn)/rin;
    I(I_out) <+ (-iout + V(I_out))/rout;
    I(Q_out) <+ (-qout + V(Q_out))/rout;
    V(I_in, I_inn) <+ white_noise(noise, "LNA_BB_i");
    V(Q_in, Q_inn) <+ white_noise(noise, "LNA_BB_q");
end
endmodule
```

empirical data. Let $\psi(A)$ be a well-behaved function that fits the data. With AM/PM conversion, (43) becomes

$$B_{\text{out}}(A, \theta) = Ae^{j\theta}\alpha(A)e^{j\psi(A)} \tag{60}$$

and the new describing function, denoted $\gamma(A)$, becomes

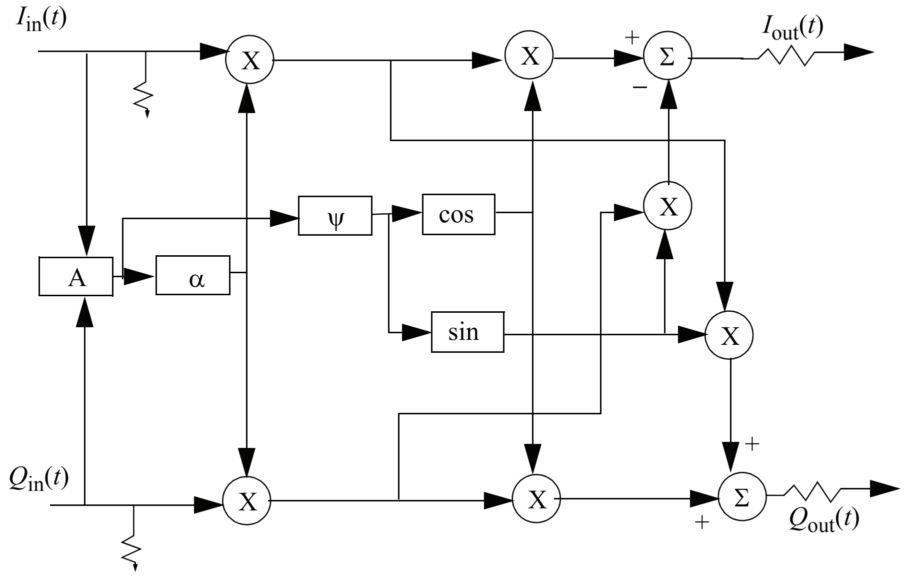
$$\gamma(A) = \alpha(A)e^{j\psi(A)}. \tag{61}$$

The Cartesian form of (60) is better for implementation because it circumvents the gain problem at the origin:

$$B_{\text{out}} = \alpha(A)(I\cos(\psi(A)) - Q\sin(\psi(A)) + j(I\sin(\psi(A)) + Q\cos(\psi(A)))). \tag{62}$$

Figure 5 shows a block diagram of the power amplifier model. The power amplifier does not include noise sources because the diagram is already fairly cluttered and because the input signal is usually much stronger than any noise in the transmitter. Listing 5 lists an implementation of a power amplifier model. The $\psi(A)$ function is somewhat arbitrary and should be replaced by whatever function best fits the data at hand. The power amplifier model is specified by gain and one dB compression point. The input and output impedances are 50 Ω. The gain is defined as power gain, not voltage gain, which means the amplifier must be loaded with 50 Ω to observe the specified gain.

FIGURE 5 *Power amplifier block diagram.*



4.2 IQ Modulator

The baseband equivalent model of the *IQ* modulator is derived by expanding the trigonometric terms in (19). More specifically, LISTING 5 *Baseband power amplifier model.*

```
//Baseband equivalent model of a power amplifier.

`include "constants.h"
`include "discipline.h"
`define K 100000
module PA_BB(I_in, I_out, Q_in, Q_out);
inout I_in, I_out, Q_in, Q_out;
electrical I_in, I_out, Q_in, Q_out;
parameter real gain = 40; // Gain in dB.
parameter real cpdb = 0; // Output referred one dB compression point in dB.
parameter real rin = 50 from (0:inf); // Input resistance in Ohms.
parameter real rout = 50 from (0:inf); // Output resistance in Ohms.
// AM/PM parameters:
parameter real pscp = 0.7 from (0:inf); // Radians of phase shift at cpdb.
parameter real psinf = 2 from (pscp:inf); // Radians of phase shift at large inputs.
parameter real shp = 2 from [0:inf); // Sharpness parameter, dimensionless.
parameter integer cw = 0 from [–1:1]; // AM/PM shifts clocwise or counterccw.
real a, cbb, alpha, cmp, rho, rhoinmax, rhooutmax, rhoout, theta, phi;
real iout, qout, beta, tmp, cosphi, sinphi;
analog begin
    // The "initial" block converts the input parameters from engineering
    // units to implementation units.
    @(initial_step) begin
        a = sqrt(pow(10,gain/10.0)∗rout/rin);
        cmp = sqrt(pow(10,(cpdb+1)/10)∗2∗rout∗0.001)/a;
        cbb = a∗(0.108749)/(cmp∗cmp);
        rhoinmax = sqrt(a/(3.0∗cbb));
        rhooutmax = (2∗a/3)∗rhoinmax;
        theta = 0;
        beta = pscp/(psinf-pscp);
    end

    // Compute the input angle and radius.
    if (V(I_in) !=0) theta = atan2(V(Q_in),V(I_in));
    else if (V(Q_in) > 0) theta = `M_PI/2;
    else theta = -`M_PI/2;
    rho = hypot(`K∗V(I_in),`K∗V(Q_in))/`K;

    // Compute the am/pm conversion.
    tmp = pow(rho/cmp,shp);
    phi = cw∗(1+beta)∗pscp∗tmp/(1+beta∗tmp);

    // Compute sine and cosine once and store the result in cosphi, sinphi.
    cosphi = cos(phi);
    sinphi = sin(phi);
```

Continued on the next page.

LISTING 5 *Baseband power amplifier model.*

```
    // Use Cartesian coordinates to apply the nonlinearity if input does not
    // exceed the critical point.
    if (rho < rhoinmax ) begin
        alpha = (a - cbb*rho*rho);
        iout = alpha*(V(I_in)*cosphi - V(Q_in)*sinphi);
        qout = alpha*(V(I_in)*sinphi + V(Q_in)*cosphi);
    end
    // Otherwise clamp use polar coordinates to clamp the output.
    else begin
        iout = rhooutmax*cos(theta + phi);
        qout = rhooutmax*sin(theta + phi);

    end
    // Assign the results and load the input terminals.
    I(I_in) <+ V(I_in)/rin;
    I(Q_in) <+ V(Q_in)/rin;
    I(I_out) <+ (–2*iout + V(I_out))/rout;
    I(Q_out) <+ (–2*qout + V(Q_out))/rout;
end
endmodule
```

$$\cos\left(\omega t + \frac{\Delta\phi}{2} + \theta\right) = \cos(\omega t)\cos\left(\frac{\Delta\phi}{2} + \theta\right) - \sin(\omega t)\sin\left(\frac{\Delta\phi}{2} + \theta\right) \tag{63}$$

and

$$\sin\left(\omega t - \frac{\Delta\phi}{2} + \theta\right) = \sin(\omega t)\cos\left(-\frac{\Delta\phi}{2} + \theta\right) + \cos(\omega t)\sin\left(-\frac{\Delta\phi}{2} + \theta\right). \tag{64}$$

To find the components of the equivalent baseband output signal, $I_{\text{out}} + jQ_{\text{out}}$, write the RF output as

$$V_{\text{rf}} = I_{\text{out}}\cos(\omega t) - Q_{\text{out}}\sin(\omega t). \tag{65}$$

Substituting (65), (63) and (64) into (19) gives

$$\begin{aligned} I_{\text{out}}\cos(\omega t) - Q_{\text{out}}\sin(\omega t) = \\ & aI_{\text{in}}\left(1 + \frac{\varepsilon}{2}\right)\left(\cos(\omega t)\cos\left(\frac{\Delta\phi}{2} + \theta\right) - \sin(\omega t)\sin\left(\frac{\Delta\phi}{2} + \theta\right)\right) - \\ & aQ_{\text{in}}\left(1 - \frac{\varepsilon}{2}\right)\left(\sin(\omega t)\cos\left(-\frac{\Delta\phi}{2} + \theta\right) + \cos(\omega t)\sin\left(-\frac{\Delta\phi}{2} + \theta\right)\right) \end{aligned} \tag{66}$$

Equating coefficients of cos(ω*t*) gives I_{out}:

$$I_{\text{out}} = a\left(1 + \frac{\varepsilon}{2}\right)\cos\left(\frac{\Delta\phi}{2} + \theta\right)I_{\text{in}} - a\left(1 - \frac{\varepsilon}{2}\right)\sin\left(-\frac{\Delta\phi}{2} + \theta\right)Q_{\text{in}}. \tag{67}$$

Equating coefficients of -sin(wt) gives Q_{out}:

$$Q_{\text{out}} = a\left(1 + \frac{\varepsilon}{2}\right)\sin\left(\frac{\Delta\phi}{2} + \theta\right)I_{\text{in}} + a\left(1 - \frac{\varepsilon}{2}\right)\cos\left(-\frac{\Delta\phi}{2} + \theta\right)Q_{\text{in}}. \tag{68}$$

Assuming the gain, *G*, defined in accordance with option four in the passband section, *a* equals *G*. With expressions for the output baseband signal in terms of the input baseband signal, the baseband equivalent model is complete.

4.2.1 Implementation

Listing 6 shows one possible implementation of the *IQ* modulator.

LISTING 6 *Baseband IQ modulator model.*

```
// Baseband equivalent model of an IQ modulator.
`include "constants.h"
`include "discipline.h"

module IQ_mod_BB(I_in , I_out, Q_in, Q_out, phase_err);
inout I_in, I_out, Q_in, Q_out, phase_err ;
electrical I_in, I_out, Q_in, Q_out, phase_err ;
parameter real gain = 0; // Average gain in dB.
parameter real epsilon = 0; // Gain mismatch in dB.
parameter real phi = 0; // Quadrature error in degrees.
parameter real I_cpdb = 100; // Output ref. I-path one dB compression pt.
parameter real Q_cpdb = 100; // Output ref. Q-path one dB compression pt.
parameter real rin = 50 from (0:inf); // Input resistance in Ohms.
parameter real rout = 50 from (0:inf); // Output resistance in Ohms.
real quad_error, I_a, Q_a, I_cbb, Q_cbb, I_rho, Q_rho;
real I_rhooutmax, Q_rhooutmax, I_rhoinmax, Q_rhoinmax;
real I_rhoout, Q_rhoout, I_theta, Q_theta;
real I_cp, Q_cp, I_cmp, Q_cmp;
analog begin
    // The initial block converts the input parameters from engineering
    // units to implementation units.
    @(initial_step) begin
        quad_error = `M_PI*phi/180;
        I_a = pow(10,gain/20)*(1+epsilon/200);
        I_cmp = sqrt(pow(10,(I_cpdb+1)/10)*2*rout*0.001)/I_a;
        I_cbb = I_a*(0.145)/(I_cmp*I_cmp);
        I_rhoinmax = sqrt(I_a/(3*I_cbb));
        I_rhooutmax = (2*I_a/3)*I_rhoinmax;
        Q_a = pow(10,gain/20)*(1-epsilon/200);
        Q_cmp = sqrt(pow(10,(Q_cpdb+1)/10)*2*rout*0.001)/Q_a;
        Q_cbb = Q_a*(0.145)/(Q_cmp*Q_cmp);
        Q_rhoinmax = sqrt(Q_a/(3*Q_cbb));
        Q_rhooutmax = (2*Q_a/3)*Q_rhoinmax;
    end
    I_rho = V(I_in);
    Q_rho = V(Q_in);
```

Continued on the next page.

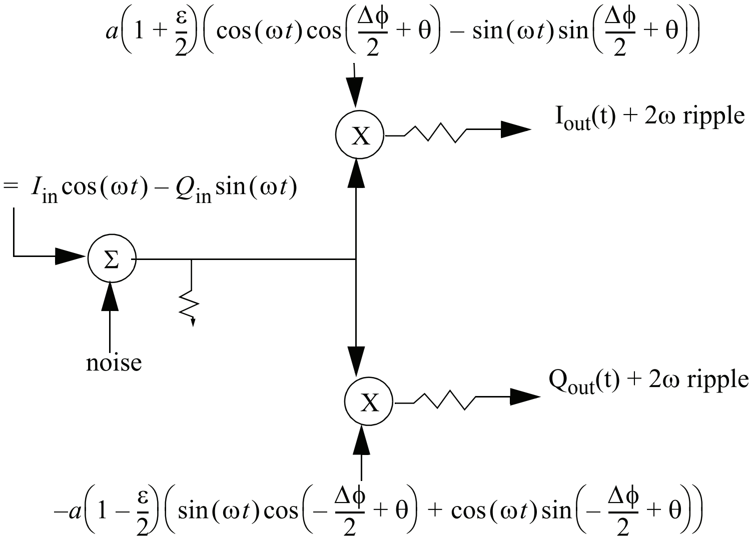
LISTING 6 *Baseband IQ modulator model.*

```
    // Apply the third order non-linearity. Clamp the output for extreme inputs.
    if ( abs(I_rho) < I_rhoinmax ) I_rhoout = (I_a - I_cbb*I_rho*I_rho)*I_rho;
    else if (I_rho < 0) I_rhoout = -I_rhooutmax;
    else I_rhoout = I_rhooutmax;
    if ( abs(Q_rho) < Q_rhoinmax ) Q_rhoout = (Q_a - Q_cbb*Q_rho*Q_rho)*Q_rho;
    else if (Q_rho < 0) Q_rhoout = -Q_rhooutmax;
    else Q_rhoout = Q_rhooutmax;
    // Phase computations.
    I_theta = V(phase_err)+quad_error/2;
    Q_theta = V(phase_err)-quad_error/2;
    I(phase_err) <+ V(phase_err)/50;
    I(I_in) <+ V(I_in)/rin;
    I(Q_in) <+ V(Q_in)/rin;
    I(I_out) <+ (-(I_rhoout*cos(I_theta)-Q_rhoout*sin(Q_theta)) + V(I_out))/rout;
    I(Q_out) <+ (-(I_rhoout*sin(I_theta)+Q_rhoout*cos(Q_theta)) + V(Q_out))/rout;
end
endmodule
```

4.3 IQ Demodulators

To facilitate the derivation of a baseband equivalent model of the *IQ* demodulator, Figure 2 is redrawn below with the local oscillator expressions expanded according to (63) and (64). Since the ripple terms are filtered out, either by parasitics or deliberate filters, the only relevant products resulting from the multipliers are those that produce zero-harmonics of the carrier. In summary, after performing the multiplications, we can drop all but the $\cos^2(\omega t)$ and $\sin^2(\omega t)$ terms and replace $\cos^2(\omega t)$ and $\sin^2(\omega t)$ each with 1/2. Neglecting noise for now, the filtered output of the *I*-multiplier is:

FIGURE 6 *IQ demodulator with expanded LO expressions.*



$a\left(1+\frac{\varepsilon}{2}\right)\left(\cos(\omega t)\cos\left(\frac{\Delta\phi}{2}+\theta\right)-\sin(\omega t)\sin\left(\frac{\Delta\phi}{2}+\theta\right)\right)$
X
$I_{out}(t) + 2\omega$ ripple
$V_{rf} = I_{in}\cos(\omega t) - Q_{in}\sin(\omega t)$
Σ
noise
X
$Q_{out}(t) + 2\omega$ ripple
$-a\left(1-\frac{\varepsilon}{2}\right)\left(\sin(\omega t)\cos\left(-\frac{\Delta\phi}{2}+\theta\right)+\cos(\omega t)\sin\left(-\frac{\Delta\phi}{2}+\theta\right)\right)$

$$I_{\text{out}} = \frac{a}{2}\left(1+\frac{\varepsilon}{2}\right)\cos\left(\frac{\Delta\phi}{2}+\theta\right)I_{\text{in}} + \frac{a}{2}\left(1+\frac{\varepsilon}{2}\right)\sin\left(\frac{\Delta\phi}{2}+\theta\right)Q_{\text{in}}\,. \tag{69}$$

The filtered output of the *Q*-multiplier is:

$$Q_{\text{out}} = -\frac{a}{2}\left(1-\frac{\varepsilon}{2}\right)\sin\left(-\frac{\Delta\phi}{2}+\theta\right)I_{\text{in}} + \frac{a}{2}\left(1-\frac{\varepsilon}{2}\right)\cos\left(-\frac{\Delta\phi}{2}+\theta\right)Q_{\text{in}}\,. \tag{70}$$

Recall from the passband discussion, *a* equals twice the specified gain.

There is a pitfall to avoid when simulating noise figure with baseband equivalent models. To simulate noise figure, the simulator must inject some input noise. Some simulators inject single sideband noise. However, to get the correct RMS noise at the outputs, the above discussions explain that we need to double the input noise. In baseband modeling, we must sometimes decide between noise figure or RMS noise. One convenient way to switch between the two analyses is to double the temperature of the input noise source for noise figure computations but not for estimating RMS noise.

4.3.1 Implementation

Listing 7 shows how an *IQ* demodulator could be coded.

4.4 Time Domain Noise: A Phase Noise Example

We occasionally wish to see how noise effects a constellation or eye-diagram and that requires time domain simulation. In general, we can not make a node noisy simply by adding a random variable at every time point. Analog simulators use variable time steps. The step size affects noise bandwidth and therefore also affects noise power. However, if we add noise at a fixed sample rate we can control the noise power. We can also shape the PSD by passing the noise through a digital filter.

As an example, we show how to generate phase noise with a desired power spectral density. The example at the end of this paper shows how to inject the phase noise into an *IQ* modulator. Here, we focus on generating the noise. Phase noise can be generated by passing additive nearly-white Gaussian noise through a digital filter. The input noise is “nearly” white because the noise is only assumed white over a bandwidth equal to the rate at which the noise is updated (i.e. the sampling frequency). The noise is assumed to be zero outside that bandwidth.

Let the desired PSD have a Lorentzian shape. The Lorentzian spectrum is constant at low frequencies and rolls off with a first order slope past some corner frequency. We will design an IIR (infinite impulse response filter) using a bilinear transformation [17] and drive it with a white noise source with the desired low frequency PSD. Let’s design the filter first. The target continuous time PSD is described by the squared magnitude of *H*(*s*) where

$$H(s) = \frac{p}{(s+p)}\,. \tag{71}$$

p is the desired corner frequency in rad/sec. To move to discrete time, we transform from *s* to *z* according to

LISTING 7 *Baseband IQ demodulator model.*

```
// Baseband equivalent model of an IQ demodulator.

`define K 1000
`include "constants.h"
`include "discipline.h"

module IQ_demod_BB(I_in , I_out, Q_in, Q_out, phase_err);
inout I_in, Q_in, I_out, Q_out, phase_err;
electrical I_in, Q_in, I_out, Q_out, phase_err;
electrical I_inn, Q_inn;

parameter real gain = 40; // Gain from input to one output in dB.
parameter real epsilon = 0; // Gain mismatch in percent.
parameter real phi = 0; // Quadrature error in degrees.
parameter real I_ip3 = 100; // Input ref. 3rd order intercept in the I-path.
parameter real Q_ip3 = 100; // Input ref. 3rd order intercept in the Q-path.
parameter real rin = 50 from (0:inf); // Input resistance in Ohms.
parameter real rout = 50 from (0:inf); // Output resistance in Ohms.
parameter real nf = 0 from [0:inf]; // Noise figure in dB.
real quad_error, I_gain, Q_gain, I_a, Q_a, I_cbb, Q_cbb;
real rho, I_rhooutmax, Q_rhooutmax, I_rhoinmax, Q_rhoinmax;
real theta, I_phi, Q_phi, I_alpha, Q_alpha, iout, qout;
real noise_voltage_squared, rnf;
analog begin
    // The initial block converts the input parameters from engineering
    // units to implementation units.

    @(initial_step) begin
        quad_error = phi*`M_PI/180;
        I_a = pow(10,gain/20)*(1+epsilon/200);
        I_cbb = I_a/(pow(10,I_ip3/10.0)*2*50*0.001);
        I_rhoinmax = sqrt(I_a/(3.0*I_cbb));
        I_rhooutmax = (2.0*I_a/3.0)*I_rhoinmax;
        Q_a = pow(10,gain/20)*(1–epsilon/200);
        Q_cbb = Q_a/(pow(10,Q_ip3/10)*2*50*0.001);
        Q_rhoinmax = sqrt(Q_a/(3*Q_cbb));
        Q_rhooutmax = (2*Q_a/3)*Q_rhoinmax;

        // voltage gain must be vrms/vrms
        rnf = pow(10,nf/10);
        noise_voltage_squared = 8.0*(rnf–1)*`P_K*$temperature*50;
    end

    // Compute the input angle and radius.
    if (V(I_inn) !=0) theta = atan2(V(Q_inn),V(I_inn));
    else if (V(Q_inn) > 0) theta = `M_PI/2;
    else theta = –`M_PI/2;
    rho = hypot(`K*V(I_inn),`K*V(Q_inn))/`K;

    // Phase computations.
    I_phi = V(phase_err) + quad_error/2;
    Q_phi = V(phase_err) – quad_error/2;
```

Continued on the next page.

LISTING 7 *Baseband IQ demodulator model.*

```
    // Apply the third order non-linearity and clamp the output for extreme
    // inputs. I first:
    if (rho < I_rhoinmax) begin
        I_alpha = (I_a – I_cbb∗rho∗rho);
        iout = I_alpha∗(V(I_inn)∗cos(I_phi) + V(Q_inn)∗sin(I_phi));
    end else begin
        iout = I_rhooutmax∗cos(theta–I_phi);
    end
    // Now Q:
    if (rho < Q_rhoinmax ) begin
        Q_alpha = (Q_a – Q_cbb∗rho∗rho);
        qout = Q_alpha∗(–V(I_inn)∗sin(Q_phi) + V(Q_inn)∗cos(Q_phi));
    end else begin
        qout = Q_rhooutmax∗sin(theta–Q_phi);
    end

    // Assign values to the output pins.
    I(phase_err) <+ V(phase_err)/50;
    I(I_inn) <+ V(I_inn)/rin;
    I(Q_inn) <+ V(Q_inn)/rin;
    I(I_out) <+ (–iout + V(I_out))/rout;
    I(Q_out) <+ (–qout + V(Q_out))/rout;

    // Add the noise.
    V(I_in, I_inn) <+ white_noise(noise_voltage_squared,"IQ_demod_BB_i");
    V(Q_in,Q_inn) <+ white_noise(noise_voltage_squared,"IQ_demod_BB_q");
end
endmodule
```

$$\frac{2}{t_s}\left(\frac{z-1}{z+1}\right) = s\,, \tag{72}$$

where t_s is the noise update period. After some rearranging, the discrete transfer function, $H(z)$, is

$$H(z) = \frac{\gamma}{1+\gamma}\left(\frac{z+1}{z-\left(\frac{1-\gamma}{1+\gamma}\right)}\right), \tag{73}$$

where

$$\gamma = \frac{pt_s}{2}\,. \tag{74}$$

$H(z)$ can be entered directly in VerilogAMS as a z-domain transfer function. The next step is to drive the IIR filter with the proper white noise source. Let the desired double-sided low frequency phase noise level be β radians squared per Hz. This is the same as the single side band density near the carrier. A random number generator with a Gaussian distribution drives the filter and to specify it, we need the standard deviation, σ. We solve for the variance, σ^2, of the input noise by equating the variance to the area under the PSD:

$$\sigma^2 = \frac{\beta}{t_s}. \tag{75}$$

The sample rate, t_s, should be twice as small as any time constant of interest but not so small that it significantly slows execution.

4.4.1 Implementation

Listing 8 shows how one could code the phase noise generator.

LISTING 8 *Phase noise generator.*

```
//Baseband phase noise model
`include "constants.h"
`include "discipline.h"

module phase_noise(phase_noise);
inout phase_noise;
electrical phase_noise;

parameter real bw = 200000 from (0:inf); // Corner frequency in Hz.
parameter real alpha = –75; // SSB low frequency phase noise, dBc/Hz.
parameter real sr = 40e6; // Sample rate in Hz.
parameter integer seedd = 23; // Random number generator seed.

real a, b, ts, pole, beta, x, y;
real p[1:2];
integer seed;
analog begin
    @(initial_step) begin
        seed = seedd;
        ts = 1/sr;
        pole = 2∗`M_PI∗bw;
        a = pole∗ts/2/(1+pole∗ts/2);
        b = (1–pole∗ts/2)/(1+pole∗ts/2);
        beta = pow(10,alpha/20.0)/sqrt(ts);
        p[1] = 1;
        p[2] = –b;
    end
    @(timer(0,ts)) x = a∗beta∗$dist_normal(seed, 0, 1);
    y = zi_nd(x, {1,1},p,ts,ts,10p);
    I(phase_noise) <+ (–2∗y + V(phase_noise) )/50;
end
endmodule
```

4.5 Reactive Elements

Baseband equivalent models of passband filters are trivial to implement in the frequency domain [2,10,11]. Simply slide the passband transfer function to the left to make the positive frequency transfer function straddle zero Hz instead of the carrier. Frequency translation is easy enough to implement in the frequency domain but not in VerilogAMS. VerilogAMS is primarily a time domain modeling tool. We therefore seek a way to slide the transfer function down to baseband with a time domain model. The modeling method described in this section assumes the filter is described as lumped network. The method replaces each lumped element with a baseband equivalent.

Consider an inductor in a passband circuit. The constitutive relationship between current and voltage for an inductor is

$$v = L\frac{d}{dt}i\,. \tag{76}$$

Working at passband, we have

$$v(t) = v_{\mathrm{i}}(t)\cos(\omega t) - v_{\mathrm{q}}(t)\sin(\omega t) \tag{77}$$

and

$$i(t) = i_{\mathrm{i}}(t)\cos(\omega t) - i_{\mathrm{q}}(t)\sin(\omega t)\,. \tag{78}$$

Substituting (77) and (78) into (76) and carrying out the differentiation gives

$$\begin{aligned} v_{\mathrm{i}}(t)\cos(\omega t) - v_{\mathrm{q}}(t)\sin(\omega t) = {} & \left(L\frac{d}{dt}i_{\mathrm{i}}(t)\right)\cos(\omega t) - L\omega\sin(\omega t)i_{\mathrm{i}}(t) - \\ & \left(L\frac{d}{dt}i_{\mathrm{q}}(t)\right)\sin(\omega t) – L\omega i_{\mathrm{q}}(t)\cos(\omega t)\,. \end{aligned} \tag{79}$$

Equating the coefficients of the trigonometric terms gives the baseband equivalent relationships for the inductor.

$$v_{\mathrm{i}}(t) = L\frac{d}{dt}i_{\mathrm{i}}(t) - L\omega i_{\mathrm{q}}(t) \tag{80}$$

and

$$v_{\mathrm{q}}(t) = L\frac{d}{dt}i_{\mathrm{q}}(t) + L\omega i_{\mathrm{i}}(t)\,. \tag{81}$$

Using complex notation, (80) and (81) become

$$v_{\mathrm{i}} + jv_{\mathrm{q}} = L\frac{d}{dt}(i_{\mathrm{i}} + ji_{\mathrm{q}}) + jL\omega(i_{\mathrm{i}} + ji_{\mathrm{q}})\,. \tag{82}$$

This is consistent with frequency translation. If we replace the derivative operator with the Laplace transform variable, *s*, we have the familiar impedance of an inductor but with *s* replaced with *s*+*j*ω. The same procedure can be carried out for capacitors, resulting in the same substitution for *s*. Thus, the original transfer function *H*(*s*), becomes *H*(s+*j*ω), which is exactly the frequency translation mentioned earlier.

The terms involving ω in (80) and (80) are like the back EMF of a 2-phase motor. The same relationships can be derived using a time varying coordinate transformation. Such transformations date back to the 1920s in modeling electric motors [18,19,20,21].

4.5.1 Implementation

Listing 9 and Listing 10 show how one might code up the baseband equivalent models of inductors and capacitors.

LISTING 9 *Baseband equivalent inductor.*

```
// Baseband equivalent model of an inductor.
`include "constants.h"
`include "discipline.h"

`module ind_BB(iin, iout, qin, qout);
inout iin, iout, qin, qout;
electrical iin, iout, qin, qout;
parameter real ind = 1n; // Inductance in Henrys.
parameter real carrier = 1G; // Carrier frequency in Hz.

analog begin
    V(iin,iout) <+ ind∗(ddt(I(iin,iout)) – 2∗`M_PI∗carrier∗I(qin,qout));
    V(qin,qout) <+ ind∗(ddt(I(qin,qout)) + 2∗`M_PI∗carrier∗I(iin,iout));
end
endmodule
```

LISTING 10 *Baseband equivalent capacitor.*

```
// Baseband equivalent of a capacitor.
`include "constants.h"
include "discipline.h"

module cap_BB(iin, iout, qin, qout);
inout iin, iout, qin, qout;
electrical iin, iout, qin, qout;
parameter real cap = 1p; // Capacitance in Farads.
parameter real carrier = 1G; // Carrier frequency in Hz.

analog begin
    I(iin,iout) <+ cap∗(ddt(V(iin,iout)) – 2∗`M_PI∗carrier∗V(qin,qout));
    I(qin,qout) <+ cap∗(ddt(V(qin,qout)) + 2∗`M_PI∗carrier∗V(iin,iout));
end
endmodule
```

5 Example

5.1 Preamble “Pretzel” Diagnostics

As stated in Section 2, the models discussed in this paper can play a major role in the top-down design of a modern wireless link. However, the diagnostic application discussed below better illustrates the models’ capabilities and demonstrates another application.

When it comes to testing an RF chip in the lab, the relationship between the engineer and the chip is much like the relationship between a doctor and his patient. Definitive diagnostic tests can be time consuming and require expensive equipment. However, observing only the most outward symptoms, an experienced doctor can diagnose a problem in minutes.

With some of the new communications standards, the only conventional symptom of poor performance is a bad packet error rate (PER). A bad PER is about as helpful as simply telling the doctor you don't feel good. The 802.11a standard for wireless LAN is based on OFDM (orthogonal frequency domain multiplexing). We needn't get into 802.11a details here except to say that the data comes in bursts and that only the beginning of each burst contains anything remotely regular. The beginning of the burst is called the preamble. The preamble begins with a known waveform called a short training sequence (STS). The STS is a periodic baseband waveform, $I(t) + jQ(t)$, lasting 8 us. The $I(t)$ and $Q(t)$ waveforms individually do not reveal much. However, plotted against each other, the waveforms trace out a trajectory rich in easily recognizable symptoms. The trajectory is easily produced in the lab. Since a healthy trajectory happens to resemble a pretzel, I refer to these tests as "pretzel diagnostics".

Behavioral models can quickly train an engineer in how to diagnose STS pretzels. Based only on a visual inspection, a good pretzel doctor can not only identify the distortion mechanism, he/she can quantify within a few percent, the standard metric associated with the distortion. This section shows how baseband models can quickly create a catalog of pretzel symptoms associated with a few common problems in a direct conversion wireless OFDM link. This example is not meant to create an exhaustive catalog of symptoms; it only demonstrates the basic idea.

5.1.1 Normal Pretzel

I generated the pretzels in this section primarily with baseband equivalent models of a direct conversion transmitter and a direct conversion receiver. I constructed both models from the models described in this paper. The transmitter drove the receiver through a simple attenuation block that modeled the loss between the transmitter and receiver antennae. All pretzels were observed at the receiver output.

Figure 7 overlays two pretzels, one simulated with passband models and the other simulated with baseband models. Aside from the initial conditions, the two models are in good agreement but the baseband model ran 221 times faster than the passband model. The initial conditions differ because the DC analysis from the passband model does not produce the correct output initial conditions. In the passband model, no meaningful DC operating point exists because the mixers render the system time-varying. Since the baseband model suppresses the carrier, the model is time-invariant and the DC analysis computes true steady state operating conditions.

The remaining pretzels were generated with baseband equivalent models alone.

5.1.2 Frequency Offset

The transmitter and receiver local oscillators are usually independent and therefore never at exactly the same frequency. The frequency offset causes the observed receiver pretzel to rotate at the offset frequency. Figure 8 shows what frequency offset does to the received pretzel.

5.1.3 IQ Gain Mismatch

The transmitter and/or receiver can suffer from *IQ* mismatch. *IQ* mismatch can be caused by asymmetry in the mixer gains or quadrature error between the local oscillators driving the mixers. Quadrature error refers to the *I*- and *Q*- local oscillator signals being out of phase by something other than 90 degrees. Let's consider gain mismatch first. Figure 9 shows the received pretzel in the presence of transmitter gain mismatch

FIGURE 7 *Normal Pretzel simulated with passband and baseband models.*

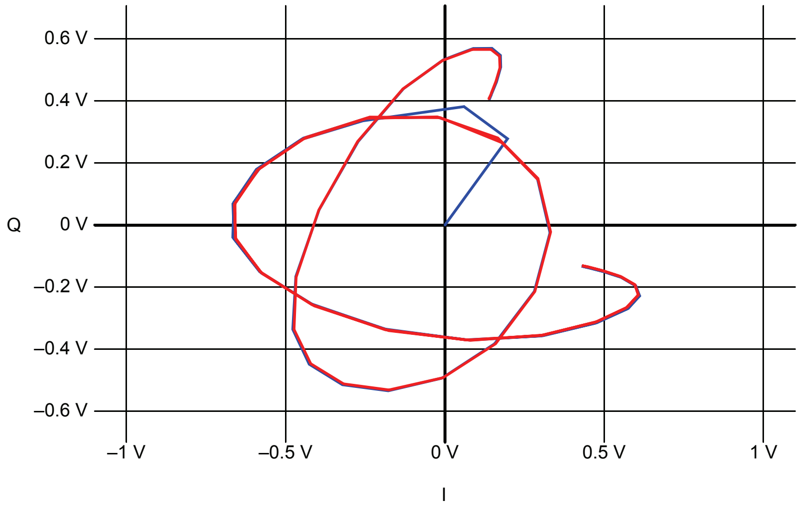
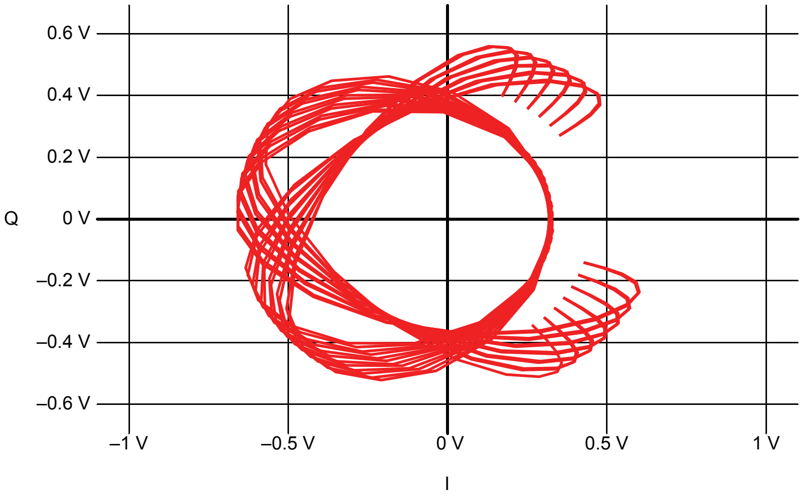


FIGURE 8 *Frequency offset.*



and a considerable amount of frequency offset. If there is no frequency error, gain mismatch compresses the pretzel in one direction. If the mismatch is in the transmitter, the compression can occur along any direction. With frequency offset and transmitter mismatch, the aspect ratio of the pretzel stays the same and with sufficient frequency error, the pretzel spins to sweep out a disk.

Both the *IQ* modulator and demodulator models have mismatch parameters. They also have pins for injecting phase noise and frequency offset. Frequency offset is introduced by driving the pin with a ramp or circular integrator that integrates a constant.

Figure 10 shows the pretzel resulting from receiver gain mismatch instead of transmitter mismatch. Note that with receiver gain mismatch, the compression occurs along either the vertical or horizontal axis, regardless of the orientation of the pretzel.

FIGURE 9 *Transmitter gain mismatch.*

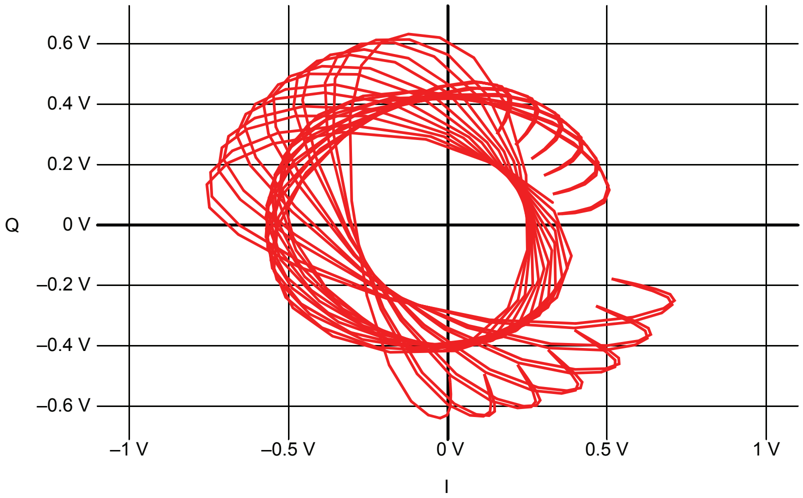
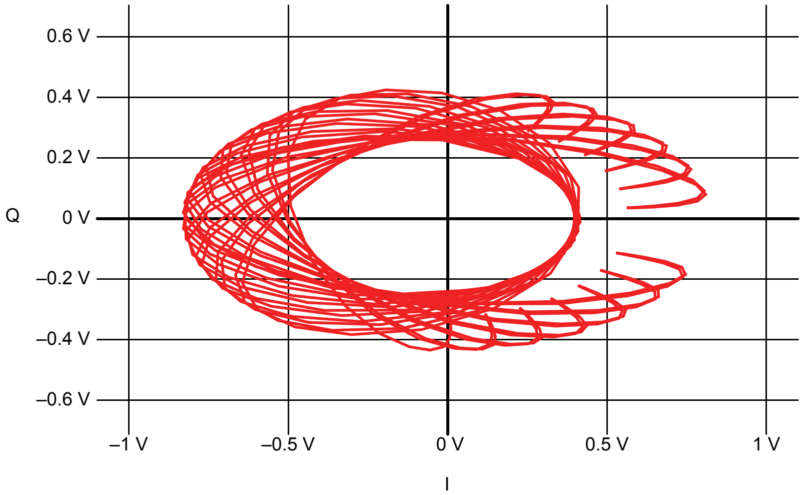


FIGURE 10 *Receiver gain mismatch.*



5.1.4 IQ Phase Mismatch (Quadrature Error)

Both the *IQ* modulator and demodulator have parameters for quadrature error. Figure 11 shows what quadrature error does to the pretzel. If the quadrature error has the opposite sign, the compression occurs along the other diagonal.

5.1.5 AM/AM Conversion

Large input signals can drive either the transmitter or receiver into saturation. Figure 12 shows how AM/AM conversion affects the pretzel.

FIGURE 11 *Quadrature error.*

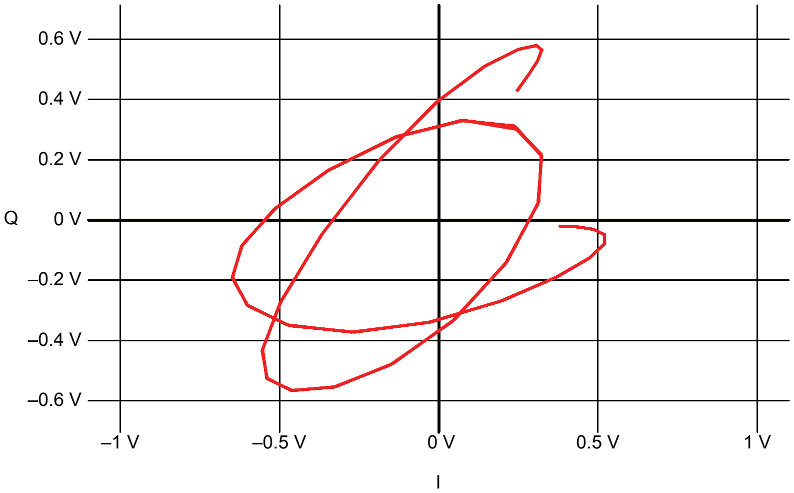
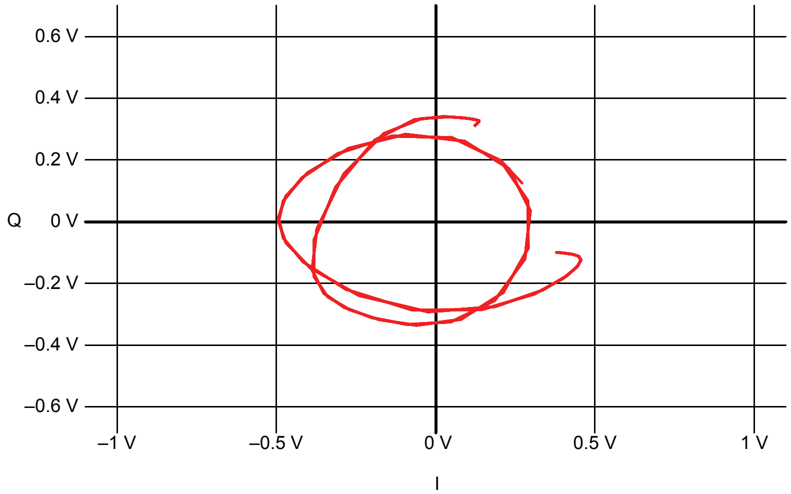


FIGURE 12 *AM/AM conversion.*



5.1.6 Reduced Filter Bandwidth

In the lab, it can be easy to set the bandwidth of the instrumentation too low. Figure 13 shows what happens to the pretzel when the receiver baseband low pass filter is 6.6MHz instead of 8.8MHz.

5.1.7 Phase Noise

All transceivers have some amount of phase noise. In a 54MB/s OFDM system, phase noise can be a serious problem. Figure 14 shows what a moderate amount of phase noise does to the pretzel.

FIGURE 13 *Low pass filter bandwidth too small.*

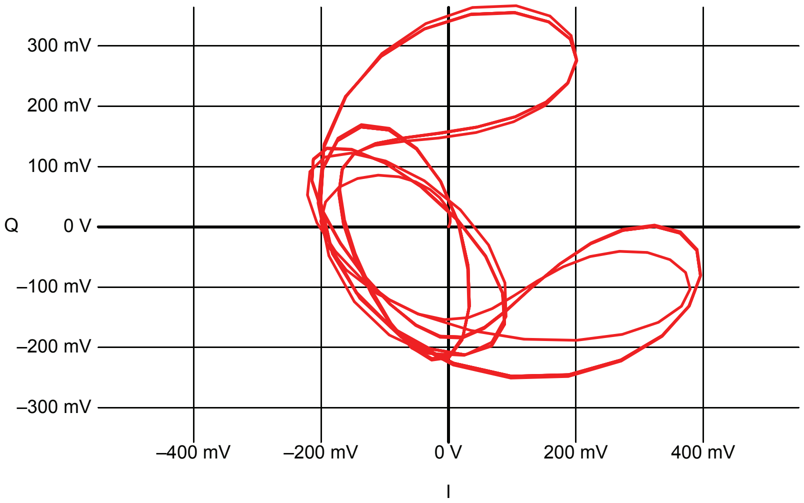
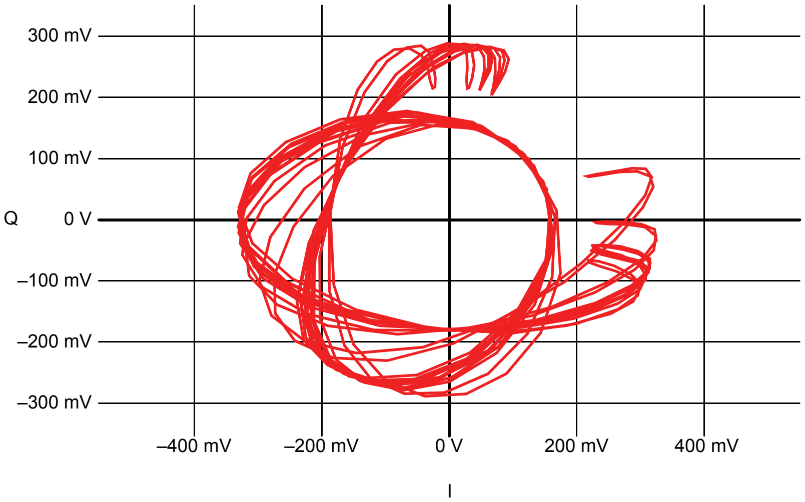


FIGURE 14 *Phase noise.*



If You Have Questions

If you have questions about what you have just read, feel free to post them on the *Forum* section of *The Designer's Guide Community* website. Do so by going to *www.designers-guide.org/Forum*.

References

[1] Thomas R. Turlington. *Behavioral Modeling of Nonlinear RF and Microwave Devices*. Artech House. 2000.

[2] Michel C. Jeruchim, Philip Balaban, K. Sam Shanmugan. *Simulation of Communication Systems.* Plenum Press. 1992. Pages 40-45, 496-503, 149-153, 283.

[3] Aarano Parssinen. *Direct Conversion Receivers in Wide-Band Systems.* Kluwer Academic Publishers. 2001.

[4] Ken Kundert. Introduction to RF simulation and its application. *Journal of Solid-State Circuits,* vol. 34, no. 9, September 1999. Available from *www.designers-guide.org/Analysis*.

[5] M. Schetzen. *The Volterra Wiener Theories of Nonlinear Systems.* Wiley and Sons, 1980.

[6] Piet Wambacq and Willy Sansen. *Distortion Analysis of Analog Integrated Circuits*. Kluwer Academic Publishers. 1998.

[7] Behzad Razavi. *RF Microelectronics.* Prentice Hall. 1998. Pages 136-137.

[8] Ralph S. Carson. *Radio Concepts Analog.* John Wiley and Sons. 1990.

[9] Ralph J. Smith. Circuits Devices and Systems. Third edition. John Wiley and Sons. 1976.

[10] Edward A. Lee and David G. Messerschmitt. *Digital Communication*, Second Edition. Kluwer Academic Publications. 1994. Pages 208-209.

[11] Floyd Gardner and John D. Baker. *Simulation Techniques: Models of Communication Signals and Processes*. John Wiley and Sons, Inc. 1997.

[12] M. Vidyasagar. *Nonlinear Systems Analysis*. Prentice-Hall. 1978. Pages 103-104.

[13] Athanasios Papoulis. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill Book Company. 1965.

[14] Tri T. Ha. *Solid-State Microwave Amplifier Design.* Krieger Publishing Co. 1991.

[15] Stephan A. Maas. *Nonlinear Microwave Circuits*. IEEE Press. 1988.

[16] Steve C. Cripps. *RF Power Amplifiers for Wireless Communications*. Artech House. 1999

[17] Gene F. Franklin, J. David Powell, Michael L. Workman. Digital Control of Dynamic Systems. Second Edition. Addison-Wesley Publishing Company. 1990.

[18] Peter Vas. *Vector Control of AC Machines*. Oxford Science Publications. 1990. Pages 135-143.

[19] Paul C. Krause and Oleg Wasynczuk. *Electromechanical Motion Devices.* McGraw Hill. 1989. Pages 172-176

[20] W. Leonard. *Control of Electrical Drives*. Springer-Verlag. 1990. Pages 142-143.

[21] Robert Stein and William T. Hunt Jr. *Electric Power System Components.* Van Nostrand Reinhold. 1979. Page13