cadence®

Virtuoso Spectre Circuit Simulator RF Analysis Library Reference

Product Version 7.2
December 2009

Printed in the United States of America.

Cadence Design Systems, Inc. (Cadence), 2655 Seely Ave., San Jose, CA 95134, USA.

Contents

Preface

Virtuoso® Spectre® circuit simulator RF analysis (Spectre RF) provides functionality designed for the needs of RF designers. Spectre RF

- Supports the efficient calculation of the operating point, transfer function, noise, and distortion of common RF and communication circuits, such as mixers, oscillators, sample and holds, and switched capacitor filters.
- Supports a multi-technology simulation (MTS) mode that enables the simulation of a system consisting of blocks designed with different processes, such as RF System-in-Package (SIP).

This user guide assumes that you are familiar with:

- RF circuit design.
- SPICE simulation.
- The Virtuoso® analog design environment (ADE).

Spectre and Spectre RF features are available in different tiers. The L tier offers basic design creation and implementation capabilities. The XL tier introduces new technologies and advancements in automation. The following table provides an overview of the features supported by each tier of products.

Features Supported in Spectre L and Spectre XL Tiers

Features	L Tier	XL Tier	GXL Tier
DC, AC, and transient analysis.	X	X	X
Noise, transfer function, and sensitivity analysis.	X	X	X
Transient noise analysis.	X	X	X
Monte Carlo and parametric statistical support.	X	X	X
Built-in measurement description language (MDL). However, MDL does not support RF analysis.	X	X	X
Parametric sweep.	X	X	X

Features Supported in Spectre L and Spectre XL Tiers, *continued*

Features	L Tier	XL Tier	GXL Tier
Multi-threading. (L Tier is limited to 4 CPUs; XL Tier to 8 CPUs; GXL Tier has no limit.)	X	X	X
Periodic and quasi-periodic steady state analysis (PSS and QPSS) based on harmonic balance and shooting Newton.		X	X
Periodic and quasi-periodic noise analysis (PNoise, QPNoise).		X	X
Periodic and quasi-periodic small signal analysis (PAC, PXF, PSP, QPAC, QPXF, QPSP).		X	X
Periodic stability analysis (PSTB).		x	x
Time-domain and frequency-domain envelope analysis.		X	X
Perturbation-based rapid IP2 and IP3.		X	X
Cosimulation with Simulink® from The MathWorks.		X	X
MMSIM Toolbox for MATLAB® from The MathWorks.		X	X
Spectre Turbo.		X	X
Spectre Parasitics.			X

Licensing for Spectre RF

To run the G, GX, and GXL tier features of Spectre RF, you must have access to a corresponding license or combination of licenses. The order in which these licenses are used is determined either by default or by using the `+lorder` option.

The `+lorder` option lets you specify a custom license checkout order for simulation. Spectre checks for a license in the specified order.

```
+lorder licenseList
```

`licenseList`	A list of licenses. Use `:` between the license names when defining the order. For example, `+lorder Virtuoso_Multi_mode_Simulation:Virtuoso_Spectre` specifies that the token license is checked before the `Virtuoso_Spectre` license.

The default license checkout order for Spectre RF is

```
Virtuoso_Multi_mode_Simulation:Virtuoso_Spectre_XL:Virtuoso_Spectre_GXL
```

where the `Virtuoso_Multi_mode_Simulation` license takes 1 token.

If the `Virtuoso_Spectre_XL` license or the `Virtuoso_Spectre_GXL` license is used, no other license is required to run Spectre RF. Otherwise, one of the following add-on licenses is required and the default checkout order for them is

```
Virtuoso_Spectre_RF:Virtuoso_Multi_mode_Simulation
```

where the `Virtuoso_Multi_mode_Simulation` license takes one additional token.

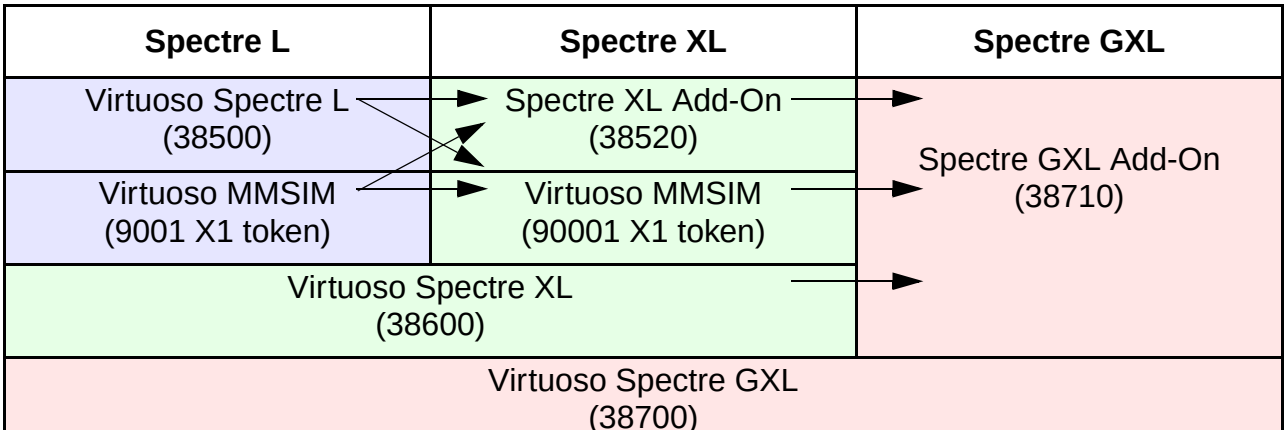
To summarize these rules, Spectre RF can run with two types of licenses, either a Spectre XL license or 2 MMSIM tokens. The Spectre XL license is available as the `Virtuoso_Spectre_XL` license or as the `Virtuoso_Spectre_RF` add-on license.

The following table lists the licenses for Spectre RF.

License Name	Feature
`Virtuoso_Spectre_RF`	License associated with *Virtuoso Spectre Circuit Simulator XL Add on Option to 38500* (38520).
`Virtuoso_Multi_mode_Simulation`	Token license associated with *Virtuoso Multi-mode Simulation Product* (90001). Two tokens are checked out simultaneously for Spectre Turbo, MathWorks, or RF analysis.
`Virtuoso_Spectre`	License associated with the *Virtuoso Spectre Circuit Simulator L* (38500).
`Virtuoso_Spectre_XL`	License associated with *Virtuoso Spectre Circuit Simulator XL* (38600). This license is a superset of `Virtuoso_Spectre` and `Virtuoso_Spectre_RF`.
`Virtuoso_Spectre_GXL`	License associated with *Virtuoso Spectre Circuit Simulator GXL* (38700). This license is a superset of `Virtuoso_Spectre` and `Virtuoso_Spectre_RF`.

License Name	Feature
`Virtuoso_Spectre_GXL_Option`	License associated with *Virtuoso Spectre Simulator GXL ADD On Option to 38600* (38710)

The following illustration summarizes the possible licensing combinations.



The following licenses are no longer sold but might be available at some sites.

License Name	Feature
`32500`	License associated with Virtuoso Spectre Circuit Simulator (32500).
`SpectreRF`	License associated with Virtuoso Spectre RF Simulation Option (32520).

Related Documents for Spectre RF

This user guide contains information about the RF functionality. The following documents provide more information about Spectre RF and related products.

- For a complete description of Spectre RF functionality you also need to refer to the Virtuoso Spectre Circuit Simulator documentation set. See
  - *Virtuoso Spectre Circuit Simulator Reference*

- *Virtuoso Spectre Circuit Simulator User Guide*

- For in-depth information and detailed examples of Spectre RF usage, see the documents listed in Appendix M, "Documents That Ship in the Software Hierarchy." So that the listed documents can be as up-to-date as possible, they are shipped with the MMSIM hierarchy rather than with the standard Cadence document set.
- To learn more about the Analog Circuit Design Environment, consult the *Virtuoso® Analog Design Environment User Guide*.
- To learn more about Spectre RF, see the reference information and theoretical concepts in *Virtuoso Spectre Circuit Simulator RF Analysis Theory*.

Third Party Tools

To view any `.swf` multimedia files, you need:

- Flash-enabled web browser, for example, Internet Explorer 5.0 or later, Netscape 6.0 or later, or Mozilla Firefox 1.6 or later. Alternatively, you can download Flash Player (version 6.0 or later) directly from the Adobe website.
- Speakers and a sound card installed on your computer for videos with audio.

Typographic and Syntax Conventions

The following typographic and syntax conventions are used in this manual.

`text`	Indicates text you must type exactly as it is presented.
`argument`	Indicates text that you must replace with an appropriate argument.
`[ ]`	Denotes an optional argument. When used with vertical bars, they enclose a list of choices from which you can choose one.
`{ }`	Used with vertical bars, they denote a list of choices from which you must choose one.
`|`	Separates a choice of options.
text	Indicates names of manuals, menu commands, form buttons, and form fields.

1

rfLib Library

The elements contained in the RF library, `rfLib`, are organized into the following categories:

Categories	Description of Category and Link
`Everything`	Lists all elements in `rfLib`.
`Baseband_components`	“Baseband_components Category” on page 18
`Butterworth_filters`	“Butterworth_filters Category” on page 50
`Chebyshev_filters`	“Chebyshev_filters Category” on page 67
`Measurements`	“Measurements Category” on page 80
`Passband_components`	“Passband_components Category” on page 105.
`RF_components`	“RF_components Category” on page 118
`Testbenches`	“Testbenches Category” on page 151
`WCDMA_components`	“WCDMA_components Category” on page 213

The `rfLib` elements support the design of both RF circuits and RF systems.

This chapter also describes how to use the Modelwriter to modify the baseband signal generators. For information, see “Modifying the BB Signal Generators Using Modelwriter” on page 223.

Baseband_components Category

The `Baseband_components` category contains the top-down baseband models of common architectural function blocks. The default view of these models is the baseband view (called `veriloga`) but most models in this category also have a differential passband view (called `veriloga_PB`). The `BB_loss` and `VGA_BB` models are exceptions because they are meant only for baseband analysis and have no passband view.

The `Baseband_components` models provide a fast method of mapping RF system specifications into detailed RF designs. The baseband models facilitate fast evaluation of candidate RF architectures specified with DSP metrics.

Baseband models are behavioral models and all behavioral models sacrifice some accuracy for increased simulation speed. Such sacrifices are usually acceptable in architectural studies because many implementation-dependent details do not affect high level decisions. The modeling approach taken in top-down design is to simulate only those effects that drive the decisions at hand.

Baseband modeling does not replace passband modeling because some effects missed by equivalent baseband models can affect high level decisions. However, the application of baseband models early followed by passband models later minimizes the number of slow simulations needed at low levels of design abstraction. Baseband models help you to quickly weed out designs that would surely fail tests simulated with passband models.

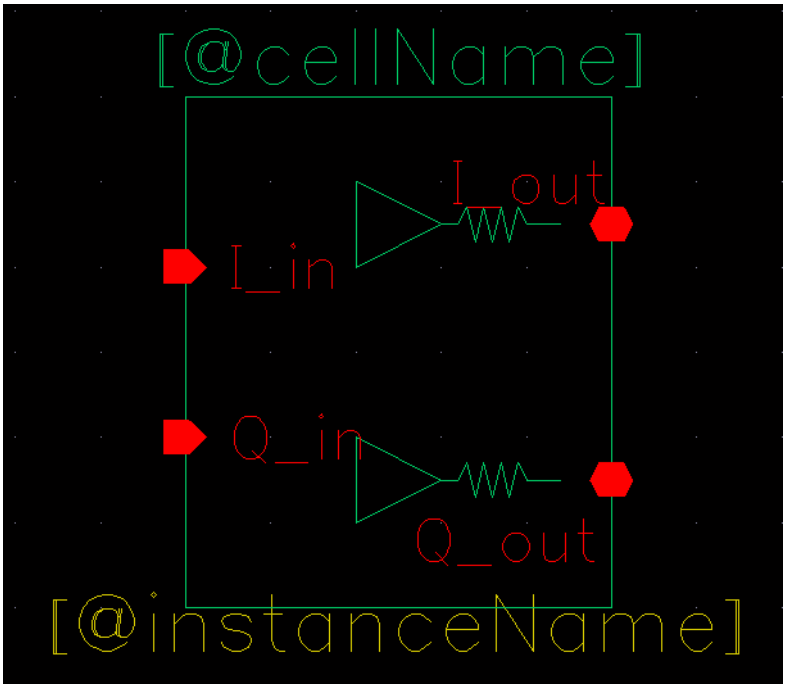
The cells in the `Baseband_components` category are:

- BB_driver
- BB_loss
- BB_shifter_combiner
- BB_shifter_splitter
- BB_xfmr
- cap_BB
- dwn_cnvrt
- HilbertTr_BB
- ind_BB
- IQ_demod_BB
- IQ_mod_BB

- LNA_BB
- PA_BB
- res_BB
- up_cnvrt
- VGA_BB

BB_driver

(Baseband Driver)

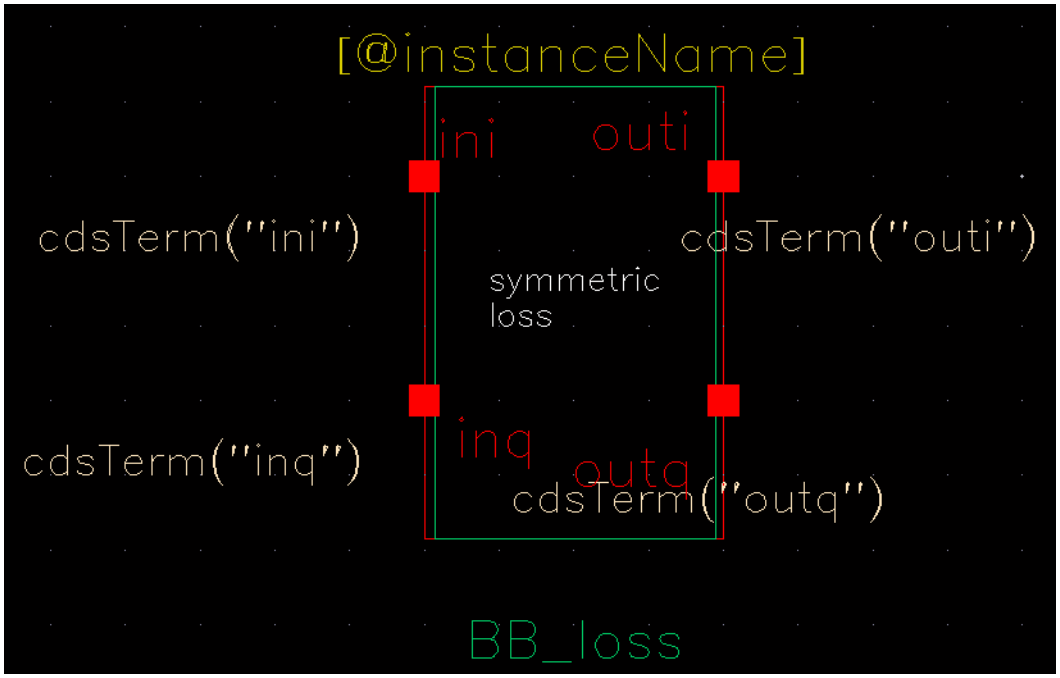


The BB_driver element senses a baseband voltage signal and amplifies it.

The parameters are:

`rout`	Output impedance.
`power_level`	

BB_loss

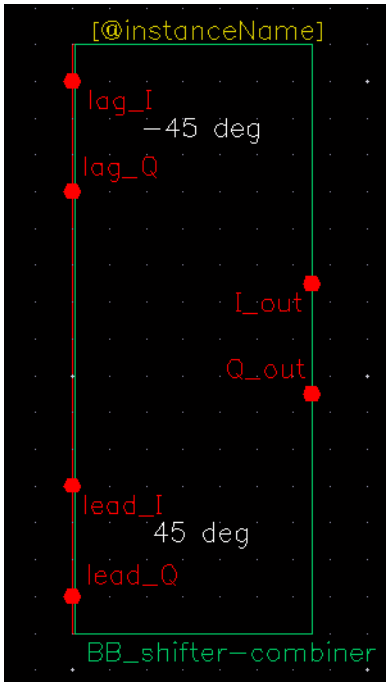


The `BB_loss` element is designed to be used with error vector magnitude (EVM) calculations. EVM is defined in terms of an ideal receiver or transmitter. If you want to remove a filter's response from the ideal receiver model while leaving only the passband attenuation, replace the filter with a `BB_loss` element and give it the same insertion loss as the filter. There is no passband view or counterpart for this model.

The parameters are:

`loss`	Filter insertion loss [dB].
`r2`	Reference impedance at port 2 [ohm].

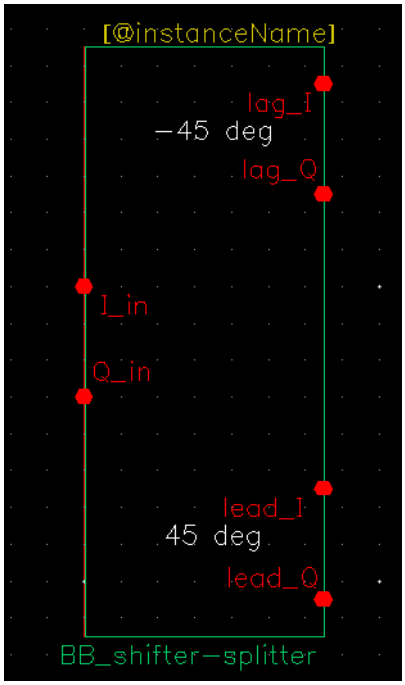
BB_shifter_combiner



The parameters are:

`fcr`	Carrier frequency.
`freq`	Frequency at which the magnitudes are balanced.
`gain`	Linear scale factor that multiplies the input voltage.
`r`	Impedance of the internal resistor.
`rin`	Input terminal impedances.
`rout`	Output impedance.

BB_shifter_splitter

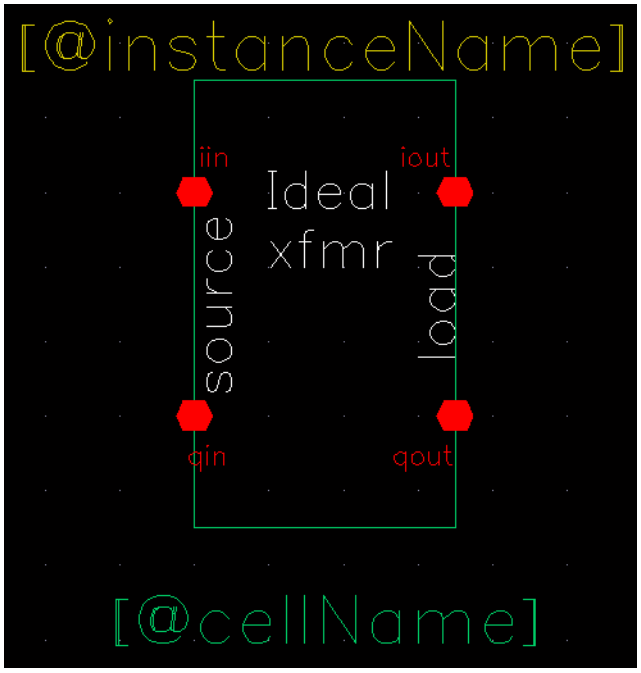


The parameters are:

`fcr`	Carrier frequency.
`freq`	Frequency at which the magnitudes are balanced.
`gain`	Linear scale factor that multiplies the input voltage.
`r`	Impedance of the internal resistor.
`rin`	Input terminal impedances.
`rout`	Output impedance.

BB_xfmr

(Ideal Transformer)



The purpose of the ideal transformer is to help designers transform between different resistances.

The parameters are:

`rsource`	`Rs`, source resistance
`rload`	`RI`, load resistance

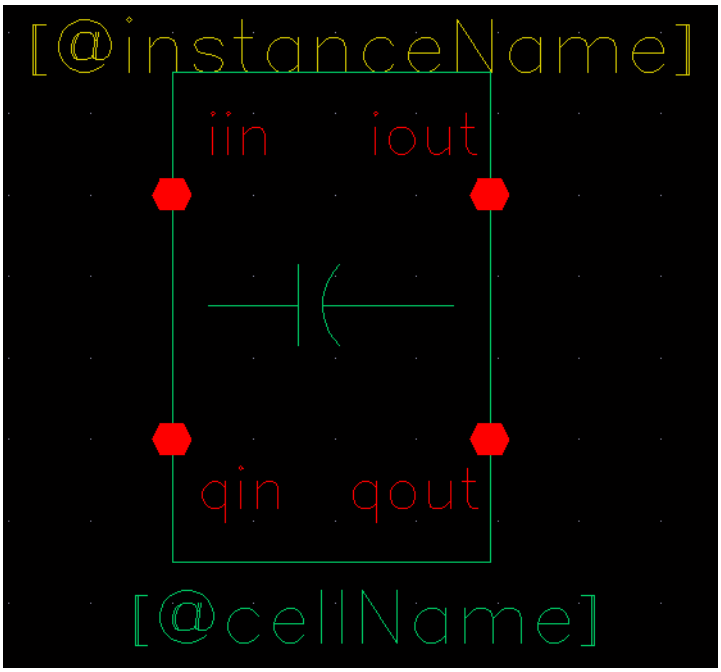
- Inputs: i and q input voltages, i and q output currents.
- Outputs: i and q output voltages, i and q input currents, defined as:

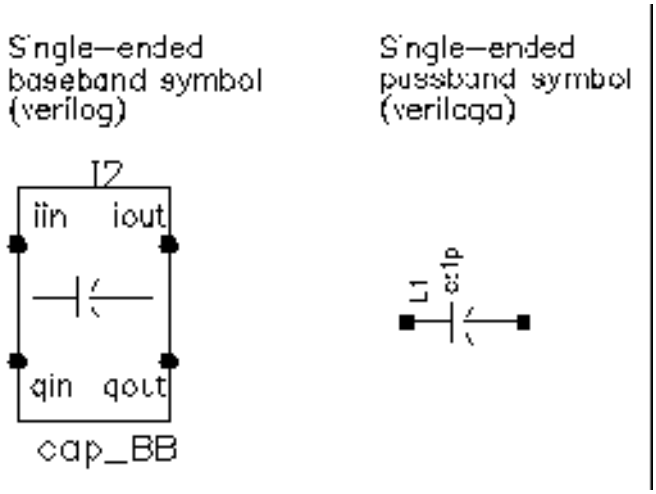
$$\begin{bmatrix} i_{iin} \\ i_{qin} \end{bmatrix} = \sqrt{\frac{RI}{Rs}} \begin{bmatrix} i_{iout} \\ i_{qout} \end{bmatrix}$$

$$\begin{bmatrix} v_{iout} \\ v_{qout} \end{bmatrix} = \sqrt{\frac{RI}{Rs}} \begin{bmatrix} v_{iin} \\ v_{qin} \end{bmatrix}$$

cap_BB

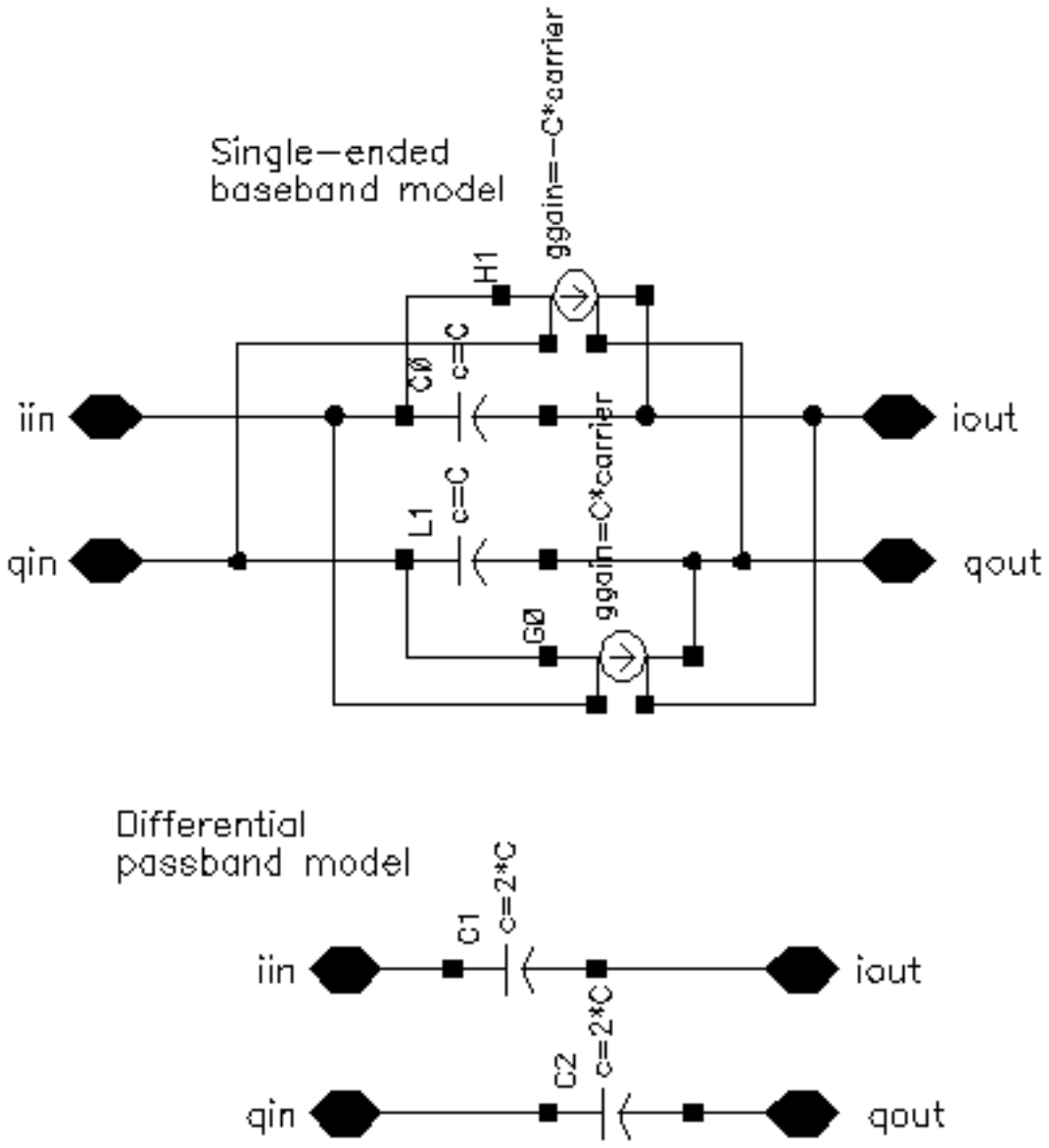
Figure 1-1 Circuit





The capacitor is the mathematical dual of the inductor. Figure 1-2 shows the baseband and differential passband capacitor models.

Figure 1-2 Capacitor Model



The parameters are:

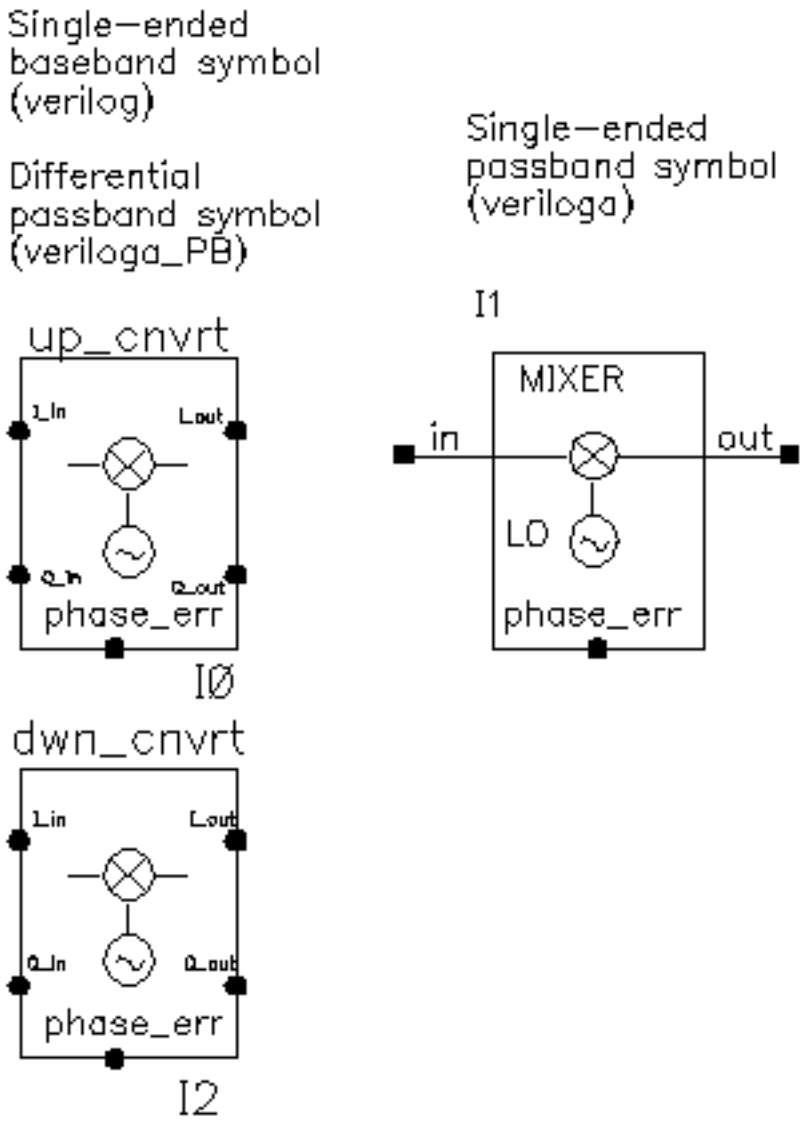
`cap`

`carrier`

dwn_cnvrt

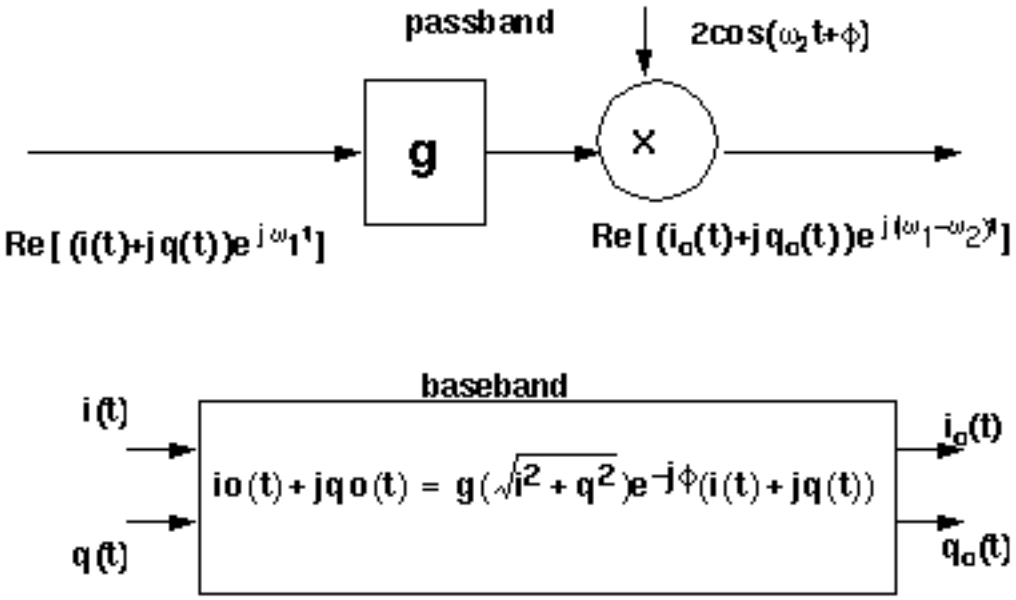
```
(baseband = dwn_cnvrt)
```

Figure 1-3 Baseband and Passband Mixer Models



`dwn_cnvrt` model is a baseband equivalent model of a mixer used to convert from RF to IF. There are some minor differences in the baseband models that depend on whether conversion is up or down. Figure 1-4 shows what the model does.

Figure 1-4 Calculations for dwn_cnvrt Mixer

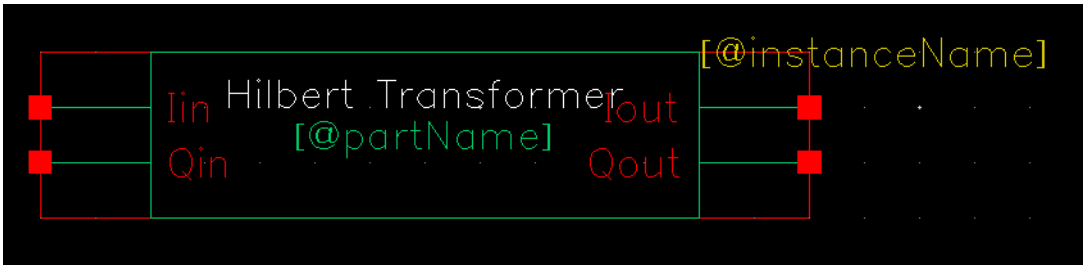


The parameters are:

`cmp`	Input power pt for phase point [dBm].
`cw`	{1,0,-1} for {cw, none, ccw}. Defines the direction of the phase shift. 1 for clockwise, 0 for no phase shift, -1 for counter clockwise.
`flo`	Local oscillator frequency.
`frf`	RF frequency.
`gain`	Available power gain [dB].
`IP3`	Input referenced IP3 [dbm].
`nf`	Noise figure.
`pscp`	|radians|@cmp. Defines the absolute value of the output phase shift at the 1dB compression point for power amplifiers. This is the phase shift at an arbitrary output power level for some models.
`psinf`	|radians|@ big input. Defines the absolute value of the output phase shift as input power goes to infinity (if it could go to infinity).
`rin`	Input resistance.
`rout`	Output resistance.

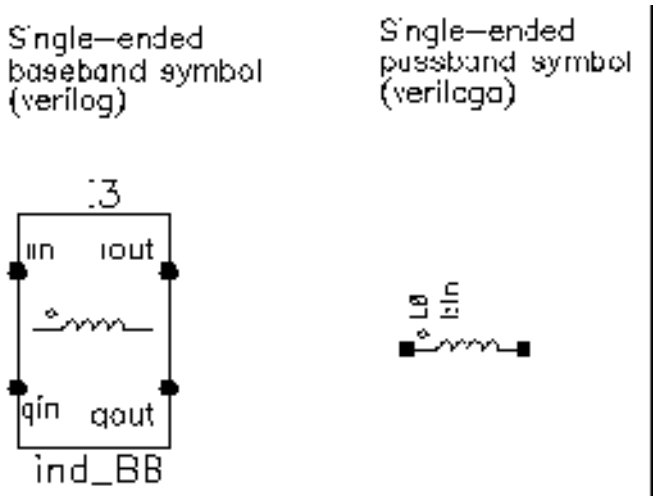
`shp`	AM/PM sharpness. Determines how fast the phase shift occurs with increasing input power. A larger number delays the shift but makes the shift rise faster as a function of input signal level.

HilbertTr_BB



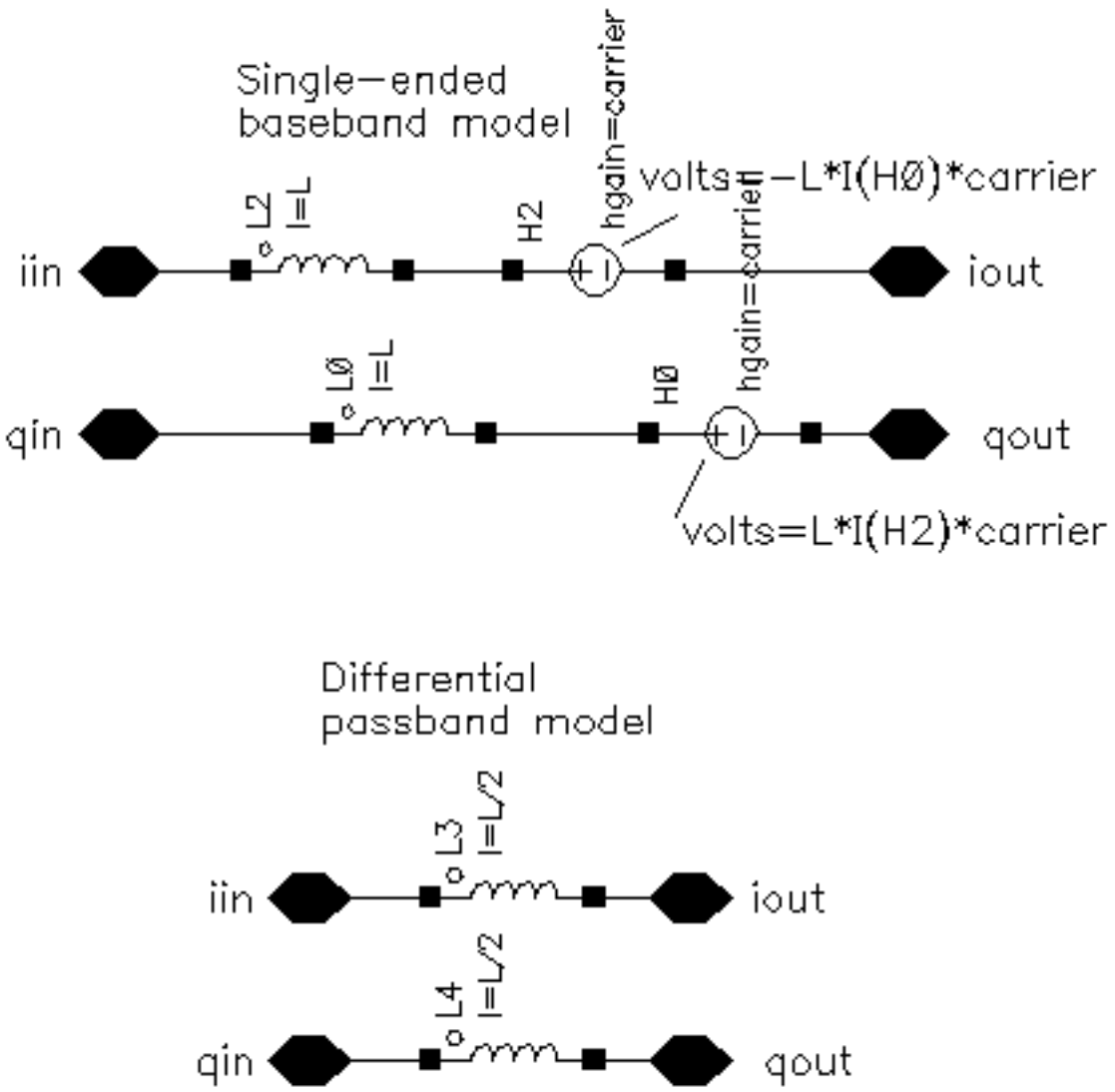
ind_BB

Figure 1-5 Circuit



The baseband inductor model requires one additional parameter besides the inductance, the carrier frequency. Figure 1-6 shows equivalent schematics of the baseband and differential passband inductor models. The inductor models are noiseless.

Figure 1-6 Inductor Model



The parameters are:

`carrier`

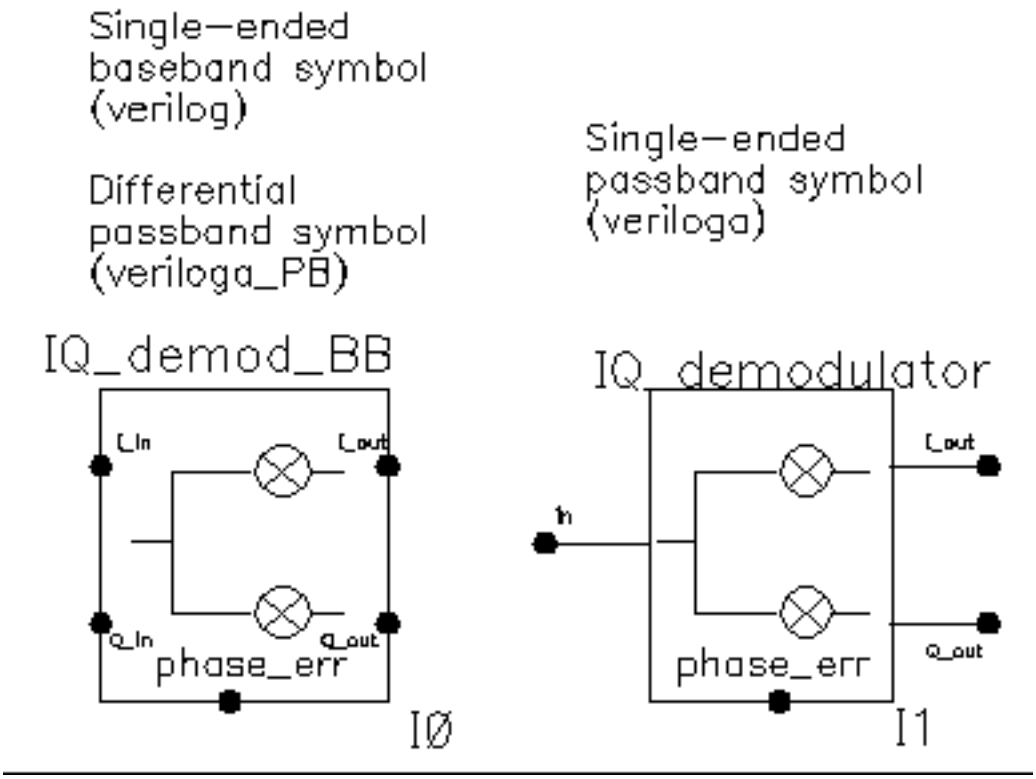
`ind`

IQ_demod_BB

(IQ Demodulator)

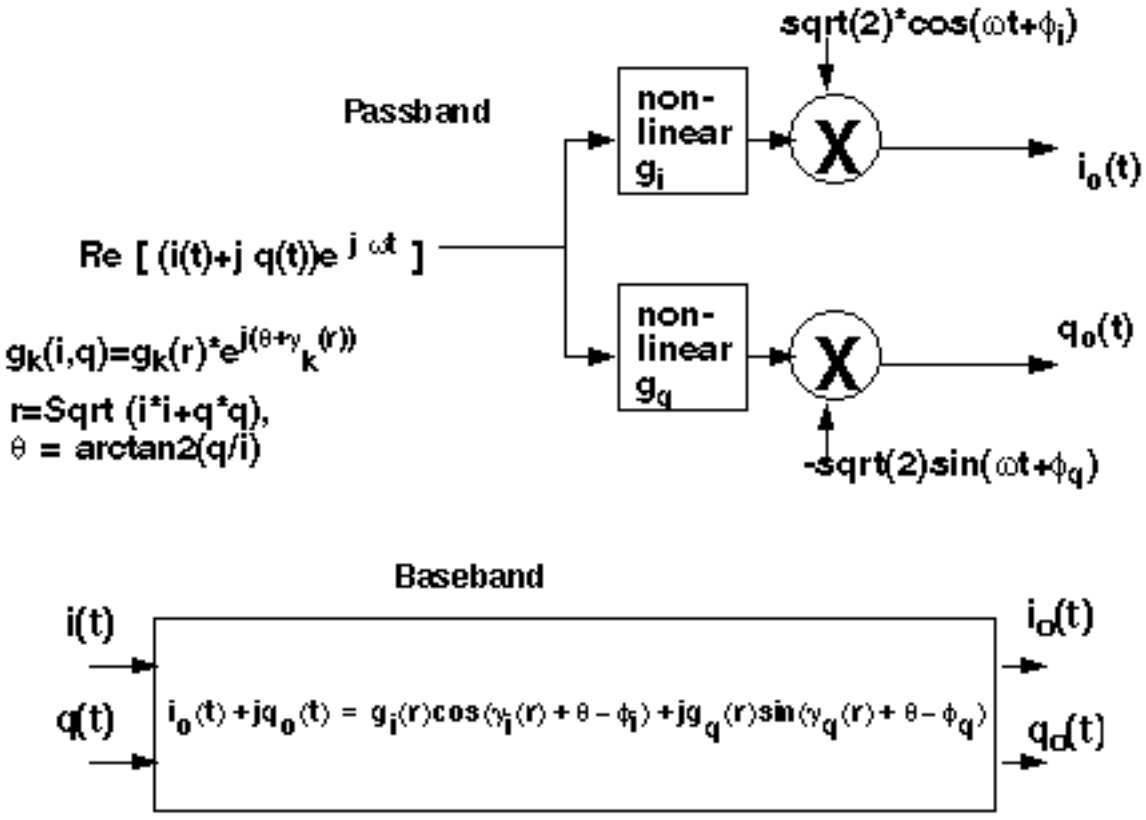
```
(baseband = IQ_demod_BB)
```

Figure 1-7 Baseband and Passband IQ Demodulator Models



The `IQ_demod_BB` converts RF (or IF) to baseband. Figure 1-8 shows exactly what the passband demodulator model does. The parameters are like those in the modulator blocks except saturation is specified by input referred IP3 instead of by 1 dB compression point. IP3 was chosen over the 1 dB compression point for specifying saturation because the demodulator usually lies in the receive path and receiver blocks are usually specified with IP3.

Figure 1-8 IQ Demodulator Calculations



The parameters are:

`I_cmp`	Input power point for phase point [dBm].
`I_cw`	Determines the direction of the phase shift. The phase shift is only in one direction. +1 means counter-clockwise, -1 means clockwise, and 0 means no phase shift (no am/pm conversion). {1,0,-1} for {cw, none, ccw}.
`I_gain`	Available power gain [dB].
`I_IP3`	Input referenced IP3 [dbm].
`I_pscp`	Output phase shift at cmp [radians]. I-radians@I_cmp.
`I_psinf`	Output phase shift as the input power goes to infinity. I-radians@big I-input.

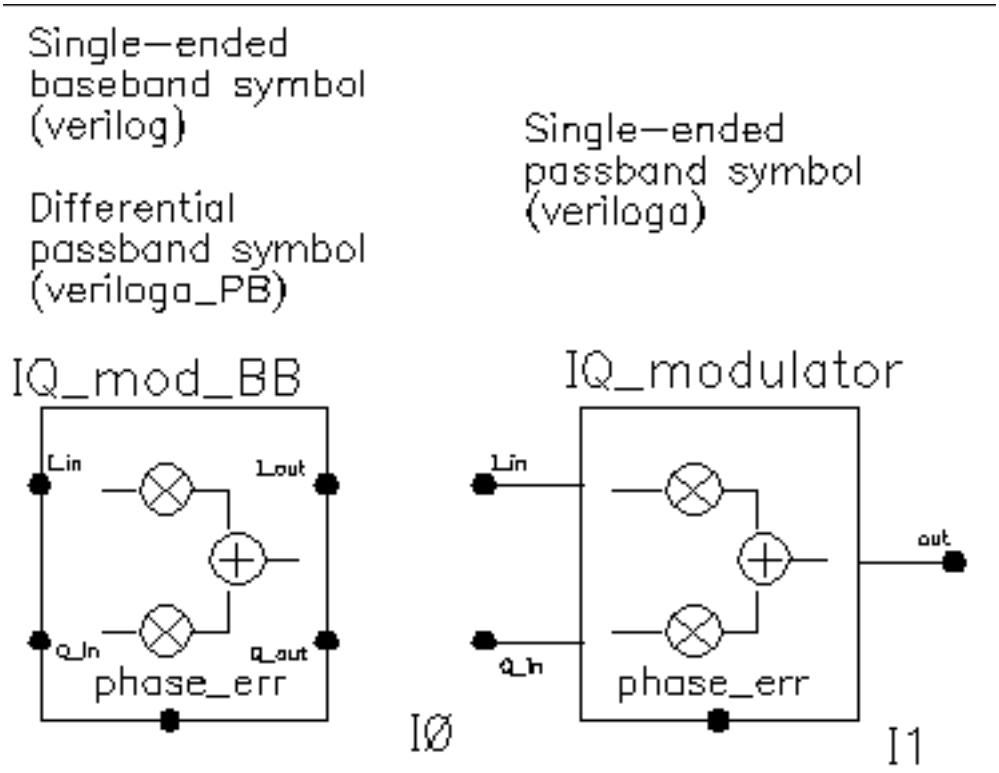
`I_shp`	Determines how fast the phase shift occurs with increasing input power. A larger number delays the shift but makes the shift rise faster as a function of input signal level. I-sharpness factor.
`nf`	Noise figure [dB].
`Q_cmp`	Input power point for phase point [dBm].
`Q_cw`	Determines the direction of the phase shift. The phase shift is only in one direction. +1 means counter-clockwise, -1 means clockwise, and 0 means no phase shift (no am/pm conversion). {1,0,-1} for {cw, none, ccw}.
`Q_gain`	Voltage gain [dB].
`Q_IP3`	Input referenced IP3 [dBm].
`Q_pscp`	Output phase shift at cmp [radians]. Q-radians@Q_cmp.
`Q_psinf`	Output phase shift as the input power goes to infinity.
`Q_shp`	Determines how fast the phase shift occurs with increasing input power. A larger number delays the shift but makes the shift rise faster as a function of input signal level. Q-sharpness factor.
`rin`	Input resistance.
`rout`	Output resistance.

IQ_mod_BB

(IQ Modulator Models)

`(baseband = IQ_mod_BB)`

Figure 1-9 Baseband and Passband IQ Modulator Models



The `IQ_mod_BB` converts baseband signals to RF or IF. Figure 1-10 summarizes exactly what the passband IQ modulator model does. The only difference between the baseband and passband models is carrier suppression. The non-linear functions, g_i and g_q, are specified by their available power gain and 1dB compression points just as in the power amplifier. The functions γ_i and γ_q characterize AM/PM effects in each mixer and are specified by the same parameters that specify power amplifier AM/PM conversion. Because noise is always added at the input, and the input is at baseband in this case, the noise sources are not doubled as they are in the power amplifier or LNA models. Noise figure is defined with reference to one input. Noise is injected at both inputs but the noise injected at just one input alone produces the specified noise figure. Thus, the noise figure parameter should be interpreted as noise figure per input. This model also includes a parameter called `quadrature error` which specifies how far away the two local oscillators signals are from being exactly in quadrature.

`Phase error` is the voltage on the phase error pin. The phase error pin has a fixed noiseless resistive input impedance of 50 ohms. The phase error pin can be used to introduce a dynamic phase error or phase noise. Phase noise can be fed into the phase error pin from a phase-domain PLL model or from a Port. Noise in Port models can be specified either by the internal resistance or by a data file that tabulates a power spectral density. The phase error pin can also be driven by a ramp or circular integrator output to model a frequency error between the incoming carrier and local oscillator.

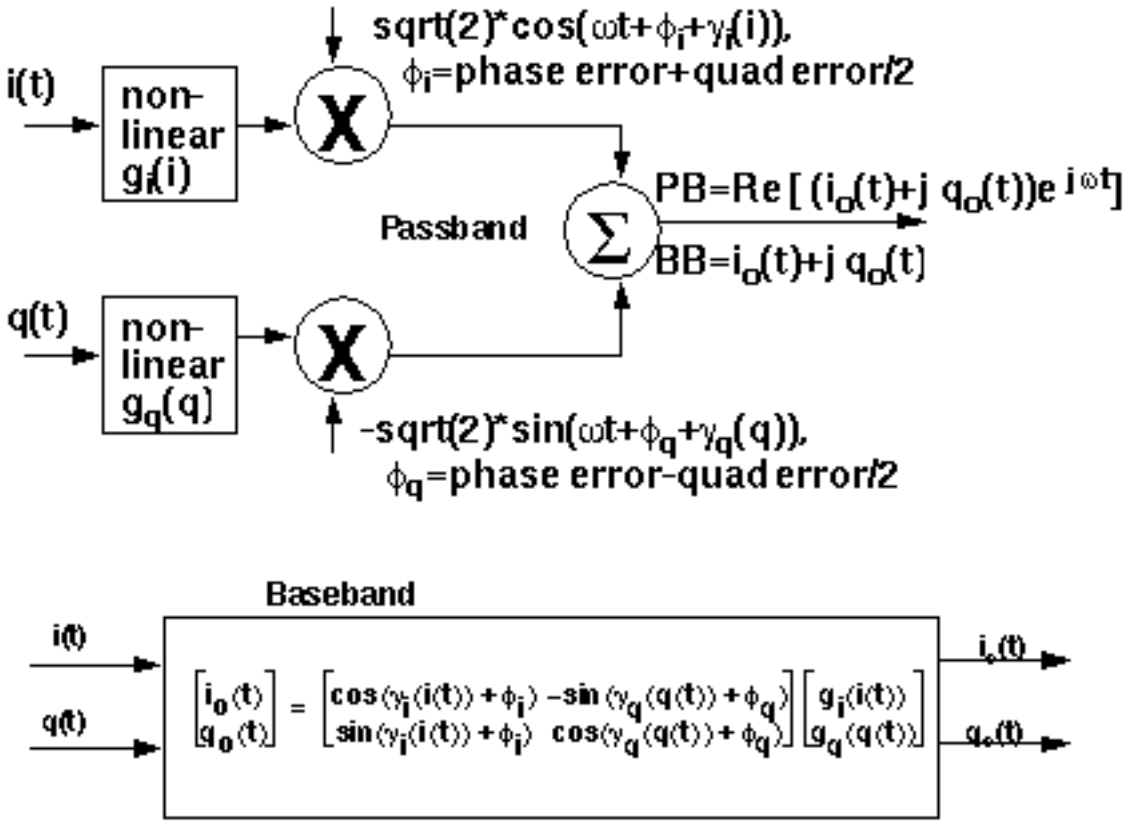
The following parameters specify the IQ modulator. The available power gain and one dB compression point are explained first. The effects of the phase_error pin and the quadrature error parameter are discussed at the end of this section.

The parameters are:

`I_cmp`	Input power point for phase point [dBm].
`I_cpdb`	I db compression point, referred to the output. I-output 1dB CP [dBm].
`I_cw`	Determines the direction of the phase shift. The phase shift is only in one direction. +1 means counter-clockwise, -1 means clockwise, and 0 means no phase shift (no am/pm conversion). {1,0,-1} for {cw, none, ccw}.
`I_gain`	Available power gain [dB].
`I_pscp`	Output phase shift at cmp [radians]. I-radians@I_cmp.
`I_psinf`	Output phase shift as the input power goes to infinity. I-radians@big I-input.
`I_shp`	Determines how fast the phase shift occurs with increasing input power. A larger number delays the shift but makes the shift rise faster as a function of input signal level. I-sharpness factor.
`nf`	Noise figure [dB].
`Q_cmp`	Input power point for phase point [dBm].
`Q_cw`	Determines the direction of the phase shift. The phase shift is only in one direction. +1 means counter-clockwise, -1 means clockwise, and 0 means no phase shift (no am/pm conversion). {1,0,-1} for {cw, none, ccw.
`Q_gain`	Voltage gain [dB].
`Q_IP3`	Input referenced IP3 [dBm].
`Q_pscp`	Output phase shift at cmp [radians]. Q-radians@Q_cmp.

`Q_psinf`	Output phase shift as the input power goes to infinity.
`Q_shp`	Determines how fast the phase shift occurs with increasing input power. A larger number delays the shift but makes the shift rise faster as a function of input signal level. Q-sharpness factor.
`rin`	Input resistance.
`rout`	Output resistance.

Figure 1-10 IQ Modulator Calculations

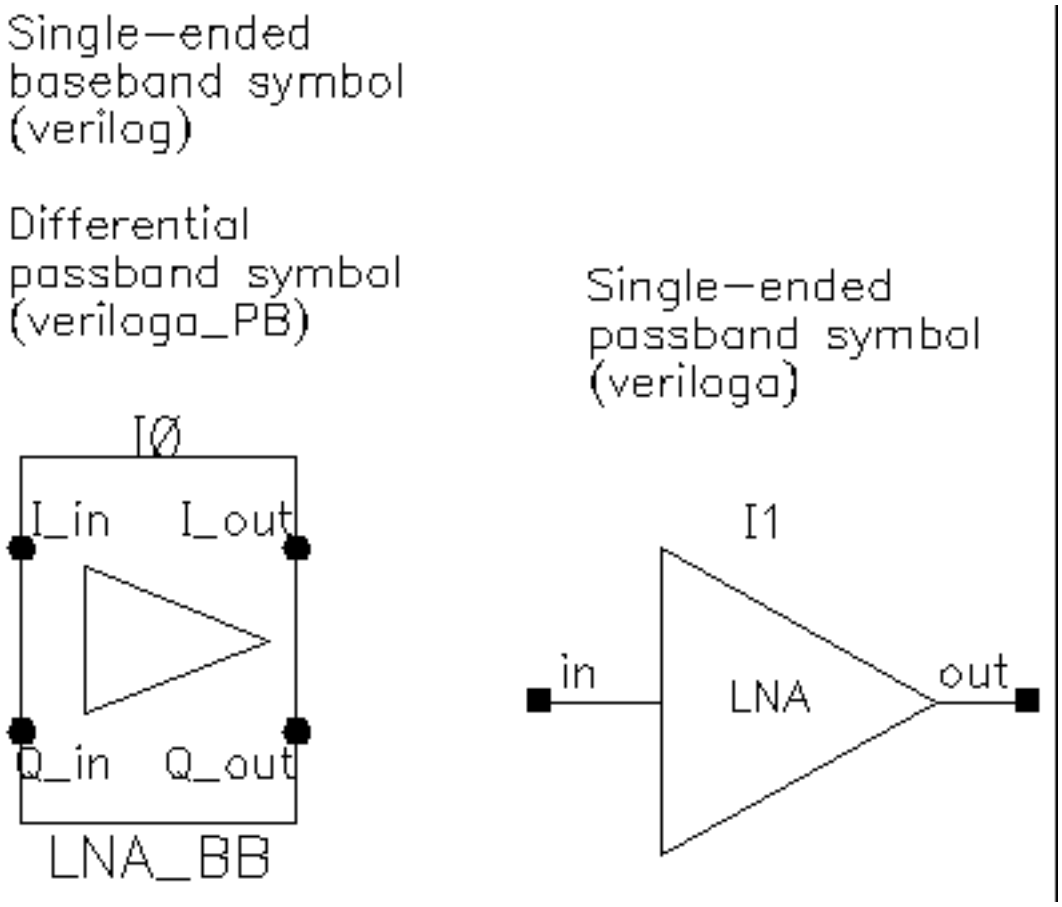


LNA_BB

(Low Noise Amplifier Models)

```
(baseband = LNA_BB)
```

Figure 1-11 Baseband and Passband Power Amplifier Models



The parameters are:

`cmp`	cmp [dBm]. Output power level where the next parameter is defined.
`cw`	{1,0,-1} for {cw, none, ccw}. Defines the direction of the phase shift. 1 for clockwise, 0 for no phase shift, -1 for counter clockwise.
`gain`	Available power gain [dB].
`IP3`	Input referred IP3 [dBm].

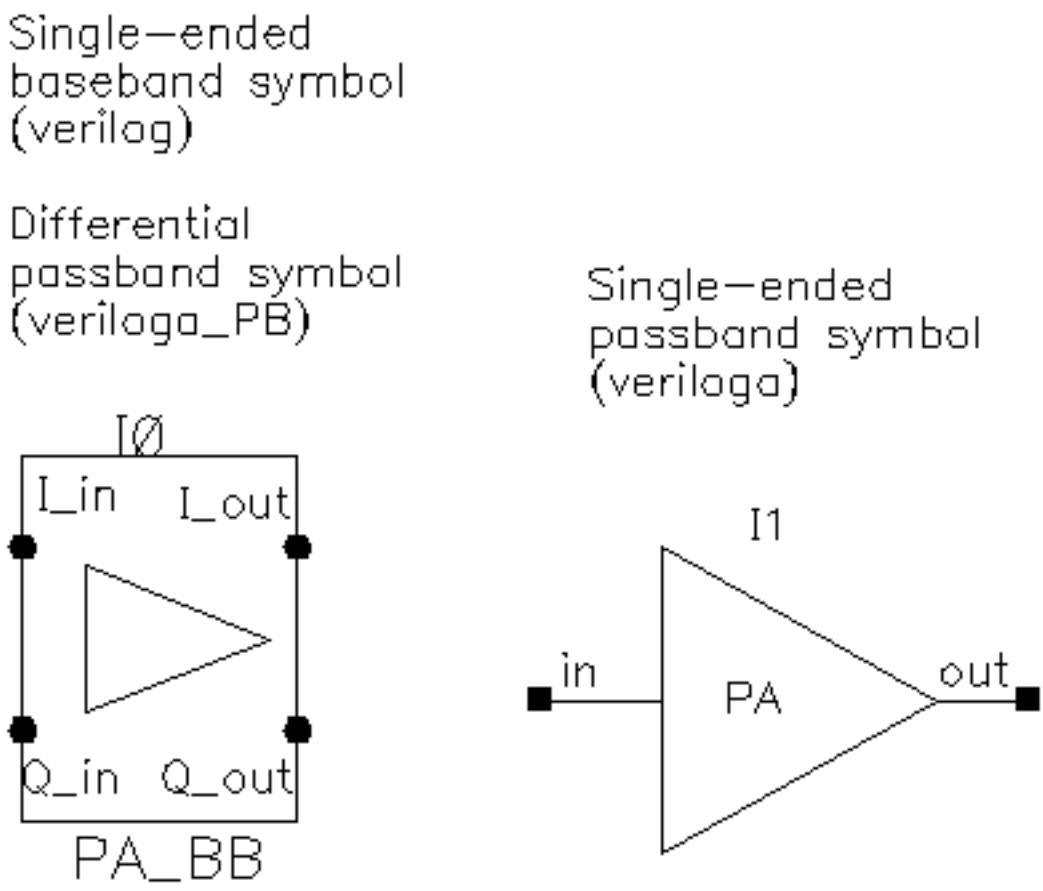
`nf`	Noise figure [dB].
`pscp`	|radians|@cmp. Defines the absolute value of the output phase shift at the 1dB compression point for power amplifiers. This is the phase shift at an arbitrary output power level for some models.
`psinf`	|radians|@ big input. Defines the absolute value of the output phase shift as input power goes to infinity (if it could go to infinity).
`rin`	Input resistance.
`rout`	Output resistance.
`shp`	AM/PM sharpness. Defines how steep the output phase shift changes are with respect to input power.

PA_BB

(Power Amplifier Model)

```
(baseband = PA_BB)
```

Figure 1-12 Baseband and Passband Power Amplifier Models



The following parameters specify the power amplifier model.

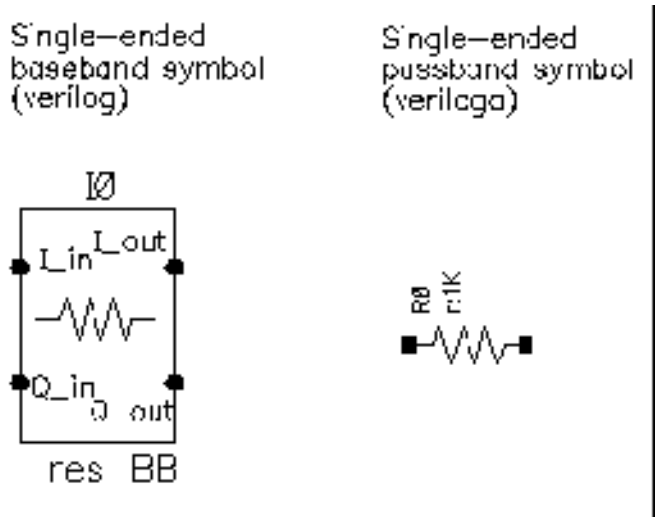
The parameters are:

`cpdb`	Output 1db cp [dBm].
`cw`	{1,0,-1} for {cw, none, ccw}. Defines the direction of the phase shift. 1 for clockwise, 0 for no phase shift, -1 for counter clockwise.
`gain`	Available power gain.

`nf`	Noise figure.
`pscp`	|radians|@cmp. Defines the absolute value of the output phase shift at the 1dB compression point for power amplifiers. This is the phase shift at an arbitrary output power level for some models.
`psinf`	|radians|@ big input. Defines the absolute value of the output phase shift as input power goes to infinity (if it could go to infinity).
`rin`	Input resistance.
`rout`	Output resistance.
`shp`	AM/PM sharpness. Defines how steep the output phase shift changes are with respect to input power.

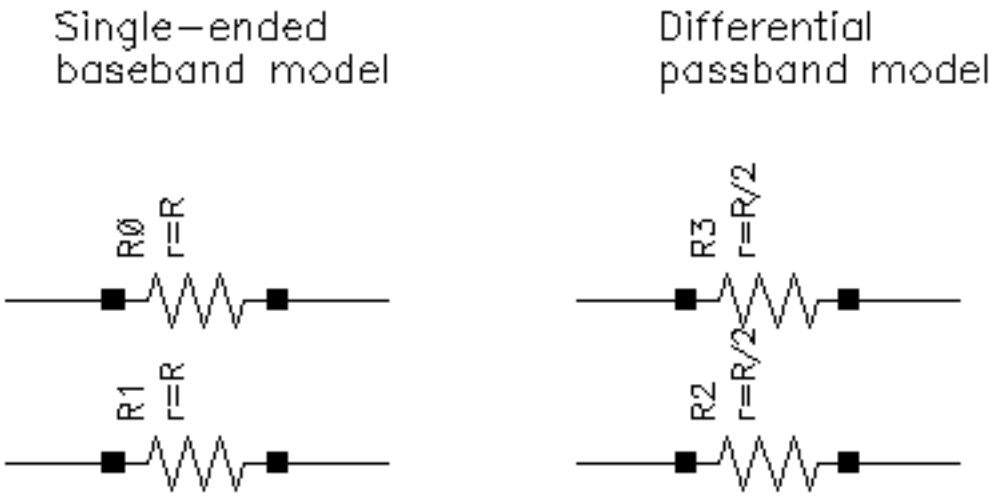
res_BB

Figure 1-13 Circuit



Besides the resistance, the baseband resistor model has a parameter for turning its thermal noise on or off. The baseband resistor is intended for use at a passband node because it's noise is doubled. (This was discussed in the section entitled "Relationship between baseband and passband noise"). Figure 1-14 shows the symbol, baseband, and passband models. The total noise in the differential passband resistor model equals the noise in one resistor of R Ohms.

Figure 1-14 Resistor Model



The parameters are:

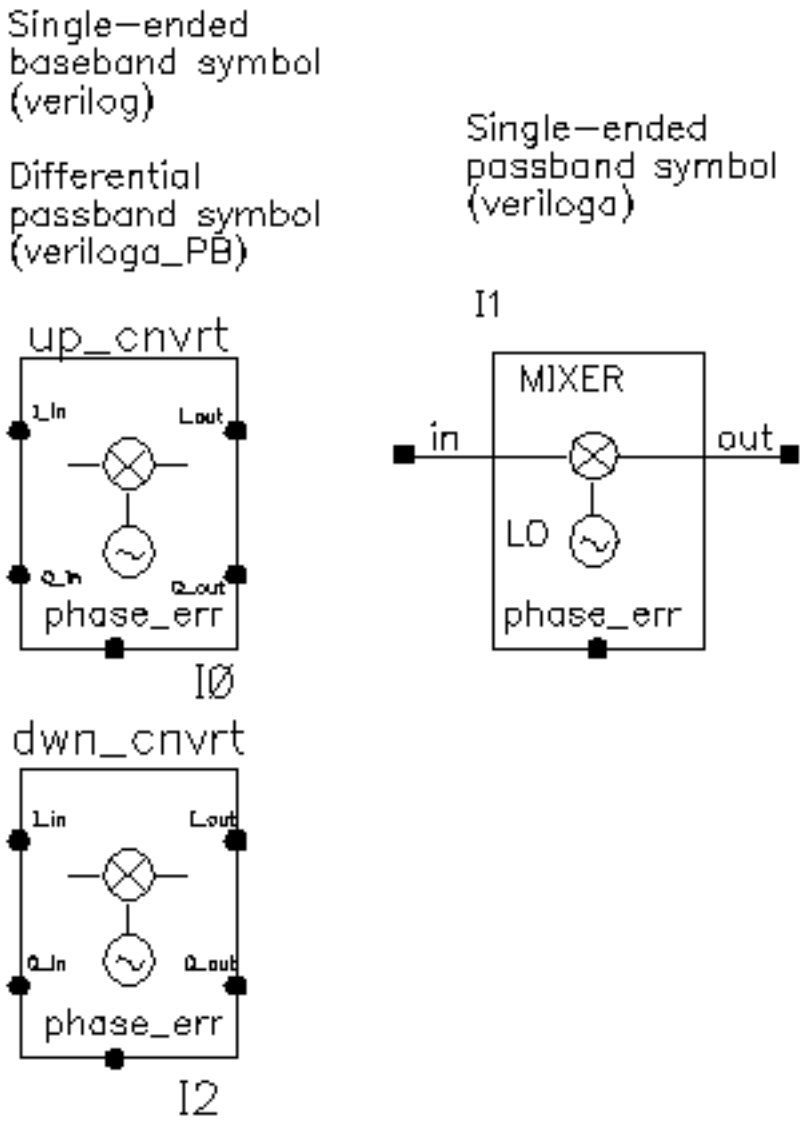
`res`

`noise_switch` 0 for noiseless, 1 for noisy.

up_cnvrt

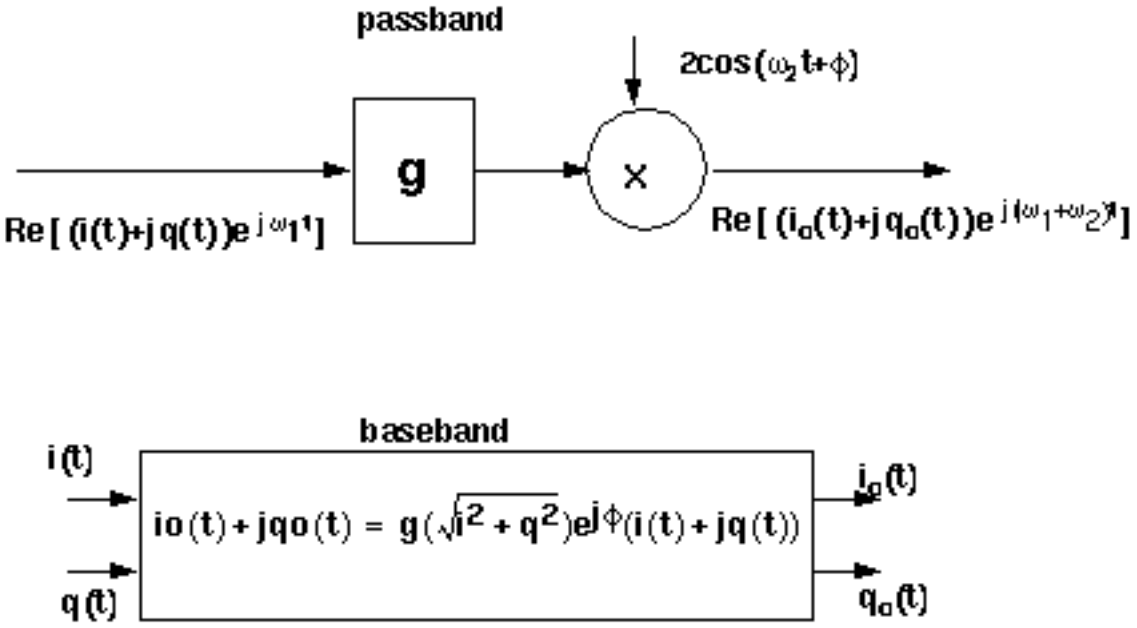
```
(baseband = up_cnvrt)
```

Figure 1-15 Baseband and Passband Mixer Models



`up_cnvrt` model is a baseband equivalent model of a mixer used to convert from IF to RF. There are some minor differences in the baseband models that depend on whether conversion is up or down. Figures 1-16 shows what the model does.

Figure 1-16 Calculations for up_cnvrt Mixer



The parameters are:

`cmp`	Input power pt for phase point [dBm].
`cw`	{1,0,-1} for {cw, none, ccw}. Defines the direction of the phase shift. 1 for clockwise, 0 for no phase shift, -1 for counter clockwise.
`gain`	Available power gain [dB].
`IP3`	Input referenced IP3 [dbm].
`nf`	Noise figure.
`pscp`	|radians|@cmp. Defines the absolute value of the output phase shift at the 1dB compression point for power amplifiers. This is the phase shift at an arbitrary output power level for some models.
`psinf`	|radians|@ big input. Defines the absolute value of the output phase shift as input power goes to infinity (if it could go to infinity).
`rin`	Input resistance.
`rout`	Output resistance.

`shp`	AM/PM sharpness. Determines how fast the phase shift occurs with increasing input power. A larger number delays the shift but makes the shift rise faster as a function of input signal level.

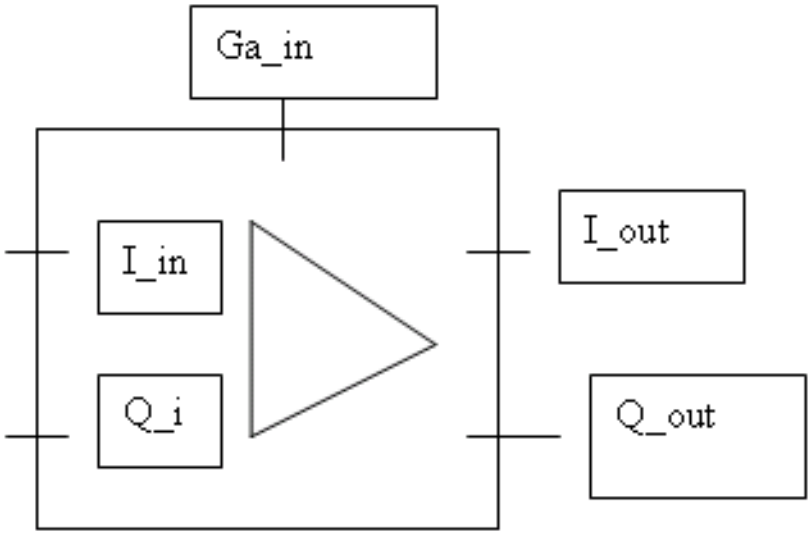
VGA_BB

(Variable Gain Amplifier Model)

```
(baseband = VGA_BB)
```

Only the Baseband view is available.

Figure 1-17 Variable Gain Amplifier Model



The parameters are:

`cpdb`	db compression point [dbm], referred to the output.
`cw`	{1,0,-1} for {cw, none, ccw}. Defines the direction of the phase shift. 1 for clockwise, 0 for no phase shift, -1 for counter clockwise.
`gpv`	Voltage gain per volt on the G_in pin.
`pscp`	|radians|@cmp. Defines the absolute value of the output phase shift at the 1dB compression point for power amplifiers. This is the phase shift at an arbitrary output power level for some models.
`psinf`	|radians|@ big input. Defines the absolute value of the output phase shift as input power goes to infinity (if it could go to infinity).
`rin`	Input resistance.
`rout`	Output resistance.

`shp`	AM/PM sharpness. Determines how fast the phase shift occurs with increasing input power. A larger number delays the shift but makes the shift rise faster as a function of input signal level.

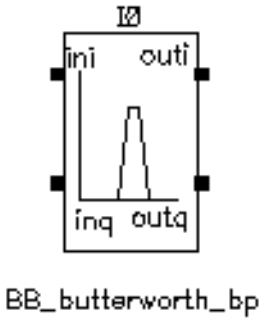
Butterworth_filters Category

The cells in the `Butterworth_filters` category are:

- BB_butterworth_bp
- BB_butterworth_bp_laplace
- BB_butterworth_bs
- BB_butterworth_bs_laplace
- BB_butterworth_hp
- BB_butterworth_hp_laplace
- BB_butterworth_lp
- BB_butterworth_lp_laplace
- butterworth_bp
- butterworth_bs
- butterworth_hp
- butterworth_lp

BB_butterworth_bp

Figure 1-18 BB_butterworth_bp



For information about the filter parameters, see “BB_butterworth and BB_chebyshev Filter Parameters” on page 52.

BB_butterworth and BB_chebyshev Filter Parameters

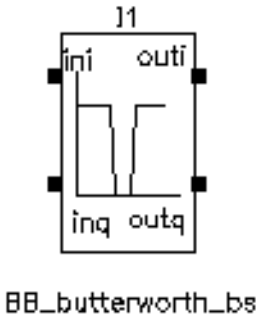
The parameters for the BB_butterworth and BB_chebyshev filters are:

`bw`	Relative bandwidth for bankpass or bandstop filter [Hz].
`f0`	Center frequency for bankpass or bandstop filter [Hz].
`fc`	Filter cutoff frequency for lowpass and highpass filter [Hz].
`fcr`	Carrier frequency.
`loss`	Filter insertion loss [dB].
`N`	Filter order (>=2). Must be defined as ‘define N.
`r1`	Reference impedance at port 1 [ohm].
`r2`	Reference impedance at port 2 [ohm].

BB_butterworth_bp_laplace

BB_butterworth_bs

Figure 1-19 BB_butterworth_bs

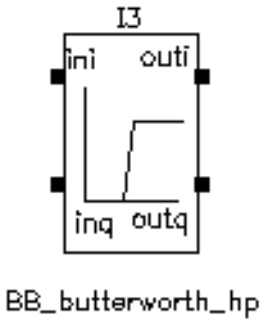


For information about the filter parameters, see “BB_butterworth and BB_chebyshev Filter Parameters” on page 52.

BB_butterworth_bs_laplace

BB_butterworth_hp

Figure 1-20 BB_butterworth_hp

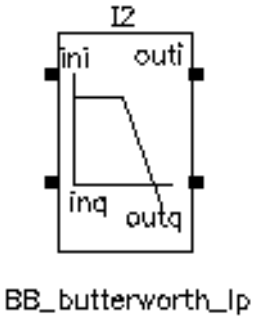


For information about the filter parameters, see "BB_butterworth and BB_chebyshev Filter Parameters" on page 52.

BB_butterworth_hp_laplace

BB_butterworth_lp

Figure 1-21 BB_butterworth_lp

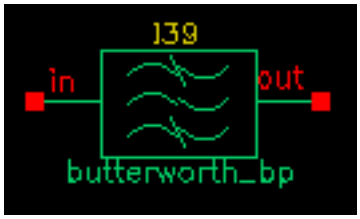


For information about the filter parameters, see "BB_butterworth and BB_chebyshev Filter Parameters" on page 52.

BB_butterworth_lp_laplace

butterworth_bp

Figure 1-22 butterworth_bp



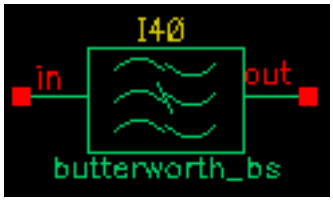
The parameters are:

`alpha`	Filter attenuation at cutoff [dB].
`bw`	Relative bandwidth for bankpass or bandstop filter [Hz].
`f0`	Center frequency for bankpass or bandstop filter [Hz].
`fc`	Filter cutoff frequency for lowpass and highpass filter [Hz].
`loss`	Filter insertion loss [dB].
`N`	Filter order (>=2). Must be defined as ‘define N.
`r1`	Reference impedance at port 1 [Ohm].
`r2`	Reference impedance at port 2 [Ohm].

For more information, see “Butterworth and Chebyshev Filter Supporting Information” on page 64.

butterworth_bs

Figure 1-23 butterworth_bs



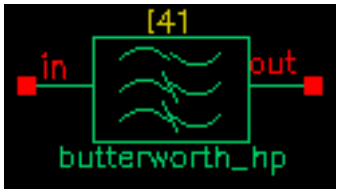
The parameters are:

`alpha`	Filter attenuation at cutoff [dB].
`bw`	Relative bandwidth for bankpass or bandstop filter [Hz].
`f0`	Center frequency for bankpass or bandstop filter [Hz].
`fc`	Filter cutoff frequency for lowpass and highpass filter [Hz].
`loss`	Filter insertion loss [dB].
`N`	Filter order (>=2). Must be defined as ‘define N.
`r1`	Reference impedance at port 1 [Ohm].
`r2`	Reference impedance at port 2 [Ohm].

For more information, see “Butterworth and Chebyshev Filter Supporting Information” on page 64.

butterworth_hp

Figure 1-24 butterworth_hp



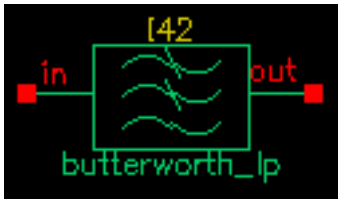
The parameters are:

`alpha`	Filter attenuation at cutoff [dB].
`bw`	Relative bandwidth for bankpass or bandstop filter [Hz].
`f0`	Center frequency for bankpass or bandstop filter [Hz].
`fc`	Filter cutoff frequency for lowpass and highpass filter [Hz].
`loss`	Filter insertion loss [dB].
`N`	Filter order (>=2). Must be defined as ‘define N.
`r1`	Reference impedance at port 1 [Ohm].
`r2`	Reference impedance at port 2 [Ohm].

For more information, see “Butterworth and Chebyshev Filter Supporting Information” on page 64.

butterworth_lp

Figure 1-25 butterworth_lp



The parameters are:

`alpha`	Filter attenuation at cutoff [dB].
`bw`	Relative bandwidth for bankpass or bandstop filter [Hz].
`f0`	Center frequency for bankpass or bandstop filter [Hz].
`fc`	Filter cutoff frequency for lowpass and highpass filter [Hz].
`loss`	Filter insertion loss [dB].
`N`	Filter order (>=2). Must be defined as ‘define N.
`r1`	Reference impedance at port 1 [Ohm].
`r2`	Reference impedance at port 2 [Ohm].

For more information, see “Butterworth and Chebyshev Filter Supporting Information” on page 64.

Butterworth and Chebyshev Filter Supporting Information

Filter properties are specified in the frequency domain, but it is not easy for Spectre RF to process frequency-domain data. Spectre RF simulation requires a large signal, time-domain model to simulate filter behavior.

As part of the RF AHDL library, filters are implemented using a network synthesis technique which consists of the following two steps:

1. Calculate the normalized low-pass filter prototype, which consists of serial inductors and parallel capacitors
2. Perform frequency transformation and scaling to synthesize the frequency responses of the filter type

The synthesized model contains many inductors and capacitors. They are implemented using the integral and differential functions of the Verilog-A language. Insertion loss is added using the S-parameter network technique. This network essentially dampens the signal flow by the specified insertion loss value.

In the current implementation of the Verilog-A language, the order and internal states of the filter cannot be dynamically allocated. You must use the `'define` directive in the Verilog-A source code to specify the order. Use S-parameters to test the filters because S-parameters capture the input/output impedance matching.

For example, the Butterworth bandpass filter, `butterworth_bp`, has the following module declaration:

```
module butterworth_bp(t1, t2);
    inout t1, t2;
    electrical in, out;
    parameter real r1 = 50 from (0:inf);
    parameter real r2 = 50 from (0:inf);
    parameter real f0 = 1e9 from (0:inf);
    parameter real bw = 0.10 from (0:0.5);
    parameter real fc = 1e9 from (0:inf);
    parameter real loss = 0 from [0:inf);
```

where `t1` and `t2` are the input and output nodes, respectively.

The parameters are:

`bw`	Relative frequency for bandpass or bandstop filter [Hz].
`f0`	Center frequency for bandpass or bandstop filter [Hz].
`fc`	Corner frequency (3 dB point) for low-pass and high-pass filter [Hz].

`loss`	Insertion loss [dB].
`r1`	Input impedance [Ω].
`r2`	Output impedance [Ω].

Figure 1-26 is the simple schematic used to test the filter. Two ports are used to obtain the S-parameters.

Figure 1-26 Schematic for Testing Filter Models

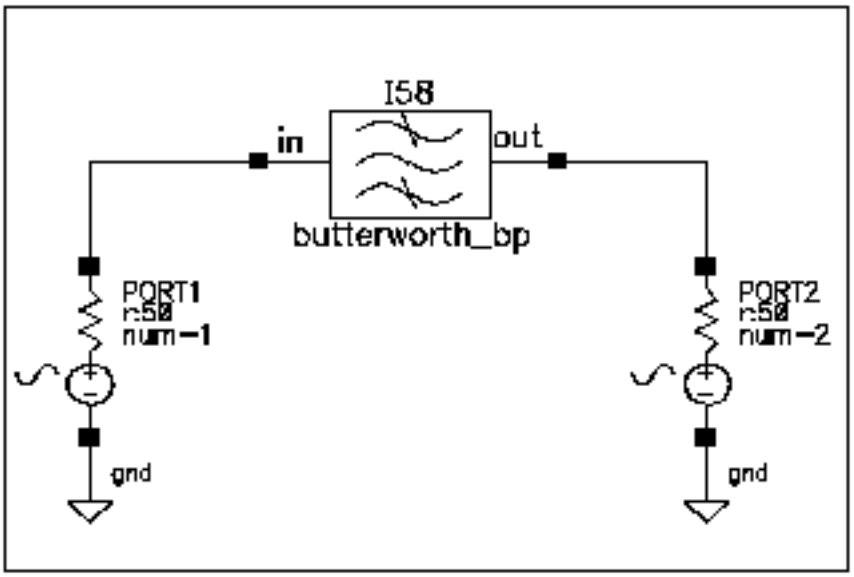
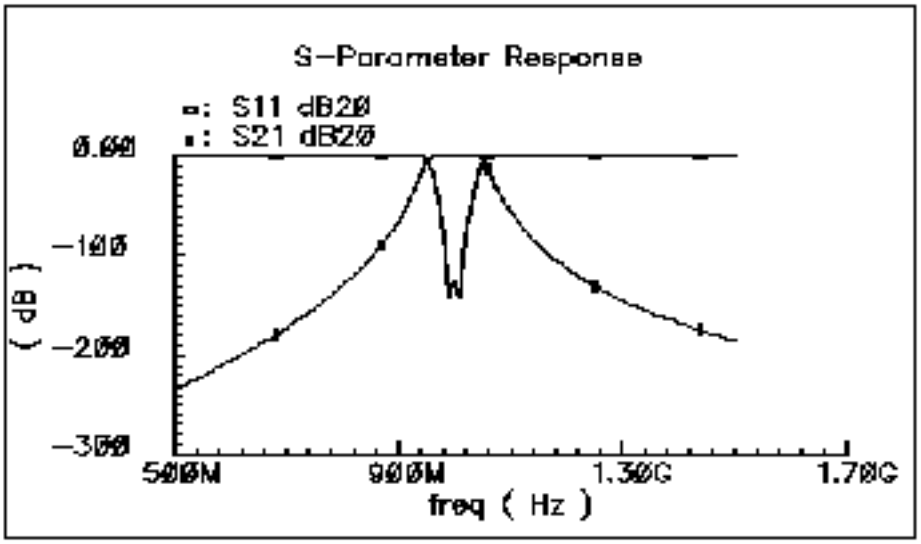


Figure 1-27 shows the calculated S-parameters of this Butterworth bandpass filter, which has a center frequency of 1 GHz and a relative bandwidth of 10 percent. The order of this specific filter is 10.

Figure 1-27 S-Parameters of a Butterworth Filter



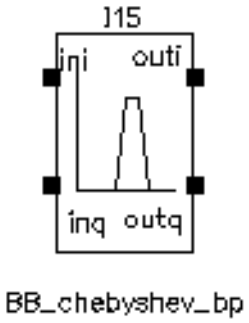
Chebyshev_filters Category

The cells in the `Chebyshev_filters` category are:

- BB_chebyshev_bp
- BB_chebyshev_bp_laplace
- BB_chebyshev_bs
- BB_chebyshev_bs_laplace
- BB_chebyshev_hp
- BB_chebyshev_hp_laplace
- BB_chebyshev_lp
- BB_chebyshev_lp_laplace
- chebyshev_bp
- chebyshev_bs
- chebyshev_hp
- chebyshev_lp

BB_chebyshev_bp

Figure 1-28 BB_chebyshev_bp

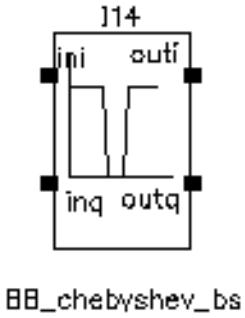


For information about the filter parameters, see "BB_butterworth and BB_chebyshev Filter Parameters" on page 52.

BB_chebyshev_bp_laplace

BB_chebyshev_bs

Figure 1-29 BB_chebyshev_bs

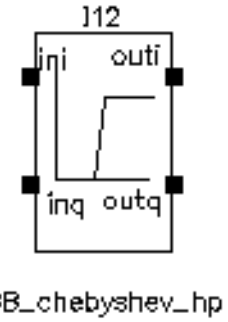


For information about the filter parameters, see “BB_butterworth and BB_chebyshev Filter Parameters” on page 52.

BB_chebyshev_bs_laplace

BB_chebyshev_hp

Figure 1-30 BB_chebyshev_hp

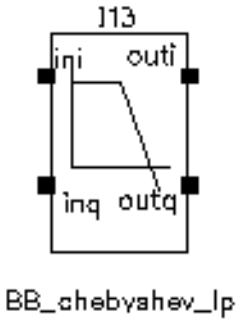


For information about the filter parameters, see "BB_butterworth and BB_chebyshev Filter Parameters" on page 52.

BB_chebyshev_hp_laplace

BB_chebyshev_lp

Figure 1-31 BB_chebyshev_lp

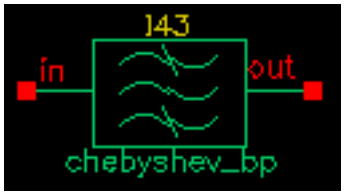


For information about the filter parameters, see "BB_butterworth and BB_chebyshev Filter Parameters" on page 52.

BB_chebyshev_lp_laplace

chebyshev_bp

Figure 1-32 chebyshev_bp



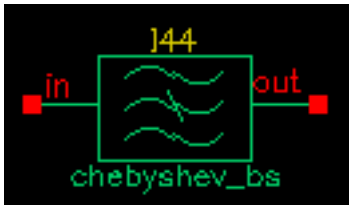
The parameters are:

`alpha`	Filter attenuation at cutoff [dB].
`bw`	Relative bandwidth for bankpass or bandstop filter [Hz].
`f0`	Center frequency for bankpass or bandstop filter [Hz].
`fc`	Filter cutoff frequency for lowpass and highpass filter [Hz].
`loss`	Filter insertion loss [dB].
`N`	Filter order (>=2). Must be defined as ‘define N.
`r1`	Reference impedance at port 1 [Ohm].
`r2`	Reference impedance at port 2 [Ohm].

For more information, see “Butterworth and Chebyshev Filter Supporting Information” on page 64.

chebyshev_bs

Figure 1-33 chebyshev_bs



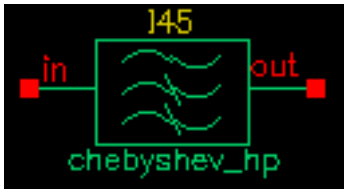
The parameters are:

`alpha`	Filter attenuation at cutoff [dB].
`bw`	Relative bandwidth for bankpass or bandstop filter [Hz].
`f0`	Center frequency for bankpass or bandstop filter [Hz].
`fc`	Filter cutoff frequency for lowpass and highpass filter [Hz].
`loss`	Filter insertion loss [dB].
`N`	Filter order (>=2). Must be defined as ‘define N.
`r1`	Reference impedance at port 1 [Ohm].
`r2`	Reference impedance at port 2 [Ohm].

For more information, see “Butterworth and Chebyshev Filter Supporting Information” on page 64.

chebyshev_hp

Figure 1-34 chebyshev_hp



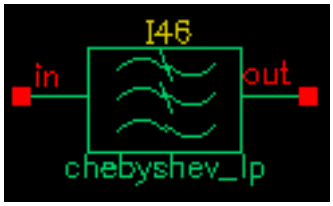
The parameters are:

`alpha`	Filter attenuation at cutoff [dB].
`bw`	Relative bandwidth for bankpass or bandstop filter [Hz].
`f0`	Center frequency for bankpass or bandstop filter [Hz].
`fc`	Filter cutoff frequency for lowpass and highpass filter [Hz].
`loss`	Filter insertion loss [dB].
`N`	Filter order (>=2). Must be defined as ‘define N.
`r1`	Reference impedance at port 1 [Ohm].
`r2`	Reference impedance at port 2 [Ohm].

For more information, see “Butterworth and Chebyshev Filter Supporting Information” on page 64.

chebyshev_lp

Figure 1-35 chebyshev_lp



The parameters are:

`alpha`	Filter attenuation at cutoff [dB].
`bw`	Relative bandwidth for bankpass or bandstop filter [Hz].
`f0`	Center frequency for bankpass or bandstop filter [Hz].
`fc`	Filter cutoff frequency for lowpass and highpass filter [Hz].
`loss`	Filter insertion loss [dB].
`N`	Filter order (>=2). Must be defined as ‘define N.
`r1`	Reference impedance at port 1 [Ohm].
`r2`	Reference impedance at port 2 [Ohm].

For more information, see “Butterworth and Chebyshev Filter Supporting Information” on page 64.

Measurements Category

The `measurement` category contains elements used to facilitate measurements and diagnostics. Elements in the `measurement` category can be used by both RF system designers and RF circuit designers.

This section also explains how to change the FIR filters inside the baseband signal generators.

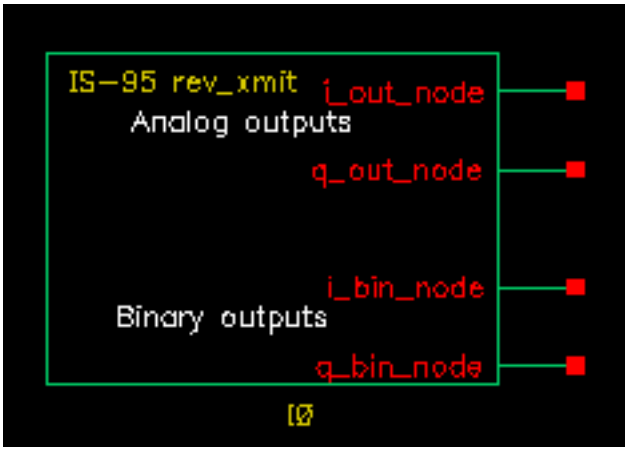
Note: All of the baseband signal sources generate digitally filtered signals. The baseband sources do not work with Spectre RF because the digital filters have hidden states.

The `measurement` category contains the following elements, discussed in the sections that follow.

- CDMA_reverse_xmit
- comms_instr
- eye_diagram_generator
- gmsk
- gsm_comms_instr
- GSM_xmtr
- instr_term
- offset_comms_instr
- phase_generator
- pi_over4_dqpsk
- polar_rect
- rect_polar

CDMA_reverse_xmit

(CDMA Signal Source)



The CDMA signal source (`CDMA_reverse_xmit`) generates a reverse-link (handset-to-base-station) IS-95 signal with the following characteristics

modulation	Offset QPSK.
symbol rate	1.2288 megasymbols per second.
sample rate	4.9152 megasamples per second.

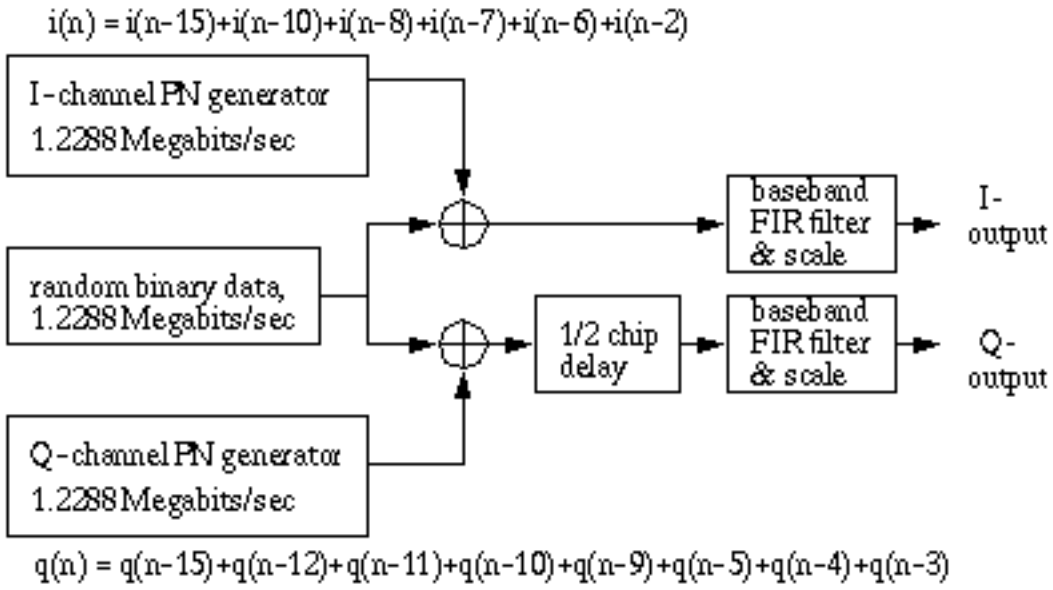
Two separate 16-bit pseudo-noise generators generate the *I* and *Q* spreading sequences operating at the sample rate.

The CDMA source

- Generates a random bit at the symbol rate
- Oversamples it by a factor of 4
- Spreads the bit with the *I* and *Q* spreading sequences
- Filters each sequence with a 48-tap FIR filter. The filter coefficients are the impulse response of a raised cosine filter.
- Generates a reverse-link (handset-to-base-station) IS-95 signal. The modulation is offset QPSK with a symbol rate of 1.2288 Mega-symbols per second and a sample rate of 4.9152 Megasamples per second. Two separate 16-bit pseudo-noise generators generate the *I* and *Q* spreading sequences operating at the sample rate.

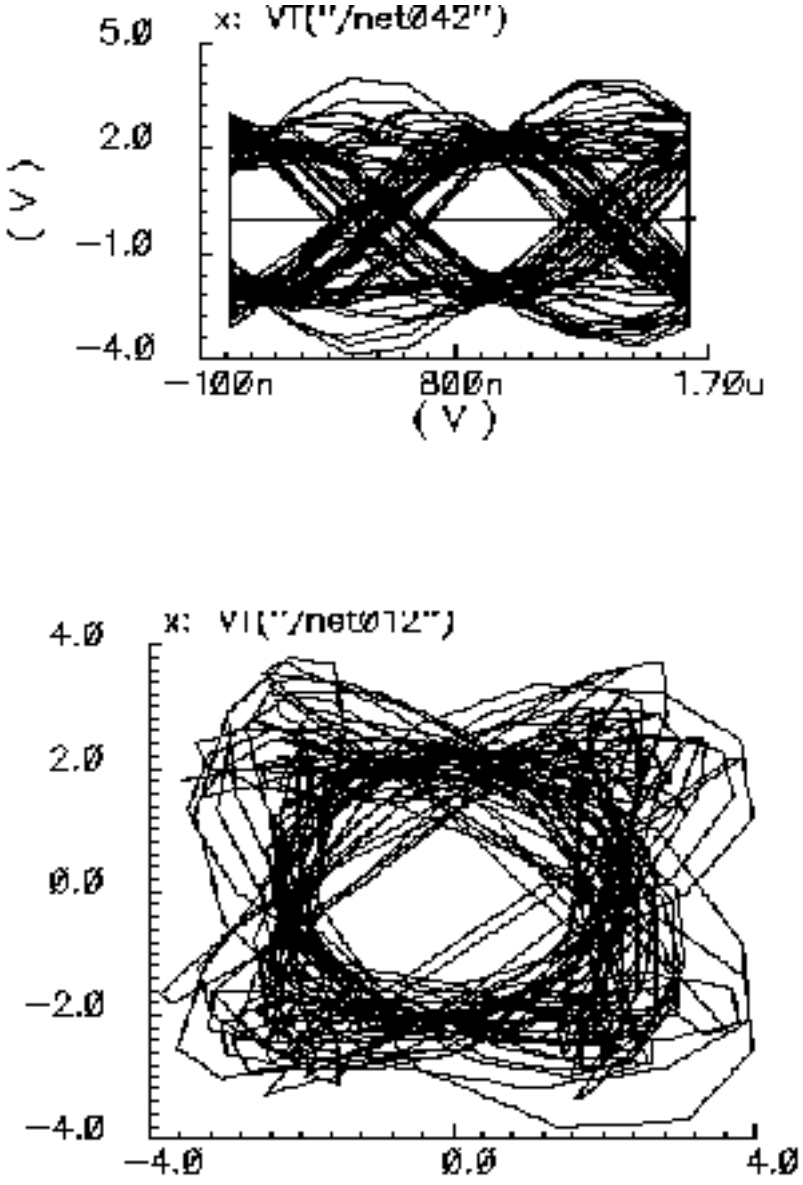
Figure 1-36 shows a block diagram of the signal generator.

Figure 1-36 CDMA Baseband Test Signal Generator



The eye-diagram generator (`eye_diagram_generator`) created the eye-diagram and trajectory. Figure 1-37 on page 83 shows the eye-diagram of one of the outputs and the trajectory of both outputs.

Figure 1-37 Eye Diagram and CDMA Trajectory



CDMA Signal Source Instance Parameters

The `amplitude` parameter sets the amplitude of the unfiltered signals. An amplitude of 1 means that each FIR filter is driven by 1 volt impulses. If you change the internal variable `IMPULSE_PULSE` to 2, the filters are driven by 1 volt pulses of four samples duration.

The `seed` parameter changes the seed for the random number generator.

CDMA Signal Source Outputs

The CDMA signal generator creates four output signals:

`i_bin_node`	The *I* unfiltered binary output.
`i_out_node`	The filtered *I* output.
`q_bin_node`	The *Q* unfiltered binary output.
`q_out_node`	The *Q* filtered output.

Changing the FIR Filter in a CDMA Signal Source

You cannot change the FIR filter, such as the tap length and tap coefficients, directly from the instance. However, you can do so using the Modelwriter as described in "Modifying the BB Signal Generators Using Modelwriter" on page 223.

CDMA Signal Source Output Transitions

The filtered outputs slew linearly from one value to the next because the rise and fall times in the transition statements equal one period. To make the outputs take abrupt steps, copy the module to your library and change the rise and fall times in the last transition statements.

comms_instr

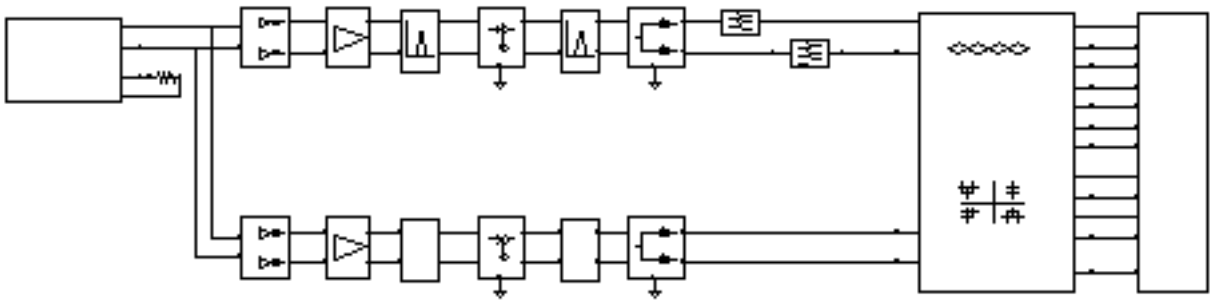
(Instrumentation Block)

The `comms_instr` block generates waveforms that can be used to create eye-diagrams, eye-diagram statistics, scatter plots, and rms error-vector-magnitudes.

For information about the parameters of the block, see "Instrumentation Block Parameters" on page 86. For information about the outputs of the block, see "Instrumentation Block Outputs" on page 87. For information about the related `offset_comms_instr` block, see "(Instrumentation Block)" on page 97. For information about the related `instr_term` block, see "instr_term" on page 96.

Figure 1-38 on page 85 shows how the `offset_comms_instr` and `instr_term` blocks should be used. The `comms_instr` block is used similarly. The circuit consists of two branches driven from a single baseband signal generator. The top branch is the non-ideal receiver model, the bottom branch is an ideal version of the top branch. The ideal version is as ideal as you like. The ideal branch computes ideal symbol locations in the complex plane. The instrumentation block compares ideal and non-ideal symbols to compute the error-vector-magnitude.

Figure 1-38 EVM setup



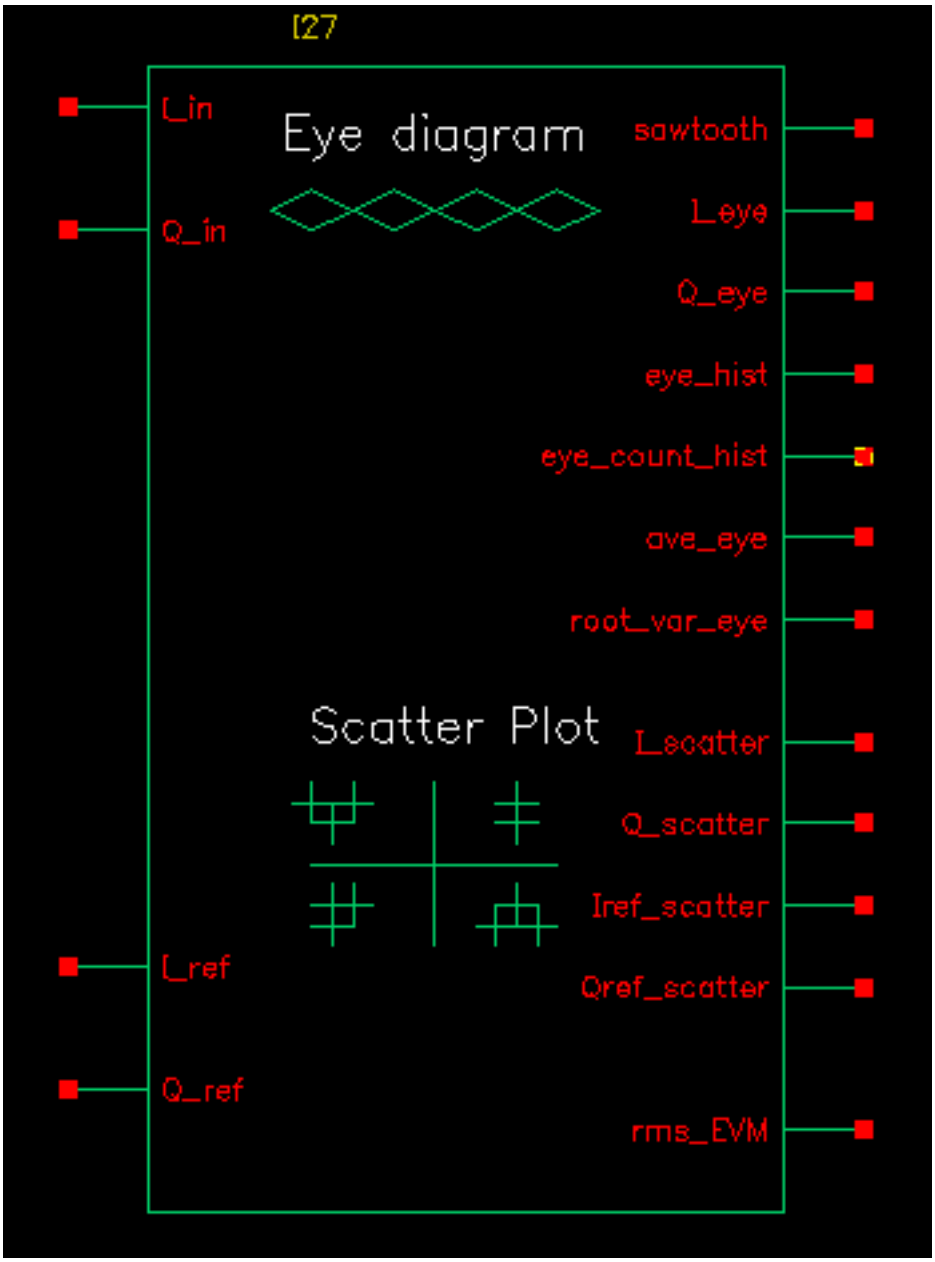
Instrumentation Block Parameters

Parameter	Meaning
I_del	**I-sampling delay.** This parameter sets the phase of the symbol sampler. It is referenced to the eye-diagram of the I-output. Estimate the optimal delay by doing one simulation just to get the eye-diagram. The optimal delay is the time from the leftmost part of the eye-diagram, which should be zero, to the time at which they eye is widest.
I_noise, Q_noise	**I-noise and Q-noise(volts^2).** These parameters set the variance of Gaussian random variables which can be added to the received symbols before anything is computed or plotted.
max_voltage, min_voltage, num_of_bins	**Max, min eye-diagram volts, and number of hstgm bins.** These parameters are used to compute the bins which define the eye-diagram histogram. The bin width equals (max voltage-min voltage)/(number of bins). The histogram shows the distribution of the I_in voltage at the sampling instant.
measurement_delay	**Statistics start time.** This parameter delays the start of any statistical computations. The purpose is to exclude start-up transients from the statistics.
number_of_symbols	**Number of symbols.** This is the number of symbols to sweep in the eye-diagram. Sweeping two symbols ensures that you see at least one continuous eye, if it is open.
rin	**Input resistance.** This parameter is the input resistance of the input terminals of the instrumentation block.
symbol_rate	**Symbols per second.** This parameter is necessary for generating the sawtooth that is used as the x-axis to generate eye-diagrams. It also determines the rate at which the input waveforms are sampled.

Instrumentation Block Outputs

Output	Meaning
ave_eye, root_var_eye	**Eye-diagram statistics.** The ave_eye output is the average absolute value of the I-input signal at the sampling instant. The root_var_eye output is the square root of the variance of the absolute value of the I-input voltage at the sampling instant. The voltages at these output pins represent running estimates of the associated statistics.
eye_hist, eye_count_hist	**Histograms.** You can only generate a histogram of the I-input signal. The histogram shows the distribution of the I-input voltage at the sampling instant. To generate a histogram, plot the eye_hist and eye_count_hist outputs in the same waveform display window. Change the x-axis to be the eye-hist signal then change the plot to *bar*.
I_scatter,Q_scatter, Iref_scatter, Qref_scatter	**Scatter plots.** A scatter plot is the I-input and Q-input samples plotted against each other. The scatter plot shows the locations of the received symbols. To generate a scatter plot, plot I_scatter and Q_scatter in the same waveform display window then change the x-axis to be the I_scatter signal. Finally, change the plot to plot data points only. A scatter plot of the reference model can be generated similarly by replacing I_scatter and Q_scatter with Iref_scatter and Qref_scatter.
rms_EVM	**RMS Error Vector Magnitude.** The rms EVM is defined as the square root of the sum of the squares of the vectorial differences between the ideal and non-ideal received symbols, normalized to the rms value of the magnitude of the ideal received symbols. The output voltage at this pin is represents a running calculation of the rms EVM. The normalized EVM is in percent.
sawtooth, I_eye, Q_eye	**Eye-diagrams.** To generate an eye-diagram of the I-input signal, plot the sawtooth and I-eye outputs in one waveform display tool. Change the x-axis to be the sawtooth. This is done through the x-axis menu in the waveform display tool. The procedure for generating an eye-diagram of the Q-output is the same except you use the Q-eye output.

eye_diagram_generator



The eye-diagram generator creates eye-diagrams and trajectories for the baseband signal generators. For more information, see "Eye-Diagram Generator Input" on page 89 and "Eye-Diagram Generator Outputs" on page 89.

Eye-Diagram Generator Input

The input to the eye-diagram generator is the *I* or *Q* output of one of the baseband signal generators.

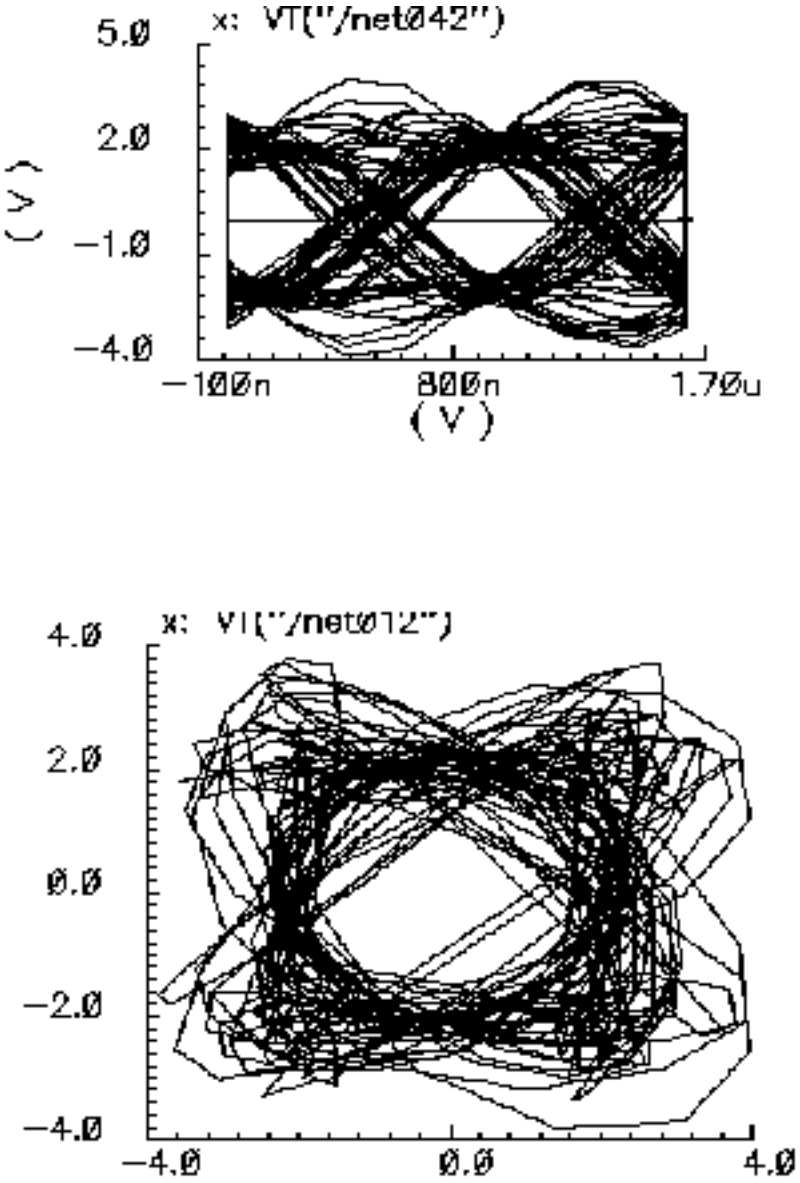
Eye-Diagram Generator Outputs

The eye-diagram generator has two outputs labeled `y-axis` and `x-axis`. The eye-diagram is generated by plotting the `y-axis` output against the `x-axis` output.

The eye-diagram generator does not work with Envelope analysis to generate similar plots. This is because the Envelope harmonic time analysis is generated by a post-processing step and the eye-diagram generator works during simulation.

Figure 1-39 shows an eye-diagram of one of the outputs and the trajectory of both outputs for the CDMA baseband signal generator.

Figure 1-39 Example Eye Diagram and CDMA Trajectory



gmsk

GMSK (Gaussian minimum shift keying) is a simple but efficient approach to digital modulation that provides the properties of narrow-band techniques, sharp cutoffs in frequency, lower overshoot impulse response, and preservation of the filter output pulse area. These qualities result in low phase distortion and make GMSK suitable for coherent demodulation. The GMSK approach is used in the Global System for Mobile Communication (GSM).

In release IC 6.1.2 and later, Cadence provides a Verilog-A module for simulating GMSK behavior. The module is located in `rfLib`.

Figure 1-40 GMSK symbol

The `inbit` input supports the use of external random generators. Usually `inbit` is disabled by specifying `enable_input` = 0.

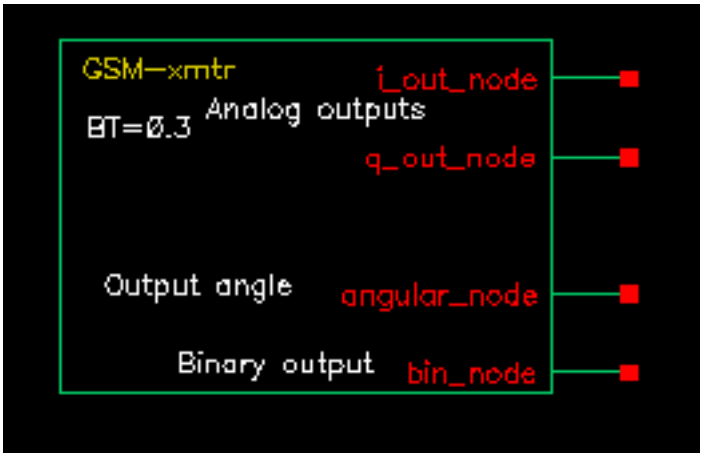
The parameters of the instance are listed in the table below:

Name	Meaning	Type	Default Value	Range
`BT`	BT product.	real	0.3	
`Period`	Input bits period.	real	6.0/1625000	
`Phase_offset`	Initial phase, normalized by p_i	real	0	
`Pulse_length`	Pulse length of Gaussian Filter	integer	3	
`Samples`	Number of samples in one T	integer	16	
`initial_sym`	Prehistory symbols used before simulation	integer	{0, 0, 0}	1 or -1
`seed`	Used for internal random generator when input is disabled	type	21	
`Enable_input`	1 if using external inputs (NRZ code)	integer	1	0 or 1

gsm_comms_instr

GSM_xmtr

(GSM Signal Source)



The GSM source generates a signal conforming to the GSM standard. The modulation is GMSK and the data is generated in frames of 3 fixed start bits, 142 random data bits, 3 fixed stop bits, and 8.25 fixed guard bits. (The embedded deterministic pattern and quarter of a bit is necessary to produce the correct spectrum.) The bit rate is 270833.333 bits per second and the sample rate is four times that.

The FIR filter is a Gaussian filter implemented with 32 taps.

Figure 1-41 shows a block diagram of the signal source.

Figure 1-41 GSM Baseband Signal Generator

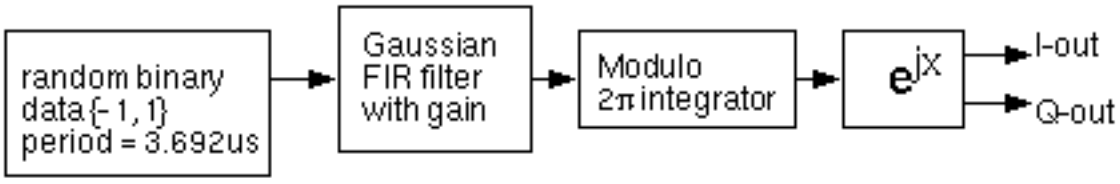
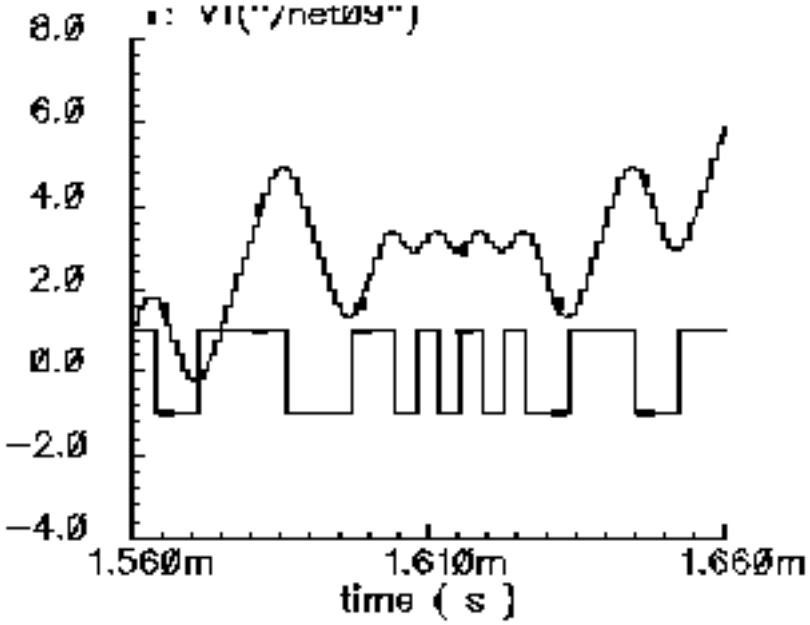


Figure 1-42 shows the binary data stream and the corresponding angle.

Figure 1-42 GSM Binary Data and Resulting Phase



GSM Signal Source Instance Parameters

The `amplitude` parameter sets the amplitude of the unfiltered signals. An amplitude of 1 means that each FIR filter is driven by 1-volt impulses. If you change the internal variable `IMPULSE_PULSE` to `2`, the filters are driven by 1-volt pulses of four samples duration.

The `seed` parameter changes the seed for the random number generator.

GSM Signal Source Outputs

The generator creates four output signals:

`angular_node`	The output signal.
`i_out_node`	The phase, multiplied by the amplitude.
`bin_node`	The bit stream being transmitted.
`q_out_node`	The phase multiplied by the amplitude.

Changing the FIR Filter in a GSM Signal Source

You cannot directly change the FIR filter, such as the tap length and tap coefficients, from the instance. However, you can make changes using the Modelwriter as described in "Modifying the BB Signal Generators Using Modelwriter" on page 223.

GSM Signal Source Output Transitions

The filtered outputs slew linearly from one value to the next because the rise and fall times in the transition statements equal one period. To make the outputs take abrupt steps, copy the module to your library and change the rise and fall times in the last transition statements.

instr_term

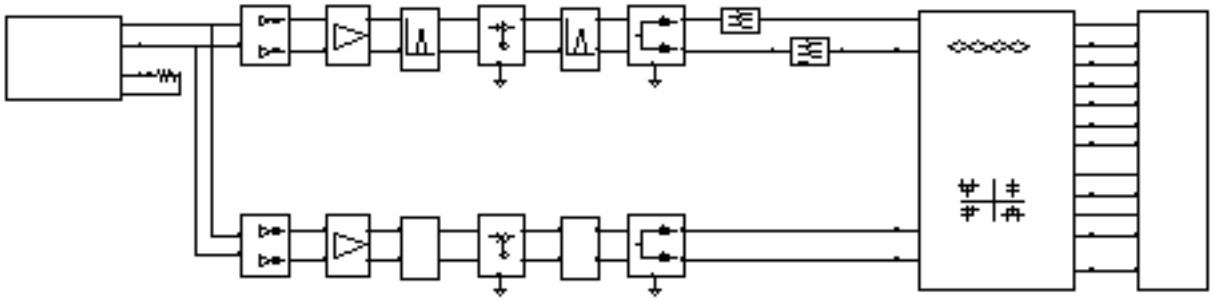
(Terminating Block)

The `instr_term` block simply loads all instrumentation output pins with 50 Ohms. The `instr_block` keeps the schematic editor from complaining about unconnected pins, nothing more.

For information about the parameters of the block, see “Instrumentation Block Parameters” on page 86. For information about the outputs of the block, see “Instrumentation Block Outputs” on page 87. For information about the related `comms_instr` block, see “(Instrumentation Block)” on page 85. For information about the related `offset_comms_instr` block, see “gmsk” on page 91.

Figure 1-43 on page 96 shows how the `offset_comms_instr` and `instr_term` blocks should be used. The `comms_instr` block is used similarly. The circuit consists of two branches driven from a single baseband signal generator. The top branch is the non-ideal receiver model, the bottom branch is an ideal version of the top branch. The ideal version is ideal as you like. The ideal branch computes ideal symbol locations in the complex plane. The instrumentation block compares ideal and non-ideal symbols to compute error-vector-magnitude.

Figure 1-43 EVM setup



offset_comms_instr

(Instrumentation Block)

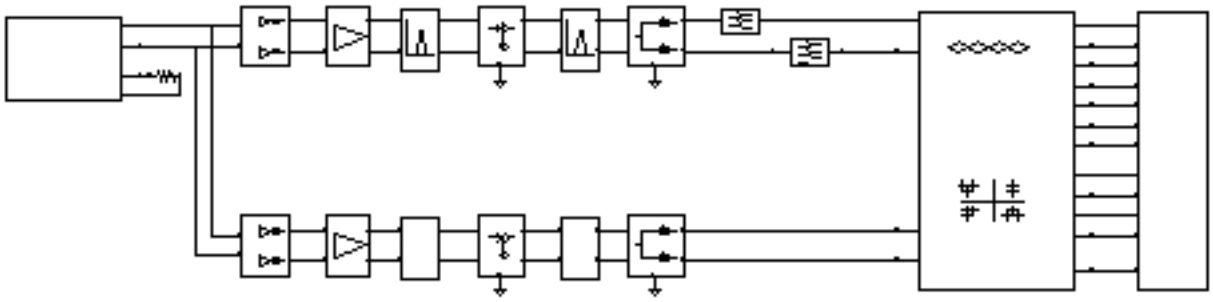
The `offset_comms_instr` block generates waveforms that can be used to create eye-diagrams, eye-diagram statistics, scatter plots, and rms error-vector-magnitudes.

The offset block is identical to the `comms_instr` except that the sampling time for scatter plots and eye-diagram statistics are delayed by half a symbol period. The delay makes it possible to plot symbols in an offset QPSK modulation scheme.

For information about the parameters of the block, see "Instrumentation Block Parameters" on page 86. For information about the outputs of the block, see "Instrumentation Block Outputs" on page 87. For information about the related `comms_instr` block, see "(Instrumentation Block)" on page 85. For information about the related `instr_term` block, see "instr_term" on page 96.

Figure 1-44 on page 97 shows how the `offset_comms_instr` and `instr_term` blocks should be used. The circuit consists of two branches driven from a single baseband signal generator. The top branch is the non-ideal receiver model, the bottom branch is an ideal version of the top branch. The ideal version is as ideal as you like. The ideal branch computes ideal symbol locations in the complex plane. The instrumentation block compares ideal and non-ideal symbols to compute error-vector-magnitude.

Figure 1-44 EVM setup



phase_generator

pi_over4_dqpsk

(Pi/4-DQPSK Signal Source)

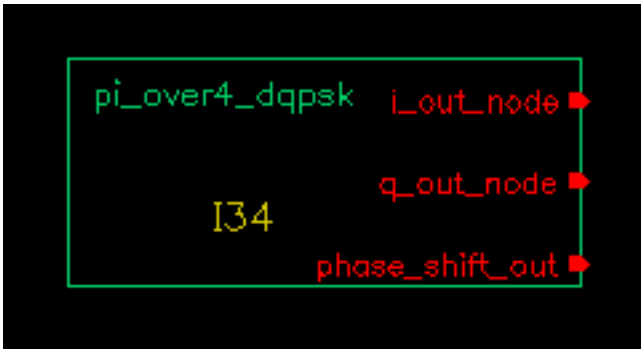


Figure 1-45 shows the block diagram for this source.

Figure 1-45 Π/4-DQPSK baseband signal generator

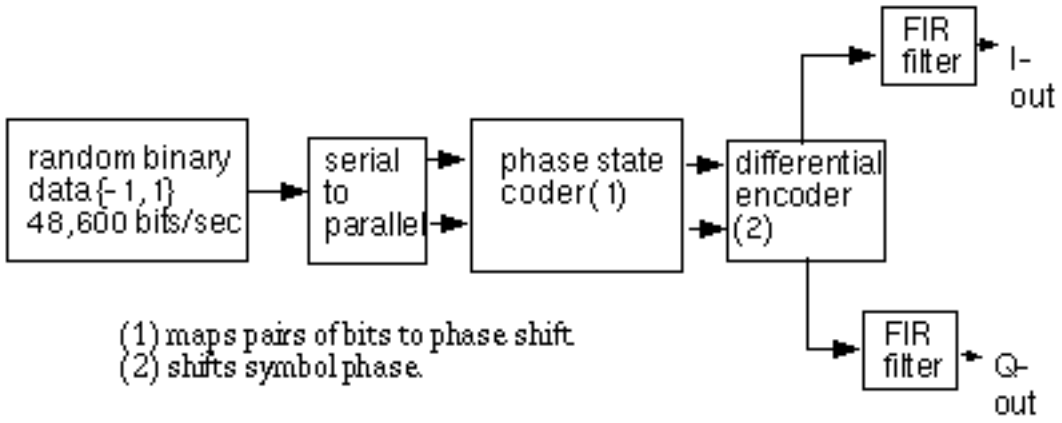


Table 1-46 shows how the phase shift is generated.

Figure 1-46 Phase Shift

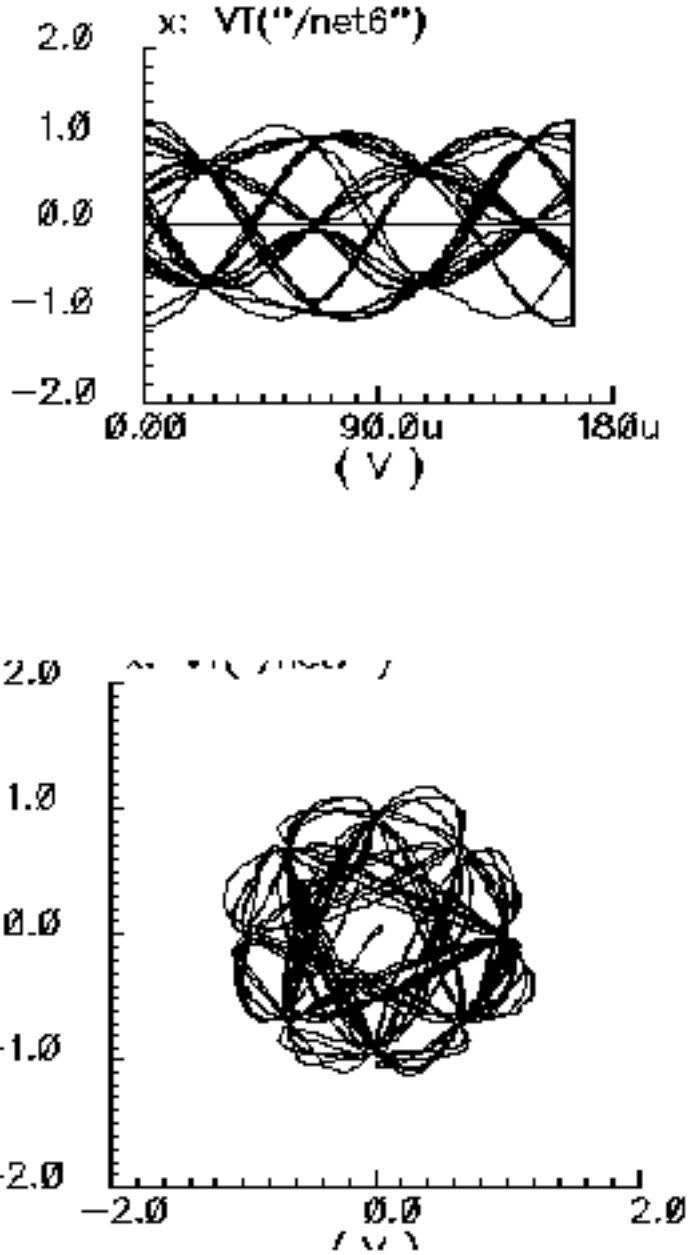
1st bit	2nd bit	Phase shift
0	0	$\pi/4$
0	1	$3\pi/4$

1st bit	2nd bit	Phase shift
1	0	-π/4
1	1	-3π/4

The symbol rate is 24300 symbols per second and the sample rate is 8 times that. The FIR filter is a raised cosine filter implemented with 64-taps.

The eye-diagram generator (`eye_diagram_generator`) created the eye-diagram and trajectory. Figure 1-47 shows the eye-diagram and trajectory for this generator.

Figure 1-47 Eye Diagram and Pi/4 Trajectory



The `amplitude` parameter lets you set the amplitude of the unfiltered signals. An amplitude of “1” means that each FIR filter is driven by 1-volt impulses. If you change the internal variable `IMPULSE_PULSE` to 2, the filters are driven by 1-volt pulses of four samples duration.

The `seed` parameter lets you change the random number generator seed.

Pi/4-DQPSK Signal Source Outputs

The generator creates three output signals.

`i_out_node`	The phase, multiplied by the amplitude.
`q_out_node`	The phase, multiplied by the amplitude.
`phase_shift_out`	The phase shift from one symbol to the next.

Changing the FIR filter in a Pi/4-DQPSK Signal Source

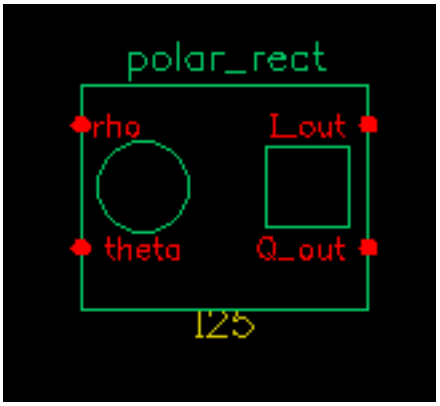
You cannot change the FIR filter, such as the tap length and tap coefficients, directly from the instance. However, you can do so using the Modelwriter as described in “Modifying the BB Signal Generators Using Modelwriter” on page 223.

Pi/4-DQPSK Signal Source Output Transitions

The filtered outputs slew linearly from one value to the next because the rise and fall times in the transition statements equal one period. To make the outputs take abrupt steps, copy the module to your library and change the rise and fall times in the last transition statements.

polar_rect

(Polar-to-Rectangular Transformation)



The polar-to-rectangular block is in the measurement category. The only parameters are input and output resistances. The inputs are the baseband signal in polar coordinates, the outputs are the baseband signal in rectangular coordinates.

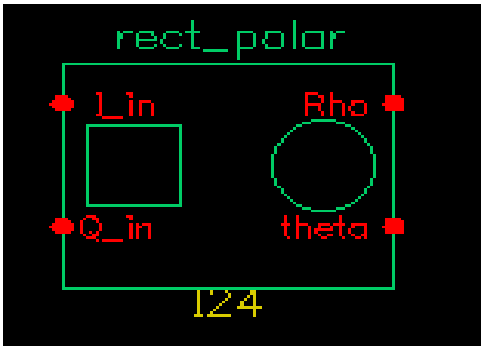
- Inputs = ρ and θ volts.
- Outputs = i and q volts, such that

$$i = \rho * \cos(\theta)$$

$$q = \rho * \sin(\theta)$$

rect_polar

(Rectangular-to-Polar Transformation)



The rectangular-to-polar block is in the measurement category. The only parameters are input and output resistances. The inputs are the baseband signal in Cartesian coordinates, the outputs are the baseband signal in polar coordinates.

Parameters: Input and output resistances.

- Inputs = `i` and `q` volts.
- Outputs = ρ and θ volts, such that

  $$\rho = \sqrt{i*i + q*q}$$

  and

  $$\theta = \text{atan}[q / i] + \pi*(1 + \text{sgn}[i]) / 2$$

  where θ is in radians and with appropriate checks for the $i = 0$ case

Passband_components Category

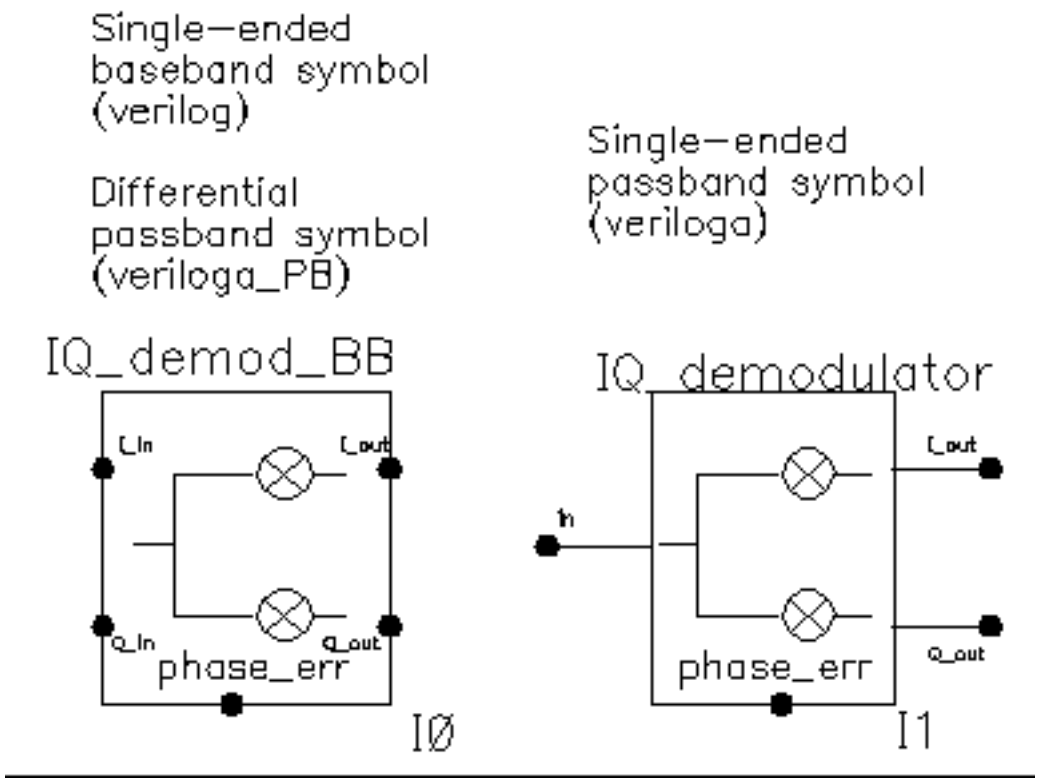
The `Passband_components` category contains the following elements, discussed in the sections that follow.

- IQ_demodulator
- IQ_modulator
- LNA_PB
- MIXER_PB
- PA_PB
- shifter_combiner
- shifter_splitter

IQ_demodulator

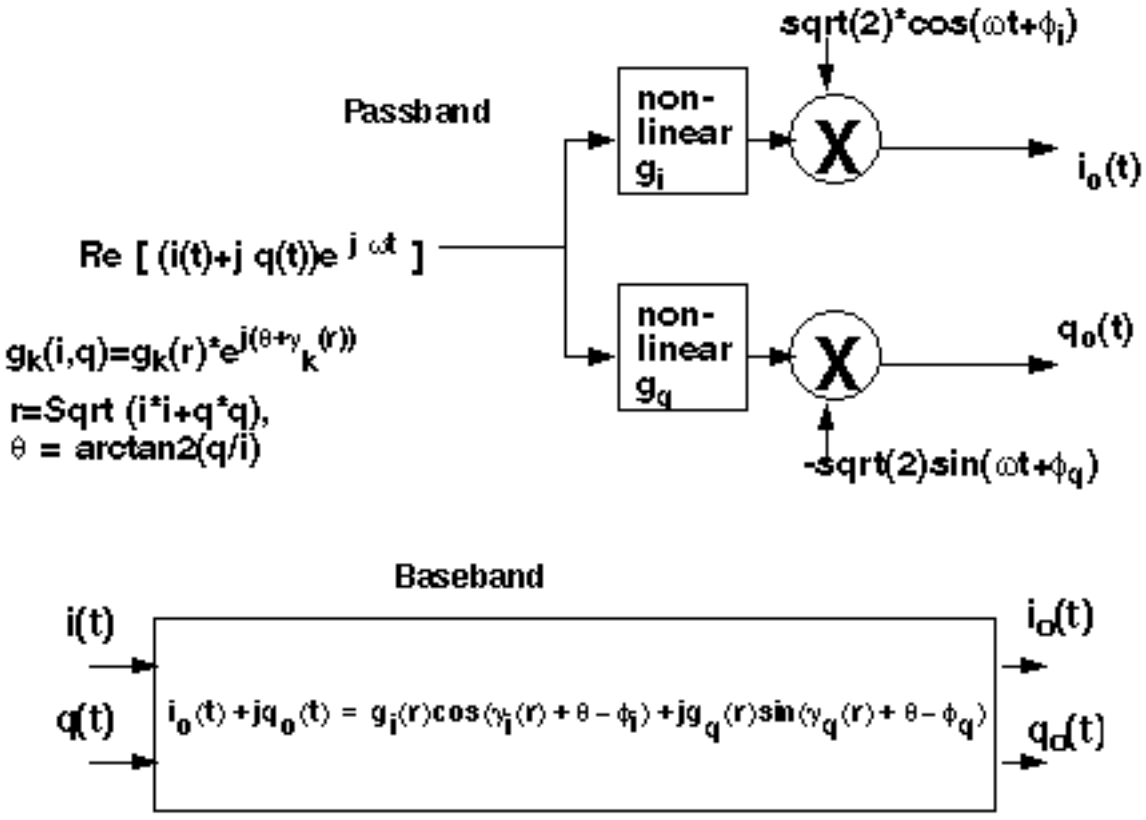
```
(passband = IQ_demodulator)
```

Figure 1-48 Baseband and Passband IQ Demodulator Models



The `IQ_demodulator` converts RF (or IF) to baseband. Figure 1-49 on page 107 shows exactly what the passband demodulator model does. The parameters are like those in the modulator blocks except saturation is specified by input referred IP3 instead of by 1 dB compression point. IP3 was chosen over the 1 dB compression point for specifying saturation because the demodulator usually lies in the receive path and receiver blocks are usually specified with IP3.

Figure 1-49 IQ Demodulator Calculations



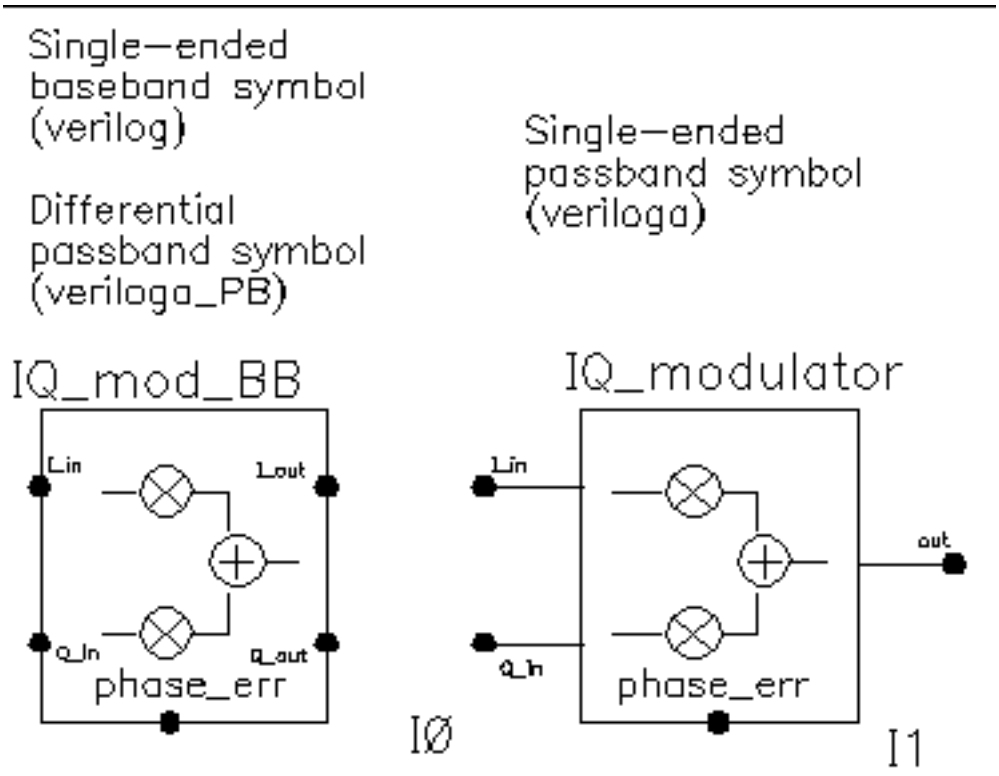
The parameters are:

`flow`	Local oscillator frequency.
`I_gain`	Available power gain [dB].
`I_IP3`	Input referenced IP3 (dbm).
`nf`	Noise figure [dB].
`Q_gain`	Voltage gain [dB].
`Q_IP3`	Input referenced IP3 [dBm].
`quad_error`	Quadrature error.
`rin`	Input resistance.
`rout`	Output resistance.

IQ_modulator

```
(passband = IQ_modulator)
```

Figure 1-50 Baseband and Passband IQ Modulator Models

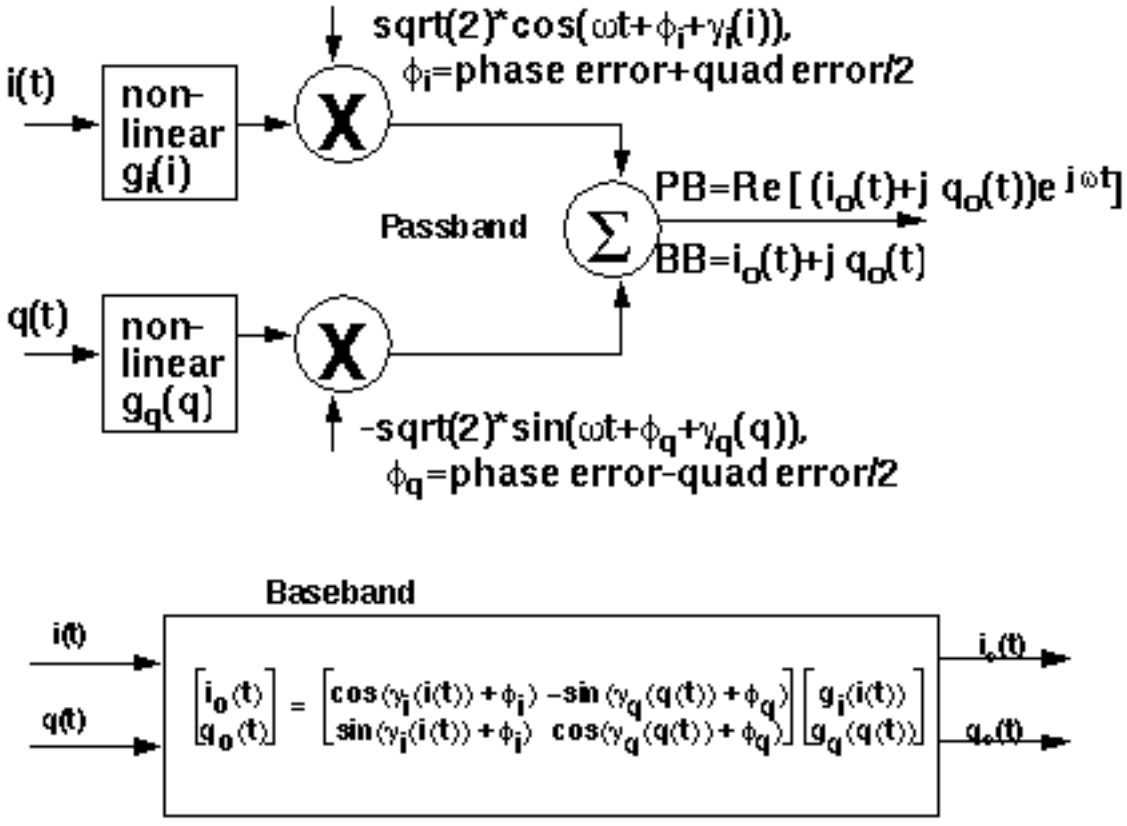


The `IQ_modulator` converts baseband signals to RF or IF. Figure 1-51 on page 109 summarizes exactly what the passband IQ modulator model does. The only difference between the baseband and passband models is carrier suppression. The non-linear functions, g_i and g_q, are specified by their available power gain and 1dB compression points just as in the power amplifier. The functions γ_i and γ_q characterize AM/PM effects in each mixer and are specified by the same parameters that specify power amplifier AM/PM conversion. Because noise is always added at the input, and the input is at baseband in this case, the noise sources are not doubled as they are in the power amplifier or LNA models. Noise figure is defined with reference to one input. Noise is injected at both inputs but the noise injected at just one input alone produces the specified noise figure. Thus, the noise figure parameter should be interpreted as noise figure per input. This model also includes a parameter called `quadrature error` which specifies how far away the two local oscillators signals are from being exactly in quadrature.

`Phase error` is the voltage on the phase error pin. The phase error pin has a fixed noiseless resistive input impedance of 50 ohms. The phase error pin can be used to introduce a dynamic phase error or phase noise. Phase noise can be fed into the phase error pin from a phase-domain PLL model or from a Port. Noise in Port models can be specified either by the internal resistance or by a data file that tabulates a power spectral density. The phase error pin can also be driven by a ramp or circular integrator output to model a frequency error between the incoming carrier and local oscillator.

The following parameters specify the IQ modulator. The available power gain and one dB compression point are explained first. The effects of the phase_error pin and the quadrature error parameter are discussed at the end of this section.

Figure 1-51 IQ Modulator Calculations



The parameters are:

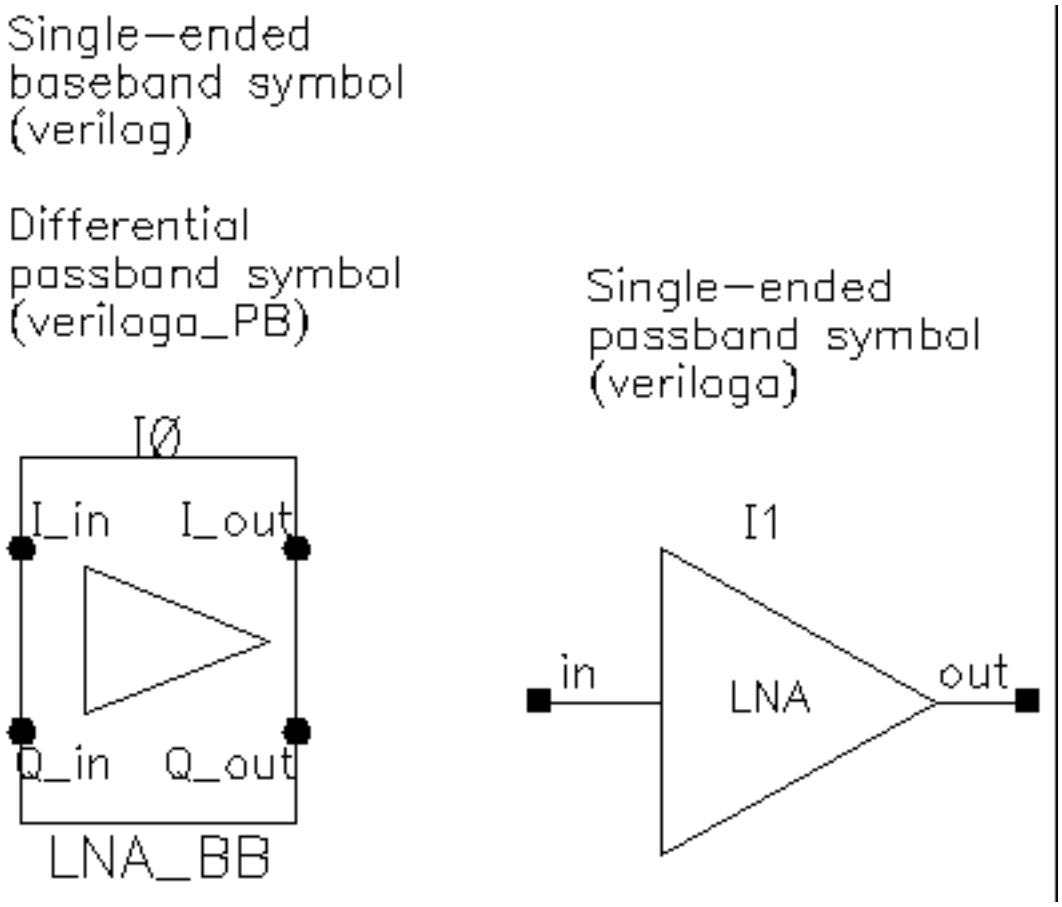
	Quadrature error.
`flo`	Local oscillator frequency.

`I_cpdb`	I-output 1dB CP [dBm].
`I_gain`	Available I-mixer gain [dB].
`nf`	Noise figure [dB].
`Q_cpdb`	Q-output 1dB CP [dBm].
`Q_gain`	Available Q-mixer gain [dB].
`rin`	Input resistance.
`rout`	Output resistance.

LNA_PB

```
(passband = LNA_PB)
```

Figure 1-52 Baseband and Passband Power Amplifier Models



The following parameters specify the low noise amplifier model.

The parameters are:

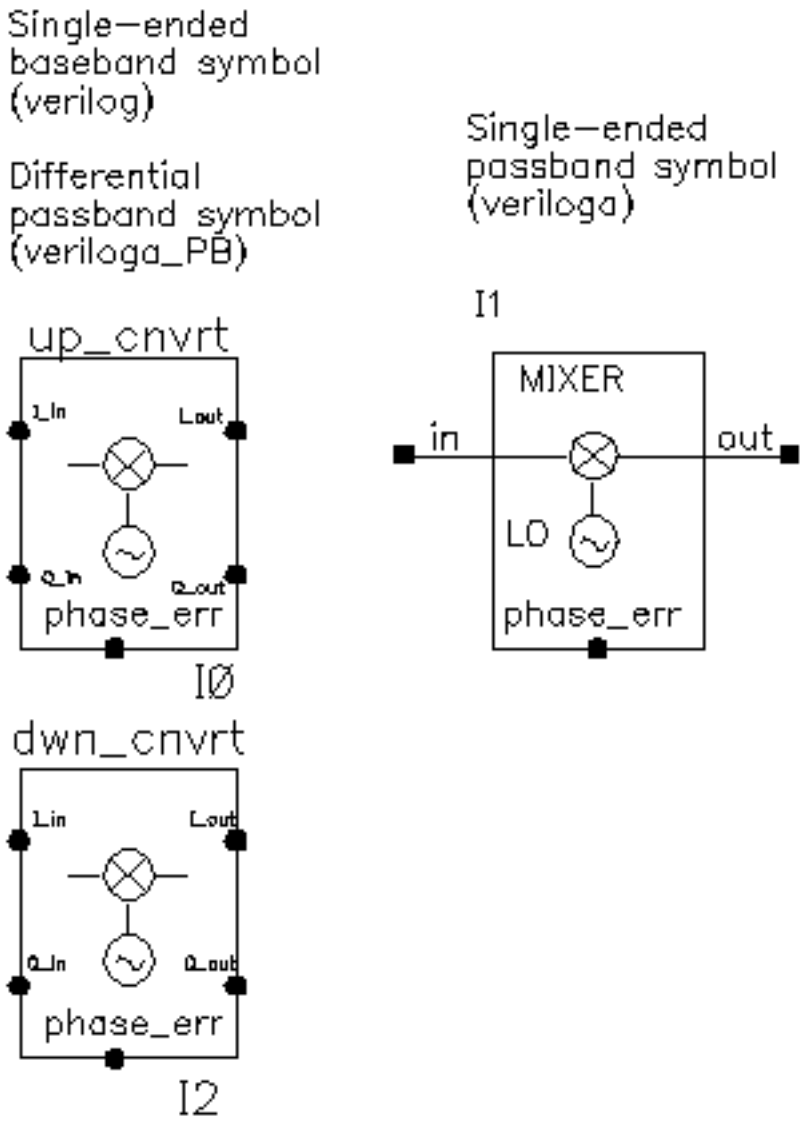
`cmp`	Input power point for phase point [dBm].
`cw`	Determines the direction of the phase shift. The phase shift is only in one direction. +1 means counter-clockwise, -1 means clockwise, and 0 means no phase shift (no am/pm conversion).
`gain`	Available power gain.

`IP3`	Input referred IP3 [dBm].
`nf`	Noise figure.
`pscp`	Output phase shift at cmp [radians].
`psinf`	Output phase shift as the input power goes to infinity.
`rin`	Input resistance.
`rout`	Output resistance.
`shp`	Determines how fast the phase shift occurs with increasing input power. A larger number delays the shift but makes the shift rise faster as a function of input signal level.

MIXER_PB

```
(passband = MIXER_PB)
```

Figure 1-53 Baseband and Passband Mixer Models



`MIXER_PB` is a passband model that converts RF to IF and IF to RF.

The parameters are:

`flo`	Local oscillator frequency.
`gain`	Available power gain [dB].
`IP3`	Input referenced IP3 [dBm].
`nf`	Noise figure [dB].
`psinf`	Output phase shift as the input power goes to infinity.
`rin`	Input resistance.
`rout`	Output resistance.

PA_PB

The parameters are:

`cpdb`	Output referenced 1dB compression [dBm].
`gain`	s21 referenced [dB].
`nf`	Noise figure [dB].
`rin`	Input impedance [Ohm].
`rout`	Output impedance [Ohm].

shifter_combiner

The `shifter_combiner` combines two signals so that they add if one leads the other by 90 degrees and so that they cancel if it lags by 90 degrees.

The parameters are:

`freq`	Frequency at which the magnitudes are balanced.
`gain`	Multiplies the input voltages.
`r`	Impedance of the internal resistor.
`rin`	Input terminal impedances.
`rout`	Output impedance.

shifter_splitter

The `shifter_splitter` splits a signal into two signals 90 degrees out of phase with each other.

The parameters are:

`freq`	Frequency at which the magnitudes are balanced.
`gain`	Multiplies the input voltages.
`r`	Impedance of the internal resistor.
`rin`	Input impedance.
`rout`	Output impedance.

RF_components Category

The cells in the `RF_components` category are:

- ACPR_source
- balun
- balun_com
- balun_ideal
- freq_divider
- gfsk
- ideal_demod
- ideal_mod
- lna
- mixer
- ofdm
- osc
- pa
- PB2BB_demod
- quadrature
- root_raised_cos
- sample_and_hold
- shifter

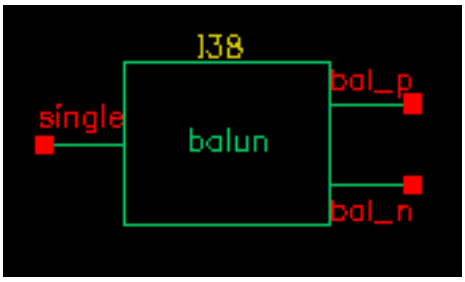
ACPR_source

The `ACPR_source` is used to generate modulated RF waveforms. It is a self-contained RF modulator. The out connection is the mmodulated RF output.

The parameters are:

`I PWL file name`	File name for the I modulation piecewise linear file
`Q PWL file name`	File name for the Q mudulation piecewise linear file
`Gain`	Linear multiplier for the amplitude
`RF Frequency`	RF carrier frequency
`RF Amplitude (dBm)`	RF amplitude in dBm
`Reference Resistance`	Output resistance (Ohms)

balun



The balun (balancing transformer) is used in circuits that require single/differential signal transformation. Although a passive network (including the transformer) is used to achieve balun, this implementation employs a three-port network. There are three ports (or nodes), because the reference nodes are always at the global ground: `single`, `bal_p`, and `bal_n`.

The three ports are

`single`	Single end.
`bal_p`	In-phase end of the balanced output.
`bal_n`	Out-of-phase end of the balanced output.

When the ports are numbered as `single(1)`, `bal_p(2)`, and `bal_n(3)`, the S-parameter for the three-port network is

$$S = \begin{bmatrix} 0 & t & -t \\ t & 0 & 0 \\ -t & 0 & 0 \end{bmatrix}$$

where

$$t = \frac{10^{-loss/\ 10}}{\sqrt{2}}$$

when loss is specified in dB.

This module can also be used in common mode cancellation applications.

The module is declared as follows

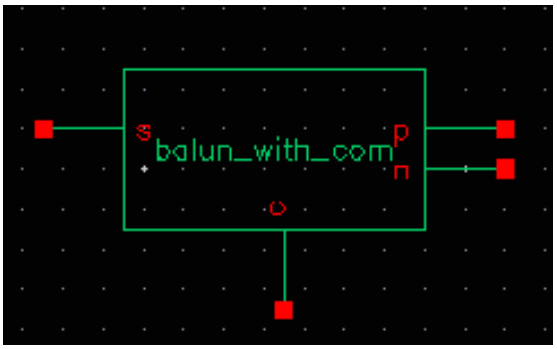
```
module balun(single, bal_p, bal_n);
  inout single, bal_p, bal_n;
  electrical single, bal_p, bal_n;
   parameter real rin = 50 from (0:inf);
   parameter real rout = 50 from (0:inf);
   parameter real loss = 0 from [0:inf);
```

Parameters include the input impedance (for single end), the output impedance (for balanced end to ground), and the insertion loss (from single end to balanced end and from balanced end to single end).

The parameters are:

`loss`	Insertion loss [dB].
`rin`	Input impedance [Ω].
`rout`	Output impedance [Ω].

balun_com



The `balun_com` has, in addition to the three ports of the `balun`, an external reference node that can be used for DC bias set up in the balanced end. The balun_com is equivalent to the balun when the voltage of the reference node `c` is set to 0.

The four ports of the `balun_com` are:

`c`	Common (reference) node for `p` and `n`.
`n`	Out-of-phase end of the balanced output.
`p`	In-phase end of the balanced output.
`s`	Single end.

The module is declared as follows

```
module balun_com(s, p, n, c);
  inout s, p, n, c;
  electrical s, p, n, c;
   parameter real rin = 50 from (0:inf);
   parameter real rout = 50 from (0:inf);
   parameter real loss = 0 from [0:inf);
```

Parameters include the input impedance (for single end), the output impedance (for balanced end to ground), and the insertion loss (from single end to balanced end and from balanced end to single end).

The parameters are:

`loss`	Insertion loss [dB].
`rin`	Input impedance [Ω].
`rout`	Output impedance [Ω].

balun_ideal

freq_divider

gfsk

(Gauss Frequency Shift Keying)

GFSK is a pre-modulation Gaussian filtered frequency modulation, which makes the pulse smoother so as to limit the spectral width and the channel inter-symbol interference (ISI). GFSK is poorer in terms of the spectral efficiency but easier to implement as compared to GMSK. GFSK is often seen in cordless phones (using DECT standard) and blue tooth.

The ports are

`inbit`	Input signal, activated only when ‘enable_input = 0’. This terminal is used when you want to use an external random generator and should otherwise be disabled.
`iout`	Output signal such as the GFSK baseband signals.
`qout`	Output signal such as the GFSK baseband signals.

Instance Parameters

The parameters are:

Name	Meaning	Type	Default Value	Range
`BT`	BT product	real	0.3	
`Period`	Initial phase	real	6.0/ 1625000	
`Phase_offset`	Power per bit in dB	real	0	
`pulse_length`	Frame time	integer	3	
`Samples`	Sampling points in one frame	integer	16	

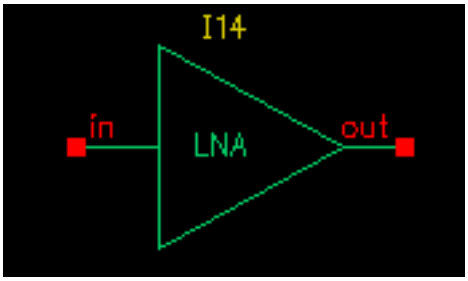
`Initial_sym`	the pulse length of the filter	integer	[0 0 0]	1 or -1
`seed`	Samples in one frame	type	21	
`enable_input`	1 if using external random input, otherwise 0	integer	1	0 or 1

ideal_demod

ideal_mod

lna

(Low-Noise Amplifier)



Low-noise amplifiers (LNAs) are commonly used in receiver designs to amplify the signal with a low noise figure. A typical LNA has the following three sets of parameters:

- Linear model
- Nonlinear model
- Noise model.

The module is declared as follows:

```
module lna(in, out);
   inout in, out;
   electrical in, out;
   parameter real nf = 2 from [0:inf);
   parameter real ip3 = -10;
   parameter real gain = 15 from [0:inf);
   parameter real isolation = 200 from (0:inf);
   parameter real rin = 50 from (0:inf);
   parameter real cin = 0 from [0:100];
   parameter real rout = 50 from (0:inf);
   parameter real cout = 0 from [0:100];
   parameter real gammain = -150 from (-inf:0];
   parameter real mismatch = 1 from [-1:1] exclude (-1:1);
   parameter real gammaout=-150 from (-inf:0];
```

The parameters are:

`cin`	Parasitic input capacitance [pF].
`cout`	Parasitic parallel output capacitance [pF].
`gain`	S21 (power gain) [dB].
`gammain`	Input return loss [dB].
`gammaout`	Output return loss [dB].

`ip3`	Input referenced IP3 [dBm].
`isolation`	S12 [dB].
`mismatch`	Mismatch sign of input. 1: input impedance > reference impedance -1: otherwise.
`nf`	Noise figure [dB].
`rin`	Reference impedance of the input port [Ω].
`rout`	Reference impedance of the output port [Ω].

Internally, a set of linear equations is constructed to satisfy the S-parameters. Furthermore, nonlinearity, expressed by a third-order polynomial function, is added to the gain (or S21) to describe the IP3. Excess white noise is added at the input port to describe the noise figure.

IP3 is the measure of the corruption of signals due to the third-order intermodulation of two nearby tones as shown in Figure 1-54. You measure this parameter using a two-tone test. Avoid the measurement of IP3 by a single tone test.

Figure 1-54 Intermodulation of Two Nearby Signals

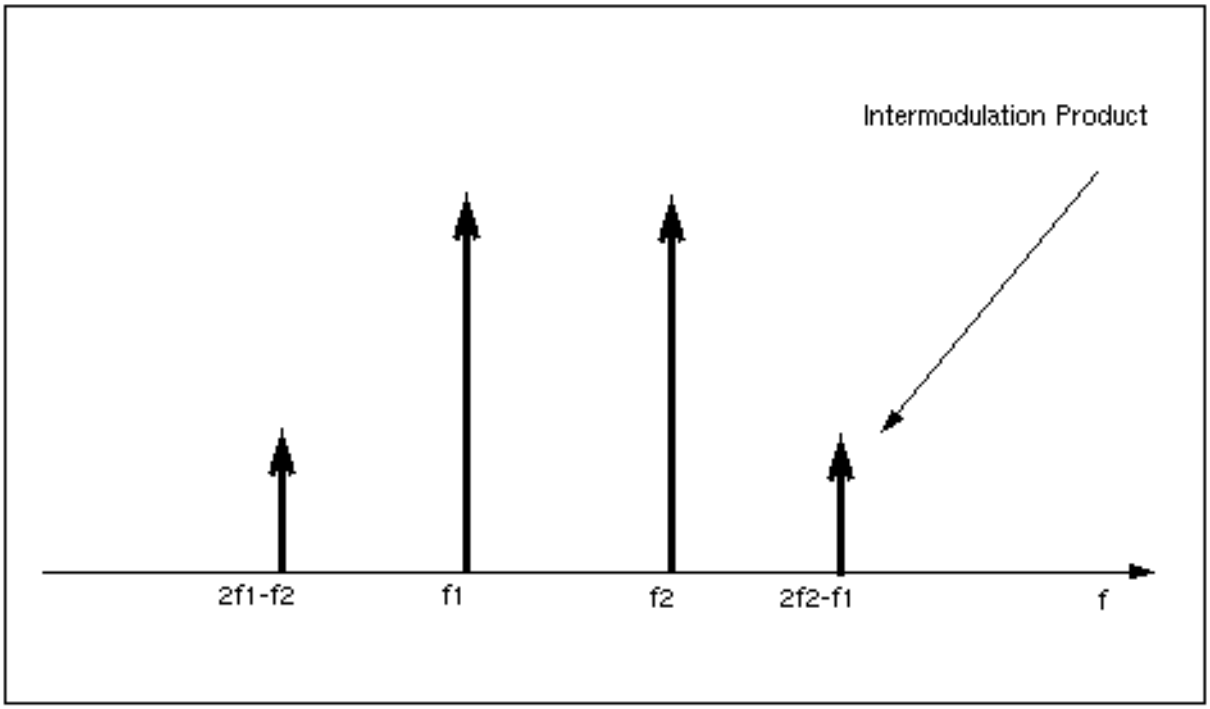
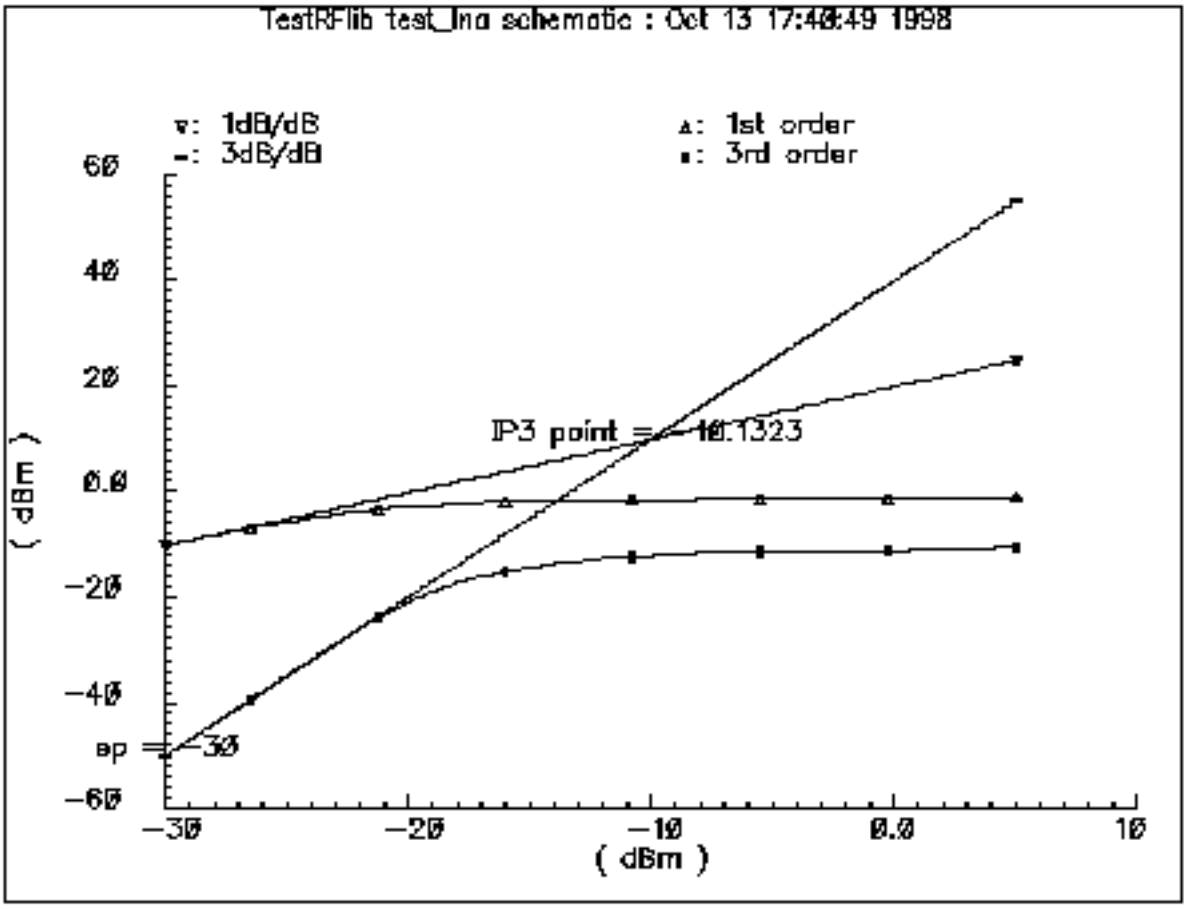
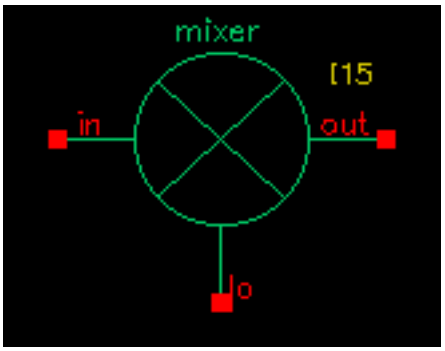


Figure 1-55 shows the captured IP3 when the requested value of IP3 is -10dBm.

Figure 1-55 IP3 from Spectre RF Simulation



mixer



Mixers are important for frequency translation in RF circuits. A typical mixer has the following three sets of parameters.

- Time-varying linear model
- Nonlinear model
- Noise model

This RF library model describes the typical behavior of integrated mixers. The LO switches the input signal on and off. Input LO power beyond the specified limit is effectively clipped off.

Declare the module as follows

```
module mixer(in, lo, out);
    electrical in, lo, out;
    parameter real gain = 10 from [-50:50];
    parameter real plo = 10 from [-100:100];
    parameter real rin = 50 from (0:inf);
    parameter real rout = 200 from (0:inf);
    parameter real rlo = 50 from (0:inf);
    parameter real ip2 = 5;
    parameter real ip3 = 5;
    parameter real nf = 2 from [0:inf);
    parameter real isolation_LO2IN = 20 from (0:inf);
    parameter real isolation_LO2OUT = 20 from (0:inf);
    parameter real isolation_IN2OUT = 20 from (0:inf);
```

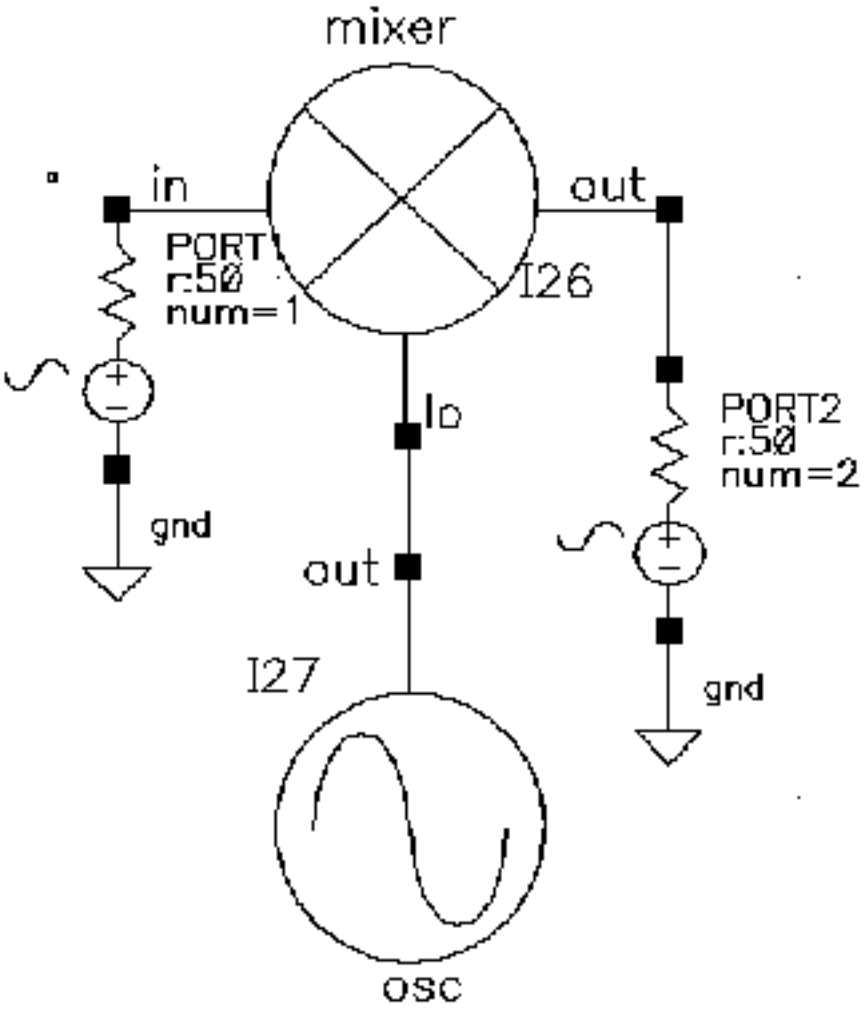
The parameters are:

`gain`	Gain from IN to OUT [dB].
`ip2`	Input referenced IP2 [dBm].
`ip3`	Input referenced IP3 [dBm].
`isolation_IN2OUT`	Isolation from IN to OUT [dB].

`isolation_LO2IN`	Isolation from LO to IN [dB].
`isolation_LO2OUT`	Isolation from LO to OUT [dB].
`nf`	Noise figure (DSB) [dB].
`plo`	Power of the LO input [dBm].
`rin`	Input impedance for IN [Ω].
`rlo`	Input impedance for LO [Ω].
`rout`	Output impedance for OUT [Ω].

Figure 1-56 is the simple schematic that tests the mixer.

Figure 1-56 Schematic for Testing the Mixer Model



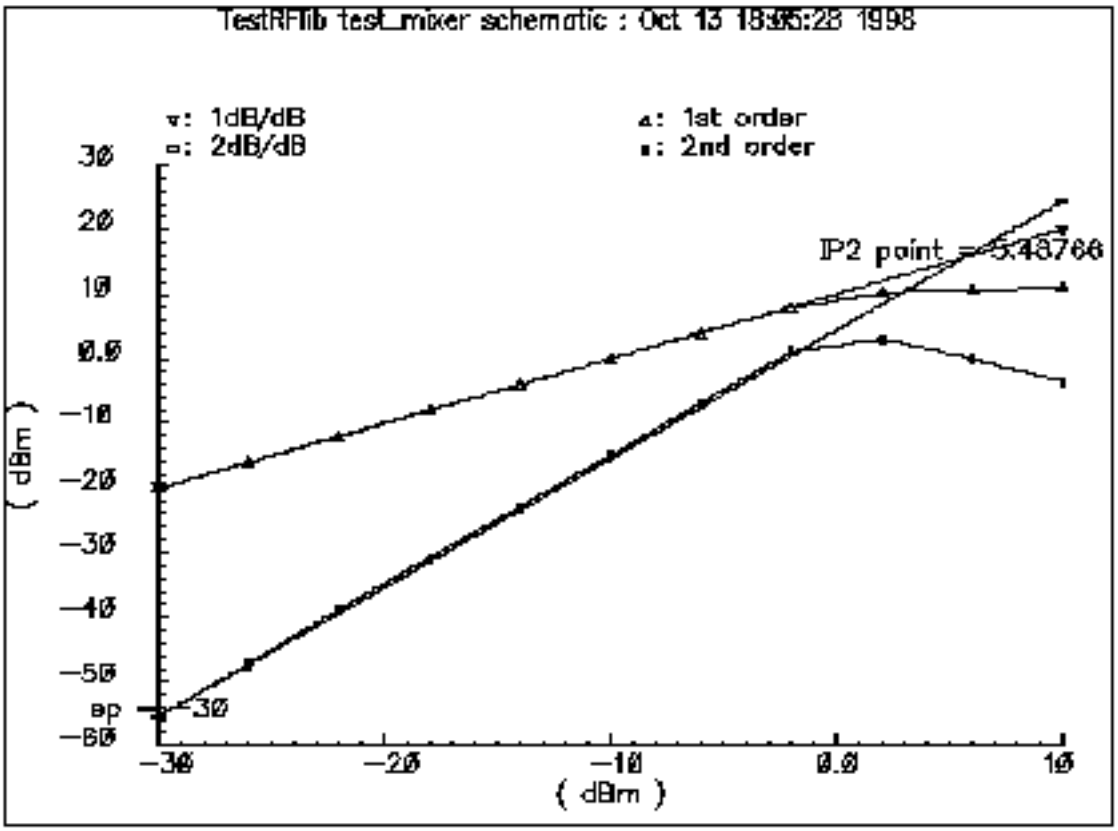
The maximum power of the fundamental frequency of the local oscillator, `plo`, can be used in the mixing process. Therefore, the gain, defined as the output power of the mixed product versus the input power of the RF signal, depends on the power level of the LO. The gain levels off, however, to the specified maximum value as the LO signal becomes larger.

You can measure both IP3 and IP2 with Spectre RF. You must select frequencies carefully when you measure IP3 to measure harmonic distortion (HD) and IP2. Testing IP3 requires two tones to measure the intermodulation distortion (IMD), while testing IP2 requires only one tone.

Assume the RF input frequencies are f_1 and f_2, and the LO frequency is f_{lo}. If the input power level at f_1 equals that at f_2, the IP3 is the intercept point of the extrapolated line of output

power at frequency $|f_{lo} - (2f_2 - f_1)|$ versus the extrapolated line of the linear output signal at $|f_{lo}-f_1|$. Input-referenced IP3, therefore, can be read as the X-axis value at the intercept point. The IP2, for the purpose of measuring the half-IF effects, is defined as the intercept point of the extrapolated line of output power at frequency $|2(f_{lo} - f_1)|$ versus the linear output signal. Figure 1-56 shows that the intercept point of the 1 dB/dB and 2 dB/dB lines is at the X-axis reading of 4.78 dBm, while the requested IP2 value is 5 dBm. The order of the intercept point is based only on the order of the RF signals. The order of LO signal is not counted in the definition of the intercept point. In the implementation of this model, the orders of LO for IP3 and IP2 are 1 and 2 respectively.

IP2 Measurement



Internally, a set of equations is built to satisfy a three-port S-parameter. A third-order polynomial describes the nonlinearity of IP3. The LO signal is further multiplied by itself to derive the second-order harmonic, which is then used to produce the IP2 effect. Excessive white noise is added in the RF input port to satisfy the noise figure. Remember, however, that the noise figure is double-sideband. If the noise at the image frequency is not filtered out, the measured noise figure is 3dB larger than the DSB noise figure.

ofdm

(Orthogonal Frequency-Division Multiplexing)

OFDM is a digital multi-carrier modulation scheme, in which closed-spaced sub-carriers are summed into main carrier. These sub-carriers are orthogonal to each other and modulated with conventional modulation scheme at a low symbol rate. The summation is performed through Fast Fourier Transform.

OFDM has the merit of robost against intersymbol interference and narrow-band co-channel interference. It is spectral-efficient. It has seen many applications such as WiMAX, MBWA, Wi-Fi and UWB and so on.

The ports are

I_in, Q_in [v]	Input signals, activated only when “input_enable” is set to 1.
I_out, Q_out [v]	Output signals, i.e., ofdm baseband signals.

Instance Parameters

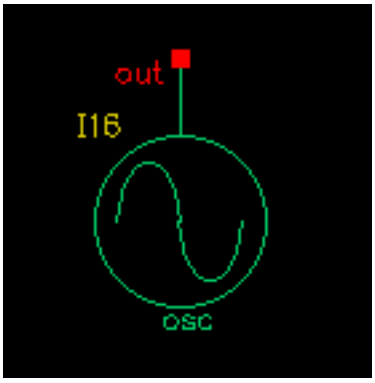
The parameters are:

`frame_time`	Frame time
`samples`	Samples in one frame
`Poly_length`	The length of vector for shift register's feedback connections
`Shift_length`	The length of vector for the delay of PN sequence
`Poly_order`	The order of polynomial function
`Init_state_size`	The length of vector of initial state

`poly`	The array of polynomial function
`state`	The array of initial state
`shift`	The array for delay of PN
`Dump_frames`	How many frames are skipped initially

osc

(Oscillator)



Oscillator models describe the essential information for a typical oscillator, more precisely, a local RF power source.

The definition of the model in the Verilog-A language is as follows:

```
module osc(out);
    electrical out;
    inout out;
    parameter real power = 10;
    parameter real f = 1e9 from (0:inf);
    parameter real rout = 50 from (0:inf);
    parameter real floor = -60 from (-inf:0);
    parameter real f1 = 1000 from (0:1e6);
    parameter real n1 = -40 from (bottom:0);
    parameter real fc = 0 from [0:f1);
```

The parameters are:

`bottom`	Noise floor [dBc/Hz].
`f1`	Frequency point for `n1` [Hz].
`fc`	Corner frequency of white phase and flicker phase [Hz].
`freq`	Output frequency [Hz].
`n1`	Phase noise at `f1` [dBc/Hz].
`power`	Output power when matched [dBm].
`rout`	Output impedance [W].

This model is not an autonomous model. Rather, it simply generates a sinusoidal wave with the specified impedance, power level, and phase noise characteristics.

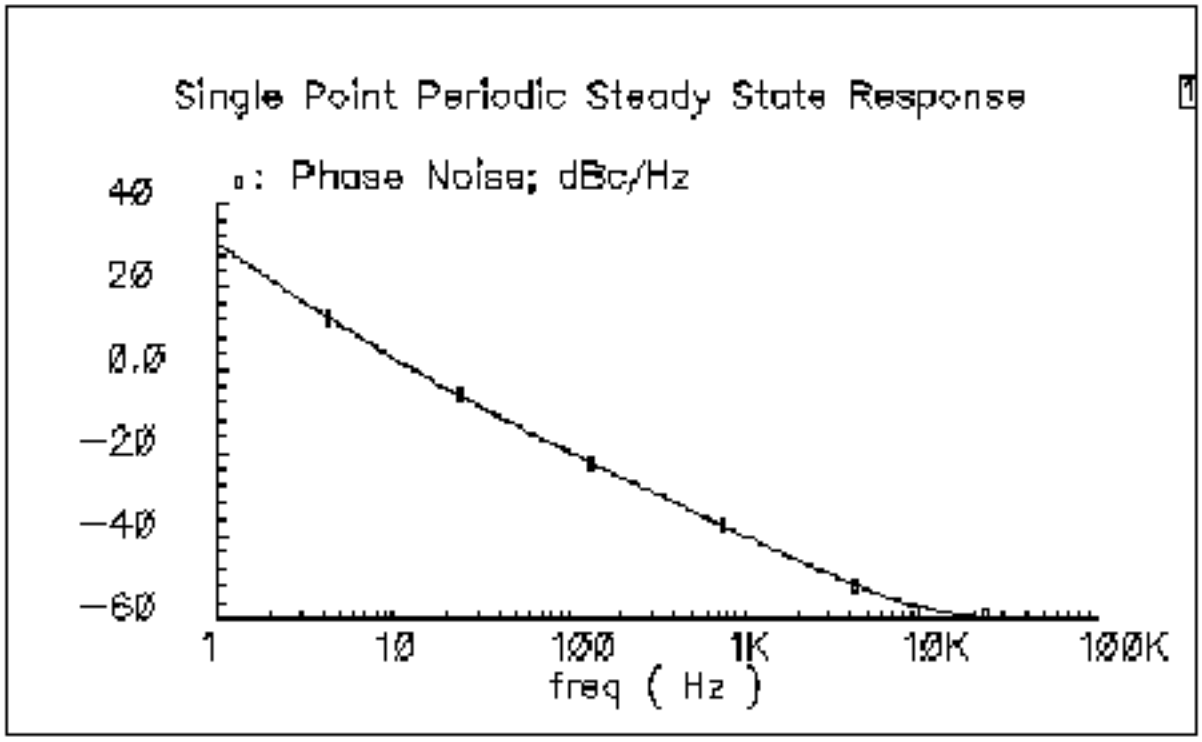
When the load is matched to the internal impedance, the load dissipates the specified output power. You can specify the noise floor of the output signal. Furthermore, by adding one point (frequency, phase noise), you can specify $1/f^2$ frequency noise (corresponding to the phase noise induced by white noise). If f_c, the corner frequency of white phase and flicker phase noise, is bigger than 0, $1/f^3$ frequency noise (flicker-noise-induced phase noise) is further specified. Otherwise, $1/f^3$ noise is not included.

The phase noise values that are symmetric around the carrier are correlated. The noise floor, however, is not correlated.

Figure 1-57 shows the phase noise of the oscillator model. In Figure 1-57, the specified parameters are:

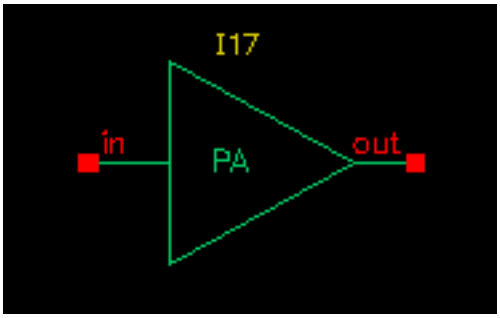
`noise floor`	-60 dBc/Hz
f_1	1 K
n_1	-40 dBc/Hz
f_c	100

Figure 1-57 Phase Noise for the Oscillator



pa

(Power Amplifier)



Power amplifiers (PAs) are used in RF transmitters to achieve output of a higher power level. The PA model differs from the LNA model in that it has greater power delivery capabilities with less stress on matching capabilities.

The Verilog-A module is declared as follows:

```
module pa(in, out);
    inout in, out;
    electrical in, out;
    parameter real nf = 2 from [0:inf);
    parameter real gain = 20 from [0:inf);
    parameter real rin = 50 from (0:inf);
    parameter real rout = 50 from (0:inf);
    parameter real p1db = 30;
    parameter real psat = 35;
    parameter real ip2 = 40;
```

The parameters are:

`gain`	S21 [dB].
`ip2`	Input-referenced IP2 [dBm].
`nf`	Noise figure [dB].
`p1db`	Output-referenced 1dB compression [dBm].
`psat`	Maximum output power [dBm].
`rin`	Input impedance [Ω].
`rout`	Output impedance [Ω].

The power amplifier model has the following three parts:

- the linear model
- the nonlinear model
- the noise model

Internally, for simplicity, the reverse isolation is assumed to be ideal. A set of linear equations is constructed to satisfy these S-parameters. Nonlinear effects are added to the gain to describe the nonlinearity. The output power of the power amplifier compresses to 1 dB less than the output of an ideal linear amplifier at the 1 dB compression point. Further increase of the input power makes the output approach the saturation power only at the fundamental operating frequency. IP2 describes the second order effects of the amplifier, so use only one tone in the test. Excess white noise is added at the input port to describe the noise figure.

The implementation of `psat` assumes a pure sinusoidal waveform. To maintain a restrained output power, the output waveform is clipped from a sinusoidal to a square wave form. Figure 1-58 shows the input and output waveforms of the power amplifier. Because of the output waveform clipping, the input sinusoidal wave should have a DC component of zero.

Figure 1-58 Input and Output Waveforms of the Power Amplifier

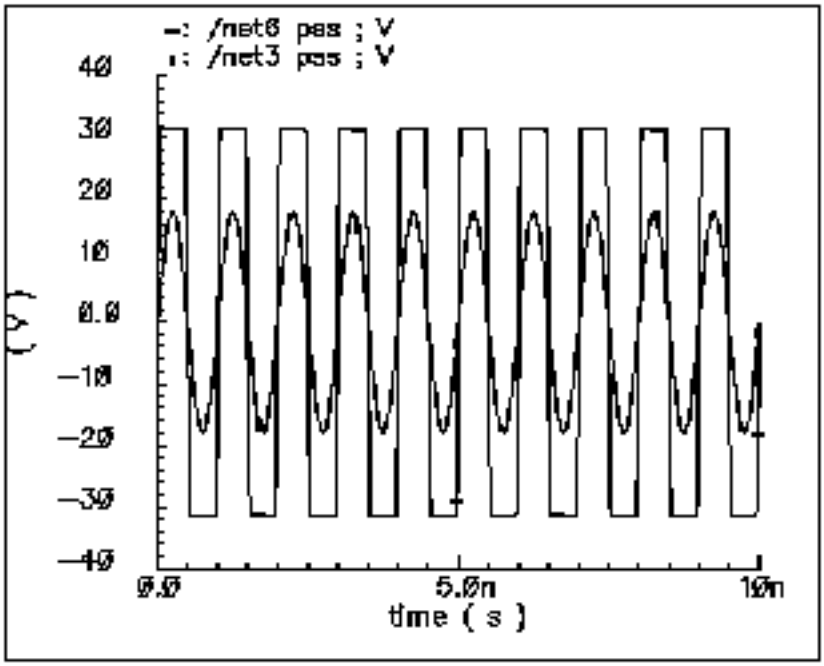
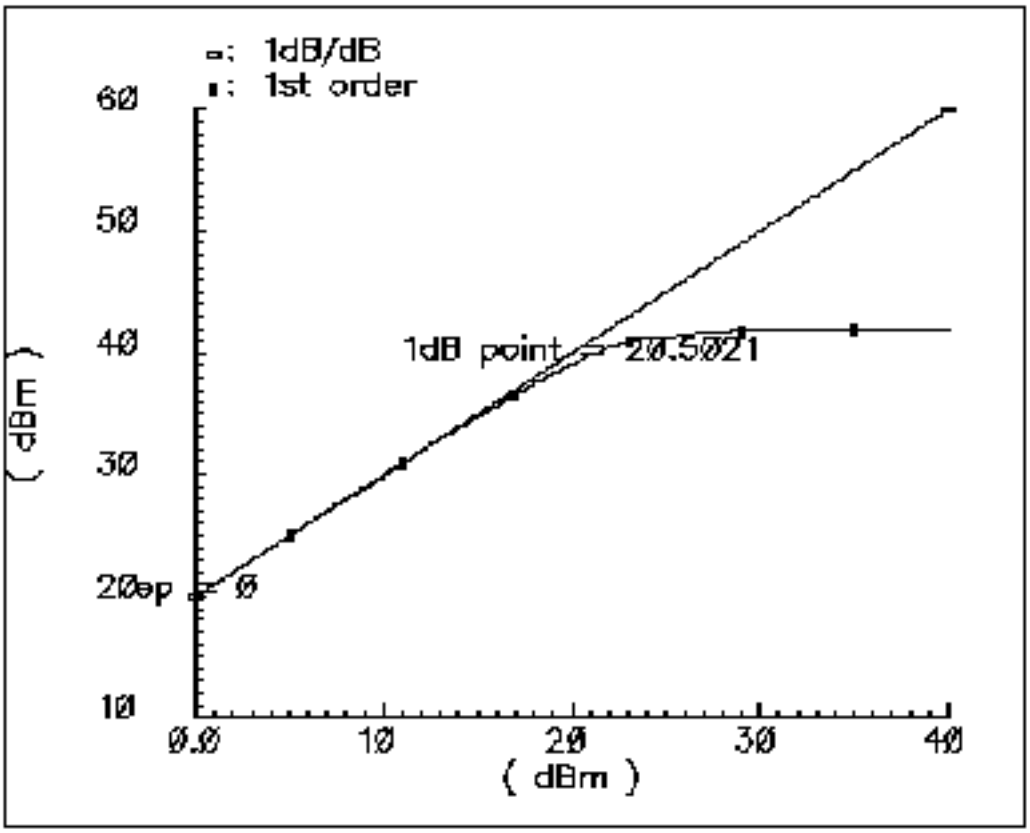


Figure 1-59 shows the 1 dB compression point and the saturation power. This difference is caused by the 50 Ωload impedance. The specified output referenced 1 dB compression point is 40 dBm, which Spectre RF captures as 39.6.

If `psat` is much larger than `p1db`, your `psat` might not be satisfied.

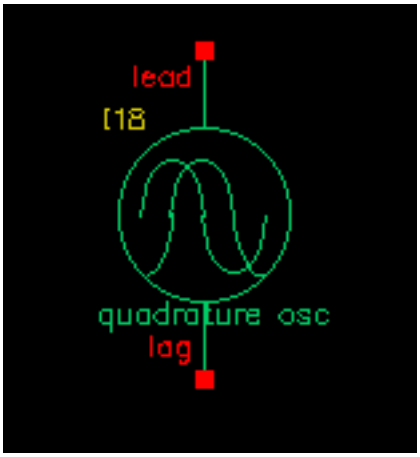
Figure 1-59 1dB Compression Point and Saturation Power



PB2BB_demod

quadrature

(Quadrature Signal Generator)



The quadrature signal generator model is included because, in quadrature receiver design, a phase shifter is ordinarily used to generate the quadrature signal from one signal source such as the VCO. However, a phase shifter is hard to implement in a wide band model.

A quadrature signal consists of two signals with a 90-degree phase difference but with identical noise and amplitude.

The Verilog-A module is declared as follows.

```
module quadrature(lead, lag);
    electrical lead, lag;
    inout out_cos, out_sin;
    parameter real power = 10;
    parameter real f = 1e9 from (0:inf);
    parameter real rout = 50 from (0:inf);
    parameter real floor = -60 from (-inf:0);
    parameter real f1 = 1000 from (0:1e6);
    parameter real n1 = -40 from (bottom:0);
    parameter real fc = 0 from [0:f1);
```

The parameters are:

`bottom`	Noise floor [dBc/Hz].
`f1`	Frequency point for `n1` [Hz].
`fc`	Corner frequency of white phase and flicker phase [Hz].
`freq`	Output frequency [Hz].
`n1`	Phase noise at `f1` [dBc/Hz].

`phase_shift`	
`power`	Output power when matched [dBm].
`rout`	Output impedance [W].

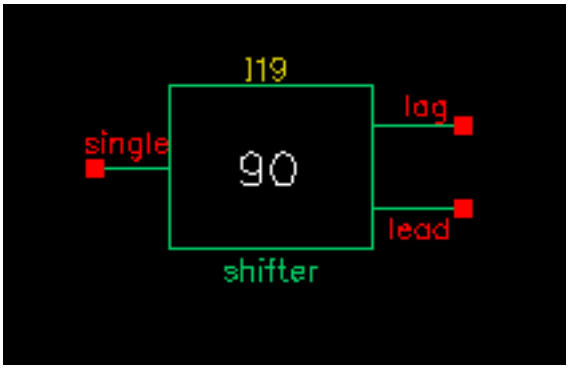
The difference between the quadrature signal generator model and the oscillator model is that the oscillator has only one output node but the quadrature signal generator has two output nodes, `lead` and `lag`. In the quadrature signal generator model, when the power levels, output impedances, and noise sources are identical, the two outputs, `lead` and `lag`, have a 90-degree phase difference.

root_raised_cos

sample_and_hold

shifter

(Phase Shifter)



In digital RF system designs, quadrature signal processing involves the phase splitting of high-frequency signals. The most common use of such components is to generate two signals that have a 90-degree phase difference based on one signal source (such as the RF signal or oscillator output). Another common use for a phase shifter is to combine two signals after adding a 90-degree phase difference, as in image-rejection receiver designs.

The Verilog-A module is declared as follows

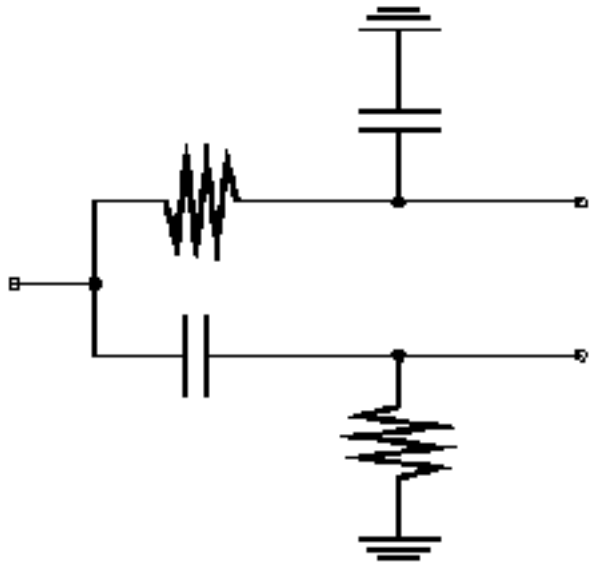
```
module shifter(single, lag, lead);
    inout single, lag, lead;
    electrical single, lag, lead;
    parameter real freq = 1e9 from (0:inf);
    parameter real r    = 50 from (0:inf);
```

The parameters are:

`freq`	Frequency of operation [Hz].
`r`	Resistance [Ω] (see Figure 1-60).

Internally, the phase shifter is implemented using the RC-CR circuit as shown in Figure 1-60. While the phase difference is also 90-degrees when the `lead` and `lag` have the same output impedance, only at the operating frequency do the magnitudes remain the same. This circuit network also generates white noise.

Figure 1-60 Phase Shifter



There are two buffered versions of the shifter:

- The `shifter_combiner` combines two signals so that they add if one leads the other by 90 degrees and so that they cancel if it lags by 90 degrees.
- The `shifter_splitter` splits a signal into two signals 90 degrees out of phase with each other.
- You specify the input and output impedances. These networks are noiseless.

Testbenches Category

The `testbenches` category contains the test circuits used to define model specifications. Where possible, the element names are in terms of standard RF measurements. The most precise way to describe a measurement is with a test circuit, set up instructions, and sample measurements. The circuits in the `testbenches` category serve this purpose.

The components in the `testbenches` category are:

- AM_PM_test_ckt
- ava_pwr_gain
- BB_ind_cap_test
- demod_ip3
- dwn_cnvt_test
- mixer_ip3
- mod_1dbcp
- mod_demod_test
- noise_figure
- one_db_cp
- PB_BB_filter_comparison
- PB_ind_cap_test
- quad_and_phase_error_demo
- shifter_combiner_test
- shifter_splitter_test
- up_cnvt_test
- view_switching

AM_PM_test_ckt

(AM/PM Conversion Parameters)

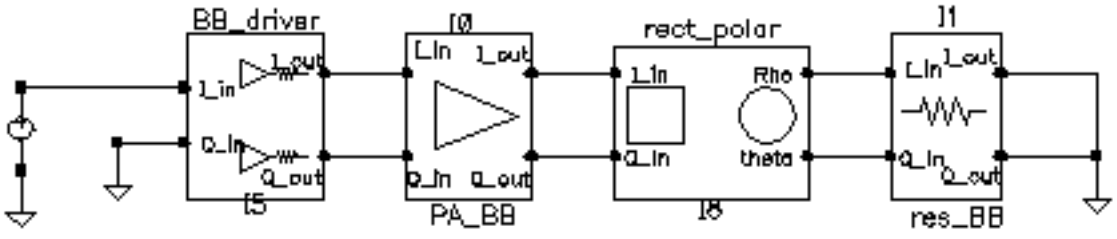
Only the baseband models include the four parameters for AM/PM conversion.

Table 1-1 AM/PM Conversion Parameters for Baseband Models

AM/PM Parameter	Definition
`AM/PM Sharpness`	Defines how steep the output phase shift changes are with respect to input power.
`{1, 0, -1}` for `{cw, none, ccw}`	Defines the direction of the phase shift. 1 for clockwise, 0 for no phase shift, -1 for counter clockwise.
`|radians|@1dB cp`	Defines the absolute value of the output phase shift at the 1dB compression point for power amplifiers. This is the phase shift at an arbitrary output power level for some models.
`|radians|@big input`	Defines the absolute value of the output phase shift as input power goes to infinity (if it could go to infinity).

The test circuit in Figure 1-61 is listed as `am_pm_test_ckt` in the `testbenches` category in `rfLib`.

Figure 1-61 The am_pm_test_ckt Circuit



In the `am_pm_test_ckt` test circuit,

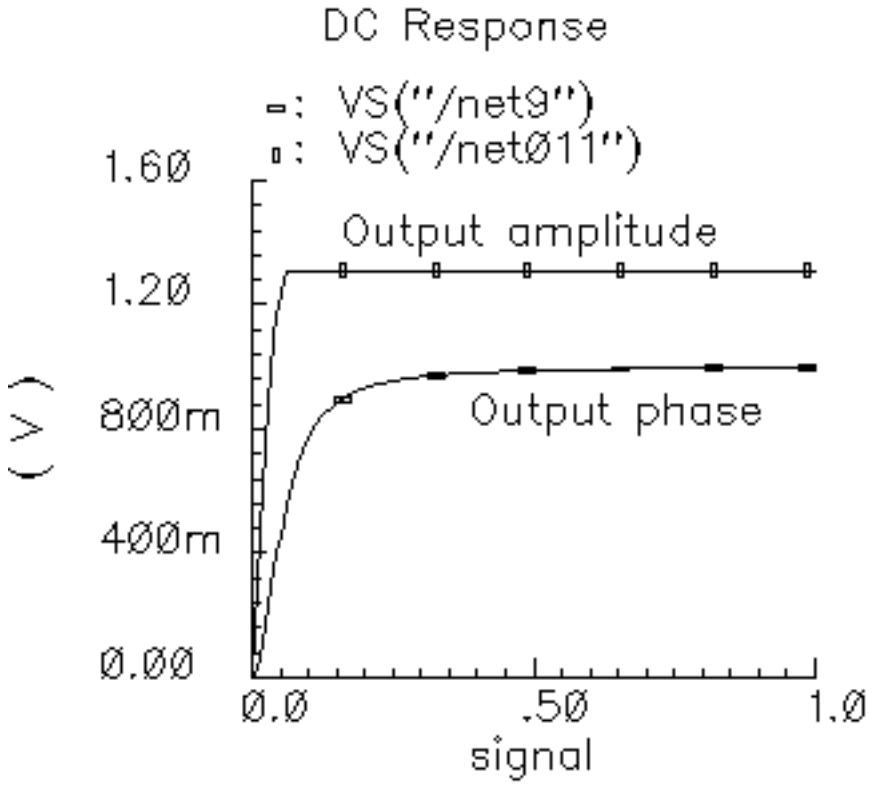
- The first block (`BB_driver`) scales the control voltage generated by the leftmost element so that the output equals the specified dBm when the control voltage equals 1

volt. This is done so you can specify maximum dBm but still sweep linearly from zero signal.

- The second block (`PA_BB`) is a power amplifier.
- The third block (`rect_polar`) transforms the rectangular description of the baseband signal into polar coordinates so you can observe the phase shift and output signal level directly.

Figure 1-62 shows the output amplitude and phase as functions of the input signal level. Generate these with a swept DC analysis. Sweep the *signal* variable from 0 to 1 in 200 linear steps and display the *rect_polar* outputs.

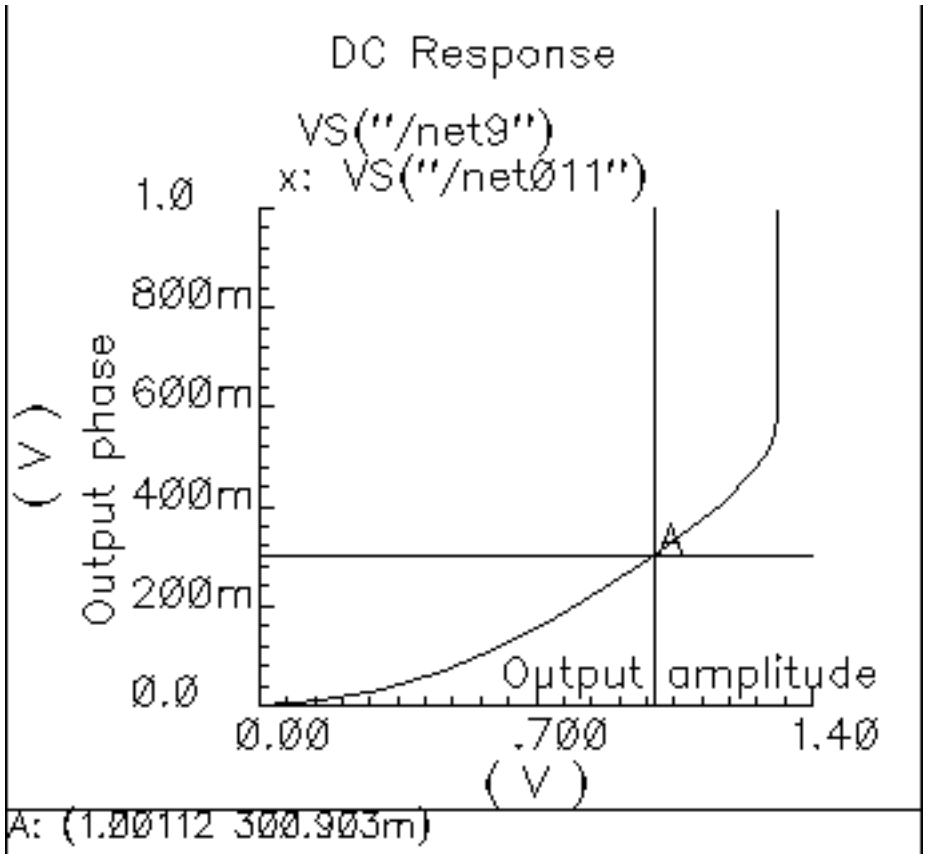
Figure 1-62 Output Amplitude and Phase



By changing the x-axis to be the output amplitude trace, you can confirm that the phase shift at the output referred 1 dB compression point of 10dBm (or 1 volt peak across a 50 ohm load) equals 0.3 radians, as specified. Figure 1-63 shows the plot.

Note that the measured power across the load is as specified only when the load matches the amplifier output resistance. If you mismatch the load you do not measure the specified phase shift at the specified output power level.

Figure 1-63 Output Phase Versus Output Amplitude



In the next three figures, output phase is plotted against input signal level. Each plot shows the effect of one of the AM/PM conversion parameters. You can generate the plots by applying the Parametric Tool to the existing analysis.

Figure 1-64 shows the effect of the `|radians|@1 db cp` parameter. Sweep `rad_cp` from 10 m to 100 m in 5 linear steps.

Figure 1-64 Output Modified by the |radians|@1 db cp Parameter

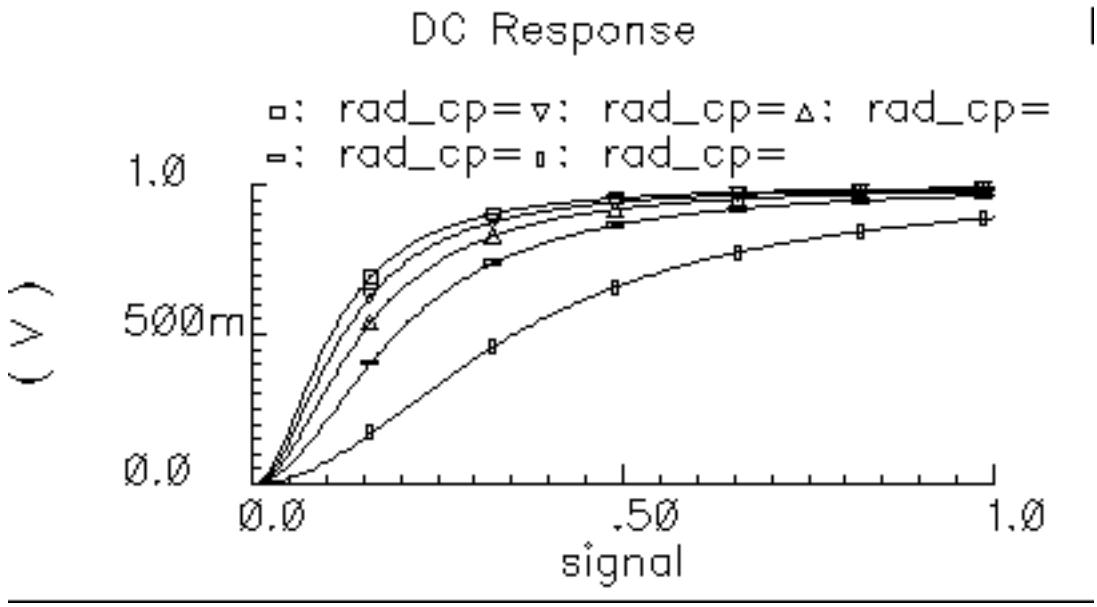


Figure 1-65 shows the effect of the `am/pm sharpness` parameter. Sweep `sharpness` from 1 to 6 in 5 linear steps.

Figure 1-65 Output Modified by the Sharpness Parameter

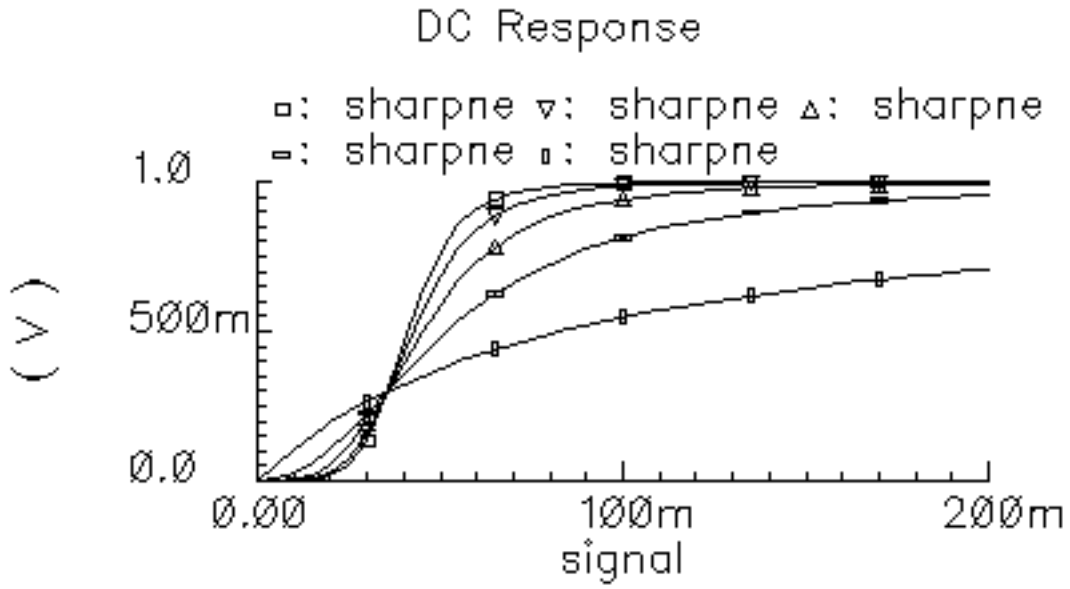
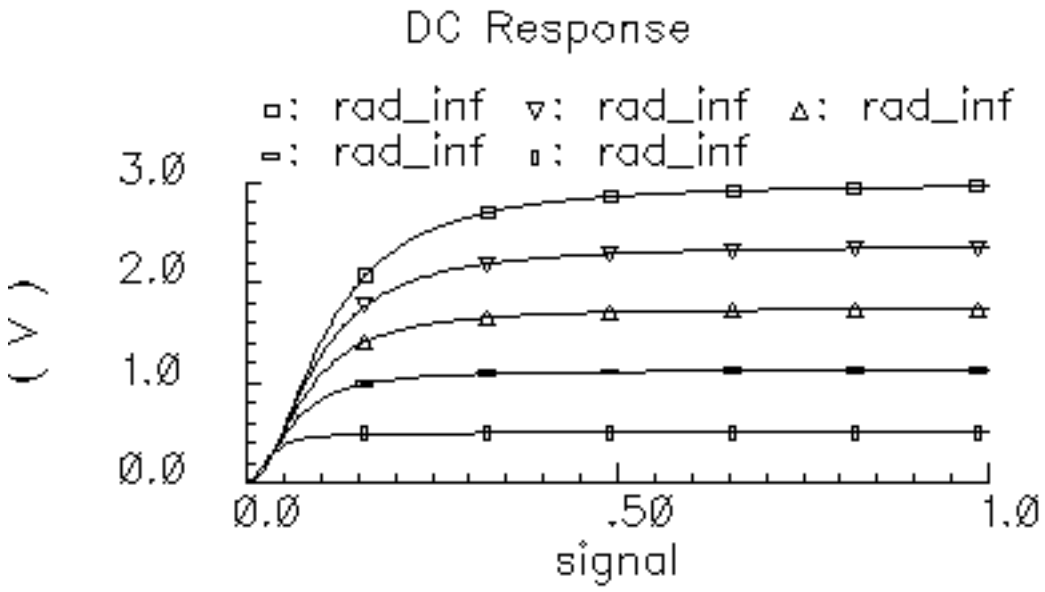


Figure 1-66 shows the effect of the `rad_inf` parameter. Sweep `rad_inf` from 0.5 to 3 in 5 linear steps.

Figure 1-66 Output Modified by the rad_inf Parameter



ava_pwr_gain

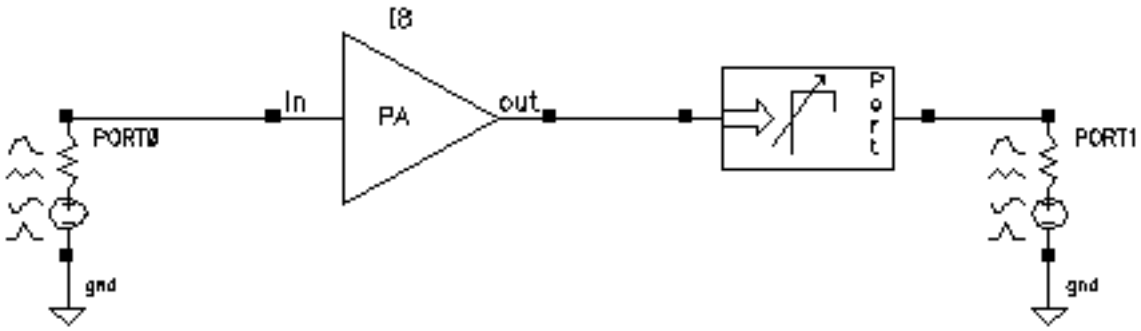
(Available Power Gain Parameter)

When an amplifier's load is equal to it's output resistance, available power gain equals the following

$$10 \times \log\left(\frac{outputpower}{inputpower}\right)$$

The test circuit in Figure 1-67 is listed as `ava_pwr_gain` in the `testbenches` category in `rfLib`.

Figure 1-67 The ava_pwr_gain Circuit



Computing Constant Power Contours

The `ava_pwr_gain` test circuit is set up to compute constant power contours. As you would expect, maximum power transfer occurs when the load and output impedances are matched. The port adapter inserts reactive elements into the signal path to load the amplifier with the specified reflection coefficient.

Figure 1-68 on page 158 shows a Smith Chart that displays how the load power varies with the load refection coefficient.

The load pull contours were computed by

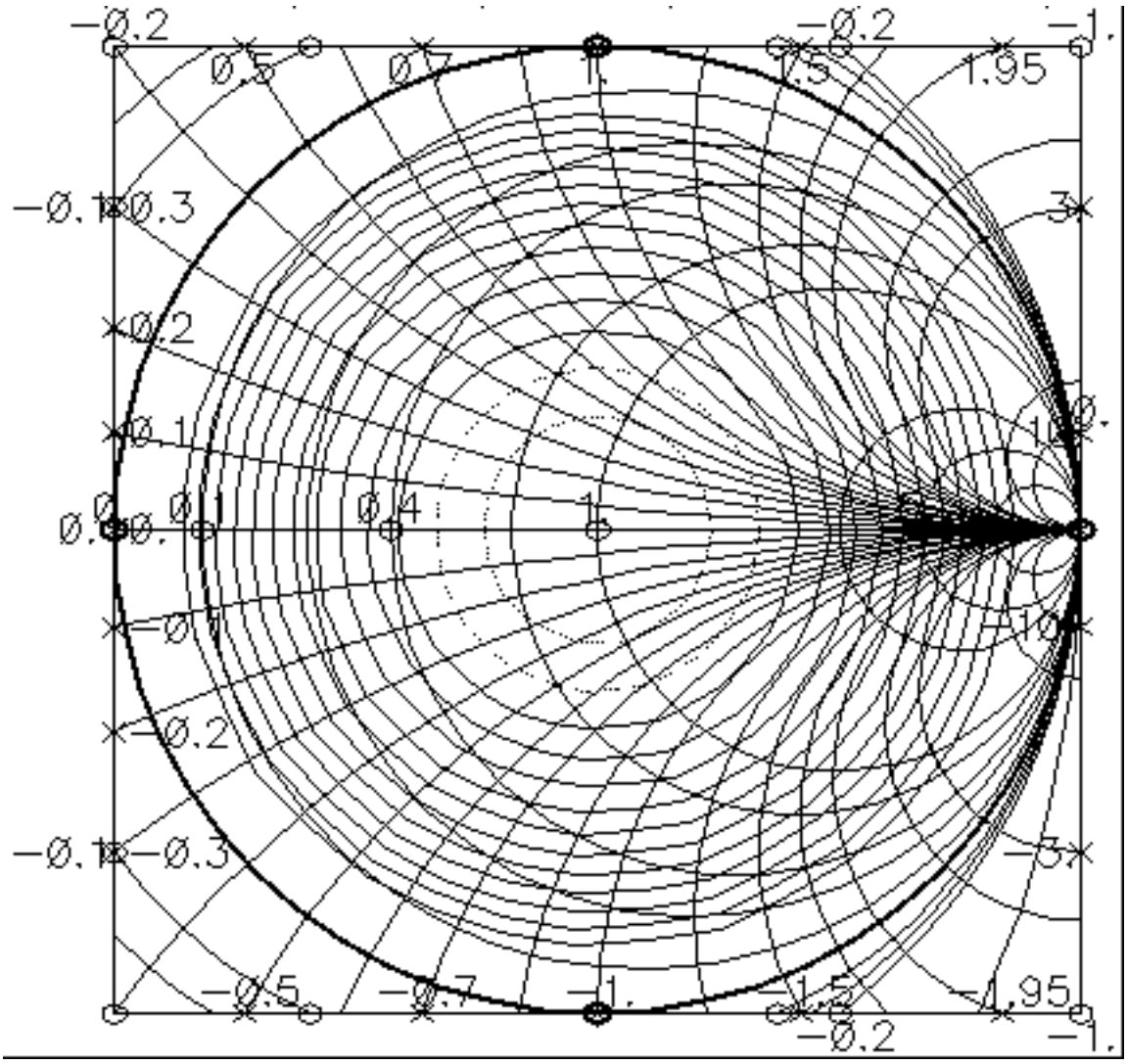
- Sweeping the `pp` parameter in a PSS analysis (`pp` is the *phase* of the reflection coefficient)

- Sweeping the `mm` parameter with the Parametric Tool (`mm` is the *magnitude* of the reflection coefficient)

The *load reflection coefficient* is defined with reference to the amplifier output resistance, `300` Ohms in this case. The amplifier input resistance is `20` Ohms. The input source resistance is `50` hms. The amplifier 1 dB compression point is set high enough to make the amplifier linear. The available power gain parameter is `20` dB.

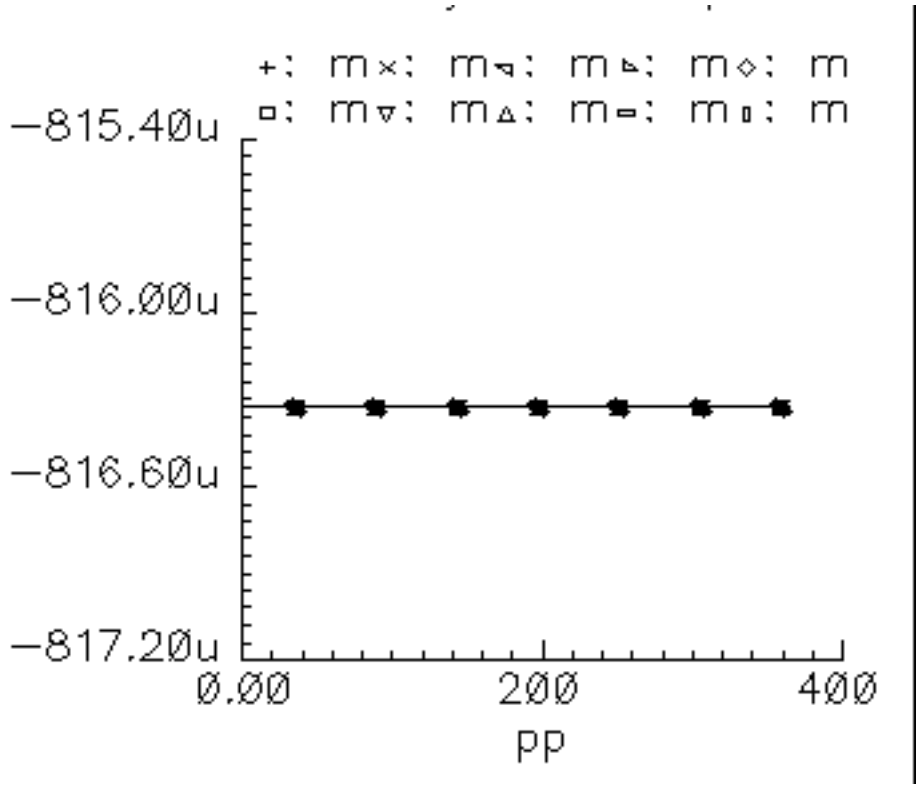
To generate the load pull contours you must save *both* the current flowing into the port adapter (`port`) and the current flowing into `Port0`.

Figure 1-68 Smith Chart



When you place the cursor on the smallest contour on the Smith Chart, you can see that the amplifier delivers a maximum power of `81.63` mW to an optimum load of `300` Ohms (reflection coefficient = 0). When you plot the magnitude of the power coming from the input port against the sweep variable (`pp`, phase of the reflection coefficient) you find that input power equals `816.3` uW, independent of load, as shown in Figure 1-69. The ratio of maximum output to input power equals `100`, or dB, as specified.

Figure 1-69 Input power



Note that the voltage gain in this test circuit does not equal 10 because the amplifier's input and output resistances are different. You can verify that the ratio of the output to input voltage is as follows

$$10\sqrt{\frac{R_{out}}{R_{in}}}$$

where, R_{out} is the amplifier output resistance and R_{in} is the amplifier input resistance. This assumes the amplifier is not driven into non-linear operation.

The input and output resistances specify the current drawn by the associated terminals as a linear function of terminal voltage. There is no test circuit for terminal resistances because the definition is so simple.

BB_ind_cap_test

(RLC Test Circuits)

The two circuits discussed below demonstrate how passband and baseband reactive elements are related. The circuit in Figure 1-70 shows a simple passband RLC circuit driven by a modulated carrier. The circuit in Figure 1-71 shows the associated baseband equivalent circuit model. The circuits are `PB_ind_cap_test` and `BB_ind_cap_test`. Both circuits reside in the `rfLib` under the `testbenches` category.

Figure 1-70 Simple Passband RLC Circuit

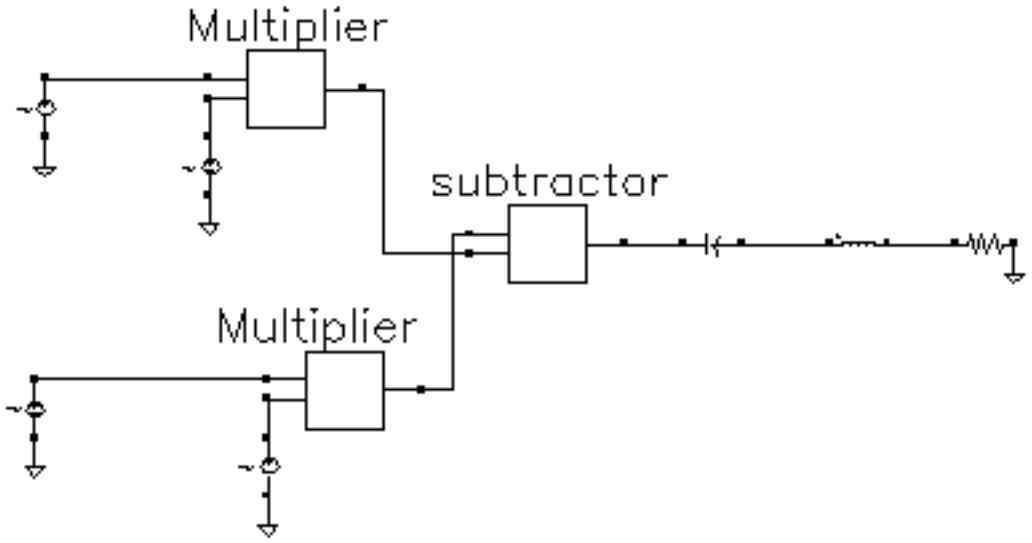
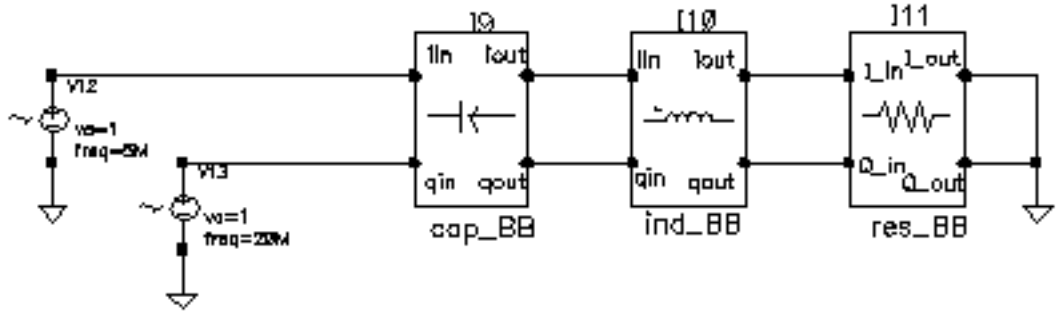


Figure 1-71 Baseband Equivalent To Figure 1-63



The following steps explain how to simulate each circuit and overlay the results.

1. Recall the `PB_ind_cap_test` circuit and bring up an analog design environment window. Set up a 200 ns Envelope analysis. Select `carrier` as the *Clock Name*. Set the *Output Harmonics* to 1.
2. Run the analysis and plot the real and imaginary parts of the *harmonic-time voltage* across the resistor. Use `1` for the harmonic number.
3. Recall the `BB_ind_cap_test` circuit and run a 200 ns transient analysis. Note the faster run time. That is the whole point to suppressing the carrier but it is only useful if the results match. Plot the `I_in` and `Q_in` voltages of the resistor model.
4. To overlay the results, bring up a waveform calculator.
5. Click the *wave* button on the calculator then click one of the Envelope waveforms. If the waveform turns yellow you may have to hit the escape button a few times and click *clear* and *clst* a couple of times in the calculator then try again.
6. Make active the waveform display tool with the transient results then click *Plot* in the calculator.
7. Repeat the last two steps for the other Envelope waveform. You should see the waveforms in Figure 1-72. The two models agree very well. The resonant frequency of the series RLC branch is just over 500 MHz. Only by riding on a carrier can the 5 MHz and 20 Mhz baseband signals propagate to the resistor at their original voltage levels. The baseband model accurately predicts the effects of the RLC circuit on the baseband signal. There are two effects, one due to phase shift at the carrier frequency and one due to filtering of the baseband signal itself.
8. In the waveform display tool that overlays the results, change the x-axis to be one of the I-signals. You should get the picture shown in Figure 1-73. The tilt in the resulting Lissajous plot indicates phase shift at the carrier frequency but not at the baseband frequencies. The aspect ratio of the Lissajous figure indicates the 20 MHz component is attenuated more than the 5 MHz component. The baseband model captures both effects well.

Figure 1-72 Waveforms

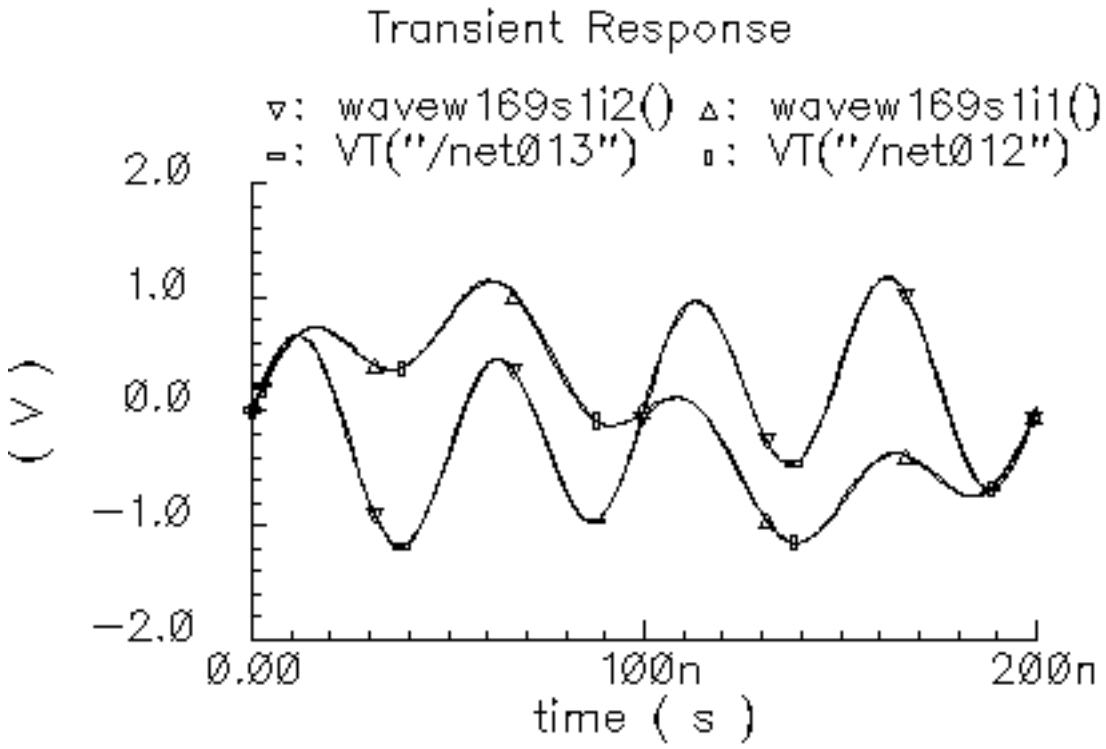
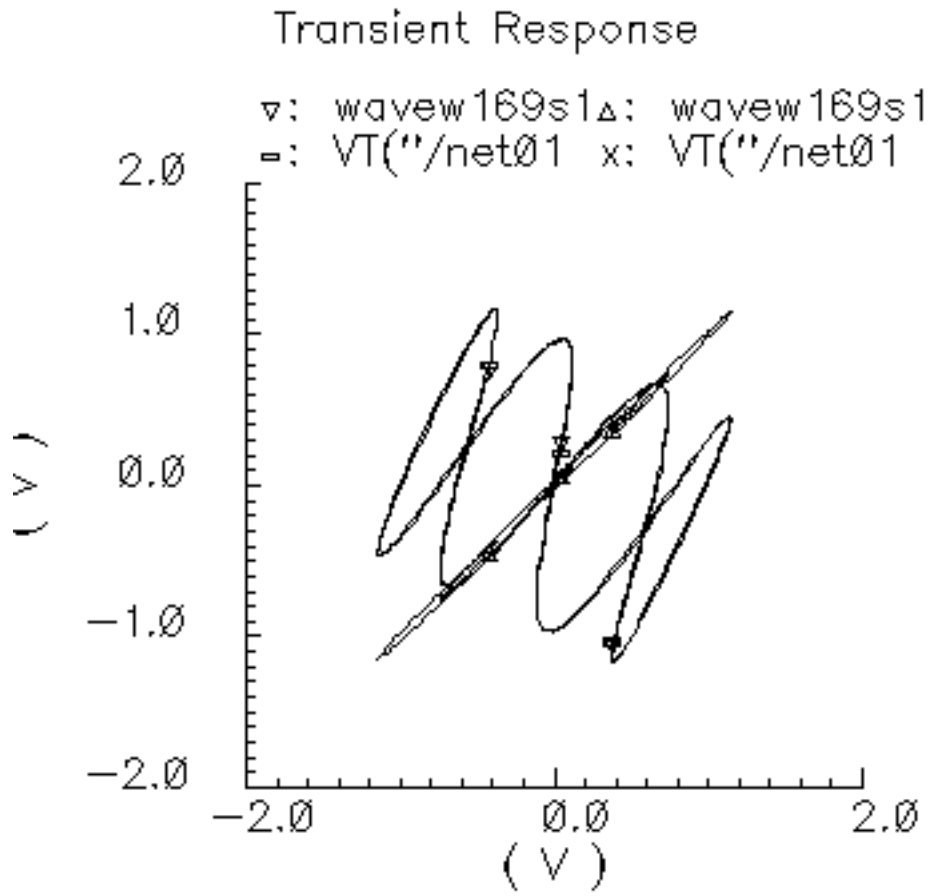


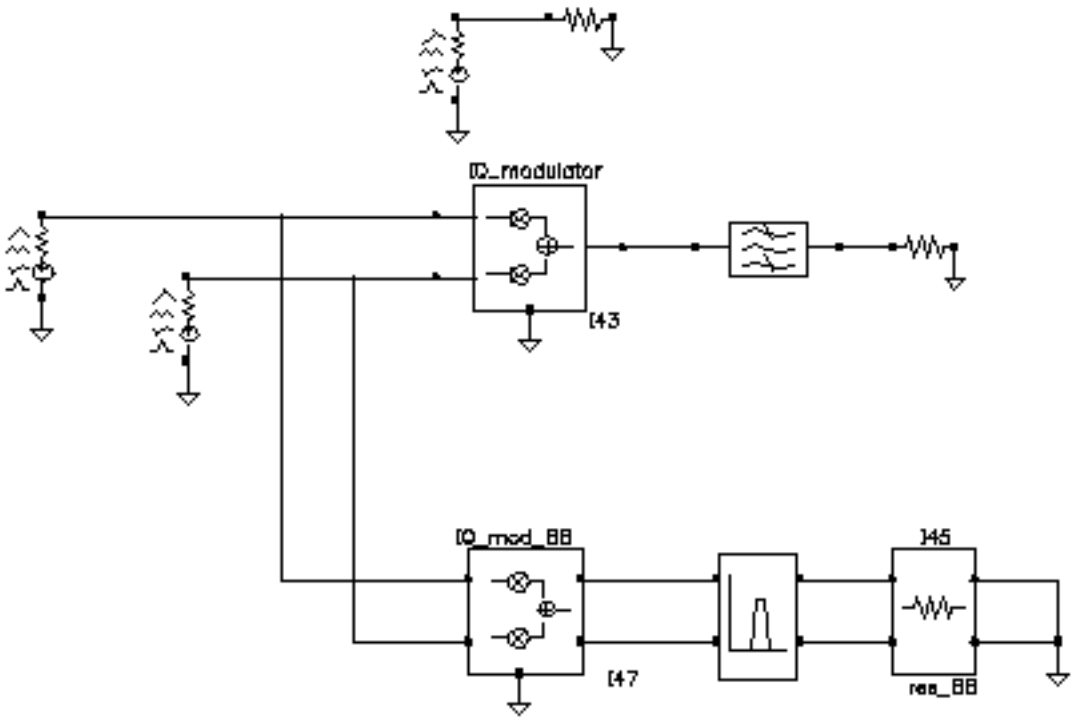
Figure 1-73 Lissajous plot



Comparison of Baseband and Passband Models

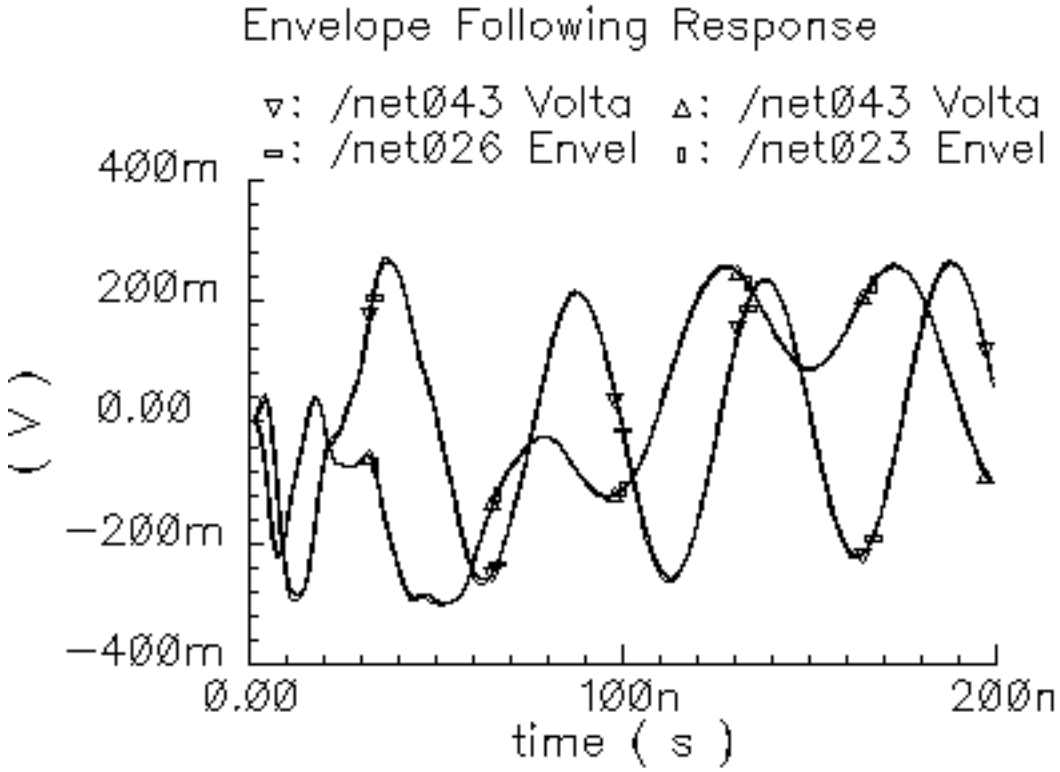
The circuit in Figure 1-74 on page 165 shows how well the baseband and passband filters agree. The I-input is a 5MHz 1 volt peak sinusoid and the Q-input signal is a 20MHz 1 volt peak signal. The filter has a center frequency of 1.1GHz and a relative bandwidth of 0.1. The modulator LO is 1GHz. To make the analysis more interesting the carrier is not exactly aligned with the filter center frequency and the terminals are not matched. The circuit is listed as `PB_BB_filter_comparison` in the testbenches category of the `rfLib`.

Figure 1-74 PB_BB_filter_comparison Circuit



1. Bring up the test circuit and an Analog Environment window.
2. Set up an Envelope analysis with “carrier” as the Clock Name. Set reltol in the analog options to 1e-5. You can use the default reltol of 1e-3 but you do not get the waveforms close to the baseband results.
3. Plot the “time” waveforms of the BB_butterworth_bp outputs. These waveforms are the response of the baseband equivalent model.
4. Plot the “harmonic time”, 1 harmonic, real and imaginary waveforms at the butterworth_lp output. These waveforms are the baseband waveforms extracted from a passband model. Figure 1-75 on page 166 overlays the baseband and passband results. The baseband and passband filter models produce identical equivalent baseband waveforms. The slight offset in time is due to the ambiguity associated with deciding whether to plot a time-varying Fourier coefficient at the beginning or at the end of a clock cycle.

Figure 1-75 I and Q Baseband Equivalent Outputs

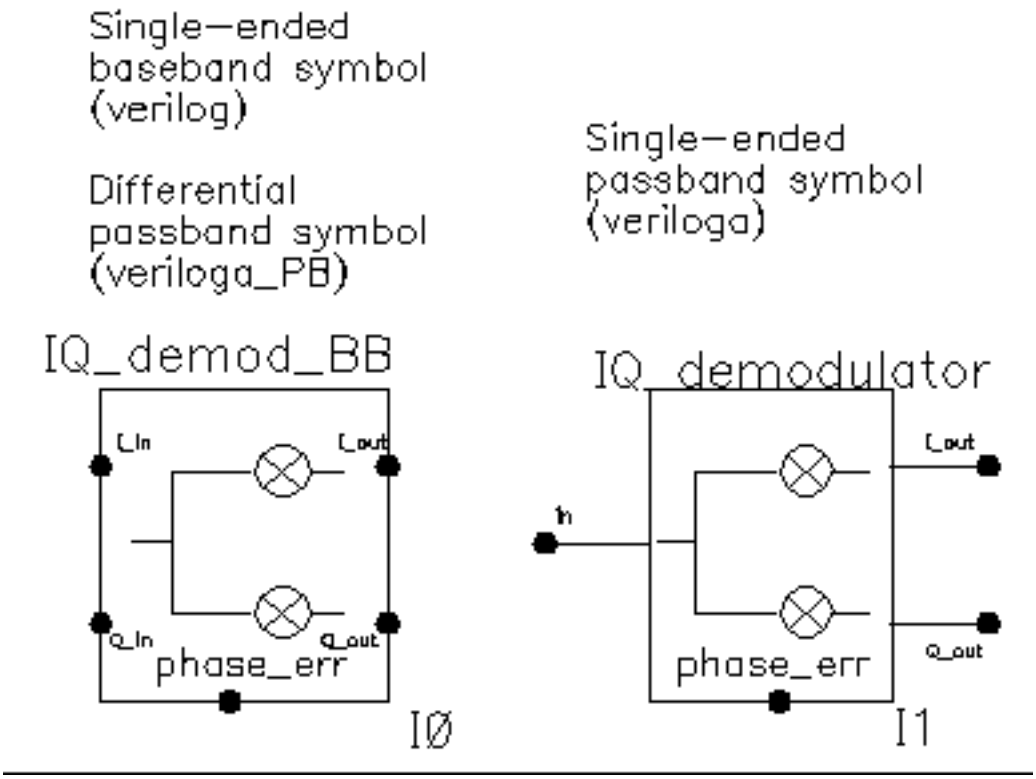


demod_ip3

(IQ Demodulator)

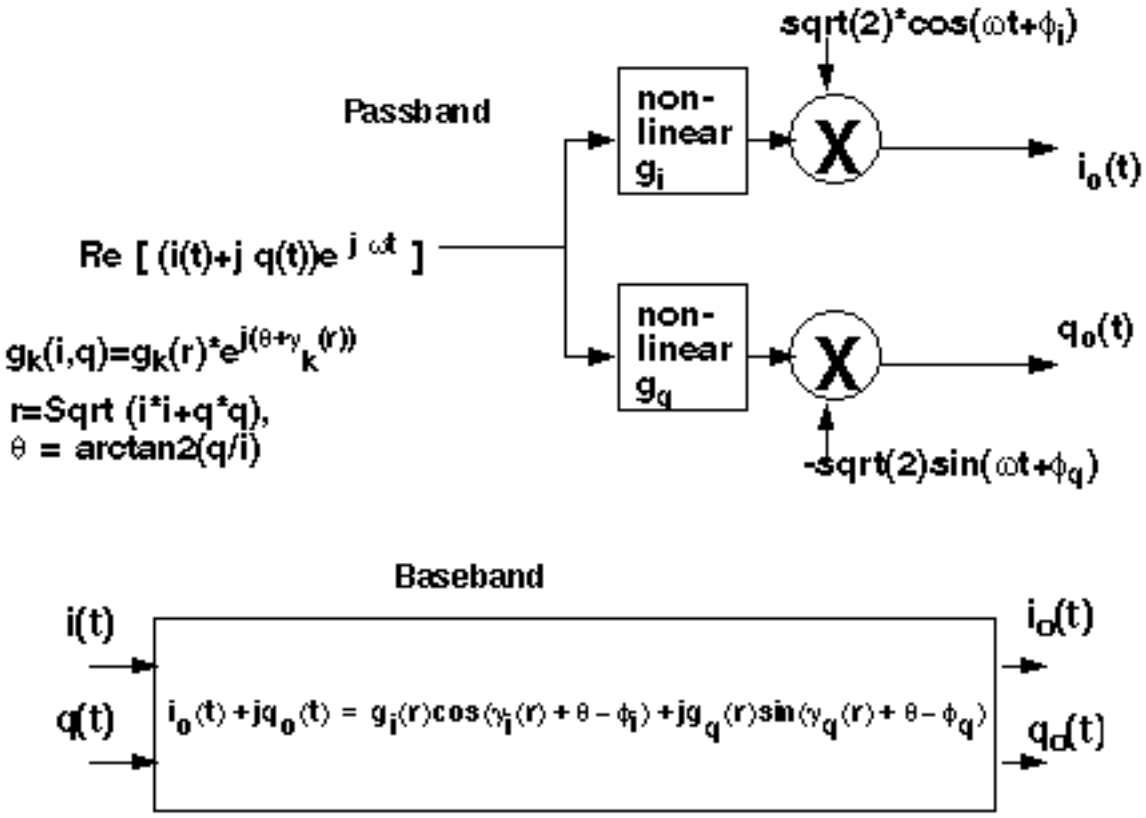
```
(baseband = IQ_demod_BB, passband = IQ_demodulator)
```

Figure 1-76 Baseband and Passband IQ Demodulator Models



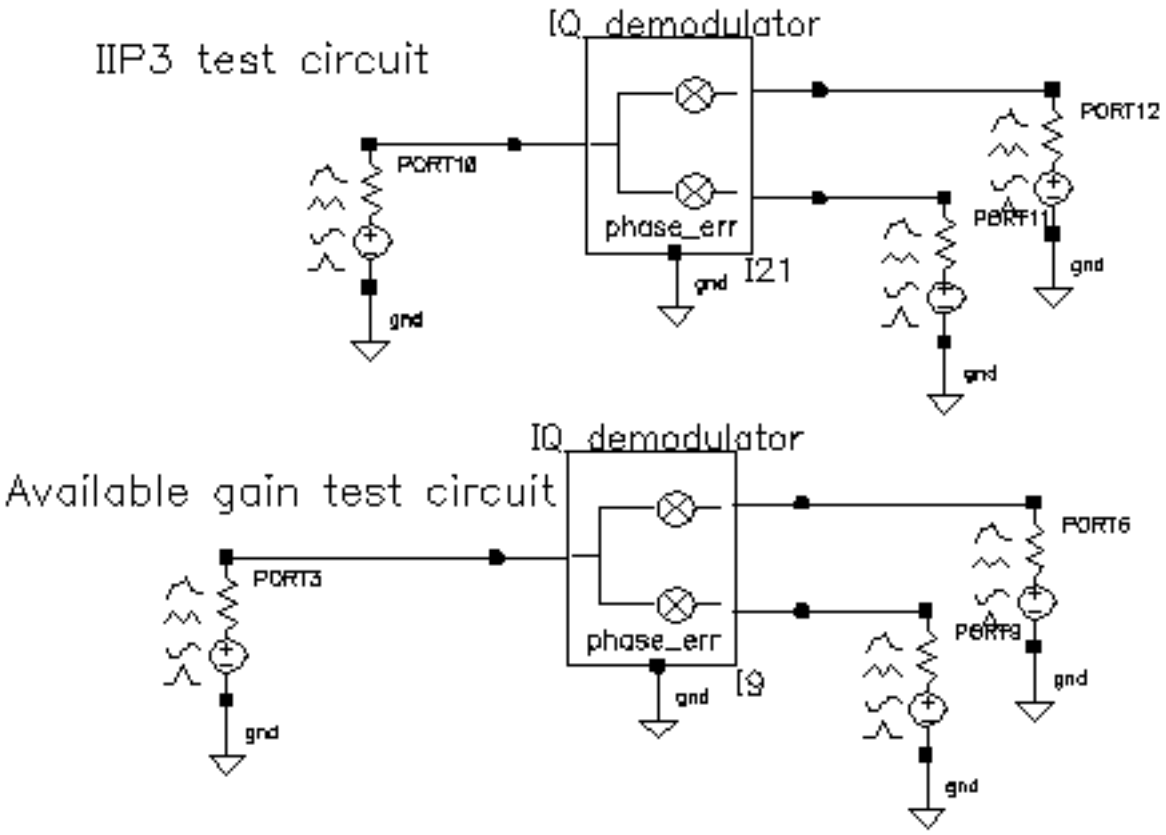
The `IQ_demodulator` converts RF (or IF) to baseband. Figure 1-77 on page 168 shows exactly what the passband demodulator model does. The parameters are like those in the modulator blocks except saturation is specified by input referred IP3 instead of by 1 dB compression point. IP3 was chosen over the 1 dB compression point for specifying saturation because the demodulator usually lies in the receive path and receiver blocks are usually specified with IP3.

Figure 1-77 IQ Demodulator Calculations



The circuit called `demod_ip3` in the `testbenches` category of the `rfLib` shows how the gain and IP3 parameters are defined. Figure 1-78 shows the schematic. Both the input and the output resistances are matched.

Figure 1-78 The demod_IP3 Schematic

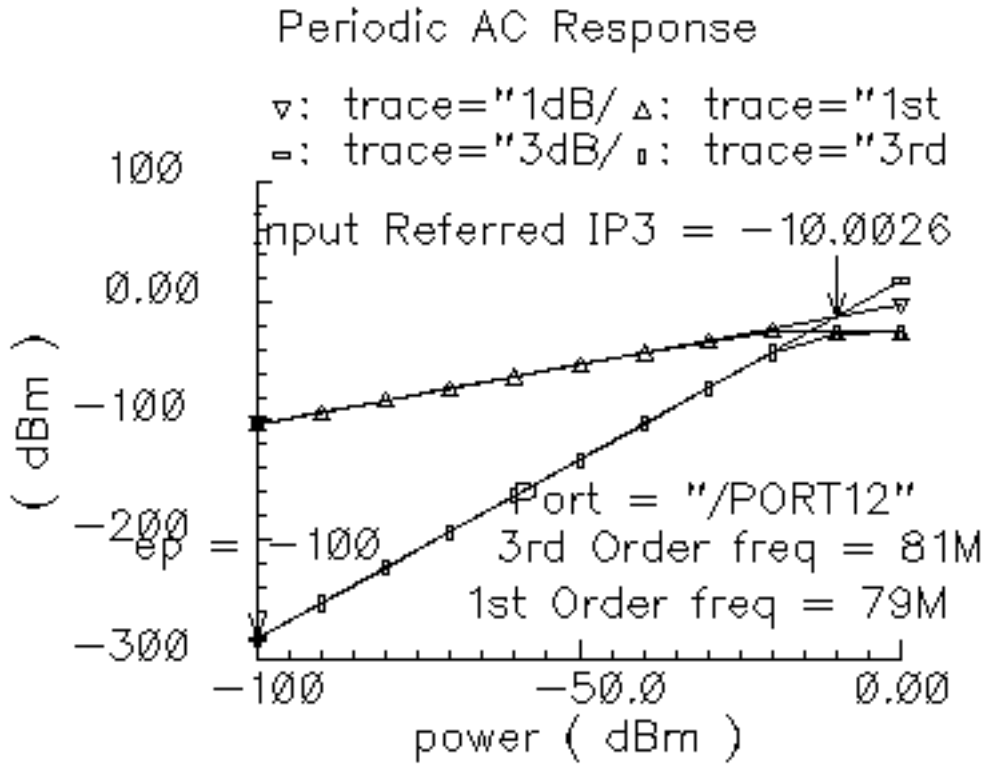


1. Recall the `demod_IP3` circuit and set up a swept PSS analysis. Let the *Beat Frequency* be *Auto Calculated*. Keep 2 harmonics. Sweep the *power* parameter from -100 to 0 in 10 linear steps.

2. Set up a single point PAC analysis at 921 MHz and keep the -25 and -21 sidebands.

3. After running the analysis, from the PAC output window plot the input-referred IP3 curves with 81 MHz as the 3rd order sideband and 79 MHz as the 1st order sideband. The procedure is similar to the mixer IP3 example covered in "(IQ Modulator Models)" on page 36. Use *Variable Sweep* for the *Circuit Input Power* and -100 for the *Extrapolation point*. Make sure to plot Input Referred IP3. Click the `I-output` port in the top circuit. You should see -10 dBm as the IP3, just as specified. Figure 1-79 shows the IP3 plot. Note that 1st order line indicates the gain is 3dB below the specified gain of 0 dB. That is because not all of the power lies at 1000 MHz-921 MHz = 79 MHz; Some of the power lies at 1000 MHz + 921 MHz = 1921 MHz. Use the bottom test circuit to measure available power gain. The bottom circuit drives the demodulator at the same frequency as the demodulator's internal local oscillator, which runs at 1 GHz. Now the output power is not split, it lies in the zero harmonic of the `I-output`.

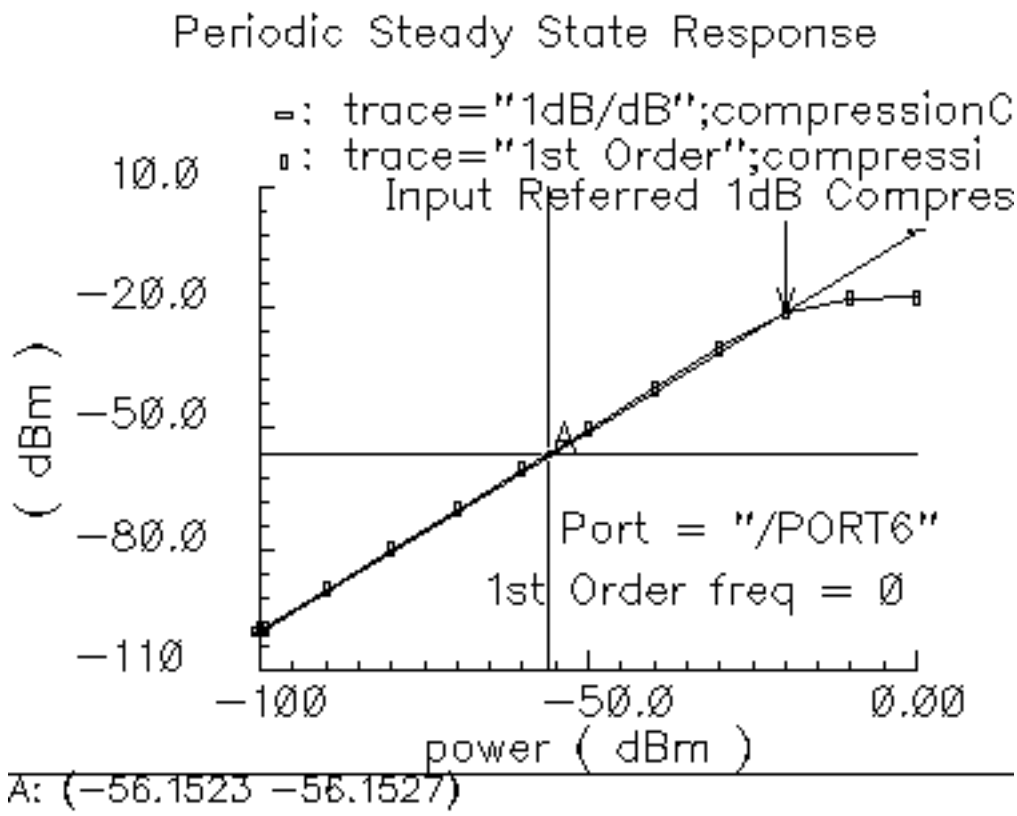
Figure 1-79 Demodulator IP3



4. Plot the 1dB compression point at the port loading the `I-output` of the bottom circuit. Use the zeroth harmonic. The ratio of output to input power should be unity in the linear region. Figure 1-80 shows the compression point plot. The measured 1dB compression point is of no use in this test. We want the gain. At low power levels where the gain is constant, the gain is as specified.

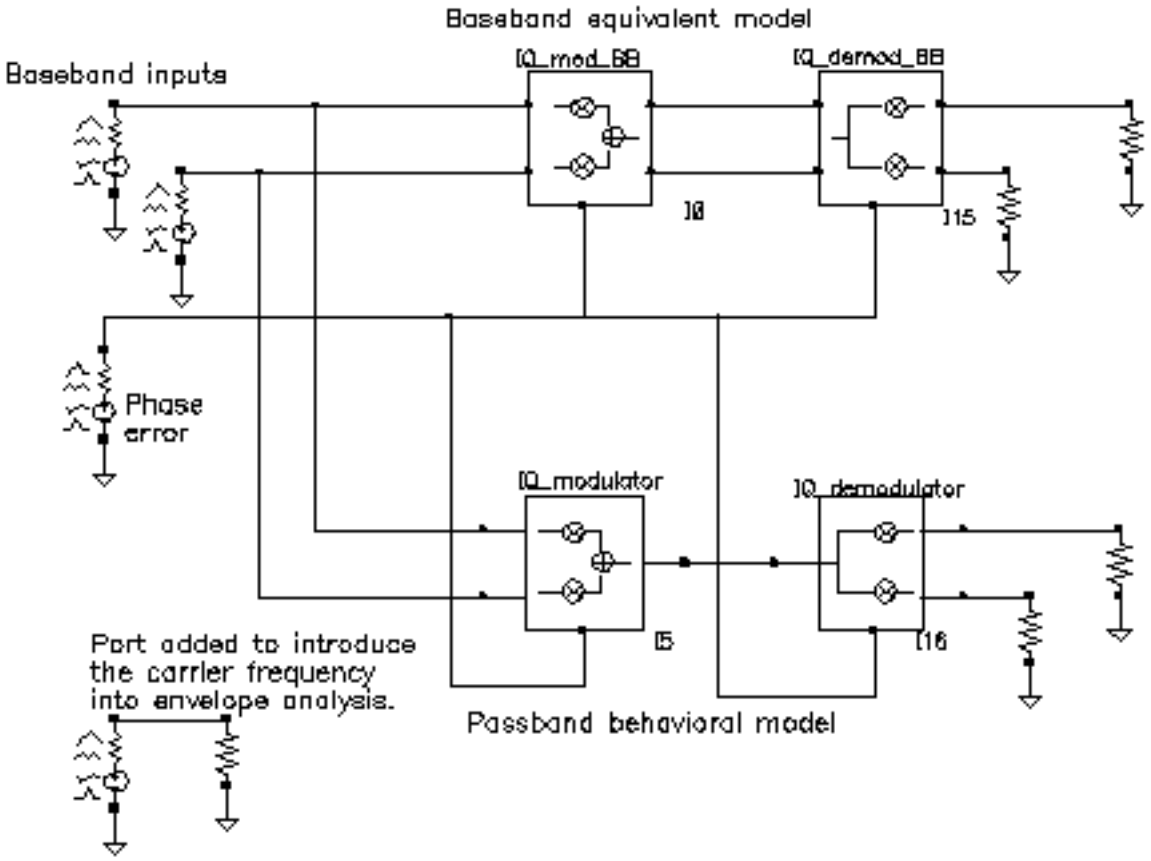
5. Remember, in this test circuit the load resistance and output resistance are equal so that the output power is maximal. Also, the input resistance equals the source resistance so that the horizontal axis truly equals input power.

Figure 1-80 Demodulator Available Power Gain



Phase errors behave like their counterparts in the modulator models except for a change of sign. Quadrature error behaves exactly as it does in the modulator models. Figure 1-81 shows a test circuit for illustrating the relationships between phase error and quadrature error in the modulators and demodulators. The test circuit is called `mod_demod_test` and is listed in the `testbenches` category. The test circuit also shows that the passband and baseband models give comparable results, as they should, as long as the passband carrier is not severely clipped. The baseband input trajectory is a complex 1 MHz tone, which produces a circular input trajectory. The demodulator outputs are not matched and are not symmetric with respect to I and Q paths. The modulators and demodulators are not perfectly linear and the non-linearities are asymmetric with respect to I and Q. The modulators and demodulators are driven by the same phase error and the quadrature error parameters are a common variable set to 0.785 radians.

Figure 1-81 mod_demod_test Circuit



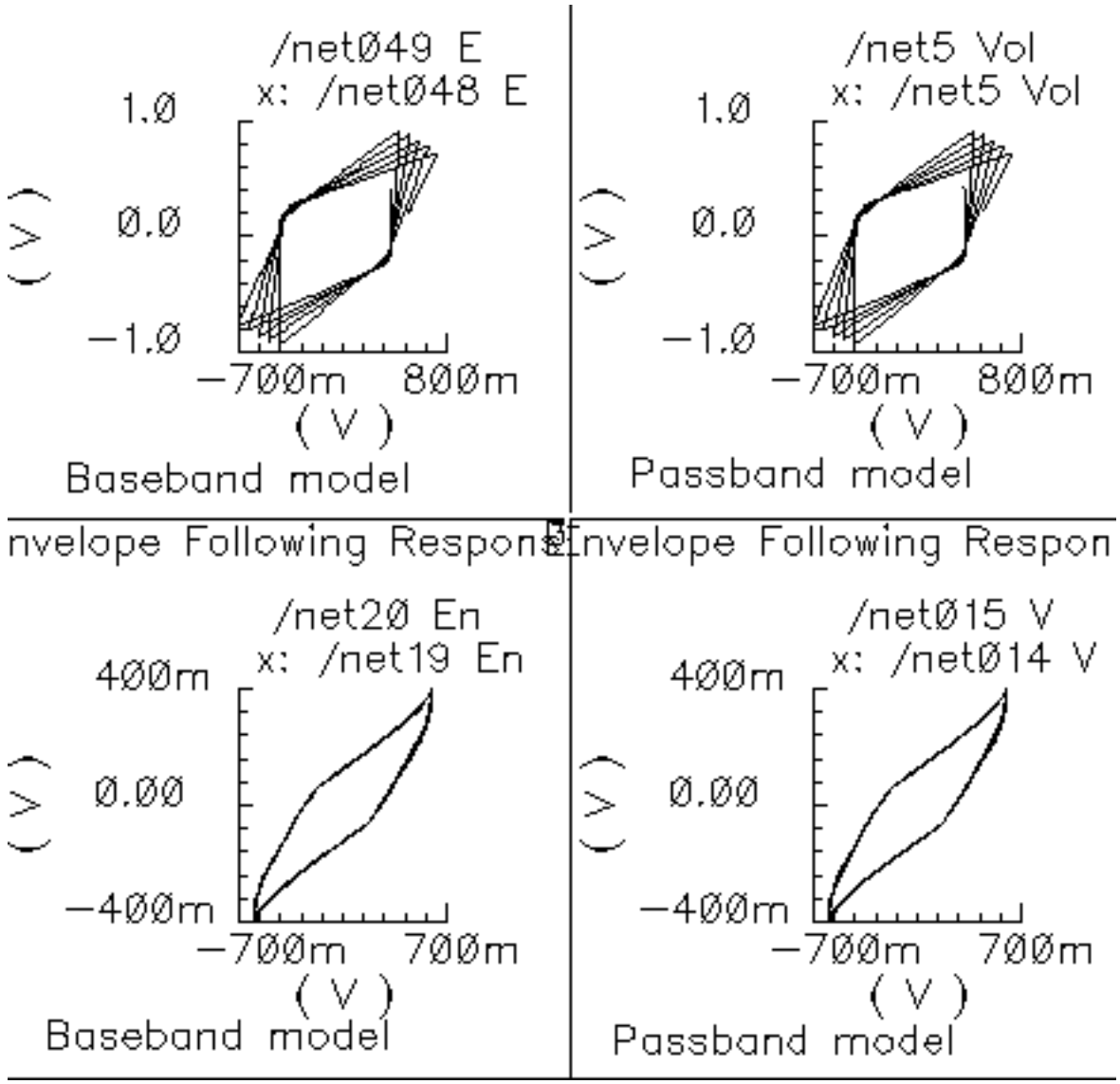
To use the `mod_demod_test` circuit:

1. Recall the circuit and set up a 5 us Envelope analysis with `carrier` as the *Clock Name*.
2. After the analysis completes, plot the `IQ_mod_BB` outputs and make the `I_out` signal the x-axis.
3. Open a subwindow and in it, plot the harmonic time waveforms of the `IQ_modulator` output. Use the first harmonic and plot the real and imaginary waveforms. Make the real waveform the x-axis.
4. Open a third subwindow and stretch the Waveform Display window so that the third subwindow appears below the first window.
5. Plot the time waveforms at the `IQ_demod_BB` outputs and make the `I_out` waveform the x-axis.

6. Open a fourth subwindow and plot the harmonic time results at the `IQ_demodulator` outputs but this time use the zeroth harmonic and only plot the real parts. Make the `I_out` waveform the x-axis. Figure 1-82 shows what you should now see.

The leftmost pictures are from the baseband models and the rightmost are from the passband models. Passband and baseband models agree quite well. The top pictures are the voltages at nodes that lie between the modulator and demodulator. Quadrature error squashes the baseband trajectory at that node. The trajectory precesses because phase error ramps up linearly with time just like in the last test. The non-linearities produce the sharp corners. The bottom trajectories do not precess because the same phase error rotates the demodulator output in the reverse direction; driven by the same phase error ramp, the demodulator undoes the precession introduced in the modulator. The demodulator outputs are nearly in phase because the quadrature errors of pi/4 in the modulators and demodulators add to give a total quadrature error of $\pi/2$, which in this case puts the baseband I and Q outputs nearly in phase with each other.

Figure 1-82 mod_demod Results

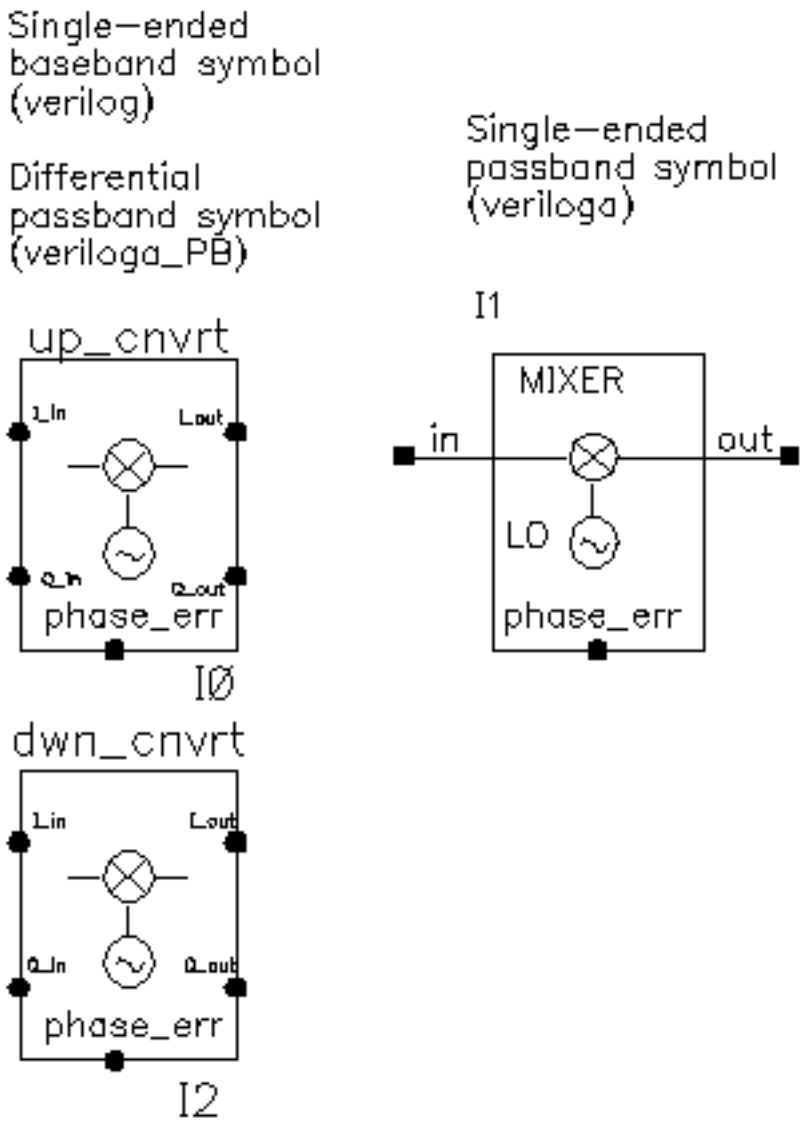


dwn_cnvt_test

(RF-to-IF and IF-to-RF Mixers)

`(passband = MIXER_PB, baseband = dwn_cnvrt` and `up_cnvrt)`

Figure 1-83 Baseband and Passband Mixer Models



`MIXER_PB` is a passband model that converts RF to IF and IF to RF. `dwn_cnvrt` model is a baseband equivalent model of a mixer used to convert from RF to IF. `up_cnvrt` model is a baseband equivalent model of a mixer used to convert from IF to RF. There are some minor

differences in the baseband models that depend on whether conversion is up or down. Figure 1-84 on page 176 and Figure 1-85 on page 176 show what the models do.

Figure 1-84 Calculations for up_cnvrt Mixer

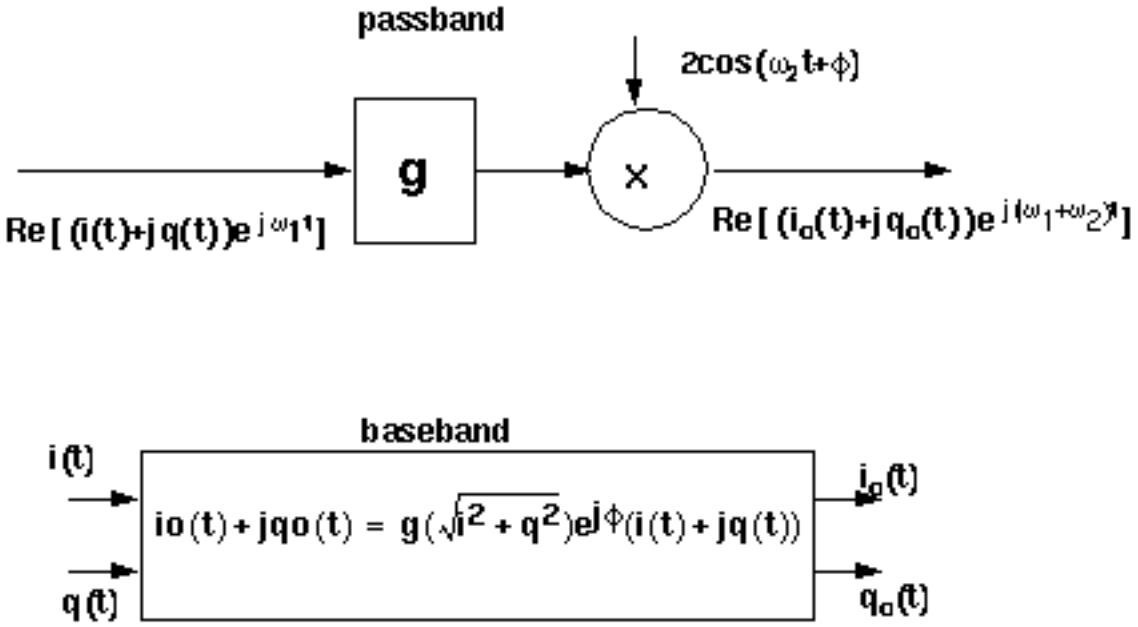
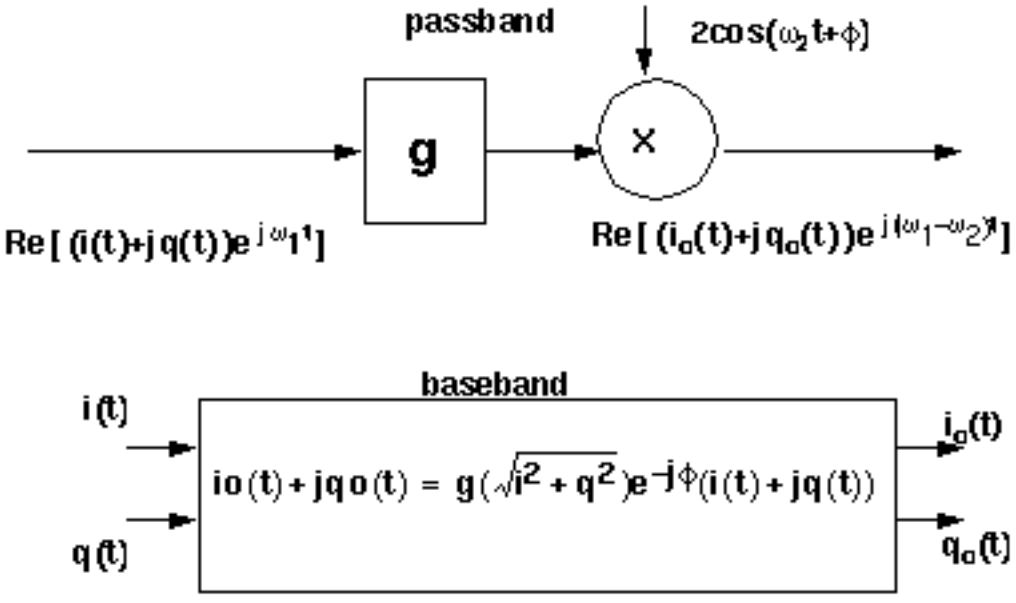


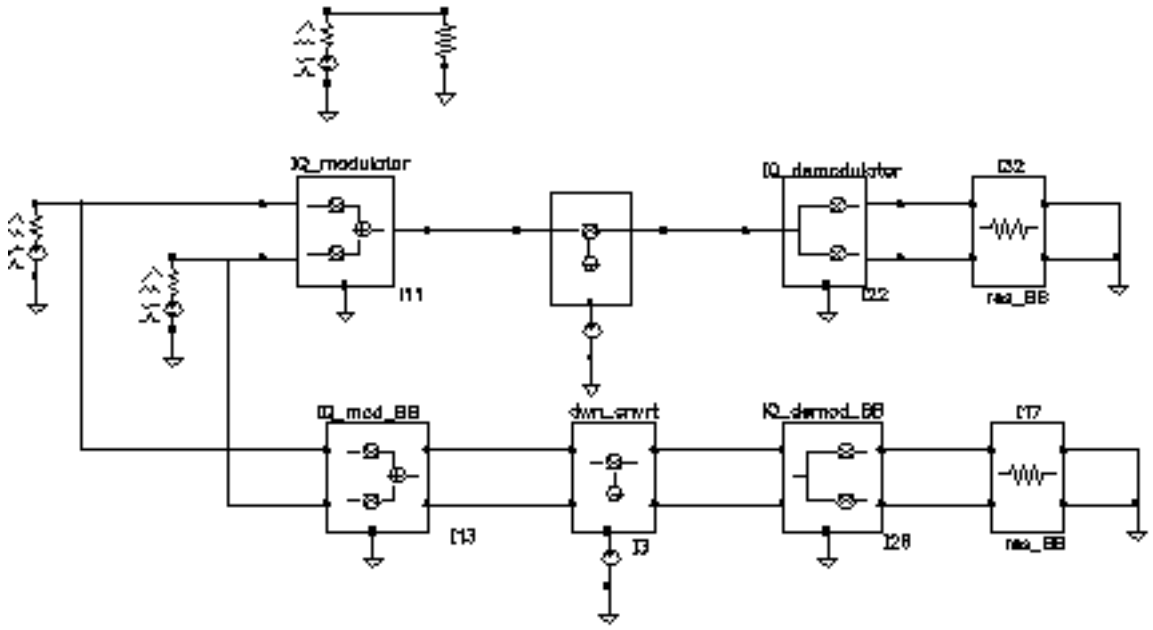
Figure 1-85 Calculations for dwn_cnvrt Mixer



The noise figure and IP3 parameters are defined in “(IP3 Parameter)” on page 180. Unlike the `IQ_demodulator`, the IP3 test circuit can be used to define the available power gain because the gain is defined from the input frequency to just one sideband.

Typically the mixer would be used to create an IF stage. In that case, it is difficult to obtain a simple (i.e. filterless) envelope analysis that overlays waveforms to show how well baseband and passband models agree. The test circuit shown in Figure 1-86, which is listed as `dwn_cnvt_test` in the `testbenches` category of the `rfLib`, shows the relationship between baseband and passband models. The top branch of the circuit consists of passband models. The bottom branch consists of baseband models.

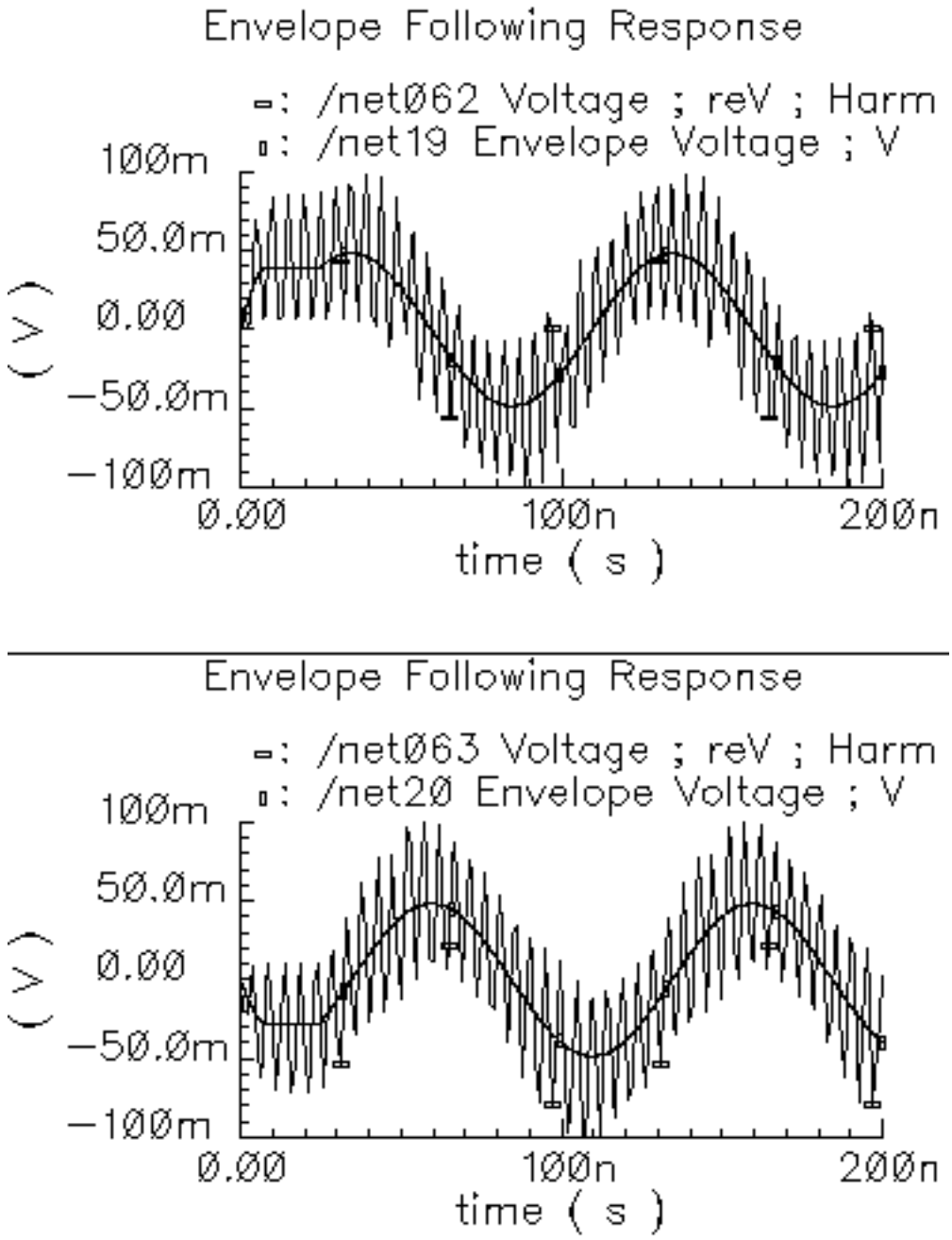
Figure 1-86 dwn_cnvrt_test Circuit



To see the relationship

1. Recall the circuit and set up a 200 ns envelope analysis with `fclck` as the *Clock Name*. Keep 1 harmonics1.
2. After the analysis completes, plot the “time” waveform at the `I_out` pin of the `IQ_demod_BB` model. Append to the plot, the harmonic-time, real part of the zero harmonic of the `I_out` pin on the `IQ_demodulator` model.
3. Open a subwindow and do the same for the `Q` outputs. You should now see a picture like the one in Figure 1-87.

Figure 1-87 Output from an Envelope analysis



To understand these results, trace the input signal through the passband branch. A complex baseband 10 MHz tone drives both branches. The modulator's local oscillator is 1 GHz so that the `IQ_modulator` output is at 1.01 GHz. There is no 990 MHz sideband because the input baseband trajectory is a circle (= sin + jcos), which represents a complex tone. The

mixer local oscillator is 900 MHz, which when mixed with 1.01 GHz, produces 110 MHz and 1.91 GHz. The `IQ_demodulator` local oscillator is 100 MHz, which produces 10 MHz, 210 MHz, 2.01 GHz, and 1.81 GHz. The 10 MHz and 210 MHz terms dominate the zero harmonic at the demodulator outputs. The higher frequencies average out to nearly zero. The baseband output is the 10 MHz term and that is what the baseband branch generates, as shown in Figure 1-87. A Transient analysis actually runs about 13 times faster than envelope on this circuit. Figure 1-112 compares the same outputs using a Transient analysis.The Transient analysis shows that the zero harmonic of the envelope analysis averaged out all frequencies above the envelope clock frequency (1 GHz).

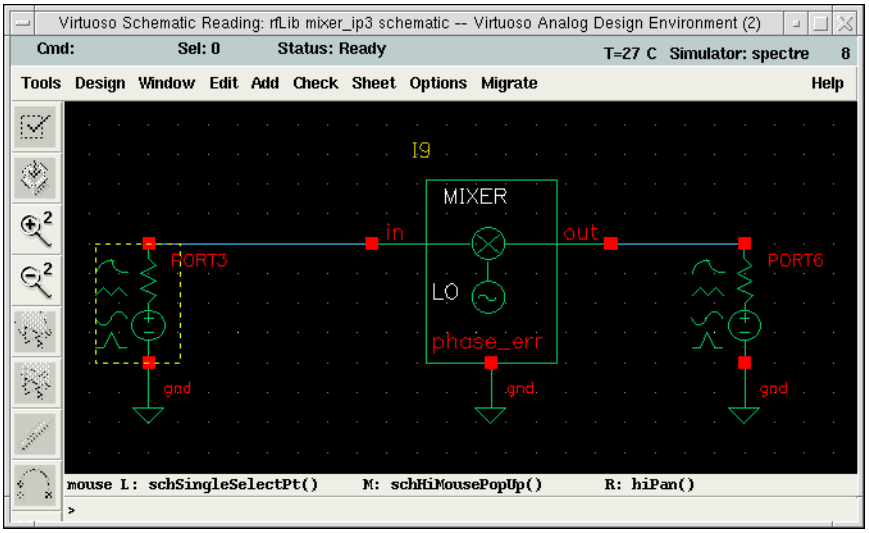
mixer_ip3

(IP3 Parameter)

IP3 is measured with a two-tone test. One tone is the fundamental PSS frequency while the other is the frequency in a single point PAC analysis. IP3 is defined as the input power level in dBm where the extrapolated power in one of the third order intermodulation terms equals the extrapolated power in the fundamental term. As with the 1dB compression point measurement, input and output terminals must be matched to the source and load respectively.

The IP3 specification is demonstrated step by step on the mixer model because the mixer IP3 measurement can be confusing. Figure 1-88 shows the test circuit. The circuit is listed as `mixer_ip3` in the `testbenches` category of the `rfLib`. For guidance on using the test circuit, see "Measuring IP3 for a Mixer" on page 180.

Figure 1-88 The mixer_ip3 Test Circuit



Measuring IP3 for a Mixer

For information about the test circuit used in this example, see "(IP3 Parameter)" on page 180.

1. Open the schematic for the circuit and bring up ADE.
2. In the Virtuoso Analog Design Environment window, choose *Analyses – Choose*.

   The Choosing Analyses form appears.

3. Set up a PSS analysis.

a. Select *pss*.

The title *Periodic Steady State Analysis* appears along with the fields required for specifying a PSS analysis.

A 920 MHz tone already appears in the form.

b. Add a *Fundamental Tone* called `eee` (the name is arbitrary) with a *Value* of 1 GHz.

c. Select *Beat Frequency*.

d. Click *Auto Calculate*.

The result is `40M` Hz.

e. For the *Number of harmonics*, type 2.

f. Select *Sweep*.

g. For the *Variable Name*, use the `power` variable.

h. Set *Frequency Variable* to *no*.

i. In the Sweep Range pane, select *Start_Stop*.

j. In the *Start* field, type -60.

k. In the *Stop* field, type 0.

l. Select *Linear*.

m. Select *Number of Steps*.

n. In the *Number of Steps* field, type 10.

4. Set up a PAC analysis.

a. Select *pac*.

The title *Periodic AC Analysis* appears along with the fields required for specifying a PAC analysis.

b. Set *Sweeptype* to *absolute*.

c. Select *Single-Point*.

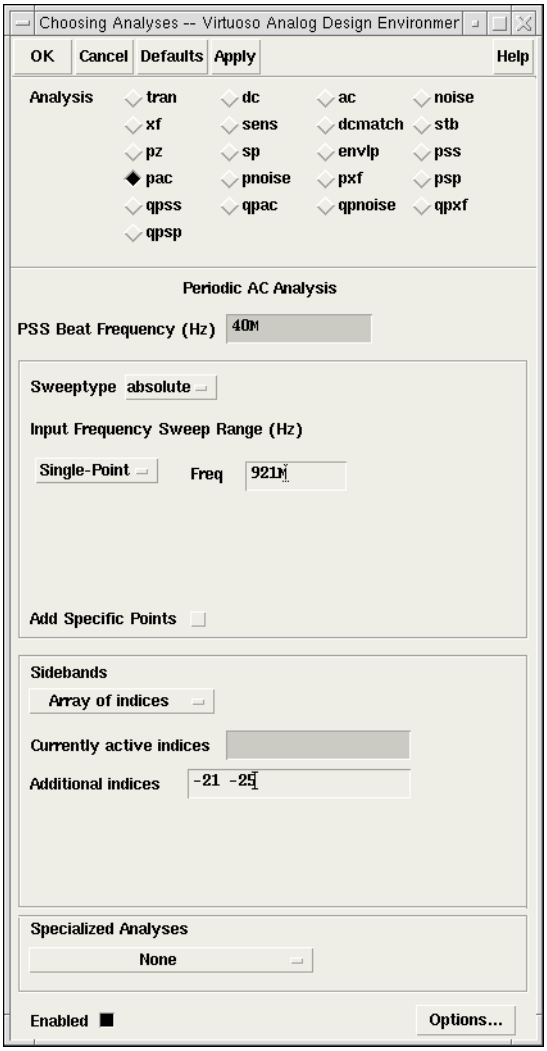
d. In the *Freq* field, type 921 M.

e. In the *sidebands* pane, select *Array of indices*.

f. In the *Additional indices* field, type -21 and -25.

After these steps, the Choosing Analyses form looks like this.

Figure 1-89 Choosing Analyses PAC Setup

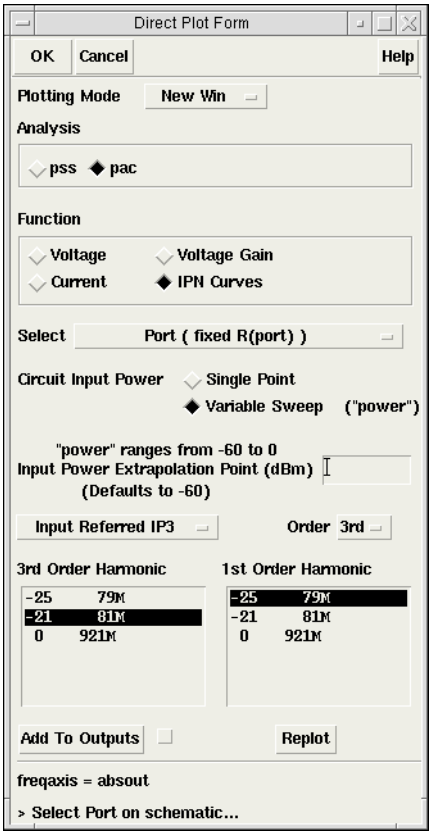


Why select the -21 and -25 sidebands? Recall from the assumptions, the non-linearity occurs before the frequency translation. The input tones to the non-linearity are the large 920 Mhz tone and the small signal 921 MHz tone. In an IP3 measurement only one tone must be large, the other can be small. PAC analysis performs small signal perturbations on the PSS solution. One perturbation term exiting the non-linearity appears at 921 MHz, right where it started. One of the third order intermodulation perturbation terms exiting the non-linearity appears at 2*920-921 = 919 MHz. The ideal mixer, driven by a pure 1 GHz local oscillator, translates the 921 MHz tone to 921-

1000=-79 MHz while translating the 919 MHz tone to 1000-919=81 Mhz. A single point 921 MHz PAC analysis produces tones displaced from harmonics of the fundamental by 921 MHz. The PAC sidebands specify which harmonics to use. You save the 79 MHz tone by saving the -25th sideband because the fundamental frequency is 40 Mhz and 921 - 40*25 = -79 MHz. You save the 81 MHz tone by saving the -21 sideband because 921-40*21 = 81 MHz. Figure 1-89 on page 182 shows the PAC setup.

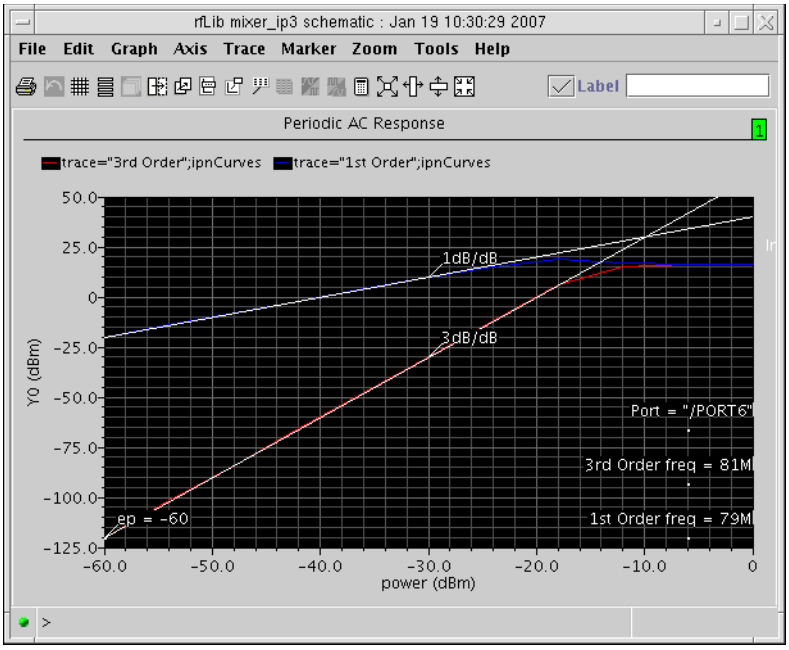
5. Run the analysis.

6. Plot the PAC results. To do this, set up the Direct Plot form like this.



7. In the Composer window, click the output Port. The results appear as shown in Figure 1-90 on page 184.

Figure 1-90 IP3 Results



The measured IP3 is, -10 dBm, as specified. The measured IP3 is as specified only if the input port resistance matches the input resistance of the device-under-test. Other input resistances produce a measured IP3 different than the one specified.

Measuring IP3 for an LNA

You can measure IP3 of an LNA by replacing the mixer with an LNA and ensuring the input terminal remains matched. In this example, remove the 1 GHz *Fundamental Tone* from the PSS analysis. The *Beat Frequency* should now be 920 MHz. In the PAC set up, change the additional indices from -21 and -25 to -1 and -2.

After the analysis completes, set up the PSS Results form as shown in Figure 1-91. As in “Measuring IP3 for a Mixer” on page 180, the input referred IP3 is 10 dBm, as specified. Figure 1-92 shows the LNA IP3 results.

Figure 1-91 Direct Plot Form for the LNA

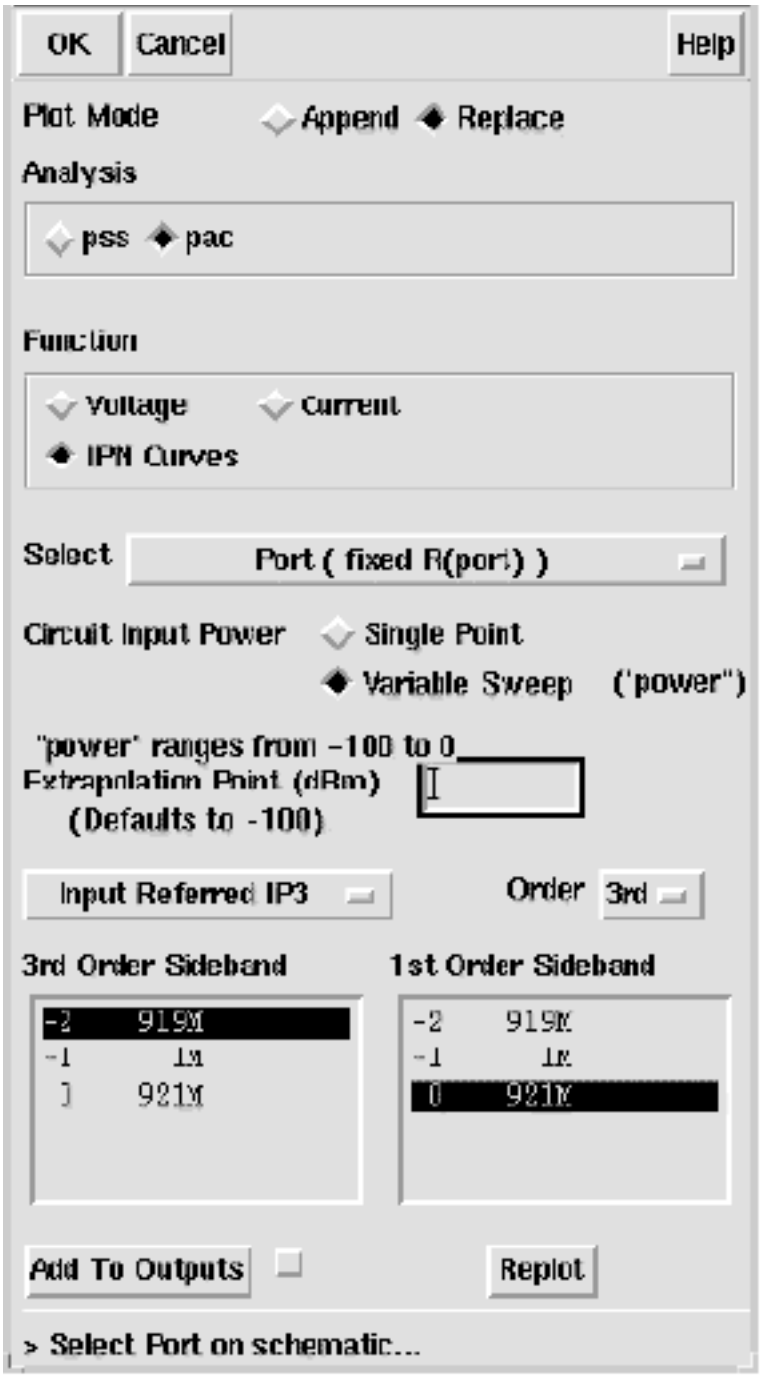
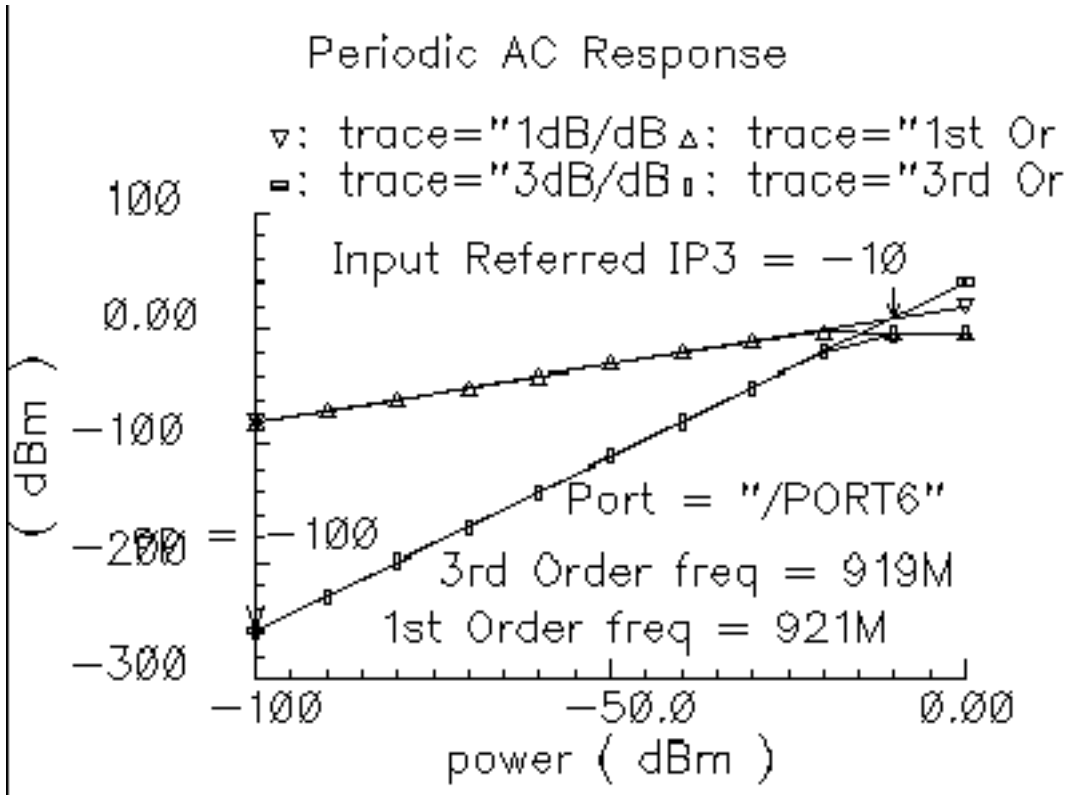


Figure 1-92 Results for the LNA



mod_1dbcp

(Available Power Gain and 1dB Compression Point)

Available power gain of the `IQ-modulator` is best explained with an example. Recall the circuit called `mod_1dbcp` listed in the `testbenches` category of the `rfLib`. The schematic contains two disjoint circuits. One shows how not to measure gain and compression point, the other shows the proper measurement.

1. Set up a PSS analysis. Both test circuits run in the same simulation. The beat frequency is 100 MHz. Save the first and 11^{th} harmonics. In the options, set `maxstep` to 50 ps. Sweep the variable `power` linearly in 50 steps from -40 to 15.
2. After the analysis completes, plot the output referred 1dB compression point of the top circuit using -40 dBm as the Extrapolation point. First select the 11^{th} harmonic (1.1 GHz) and click the output port in the top test circuit, the bad test circuit. Note that the linear gain is 3 dB lower than specified, as is the output referred 1dB compression point. The gain was specified as zero dB and the 1dB compression point was 10 dBm. The error arises from the fact that the input signal power splits between upper (1.1 GHz) and lower (900 MHz) sidebands but the ADE measurement only looks at one output sideband. The bottom test circuit resolves the ambiguity by defining the gain of the IQ-modulator as the gain from the baseband input to an ideally-demodulated baseband output. The bottom test circuit follows the IQ-modulator with an ideal IQ-demodulator. The gain of the demodulator is zero dB and the 1dB compression point is high enough to render the demodulator distortionless.
3. Repeat the steps for plotting the 1dB compression point but this time chose the first harmonic and select the output port that loads the bottom circuit. Select the first (100 MHz) harmonic and plot the 1dB compression point again. Now you should see a 1dB compression point plot that reflects the specified parameters of the IQ-modulator. The gain is now also correct, which can be computed from the ratio of the output to input power well below the compression point. Figure 1-94 shows such a plot.

Figure 1-93 1db Compression Point Test Circuit

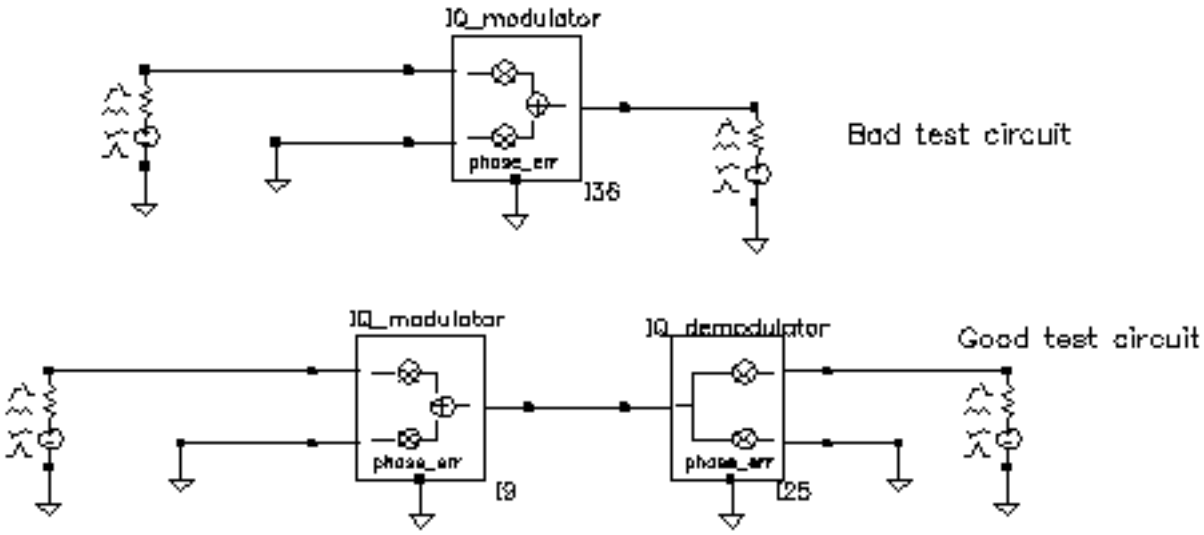
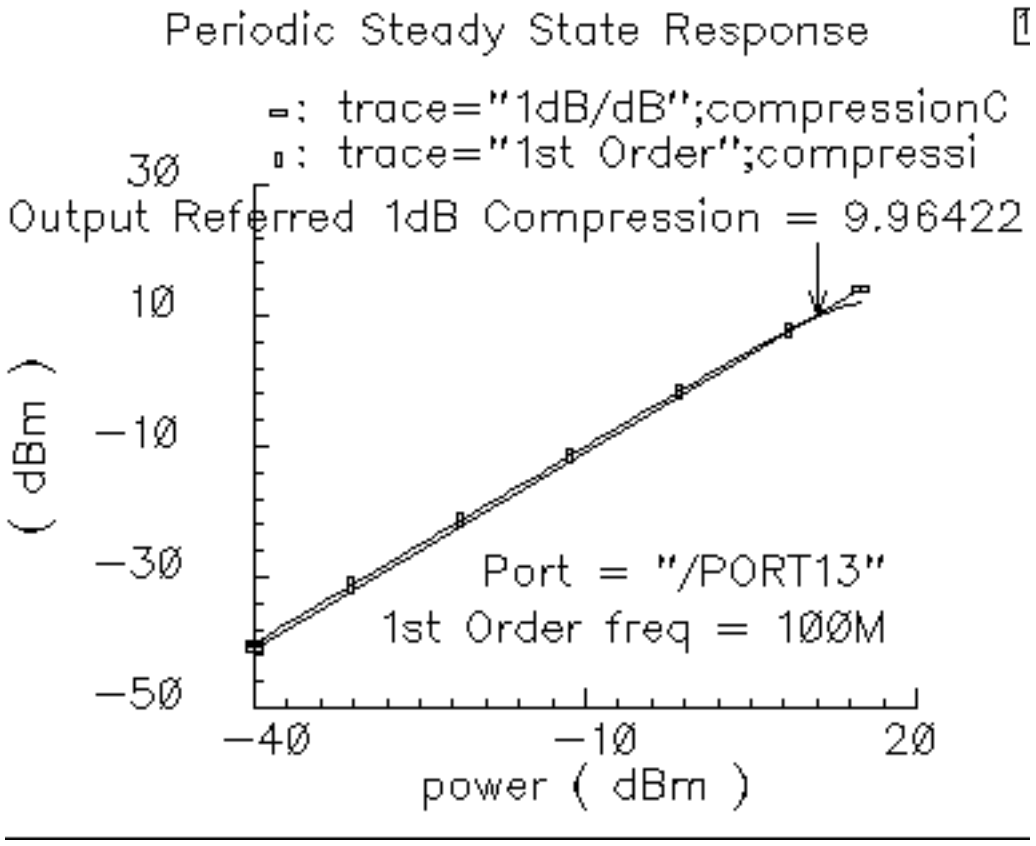


Figure 1-94 1db Compression Point Plot



mod_demod_test

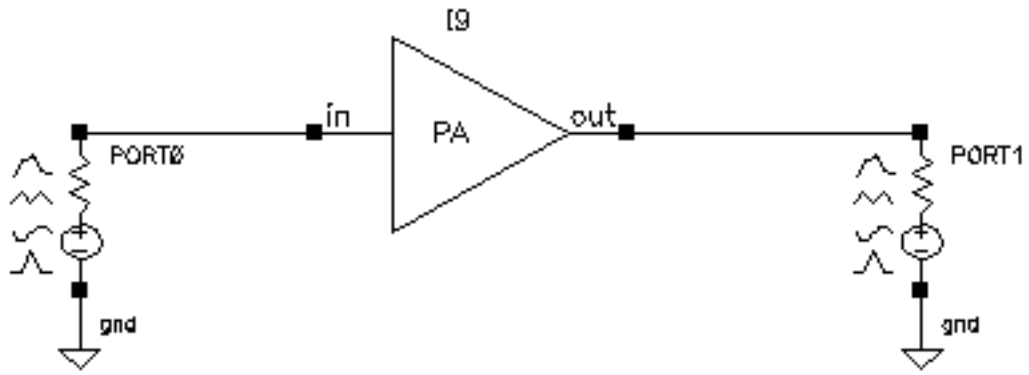
See “demod_ip3” on page 167.

noise_figure

(Noise Figure Parameter)

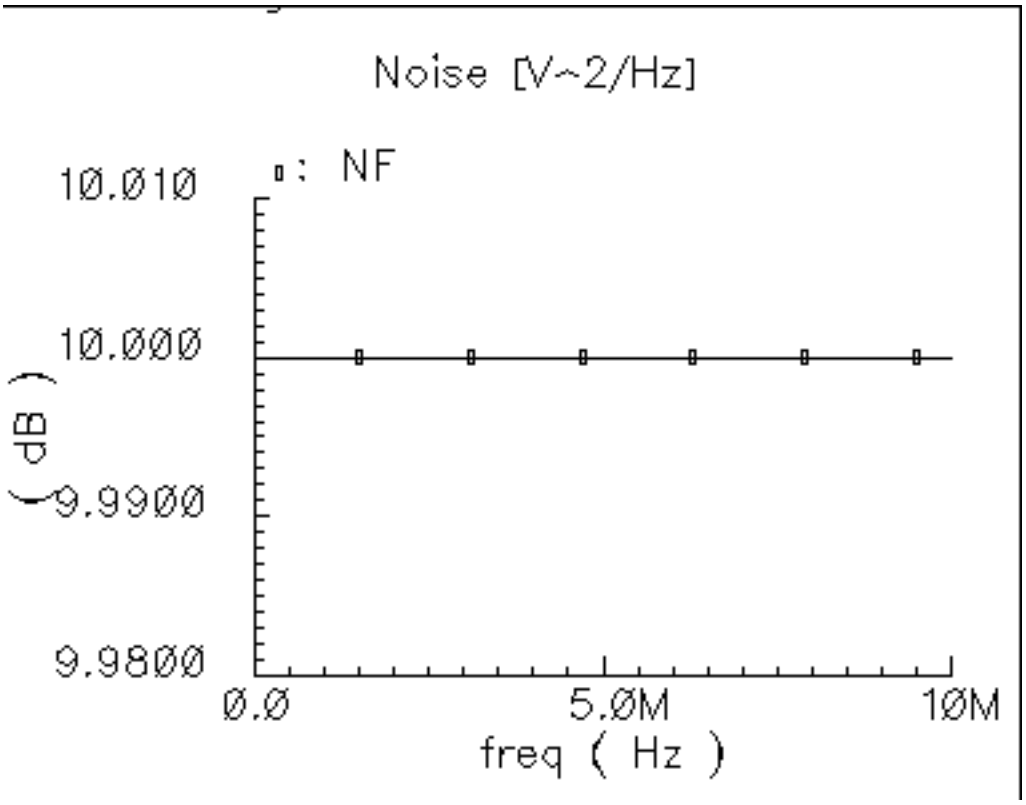
Noise figure is calculated as the input signal-to-noise ratio divided by the output signal-to-noise ratio. The test circuit for defining the noise figure parameter is shown in Figure 1-95. The circuit is listed as `noise_figure` in the `testbenches` category of the `rfLib`. It is similar to the `one_db_cp` test circuit.

Figure 1-95 The noise_figure Circuit



The specified noise figure is 10 dB. A Spectre RF Noise analysis produces the noise figure shown in Figure 1-96. To measure the specified noise figure, the driving port resistance must match the amplifier's input resistance. The port at the output does not have to match the amplifier's output resistance but the port impedance should be resistive. The input probe is the leftmost port, the output port is the rightmost port. Because the model is static, you can compute noise figure over any frequency interval.

Figure 1-96 Noise Figure Results



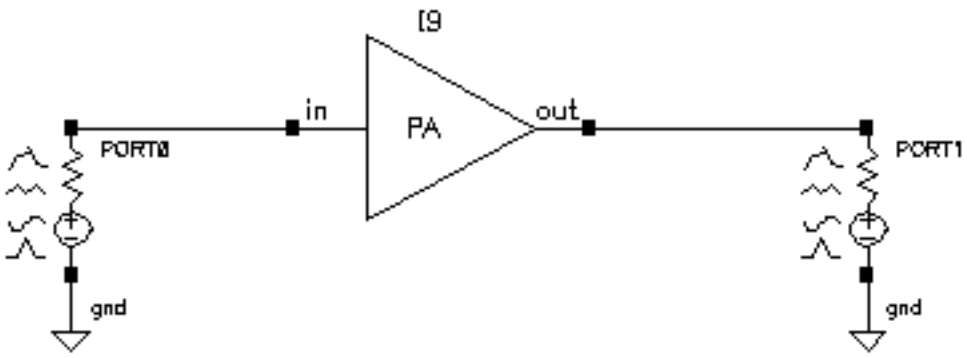
one_db_cp

(Output 1dB Compression Point Parameter)

The 1 dB compression point specifies a saturation non-linearity. It is the output power in dBm where the output power falls 1 dB below the power extrapolated linearly from the amplifier's linear region of operation.

The test circuit in Figure 1-97 is listed as `one_db_cp` in the `testbenches` category in `rfLib`.

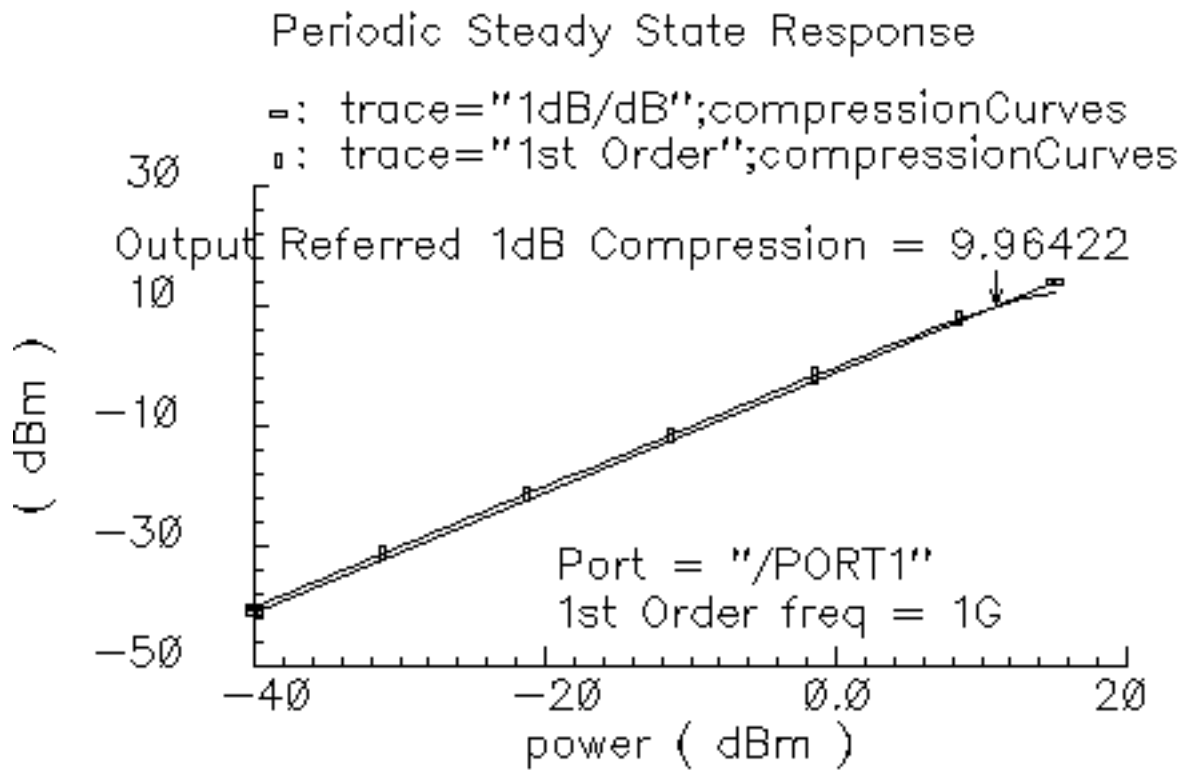
Figure 1-97 The one_db_cp Circuit



In the `one_db_cp` test circuit, *power* is the dBm of power delivered by the leftmost port. The available power gain is 0 dB. The 1dB compression point is 10 dBm. The input and output resistances are 50 Ohms and so are the port resistances.

To measure the 1dB compression point, perform a swept PSS analysis. Sweep *power* from -40 dBm to 15 dBm in 50 linear steps. The output referred 1dB compression point is computed for the 1st harmonic with an *Extrapolation Point [dBm]* of -40. Click the rightmost port device to display the output as illustrated in Figure 1-98.

Figure 1-98 Resulting 1dB Compression Point



The specified output referred compression point is 10 dBm. The measured value is 9.964 dBm, which is fairly close to the specified value. The measured 1db compression point is as specified only when the driving source resistance matches the amplifier input resistance and the load port resistance matches the amplifier's output resistance. In all compression point and IPN calculations, input power is computed from the maximum power the input Port can deliver, not from an actual power measurement. If you mismatch either terminal you do not measure the specified compression point.

PB_BB_filter_comparison

See “BB_ind_cap_test” on page 161.

PB_ind_cap_test

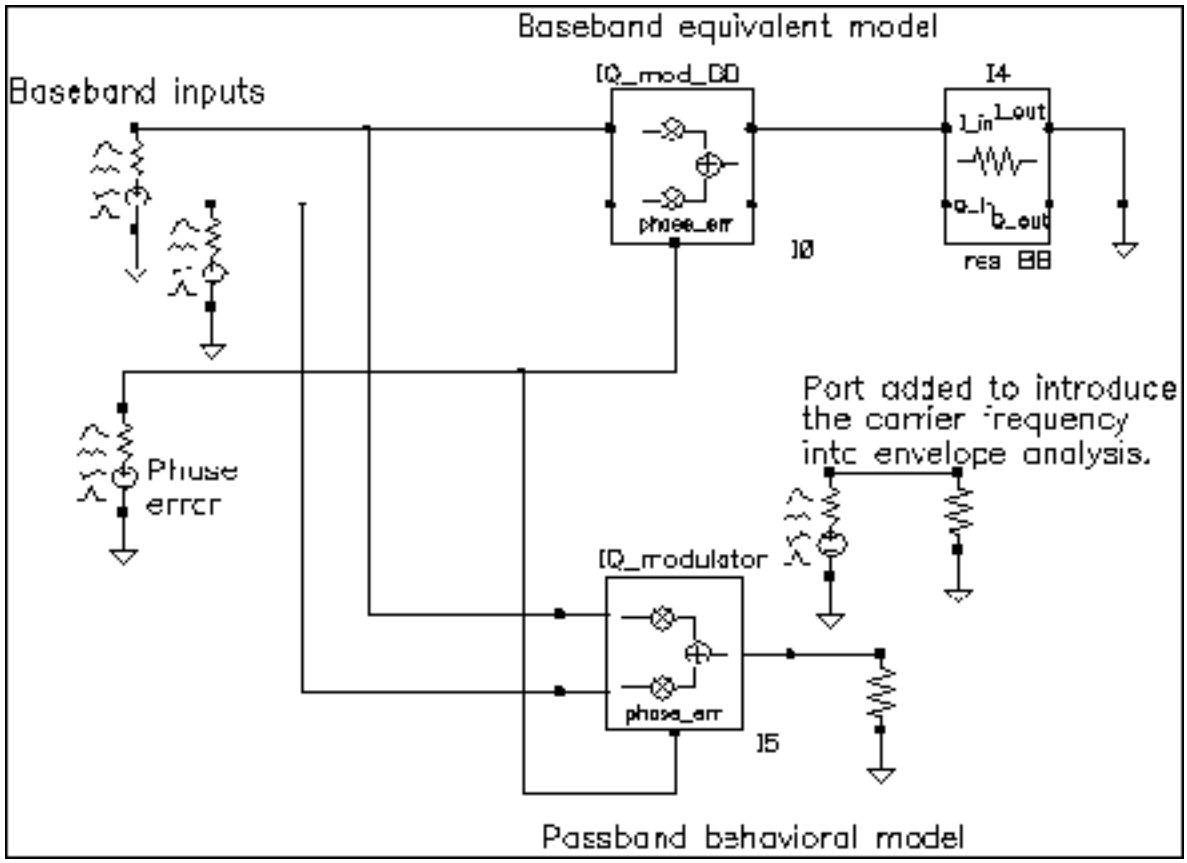
See “BB_ind_cap_test” on page 161.

quad_and_phase_error_demo

(Quadrature Error and Phase Error)

Quadrature error describes how far away from 90 degrees the two local oscillators are from each other. Ideally, they are exactly 90 degrees, or π/2 radians, apart in phase. In practice, parasitics and asymmetric delays can drive the phase shift away from π/2. Figure 1-99 on page 197 show a baseband test circuit and its passband equivalent. The schematic is listed in the `rfLib testbenches` category under the name `quad_and_phase_error_demo`. Both circuits are driven from a common set of baseband sources. The test circuit serves two purposes, it shows the correspondence between baseband and passband models and it demonstrates how quadrature error and phase error affect the baseband trajectory. The baseband input signal is a complex tone, which makes a circular input baseband trajectory. If there were no quadrature error, the baseband representation of the modulator output would also be a circle. With quadrature error, the output trajectory is an ellipse. If the `phase_err` pins are driven by a ramp, the ellipse precesses. The ramp represents a small but fixed difference between carrier and local oscillator frequencies.

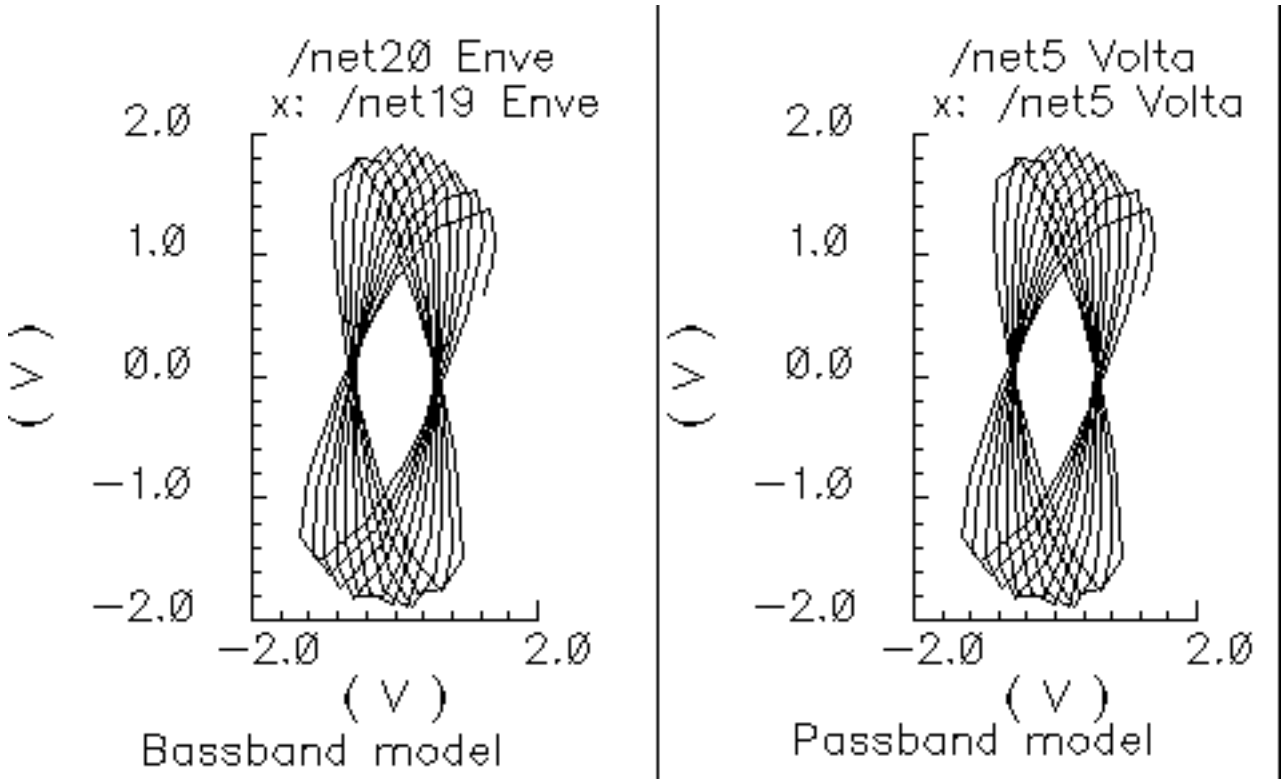
Figure 1-99 quad_and_phase_error_demo Circuit



To see these effects

1. Set up an ENVLP analysis with `carrier` as the Clock Name. Simulate 10 us of action and save the first harmonic.
2. When the analysis completes, open the Envelope Following Direct Plot form and set the sweep to *time*. Plot the two outputs of the `IQ_mod_BB` block.
3. Change the x-axis to be the `I-output`. You should see the left trajectory in Figure 1-100.
4. Add a subwindow for the passband equivalent result.
5. In the Direct Plot form, change the sweep to *harmonic time* and plot the real and imaginary parts of the first harmonic of the `IQ_modulator` output voltage.
6. Change the x-axis to be the real part of the first harmonic. You should now have two plots that match those in Figure 1-100.

Figure 1-100 Output Trajectories



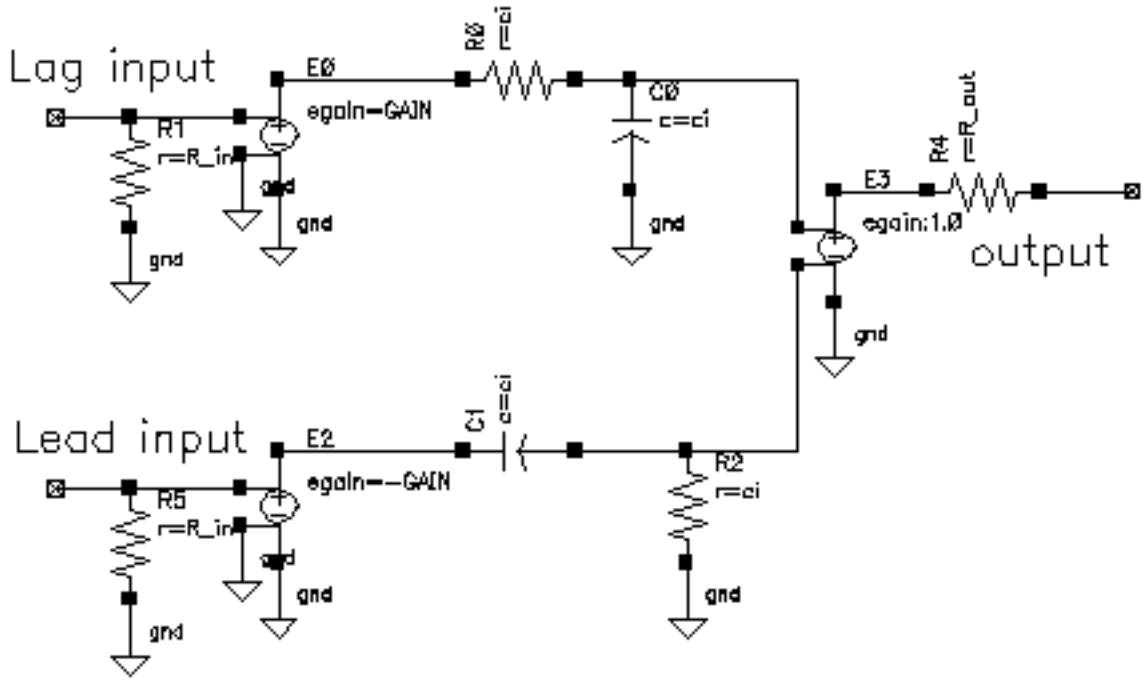
shifter_combiner_test

(Phase Shifter Combiner)

`(shifter-combiner)`

The phase shifter-combiner has two inputs and one output. The inputs are phase shifted by +/- 45 degrees then added together to form the output. All terminals are buffered and have the specified terminal resistances. The phase shifts are accomplished with Verilog-A code that does the same thing as the circuit shown in Figure 1-101 on page 199. The gains of the left-most voltage-controlled-voltage sources are user-defined. The input resistance, output resistance, intended operating frequency, and internal resistance are also user-defined. The internal resistance and operating frequency are used to calculate the capacitance necessary to provide 45 degrees of phase shifts at the operating frequency. The baseband view requires that the carrier frequency be specified.

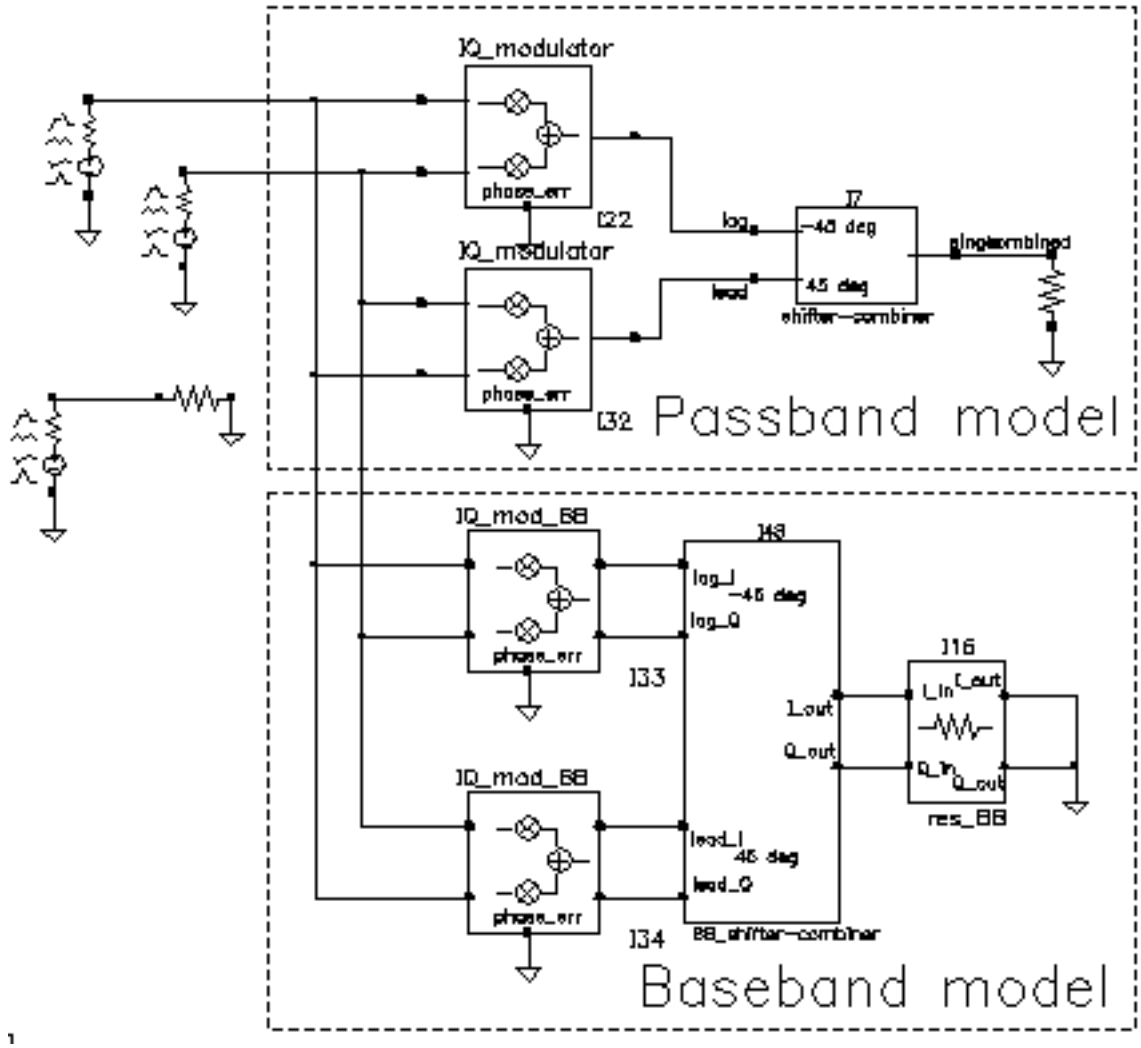
Figure 1-101 Phase Shift Circuit



The shifter-combiner can be used to eliminate one phase of the carrier. The test circuit in Figure 1-102 on page 200 shows a simple test to demonstrate the idea. The circuit is in the `rfLib` under the `testbenches` category and listed as `shifter_combiner_test`. The top circuit is a passband model and the bottom circuit is the baseband equivalent. Baseband input signals are mixed up to 1GHz then passed into the shifter-combiner. The baseband

signal contains 10MHz and 20Mhz components. The modulators and shifter-combiner are arranged to produce only a 20MHz signal riding on the carrier.

Figure 1-102 shifter_combiner_test Circuit

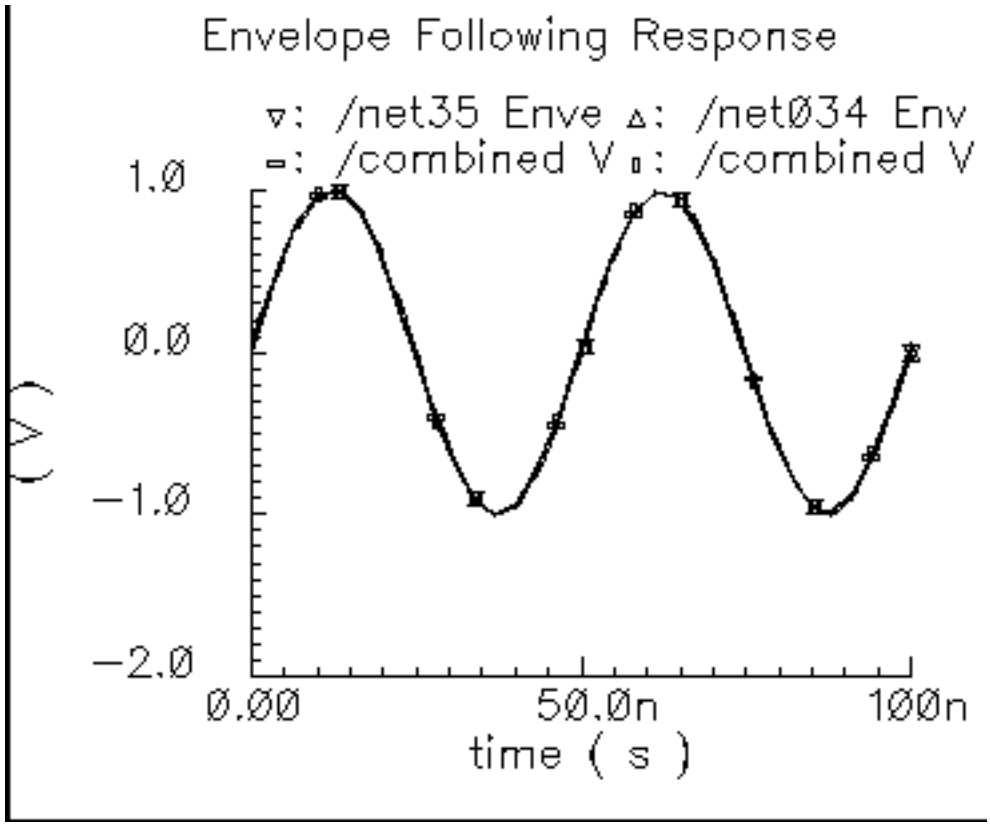


To check this assertion:

1. Bring up the test circuit and an Analog Environment window.
2. Set up a 100ns envelope analysis on the circuit with the *Clock Name* set to `carrier` and the *modulationbw* option set to 40MHz. Set the *Harmonic number* to 1.
3. Run the analysis.

4. Plot the harmonic time, 1 Harmonic, and the real and imaginary parts of the passband shifter combiner output.
5. Append to the plot, the `time` waveforms at the `I_out` and `Q_out` pins of the `BB_shifter _combiner` model. Figure 1-103 on page 201 shows what you should see in the Waveform Display window. All waveforms are the same and they contain only the 20Mhz baseband signal. The 10Mhz baseband input signal does not propagate to the output.

Figure 1-103 shifter_combiner_test results



One application of the shifter-combiner is an image rejection receiver. Figure 1-104 on page 202 shows a very simple example of an image rejection receiver. Figure 1-105 on page 202 shows the baseband equivalent model of the receiver. Both examples are in the `rfExamples` directory and are listed as `image_reject_rcvr_PB` and `image_reject_rcvr_BB`. The local oscillator runs at 1GHz and the RF carrier is 1.1GHz, which places the image at 900Mhz. This example shows one of the limitations of the baseband equivalent models.

Figure 1-104 A Simple Image Rejection Receiver

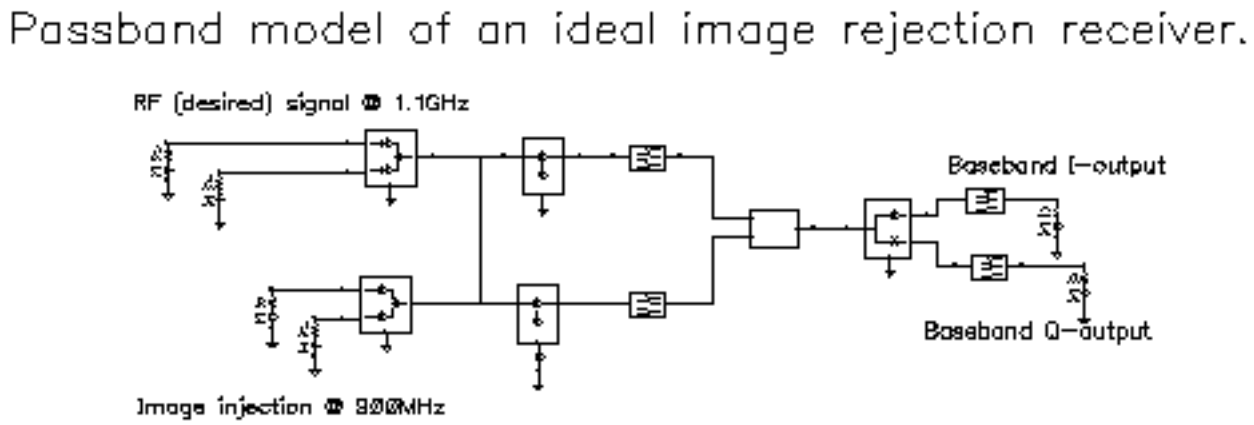
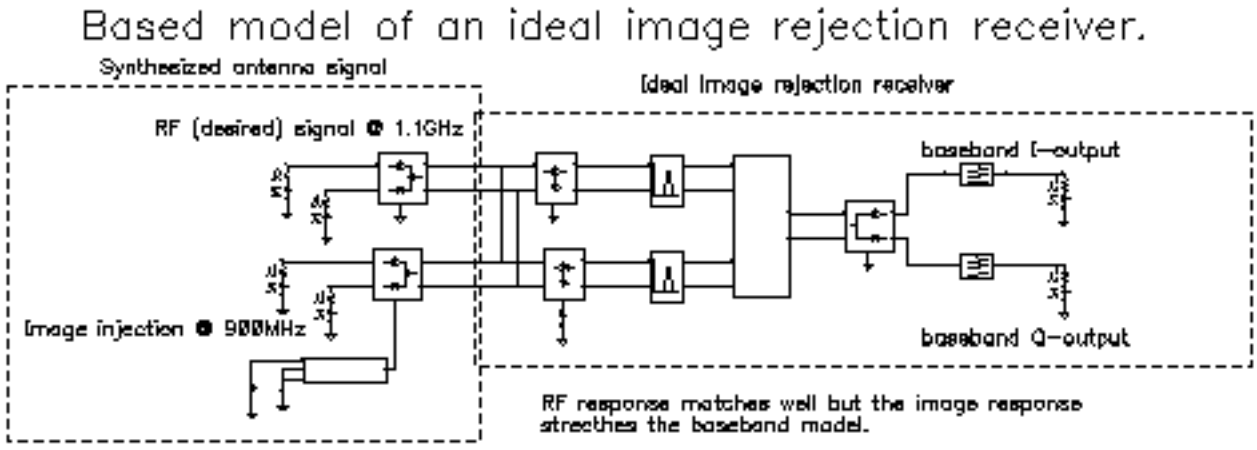


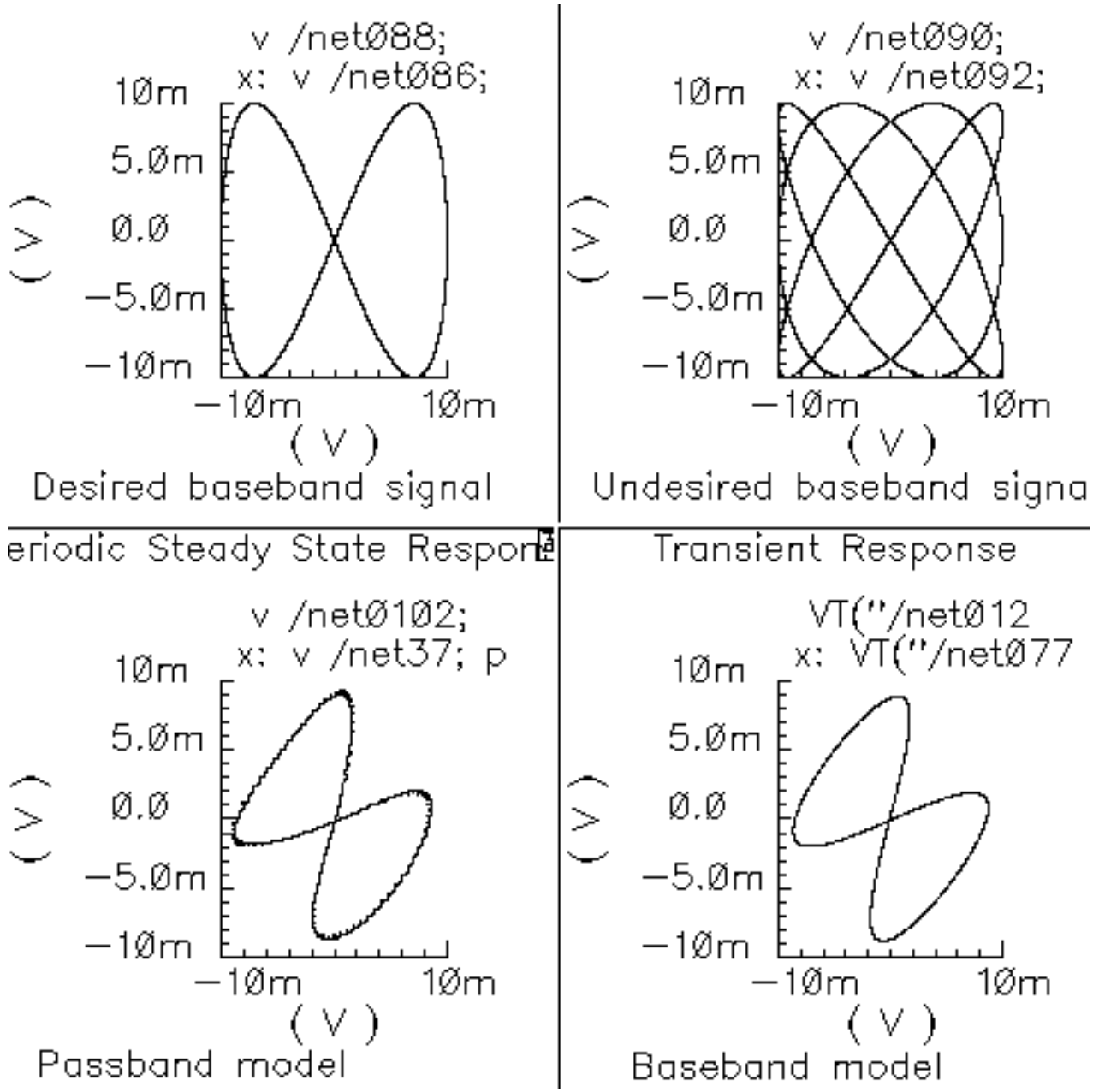
Figure 1-105 Baseband Equivalent Model of the Image Rejection Receiver



1. Bring up the passband test circuit and an Analog Environment tool.
2. Set up a PSS analysis. You need to add the 1.1GHz, 1GHz, and 900Mhz fundamental tones. Give them arbitrary but distinct names. AutoCalculate the Beat Frequency, which should be 2MHz. You need not save more than the 1st harmonic. Set the PSS maxstep option to 20ps so that it accurately simulates the oscillators hidden inside the Verilog-A modules.
3. Run the analysis.
4. Plot the voltages across Ports 5 and 6. Set the x-axis to be the voltage across Port 6. This is a Lissajous plot of the desired baseband signal, the one riding on the 1.1GHz carrier.

5. Add a subwindow.
6. Plot the voltages across Ports 8 and 7. Set the x-axis to be the voltage across Port 8. This is a Lissajous plot of the undesired baseband signal, the signal riding on the image of the carrier at 900MHz.
7. Add another subwindow.
8. Plot the I and Q- baseband outputs. Set the x-axis to be the I-output. The Lissajous plot is a tilted version of the desired baseband signal, indicating that most of the image was successfully rejected.
9. Bring up the baseband equivalent receiver model and another Analog Environment tool.
10. Run a 10us Transient analysis with 9.5us as the output start in the analysis options and maxstep set to 250ps. The phase_err pin on the image signal generator is being driven to spin the output at 200MHz, the frequency difference between the desired frequency and image frequency.
11. Add another subwindow to the Waveform Display tool showing the passband results and make sure it is active.
12. Plot the I and Q baseband outputs from the baseband equivalent receiver model. Set the x-axis to be the I-output. You might need to adjust the scales on the last two plots to make them the same. Aside from the labels, the Waveform Display tool should look like Figure 1-106 on page 204.

Figure 1-106 Lissajous Plots for Baseband Signals



The baseband equivalent receiver model indeed rejects the image but the rejection is over-estimated. If you look closely, the baseband output of the passband model contains more ripple from the image. The over-attenuated ripple in the baseband model is explained as follows.

Recall the rotating reference frame analogy for baseband modeling. With respect to the rotating 1.1GHz reference frame, the image signals rotate counter-clockwise at twice the IF, 200MHz in this case. The lower left block in the baseband receiver model spins the modulator output at -200Mhz by ramping the phase error pin. The -200MHz signal propagates through the IF bandpass filters, as it should, because the response of the baseband model of the filter peaks at DC and at minus 200MHz. The trouble occurs in the final downconversion to baseband. In the baseband model, the final low pass filters severely attenuate the -200Mhz image signal. However, in the passband model, image power at minus 100MHz contributes to the baseband signal through the low pass filters with less attenuation.

This example highlights one of the limitations of baseband equivalent models: at any point in the system, the signal should not have a bandwidth larger than any carrier (RF or IF) of the system. For this example, the baseband model is only valid for input RF signals between 1GHz and 1.2GHz.

The limitation is somewhat moot because the idea behind a baseband equivalent model is to suppress all carriers. To simulate the image response with the baseband model we had to include a 200MHz source! We would have been better off simply not suppressing the 100MHz IF carrier, i.e. using baseband models for the RF stages but passband models for the IF stages.

In summary, an all-baseband equivalent model of an image rejection receiver is only good for simulating the response to the desired RF signal, not the image response.

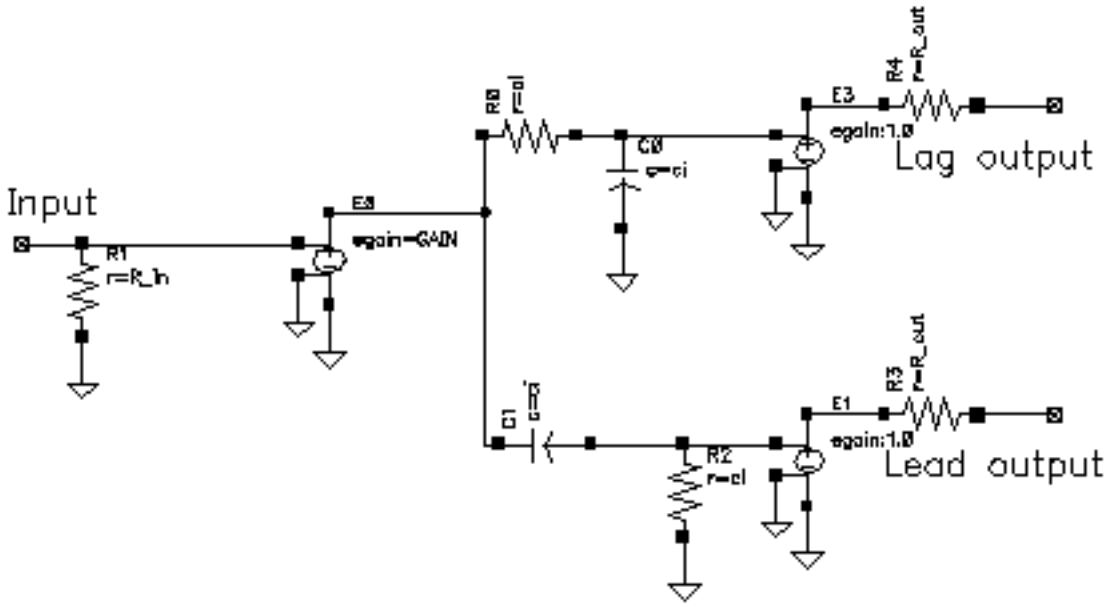
shifter_splitter_test

(Phase Shifter Splitter)

`(shifter-splitter)`

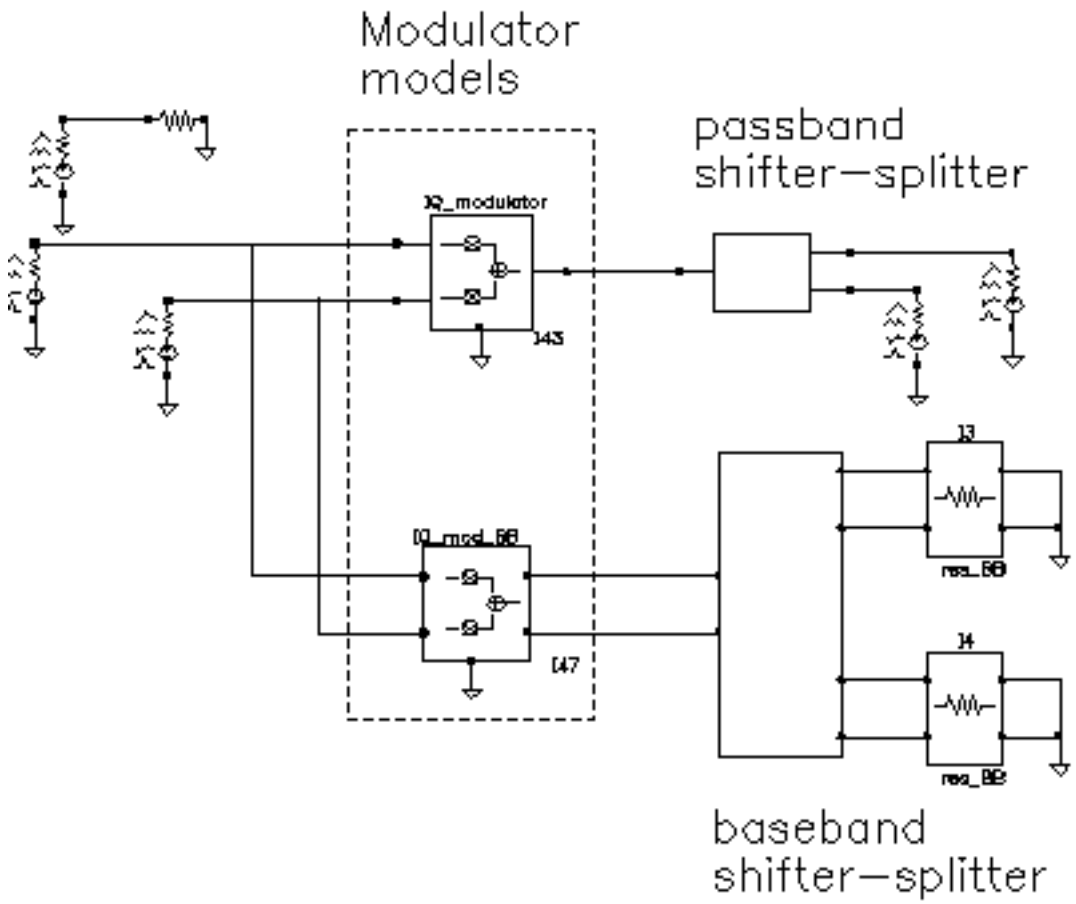
The phase shifter-splitter has one input and two outputs. The outputs are phase shifted versions of the input at the specified frequency. The phase difference between the two outputs is 90 degrees. The phase shifts are accomplished with Verilog-A code that does the same thing as the circuit shown in Figure 1-107 on page 206. The right-most voltage-controlled-voltage-sources (vcvs) are unity gain buffers. The left-most vcvs is also a buffer but the gain is a user-defined parameter. The input resistance, output resistance, intended operating frequency, and internal resistance are also user-defined. The internal resistance and operating frequency are used to calculate the capacitance necessary to provide +-45 degrees of phase shifts at the operating frequency. The baseband view requires the carrier frequency.

Figure 1-107 Phase Shift Circuit



The test circuit in Figure 1-108 on page 207 is for comparing the baseband responses of the passband and baseband equivalent models of the shifter-splitter. The circuit can be found in `rfLib` under the `testbenches` category. It is listed as `shifter_splitter_test`.

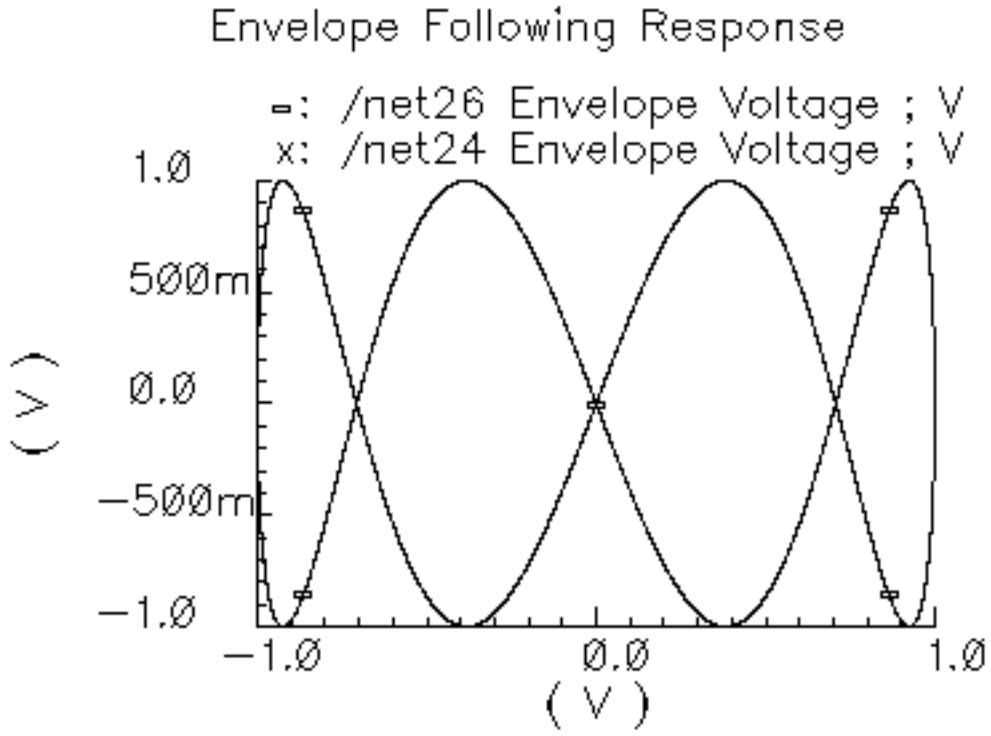
Figure 1-108 shifter_splitter_test Circuit



The following steps produce a set of Lissajous plots that show what the shifter-splitter does. You observe phase shift in the carrier by observing the tilt of the output Lissajous figures generated by the equivalent baseband signals.

1. Bring up the test circuit in Figure 1-108 on page 207 and an Analog Environment window.
2. Set up a 200ns Envelope analysis with `carrier` as the Clock Name. Set the Number of harmonics to 1. Set the Envelope analysis option called *modulationBW* equal to 100MHz.
3. Run the analysis.
4. Plot the `time` waveforms of the two input baseband signals. Change the x-axis to be the I-signal. You should see the Lissajous plot in Figure 1-109 on page 208.

Figure 1-109 Lissajous Plot for Baseband Input Signals



5. Reset the Waveform Display window and plot the harmonic time, 1 harmonic, real and imaginary parts of the voltage across Port2. Set the x-axis to be the real part. Note that the Lissajous plot is tilted -45 degrees from the one in Figure 1-109 on page 208.

6. Add a subwindow.

7. Repeat step 5 for the voltage across Port1. Notice that the Lissajous plot is tilted +45 degrees with respect to the Lissajous plot in Figure 1-109 on page 208.

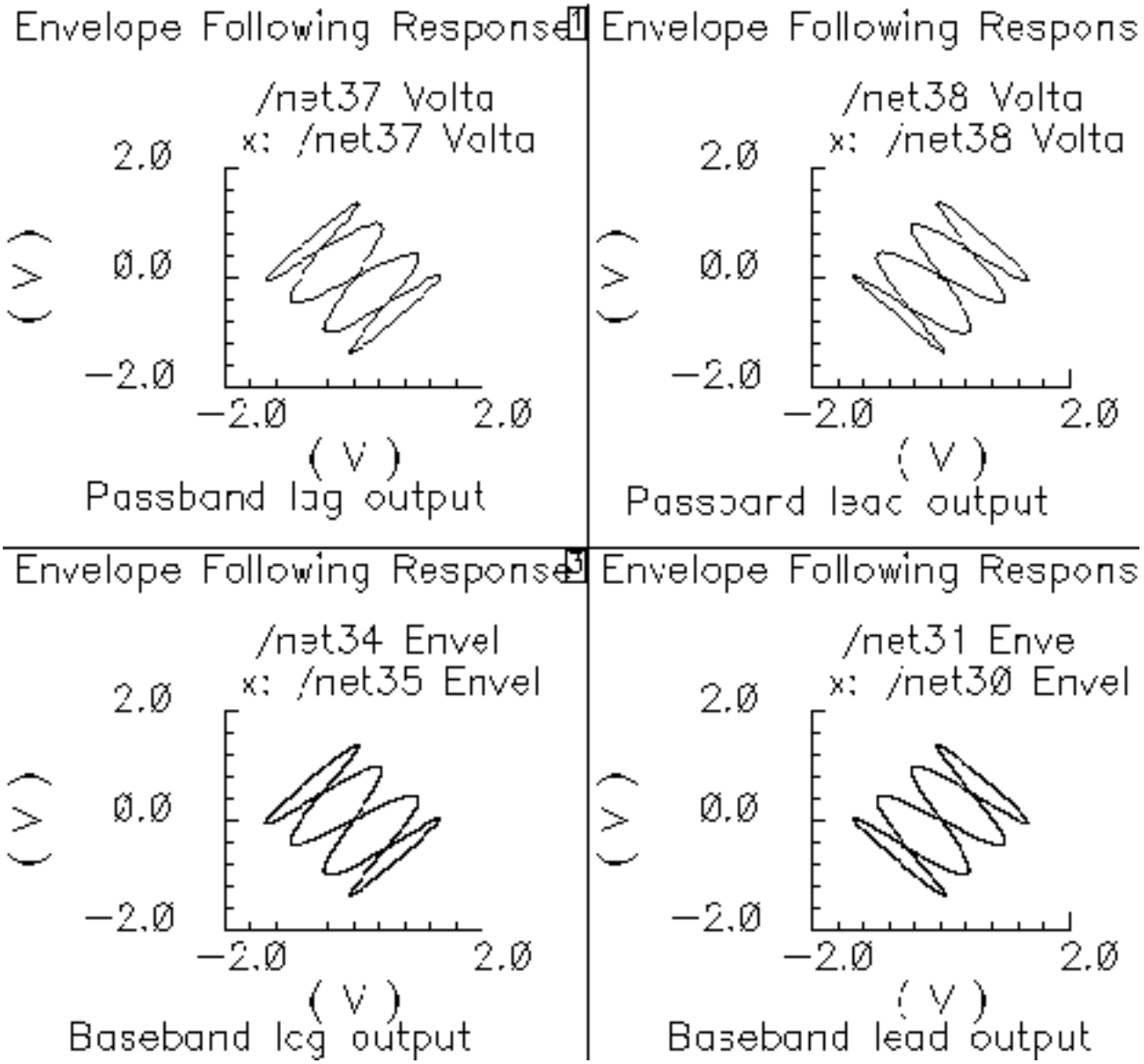
8. Add another subwindow.

9. Plot the `time` waveforms at the `lag_I` and `lag_Q` outputs of the `BB_shifter_splitter` model. Set the x-axis to be the lag_I waveform. The Lissajous plot should match the one produced in step 5.

10. Add another subwindow.

11. Repeat step 9 for the `lead` outputs of the `BB_shifter_splitter` model. The Lissajous plot should match the one produced in step 7. Aside from the labels, your Waveform Display tool should look like Figure 1-110 on page 209. The time-results of the baseband model faithfully duplicate the passband results but without simulating the carrier. The baseband model can be run with Spectre RF transient analysis.

Figure 1-110 Comparison of Lag and Lead times for Passband and Baseband Models



up_cnvt_test

(Testing the up_cnvrt Mixer)

There is a test circuit for the `up_cnvrt` model similar to the test circuit containing the `dwn_cnvrt` model. The `up_cnvrt` model is called `up_cnvt_test` and is shown in Figure 1-111. It is also in the `testbenches` category of `rfLib`. The steps parallel those for the `dwn_cnvrt` model.

Figure 1-111 up_cnvt_test Circuit

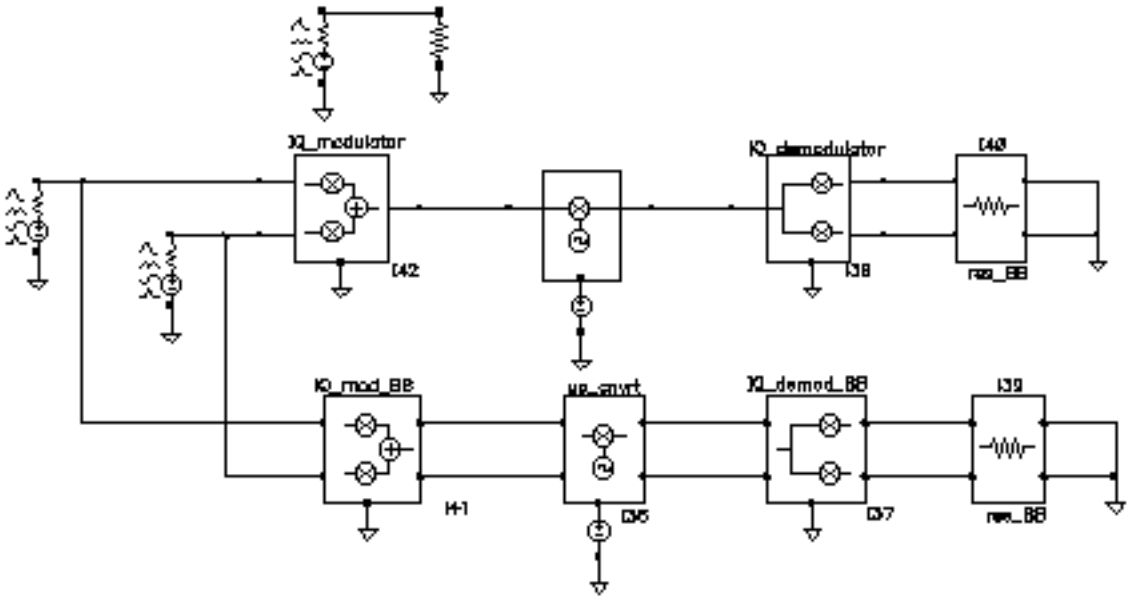
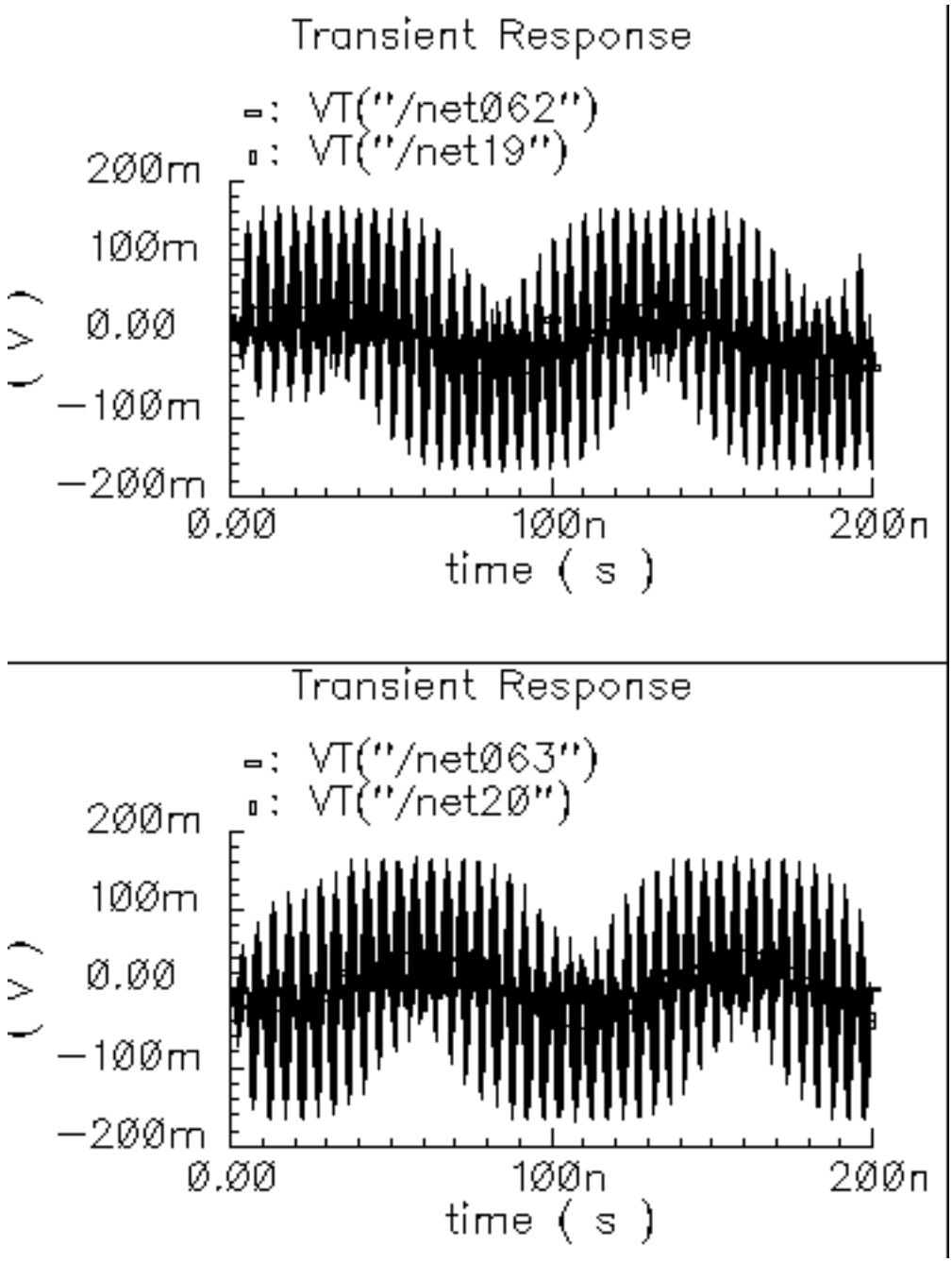


Figure 1-112 Results from a Transient Analysis



view_switching

This example illustrates how to switch between the single-ended baseband view and the differential passband view.

WCDMA_components Category

The components in the `WCDMA_components` category are:

- wcdma_dl_com_chanl
- wcdma_ocns
- wcdma_power_adjust
- wcdma_qpsk
- wcdma_sch_multiplexer
- wcdma_scrambling
- wcdma_scr_generator
- wcdma_spreading

wcdma_dl_com_chanl

(DL Common Channel Generator)

Figure 1-113 wcdma_dl_com_chanl symbol

This module, with four inputs and six outputs, generates the pCPICH (primary common pilot), PICH (paging indicator channel) and pCCPCH (primary common control physical channel). For PICH and pCCPCH, either external or internal random signals can be selected according to the value of the `enable_input` parameter. The parameters of the module are:

Name	Meaning	Type	Default Value	Range
`ccpch_seed`	Seed for CCPCH.	integer	98765	
`enable_input`	Enable input if 1; otherwise disable input.	integer	1	1, 0
`pich_seed`	Seed for PICH.	integer	12345	
`sample`	Sample time.	real	1.0/15000	

wcdma_ocns

(OCNS Generator)

Figure 1-114 wcdma_ocns symbol

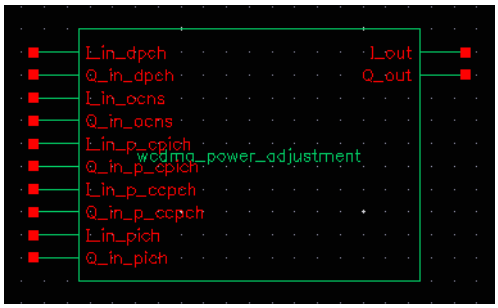
This module produces the combination of 16 dedicated data channels. The parameters of the module are:

Name	Meaning	Type	Default Value	Range
`enable_input`	Enable input if 1, otherwise disable input.	integer	1	1, 0
`frame_time`	Frame time.	Real	1.0/15000	(0:inf)
`numChipsOut`	Number of chips.	integer	256	
`sf`	Spread factor.	integer	128	[4:512]

wcdma_power_adjust

(Power Adjustment)

Figure 1-115 wcdma_power_adjust symbol



This module applies the weight to different channels. The parameters of the module are:

Name	Meaning	Type	Default Value	Range
`frame_time`	Frame time.	real	1.0/15000	(0:inf)
`numChipsOut`	Number of chips.	integer	256	
`power_ccpch`	Power for CCPCH.	real	-12	
`power_cpich`	Power for CPICH.	real	-10	
`power_dpch`	Power for DPCH.	real	-5.5	
`power_pich`	Power for PICH.	real	-15	

wcdma_qpsk

(QPSK Modulation/Mapping)

Figure 1-116 wcdma_qpsk symbol

This block has two inputs, `I_in` and `Q_in`, and two outputs, `outi` and `outq`. The inputs receive random input from outside the module. The outputs produce QPSK signals in baseband.

- This module includes the interleaving and encoding of data.
- You can use either random input from the outside or internal random bits. If `enable_input` is `on`, external data is used, otherwise internal random bits are used.

The instance parameters for the `wcdma_qpsk` block are:

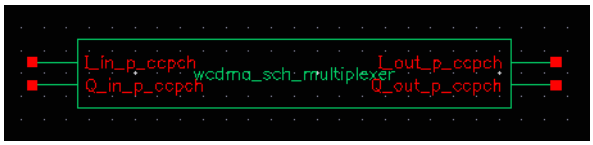
Name	Meaning	Type	Default Value	Range
`bits_per_integer`	Number of bits concerted into integer.	integer	2	[1:31]
`enable_input`	1 means to use outside input; otherwise 0.	Boolean	1	0, 1
`frame_time`	The time of one frame.	real	1.0/15000.0	(0:inf)
`mapping_mode`	Mapping type.	string	user_defined	Binary_gray, gray_binary, user_defined
`phase_offset`	Initial phase.	real	0	
`samples`	Number of samples in one frame.	integer	12345	

`seed`	Used for random.	type	2	
`usr_mapping_vec`	Bit mapping between input and output.	integer	{0, 3, 1, 2}	

wcdma_sch_multiplexer

(SCH Generator/Multiplexer)

Figure 1-117 wcdma_sch_multiplexer symbol



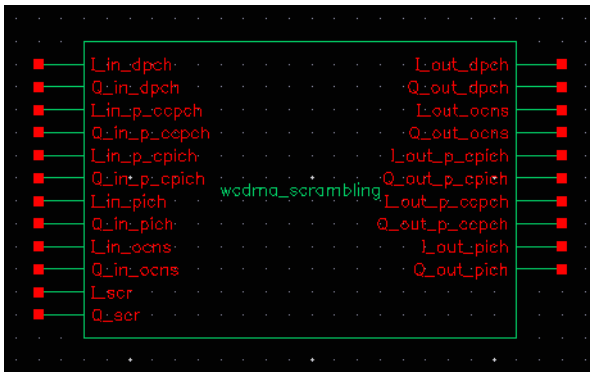
This module produces the synchronization channel and performs SCH multiplexing. The parameters of the module are:

Name	Meaning	Type	Default Value	Range
`frame_time`	Frame time	real	1.0/15000	(0:inf)
`hada_order`	Hadamard matrix order.	integer	8	
`numChipsOut`	Number of chips.	integer	256	
`ssc_num`	Scrambling code group number.	integer	64	[1:64]

wcdma_scrambling

(Scrambling/Scrambling Code)

Figure 1-118 wcdma_scrambling symbol



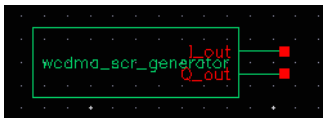
This module scrambles the spread code. The parameters of the `wcdma_scrambling` module are:

Name	Meaning	Type	Default Value	Range
`frame_time`	Frame time.	real	1.0/15000	(0:inf)
`numChipsOut`	Number of chips.	integer	256	

wcdma_scr_generator

(Square-Root Raised Cosine)

Figure 1-119 wcdma_scr_generator symbol



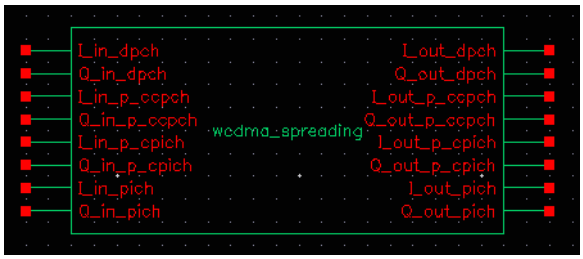
This module up-samples and filters the input. The parameters of the module are:

Name	Meaning	Type	Default Value	Range
alpha	Filter attenuation at cutoff [dB].	real	0.22	
frame_time	Frame time.	real	1.0/15000	(0:inf)
group_delay		integer	6	
numChipsOut	Number of chips.	integer	256	
over_samples		integer	8	

wcdma_spreading

(Spreading)

Figure 1-120 wcdma_spreading symbol



This module spreads the data over the OVSF codes. The parameters of the `wcdma_spreading` module are:

Name	Meaning	Type	Default Value	Range
`dpch_code`	OVSF index for DPCH.	integer	10	
`frame_time`	Frame time.	real	1.0/15000	
`oversample`	Oversample.	integer	128	
`pccpch_code`	OVSF index for PCCPCH.	integer	1	
`pcpich_code`	OVSF index for PCPICH.	integer	0	
`pich_code`	OVSF index for PICH.	integer	4	
`sf`	Spread factor.	integer	128	[4:512]

Modifying the BB Signal Generators Using Modelwriter

The baseband signal generators use FIR (finite impulse response) filters to shape their output pulses. Shaped output pulses serve several purposes:

- They help keep the transmitted signal inside the specified band.
- They work with their receiver counterparts to maximize the signal-to-noise ratio.
- Together with their receiver counterparts, they satisfy the Nyquist sampling criterion for an intersymbol-interference-free channel.

The Modelwriter gives you a convenient user-interface for creating variations on the baseband signal generators in the library. The most likely variation is in the FIR filter. This section explains how to use the Modelwriter to create a new GSM generator with different FIR filters.

1. Bring up the Modelwriter and do the following:

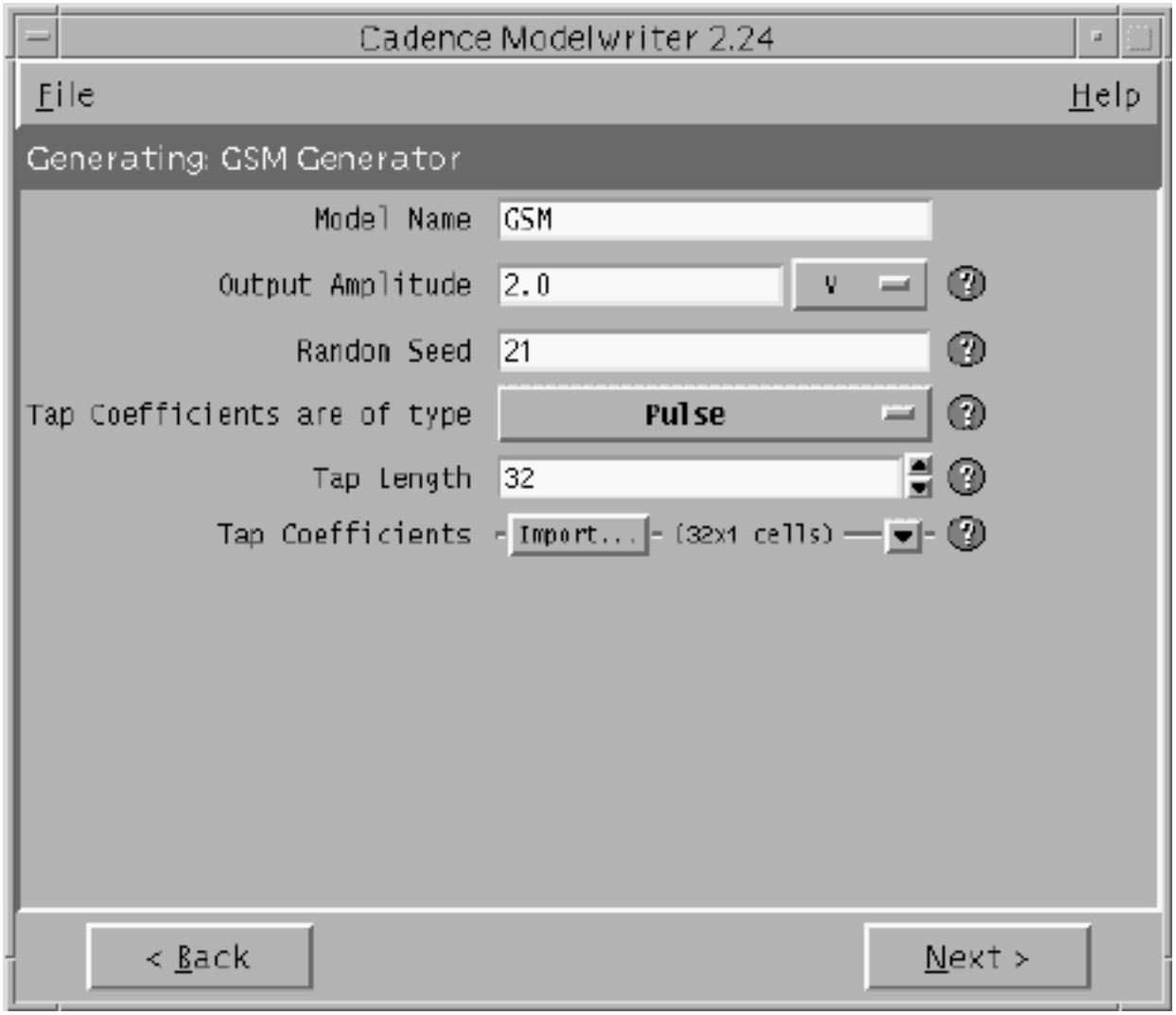
   a. Double click the Telecom folder.

   b. Select the GSM generator.

   c. Click the *Next* button in the lower right hand corner of the Modelwriter window.

   You should now see the picture in Figure 1-121 in the window.

Figure 1-121 GSM Generator



2. Specify how you plan to drive the filter by selecting the type of tap coefficients.

   FIR filters can be driven by pulses or impulses.

3. Specify the length of the FIR filter in the *Tap Length* field.

4. Specify the tap coefficients, in one of the following ways.

   - From a file
   - By direct manual entry.

   To read the coefficients from a file do the following:

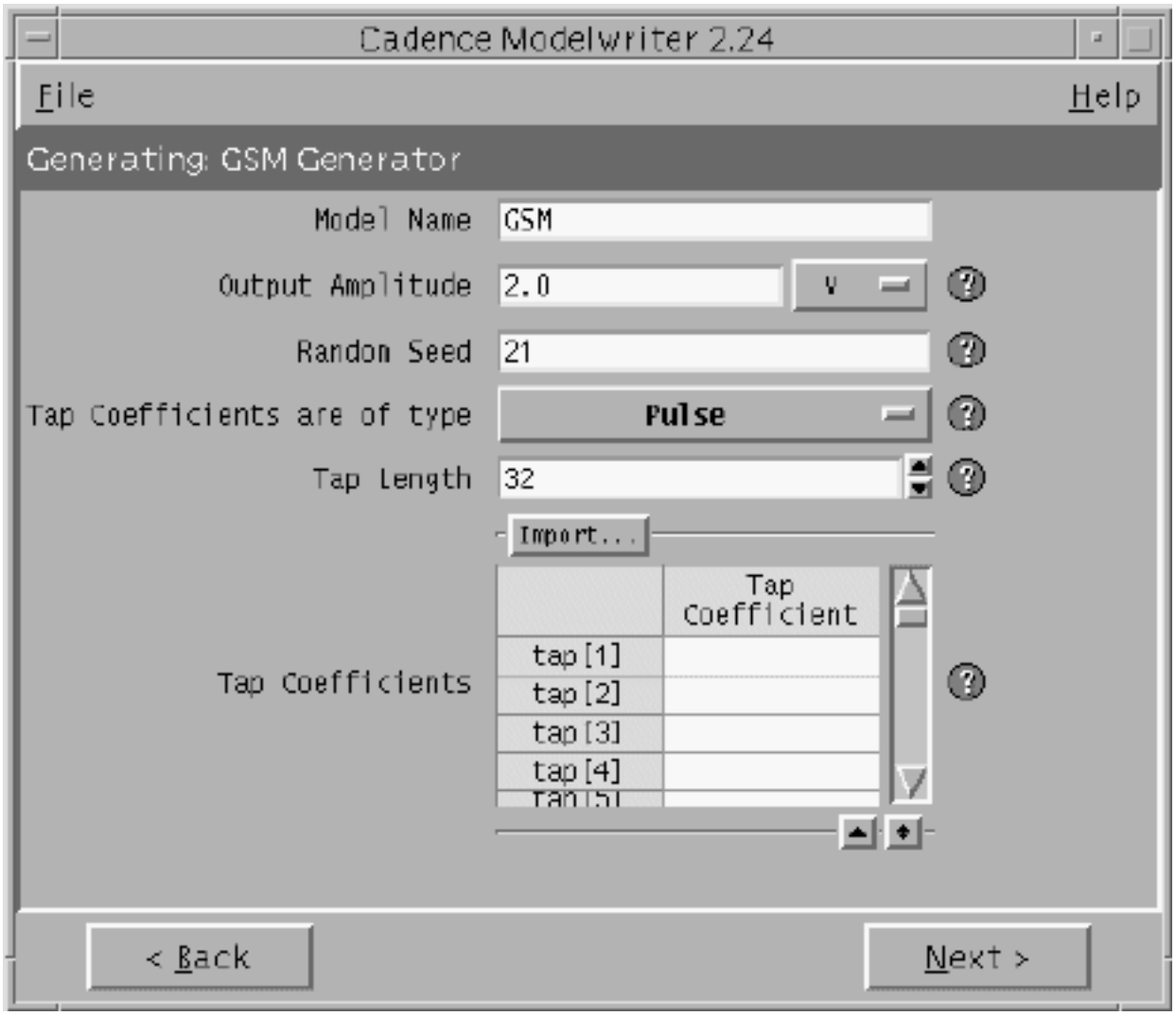
a. Select the *Import* button.

b. Enter the path to the file.

c. Click *Open*.

The coefficients appear in the window as shown in Figure 1-122. (The tap[1] coefficient multiplies the filter state with the least amount of delay, the filter state closest to the input.)

Figure 1-122 Tap Coefficients



To enter the coefficients manually do the following:

a. Click the lower rightmost button flagged in the form.

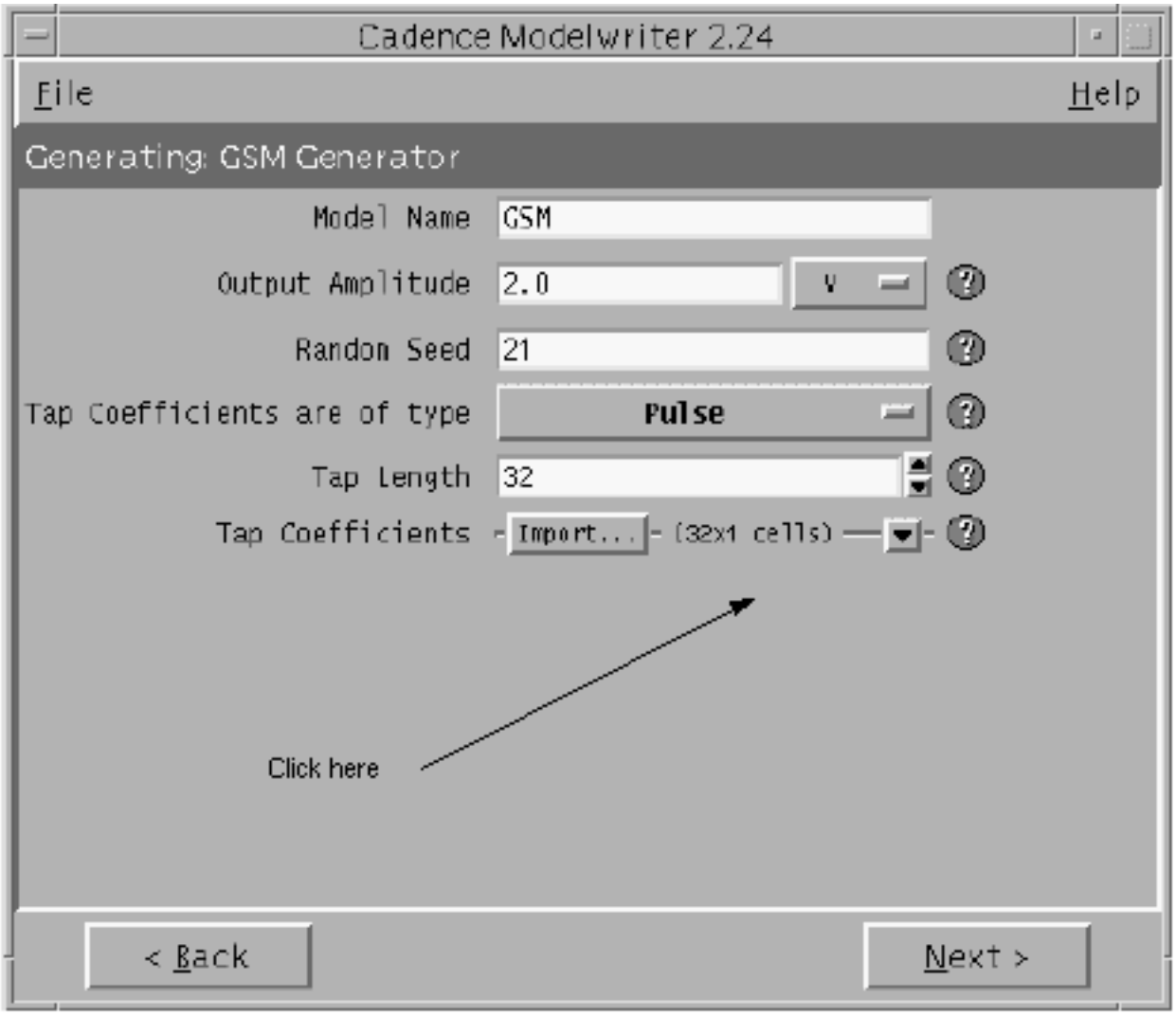
See Figure 1-123.

b. Enter the coefficients,

c. Click *next* to view the model,

5. To write the model to a file, click *Save Generated Code...*.

Figure 1-123 Manual Entry of the Tap Coefficients



2

measureLib Library

The components in the `measureLib` library provide analyses and measurements that can be instantiated in schematics, providing a convenient way of reusing and customizing measurements and analyses. This approach makes it possible to specify analyses that depend on other analyses, such as a small signal analysis that depends on a large signal analysis. This approach also facilitates plotting of results from different analyses.

The components in the `measureLib` are provided by Cadence. It is also possible to create your own measurement and analysis components and they can be instantiated just as the provided components can be.

For guidance about creating your own measurement or analysis components, see "Create New Measurement Wizard" in chapter 2 of *Virtuoso Spectre Circuit Simulator RF Analysis User Guide (Volume 1)*. For guidance about customizing and instantiating measurement or analysis components, see "Instance Measurement on schematic Wizard" in chapter 2 of *Virtuoso Spectre Circuit Simulator RF Analysis User Guide (Volume 1)*.

This chapter describes the components provided in the `measureLib` library. The descriptions are arranged by categories:

Categories	Description of Category and Link
`Everything`	Lists all elements in `measureLib`.
`Example`	"Example Category" on page 228
`Measurement`	"Measurement Category" on page 233

Example Category

The `Example` category contains designs that illustrate how to set up and use the measurement and analysis cells contained in the `measureLib` library.

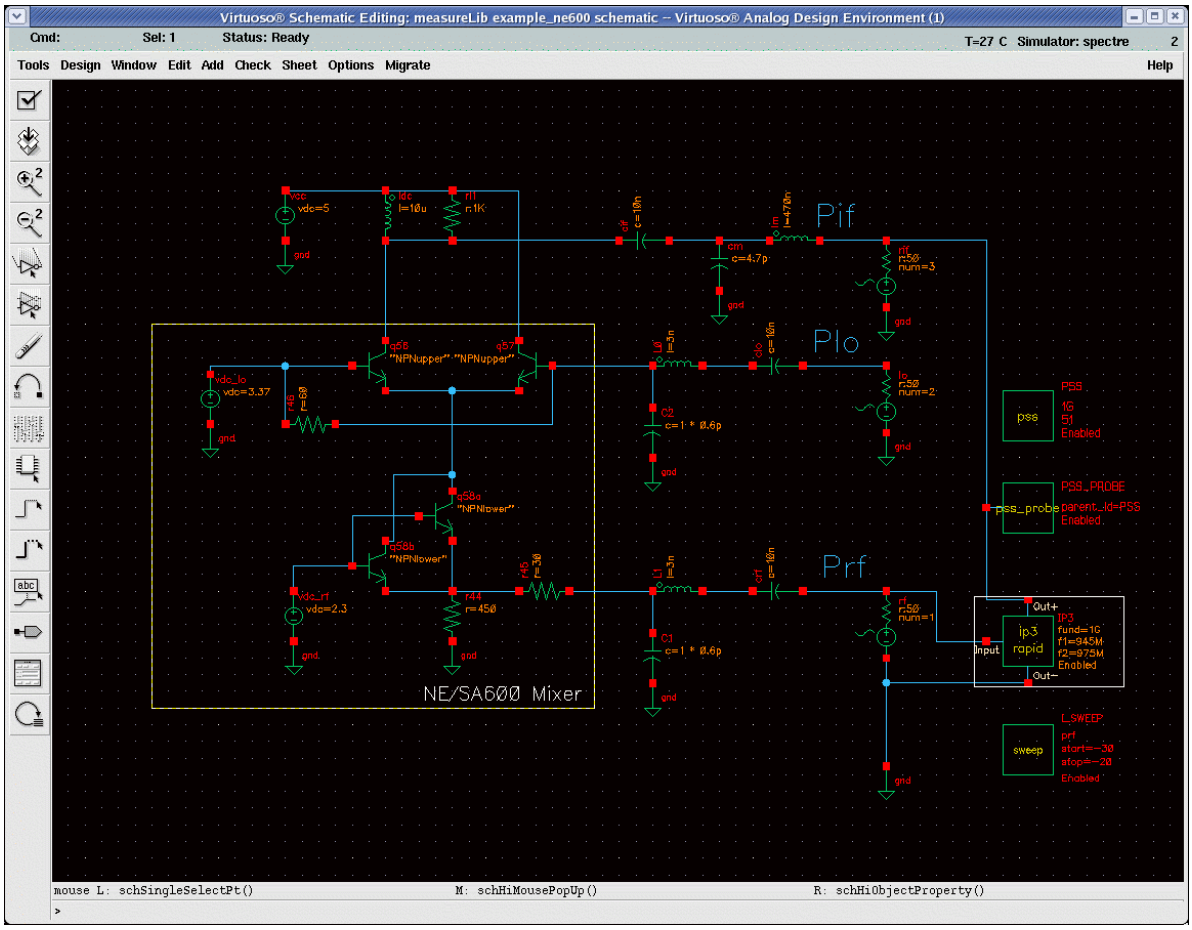
The cells in the `Example` Category are:

- example_ne600
- example_oscHartley

example_ne600

This example illustrates how to use the measurement library to implement a driven PSS with IP3 measurement.The PSS analysis uses a high harmonic to see the output nodes. The IP3 measurement uses the rapid method so only limited harmonics are set. This example also illustrates the ability to do multiple measurements on a single schematic.

The `example_ne600` design instantiates the `pss_driven` cell (labeled `pss`), the `ip3_rapid` cell (labeled `ip3_rapid`), and the `pss_vprobe` cell (labeled `pss_probe`).



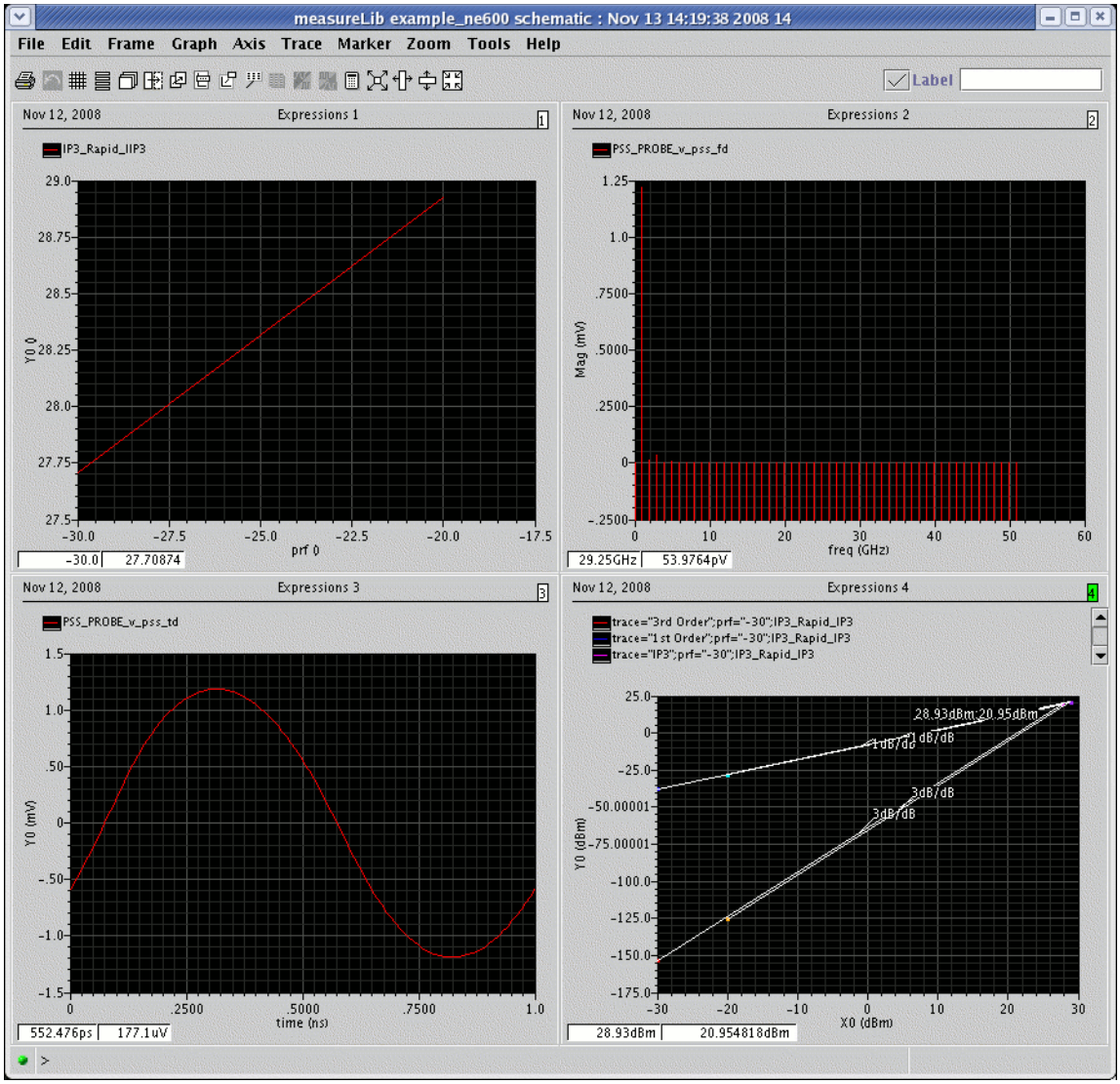
The `ip3_rapid` and `pss` instances in the schematic cause the following lines to be included in the netlist.

```
I_SWEEP_sweep sweep param=prf start=-30 stop=-20 lin=1 {
    simulator lang=spectre
    IP3_pss pss fund=1G harms=3 errpreset=moderate tstab=100n \
    annotate=status
    IP3_pac pac out2=0 out1=Pif flin_out=abs(-55e6) fim_out=abs(915e6) \
```

```
    rfdbm=prf rf2_src=rf rf1_src=rf start=945M stop=975M perturbation=ip3 \
    annotate=status
}
simulator lang=spectre
PSS_pss pss flexbalance=yes fund=1G harms=51 errpreset=moderate tstab=120n \
    lnsolver=gmres annotate=status
```

These commands in the netlist set up and run the IP3 and PSS analyses.

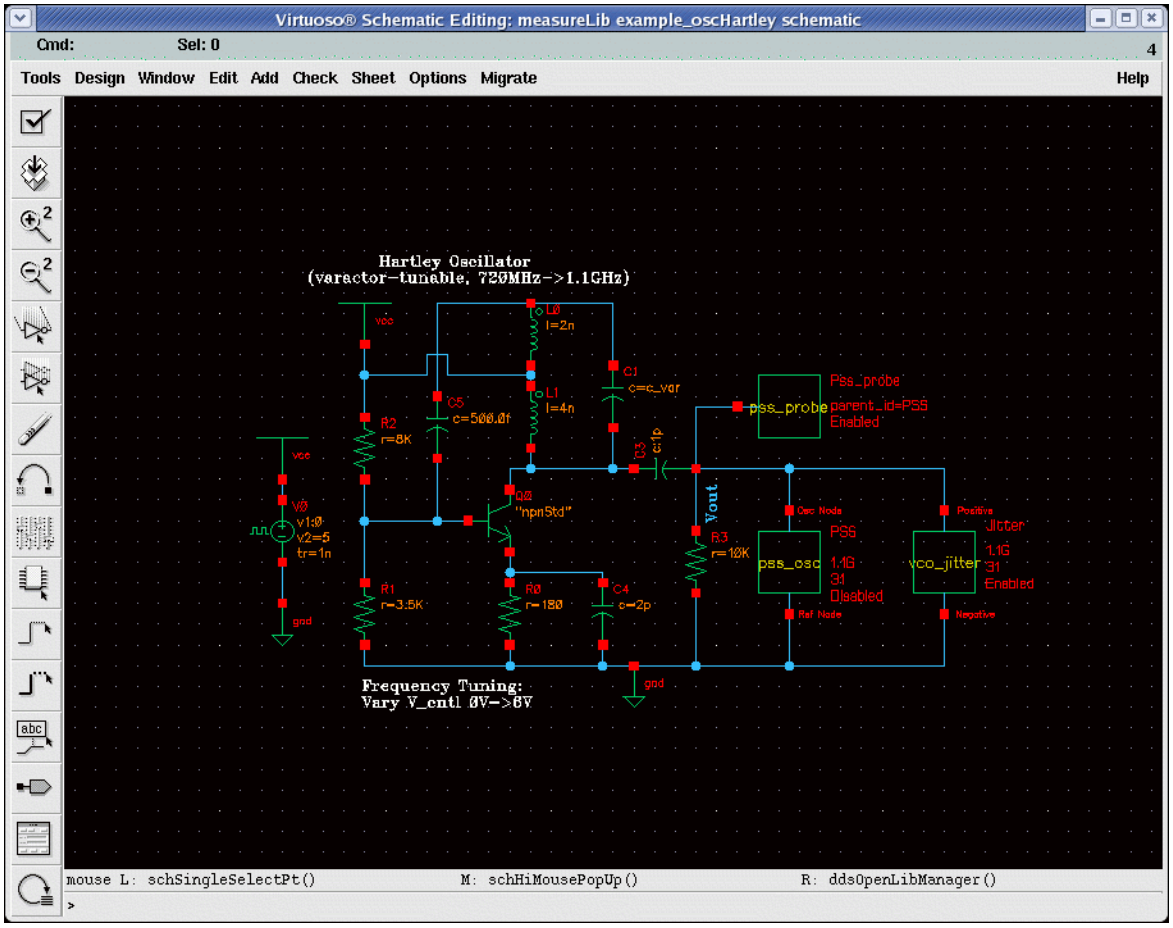
The `pss_probe` instance plots the frequency-dependent and time-dependent outputs and the ip3 point.



example_oscHartley

This example illustrates how to use the measurement library to do an autonomous PSS with a jitter measurement. The example also illustrates the ability to do multiple measurements on a single schematic.

The `example_oscHartley` design instantiates the `pss_vprobe` cell (labeled `pss_probe`), the `pss_autonomous` cell (labeled `pss_osc`), and the `vco_jitter` cell (labeled `vco_jitter`).



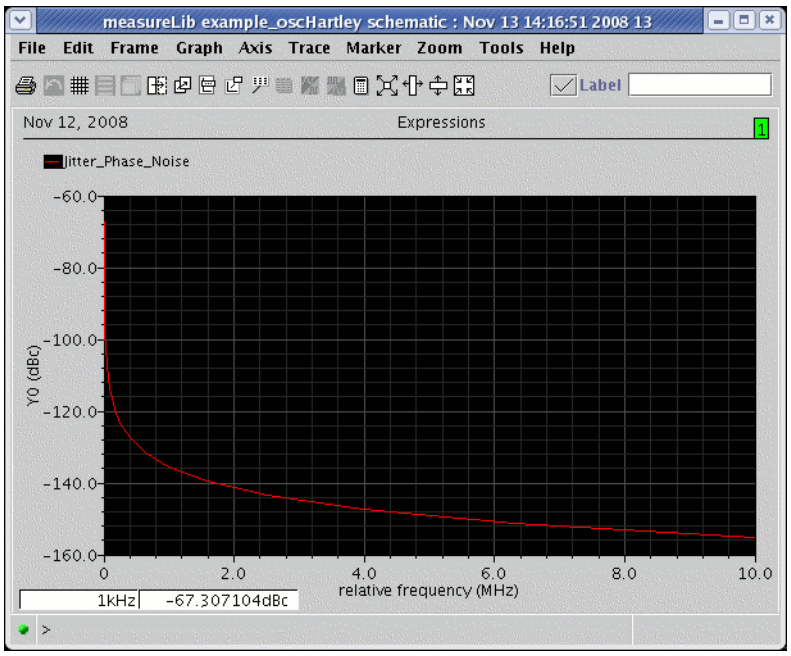
The `pss_osc` and `vco_jitter` instances in the schematic cause the following lines to be included in the netlist.

```
simulator lang=spectre
Jitter_pss ( 0 Vout ) pss flexbalance=yes fund=1.1G harms=31 \
    errpreset=moderate tstab=120n saveinit=yes method=gear2only \
    annotate=status
Jitter_jitter1 ( 0 Vout ) pnoise sweeptype=relative relharmnum=1 start=1K \
    stop=10M log=20 maxsideband=15 annotate=status
Jitter_jitter2 ( 0 Vout ) pnoise sweeptype=relative relharmnum=1 \
```

```
start=-(1K) stop=-(10M) log=20 maxsideband=15 noisetype=correlations \
cycles=[0 -2*1] annotate=status
```

These commands in the netlist set up and run the PSS and jitter analyses.

The `pss_probe` instance plots the frequency-dependent and time-dependent outputs of the PSS analysis. The instance can be connected to any net in the schematic.



Measurement Category

The `Measurement` category contains the measurement and analysis cells that Cadence provides. The cells in the `Measurement` category are further arranged as `lna`, `mixer`, `vco`, `divider`, and `PA` according to the kind of design for which the cell is suited.

For information about the Cadence-provided cells, see the following cross-references.

Divider

Measurement	Method	Cell
Jitter	Shooting PSS + Pnoise	divider_jitter on page 274
Dividing verification	Transient analysis	divider_ratio on page 276

Generic

Measurement	Method	Cell
Sweep	User-specified sweep	gen_sweep on page 277

LNA

Measurement	Method	Cell
Gain compression	One-tone HB sweep	LNA_p1db on page 236
Distortion summary	Special AC distortion summary	LNA_distortionsummary on page 238
IP2, IP3	One-tone HB + HBAC sweep	LNA_ip3_hbac on page 240
IP2, IP3	Two-tone HB	LNA_ip3_hb2 on page 242
IP2, IP3	Special AC	LNA_ip3_rapid on page 244

Measurement	Method	Cell
Gain + noise + matching	SP analysis	LNA_gain_noise_matching on page 246

Mixer

Measurement	Method	Cell
Distortion summary	one-tone HB + special HBAC distortion summary	mixer_distortionsummary on page 248
IP2, IP3	Three-tone HB sweep	mixer_ip3_hb3 on page 250
IP2, IP3	Two-tone HB + HBAC sweep	mixer_ip3_hbac on page 252
Conversion gain compression	Two-tone HB sweep	mixer_gain_compression on page 254
NF and conversion gain (blocker simulation)	Two-tone HB + HBnoise	mixer_nf_gain_blocker on page 256
NF and conversion gain	One-tone HB + HBnoise	mixer_nf_gain on page 258
IP2, IP3	One-tone HB + special HBAC rapid IP2/IP3	mixer_ip3_rapid on page 260
Conversion gain (AC)	One tone HB + HBAC	mixer_gain_hbac_LO on page 262
Conversion gain	One tone HB +HBAC	mixer_gain_hbac_RF on page 264
Conversion gain	One-tone HB + HBnoise	mixer_gain_hbnoise on page 266

PA

Measurement	Method	Cell
Power gain and power efficiency	HB sweep	PA_power on page 279

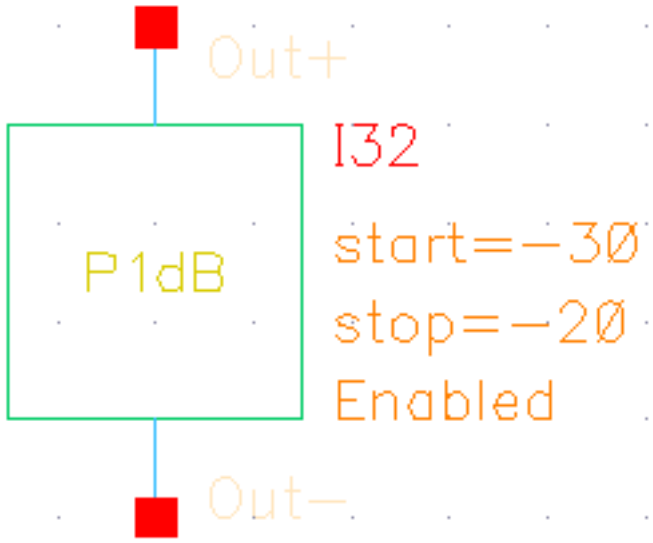
Measurement	Method	Cell
Linearity measurement	HB sweep	PA_linearity on page 281
Adjacent channel power ratio	Envelope analysis	PA_acpr on page 283

VCO

Measurement	Method	Cell
Tuning range and power consumption	Autonomous HB sweep	VCO_tuning on page 268
Phase noise	Autonomous HB + HBnoise	VCO_phasenoise on page 270
Frequency pulling	Autonomous HB sweep	VCO_freq_pulling on page 272

LNA_p1db

One-tone HB sweep



This measurement cell

1. Uses one large signal, RF, and a one-tone HB analysis sweeping the magnitude of the RF tone to get the periodic steady state.

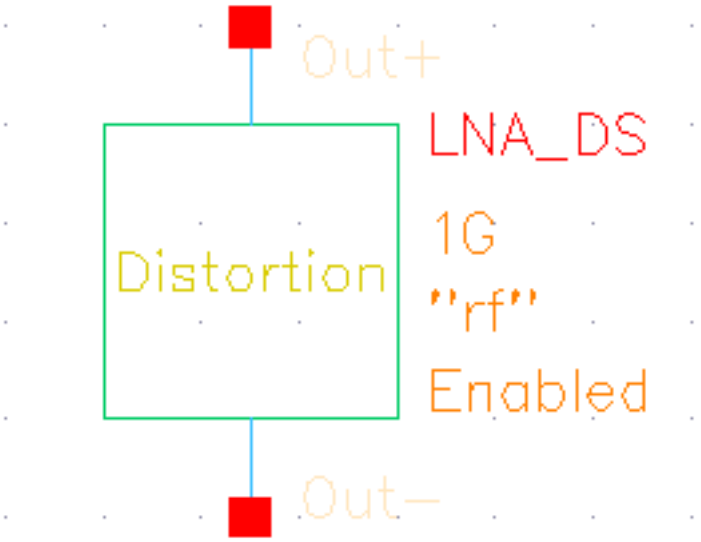
The parameters for this cell are

Parameter	Definition
Sweep param of RF tone	Name of the swept parameter for the RF source power.
Start	Starting value for the sweep.
Stop	Stopping value for the sweep.
Linear step size	The sweep step size of the linear sweep.
Number of steps	Specifies the number of steps to be used for the sweep.
Add specific points (optional)	The additional sweep points value.
Maxharms (one tone)	The number of harmonics of LO fundamental.

Parameter	Definition
Oversample	The oversample factors for each LO tone. This parameter overrides oversamplefactor.
Accuracy defaults	Quickly adjusts the simulator accuracy parameters.
Tstab	
Additional parameter for hb (optional)	
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

LNA_distortionsummary

Special AC



This measurement cell

1. Sets the RF port to DC.
2. Uses the AC distortion summary analysis to observe the summary of the linear output distortion.

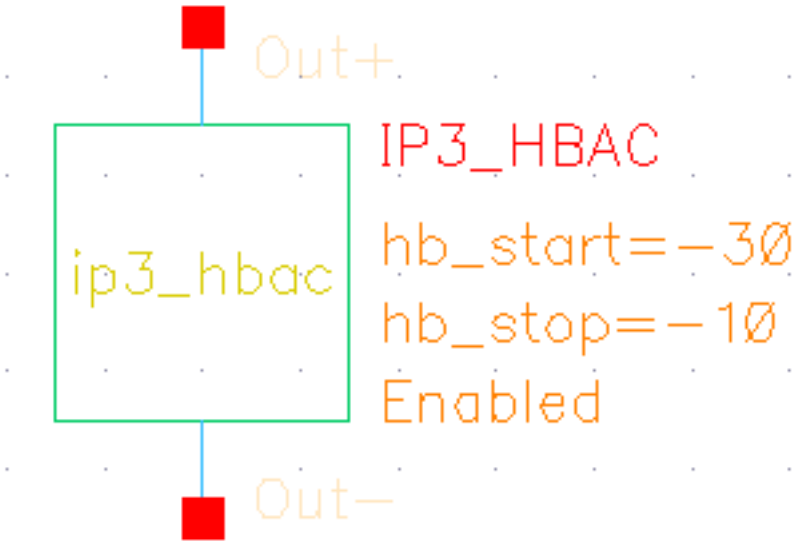
The parameters for this cell are

Parameter	Definition
Start	Starting frequency for the special AC analysis.
Maximum non-linear harms	Maximum harmonics of the input signal frequency induced by non-linear effects.
Linear output frequency	Specifies the frequency of the linear output signal.
Array of devices for distortion summary	Array of device names for the distortion summary. When the array is empty, the simulator calculates the distortion from each non-linear device.

Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

LNA_ip3_hbac

One-tone HB + HBAC, sweep



This measurement cell

1. Uses one large signal, RF1, and one small signal, RF2, set to the same amplitude.
2. Runs a one-tone HB analysis and an HBAC analysis, sweeping the amplitude to observe the
   - Second-order inter-modulation for IP2.
   - Third-order inter-modulation for IP3.

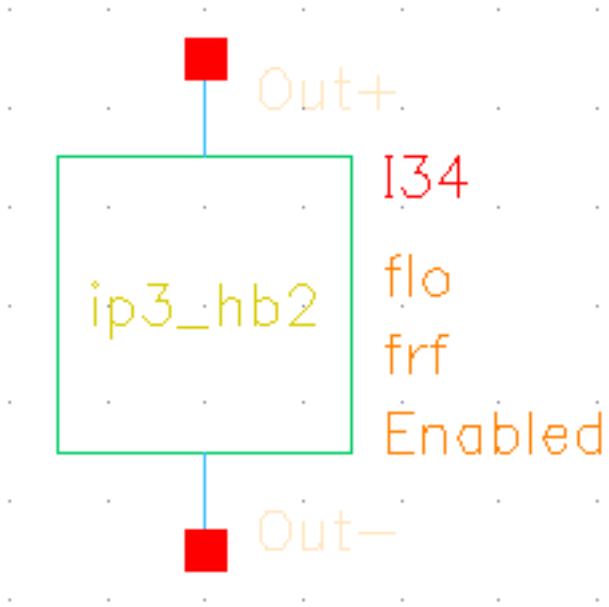
The parameters for this cell are

Parameter	Definition
Param for RF source power	Name of the swept parameter.
Start power	Beginning value for the swept parameter.
Stop power	Ending value for the swept parameter.
Linear step size	Sweep step size of linear sweep.
Number of steps	Specifies the number of steps to be used for the sweep.

Add specific points	The additional sweep points value.
Maxharms	Array of the number of harmonics of each fundamental to consider for each fundamental.
Oversample	Array of oversample factors for each tone. This parameter overrides oversamplefactor.
Tstab	
Small signal frequency	Starting frequency value for the hbac analysis.
3rd order harmonic	
1st order harmonic	
Extraction point	Starting power for extracting IP3.
Additional parameter for hb (optional)	
Additional parameter for hbac (optional)	
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

LNA_ip3_hb2

Two-tone HB



This measurement cell

1. Uses two large signals, RF1 and RF2, specified for the RF port.
2. Runs a two-tone HB analysis to observe the
   - Second-order inter-modulation for IP2.
   - Third-order inter-modulation for IP3.

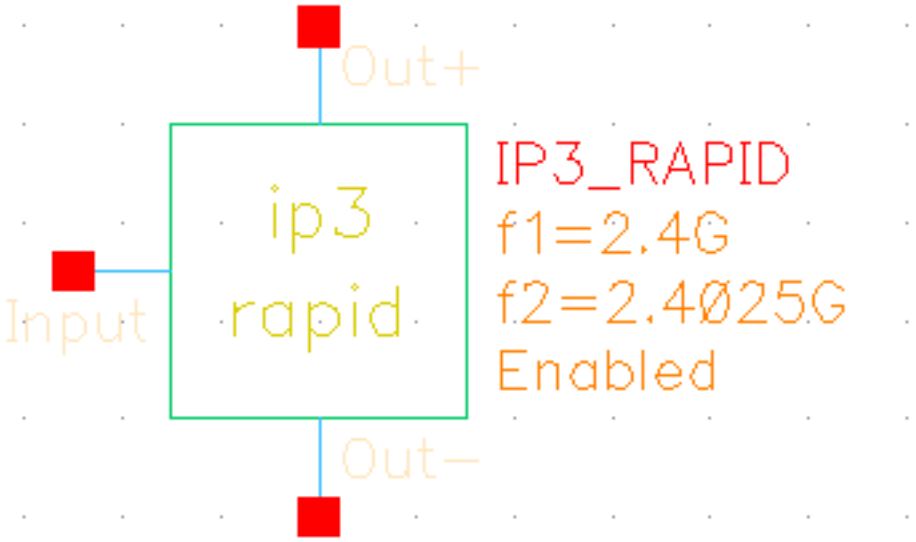
The parameters for this cell are

Parameter	Definition
Tone 1 name (LO)	The name of the LO fundamental frequency.
Tone 2 name (RF)	The name of the RF fundamental frequency.
Maxharms (two tones)	Array of the number of harmonics of each fundamental to consider for each fundamental.
Oversample	Array of oversample factors for each tone. This parameter overrides oversamplefactor.
Accuracy defaults	Quickly adjusts the simulator accuracy parameters.

Tstab	Specifies the initial stabilization time.
Accuracy defaults	Quickly adjusts the simulator accuracy parameters.
3rd order harmonic	
1st order harmonic	
Additional parameter for hb (optional)	
Extraction point	Starting power value for extracting IP3.
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

LNA_ip3_rapid

Special AC



This measurement cell

1. Sets the RF port to DC, then uses the special AC rapid IP2/IP3 analysis to observe the
   - Second-order inter-modulation for IP2.
   - Third-order inter-modulation for IP3.

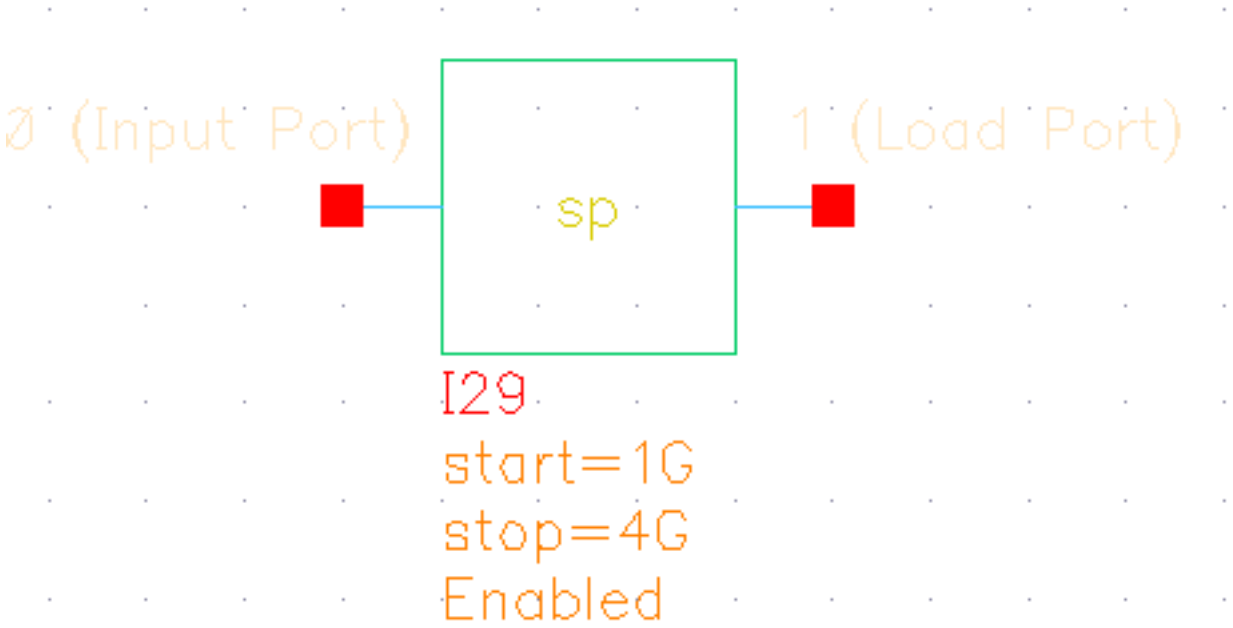
The parameters for this cell are

Parameter	Definition
RF frequency 1 iPart(f1)	The smaller rf frequency.
RF frequency 2 iPart(f2)	The larger rf frequency.
Maximum non-linear harms	Maximum harmonics of the input signal frequency induced by non-linear effects.
Linear output frequency	Specifies the frequency of the linear output signal.
IM output frequency	Specifies the frequency of the IM output signal.

Input power (dBm)	Specifies the RF source power.
ac_additional	
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

LNA_gain_noise_matching

SP analysis



This measurement cell

1. Sets the RF and output ports to DC, then sweeps the frequency to observe the gain, noise, and matching.

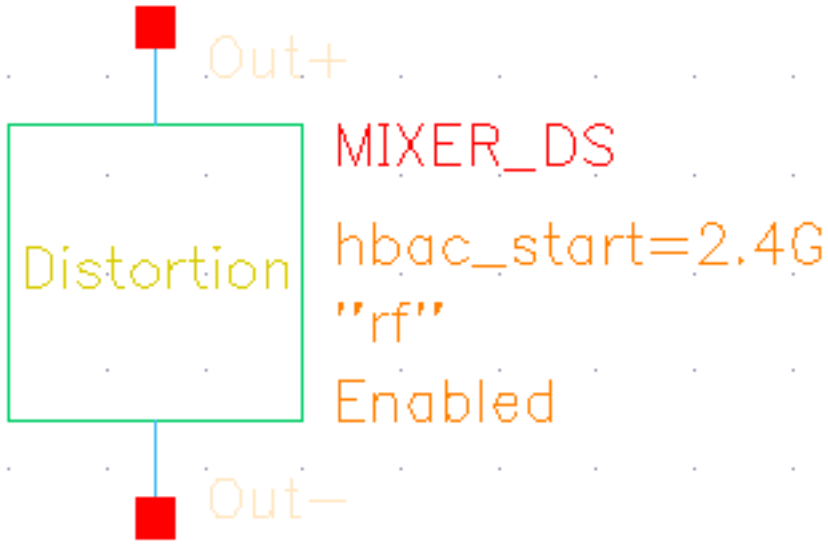
The parameters for this cell are

Parameter	Definition
Start	Starting value for the sweep.
Stop	Stopping value for the sweep.
Linear step size	
Number of linear steps	Specifies the number of steps to be used for the sweep.
Logarithmic step size	
Number of logarithmic steps	

Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

mixer_distortionsummary

One-tone HB + special HBAC (distortion summary)



This measurement cell

1. Treats LO as a large signal and runs a one-tone HB analysis to get the periodic steady state.
2. Uses a small signal, RF1, specified for the RF port, with a distortion summary HBAC analysis to get the summary of distortion at the linear output frequency (LO-RF).

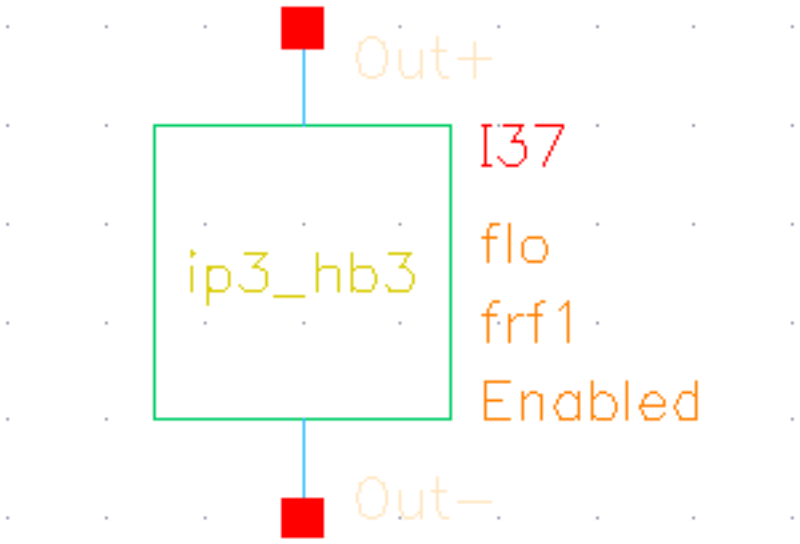
The parameters for this cell are

Parameter	Definition
Maxharms	Array of the number of harmonics of each fundamental to consider for each fundamental.
Oversample	Array of oversample factors for each tone. This parameter overrides oversamplefactor.
Accuracy defaults	Quickly adjusts the simulator accuracy parameters.
Tstab	Specifies the initial stabilization time.
RF frequency	Starting frequency value for the HBAC analysis.

Linear output frequency	Specifies the frequency of the linear output signal.
Array of devices for distortion summary	Array of device names for the distortion summary. When the array is empty, the simulator calculates the distortion from each non-linear device.
Maximum sideband	Specifies the number of frequency conversion terms to take into account.
Additional parameter for hb	
Additional parameter for hbac	
LO frequency divide ratio	Large signal frequency division.
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

mixer_ip3_hb3

Three-tone HB sweep



Assuming a down-converted mixer, this measurement cell

1. Operates on two large signals, RF1 and RF2, specified for the RF port. The signals have equal amplitude or dbm values.
2. Treats LO, RF1, and RF2 as three large signals and runs a three-tone HB analysis, sweeping the two RF tone amplitude or dbm values to observe the magnitude changes of the
   - First-order inter-modulation (RF1-LO or RF2-LO)
   - Second-order inter-modulation (RF2-RF1) for IP2
   - Third-order inter-modulation (2RF2-RF1-LO or 2RF1-RF2-LO) for IP3

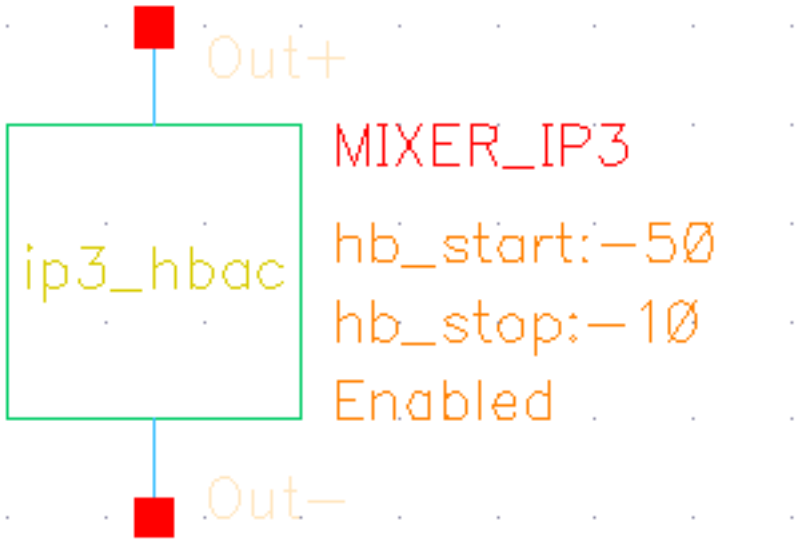
The parameters for this cell are

Parameter	Definition
Param for source power	Name of the swept parameter.
Start power	Starting value for the sweep.
Stop power	Stopping value for the sweep.

Linear step size	
Number of steps	Specifies the number of steps to be used for the sweep.
Add specific points	
Tone 1 name (LO)	
Tone 2 name (RF1)	
*Tone 3 name (RF2_*	
Maxharms (three tones)	Array of the number of harmonics of each fundamental to consider for each fundamental.
Oversample (three tones)	Array of oversample factors for each tone. This parameter overrides oversamplefactor.
Tstab	
Accuracy defaults	Quickly adjusts the simulator accuracy parameters.
LO frequency divide ratio	Large signal frequency division.
3rd order harmonic	
1st order harmonic	
Additional parameter for hb	
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

mixer_ip3_hbac

Two-tone HB + HBAC sweep



Assuming a down converted mixer, this measurement cell

1. Operates on one large signal, RF1, and one small signal, RF, specified for the RF port. The signals have equal amplitude or dbm values.
2. Treats LO and RF1 as two large signals and runs two-tone HB and HBAC analyses, sweeping the two RF signal amplitude or dbm values to observe the magnitude changes of the
   - First-order inter-modulation (RF1-LO)
   - Second-order inter-modulation (RF2-RF1) for IP2
   - Third-order inter-modulation (2RF1-RF2-LO) for IP3

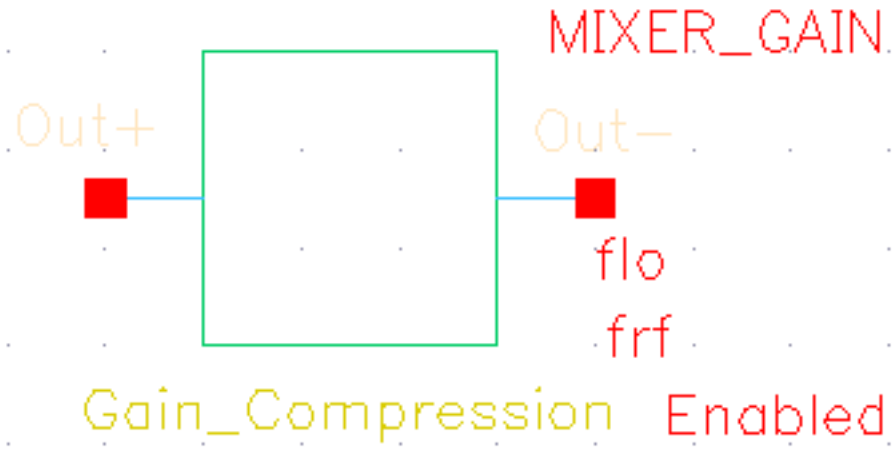
The parameters for this cell are

Parameter	Definition
Param for RF source power	Name of the swept parameter.
Start power	Starting value for the sweep.
Stop power	Stopping value for the sweep.

Linear step size	Specifies the size of the step to be used in the sweep.
Number of steps	
Add specific points	Specifies additional sweep points for the analysis.
Tone 1 name (LO)	
Tone 2 name (RF)	
Maxharms (two tones)	Array of the number of harmonics of each fundamental to consider for each fundamental.
Oversample (two tones)	Array of oversample factors for each tone. This parameter overrides oversamplefactor.
Accuracy defaults	Quickly adjusts the simulator accuracy parameters.
Tstab	Specifies the initial stabilization time.
LO frequency divide ratio	Large signal frequency division.
3rd tone frequency	Starting frequency value for the HBAC analysis.
CP 1st order harmonic	
IP3 3rd order harmonic	
IP3 1st order harmonic	
Extraction point	
Additional parameter for hb	
Additional parameter for hbac	
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

mixer_gain_compression

Two-tone HB sweep



Assuming a down converted mixer, this measurement cell

1. Treats LO and RF as two large signals and runs a two-tone HB analysis to get the periodic steady state.
2. Sweeps the RF tone amplitude from the linear region to the saturated reason to observe how the conversion gain changes.

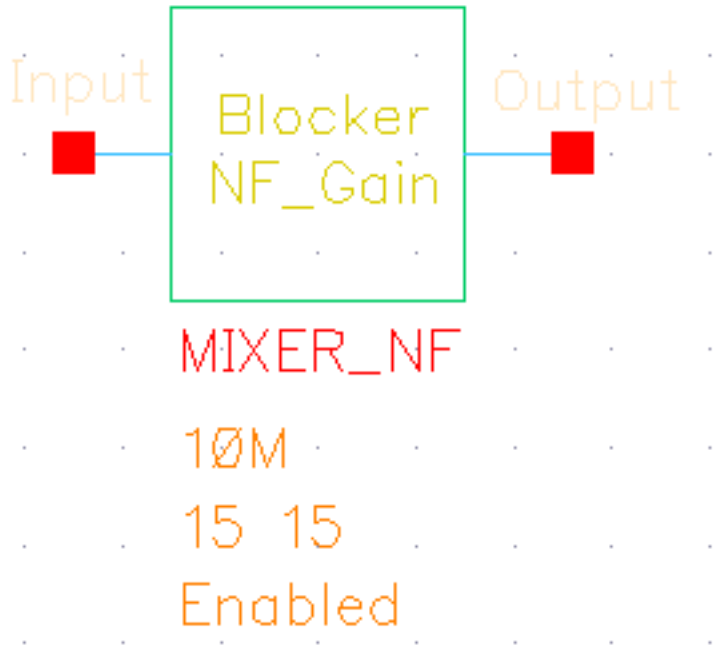
The parameters for this cell are

Parameter	Definition
Sweep param for RF	Name of the swept parameter.
Start power	Starting value for the sweep.
Stop power	Stopping value for the sweep.
Linear step size	Specifies the size of the step to be used in the sweep.
Number of steps	
Add specific points	Specifies additional sweep points for the analysis.
Tone 1 name (LO)	
Tone 2 name (RF)	

Maxharms (two tones)	Array of the number of harmonics of each fundamental to consider for each fundamental.
Oversample (two tones)	Array of oversample factors for each tone. This parameter overrides oversamplefactor.
Accuracy defaults	Quickly adjusts the simulator accuracy parameters.
Tstab	Specifies the initial stabilization time.
1st order harmonic	
LO frequency divide ratio	Large signal frequency division.
Additional parameter for hb	
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

mixer_nf_gain_blocker

Two-tone HB+HBNoise



This measurement cell

1. Treats LO and blocker as two large signals and runs a two-tone HB analysis to get the periodic steady state.
2. Uses specified input and output probes and specified reference sidebands, where

   ```
   |f(input)| = |f(out) + refsideband * fund(HB)|
   ```
3. Runs an HBnoise analysis to get the total noise at the output and the NF and conversion-gain from iprobe to oprobe. You can sweep the blocker magnitude, LO magnitude, or output frequency.

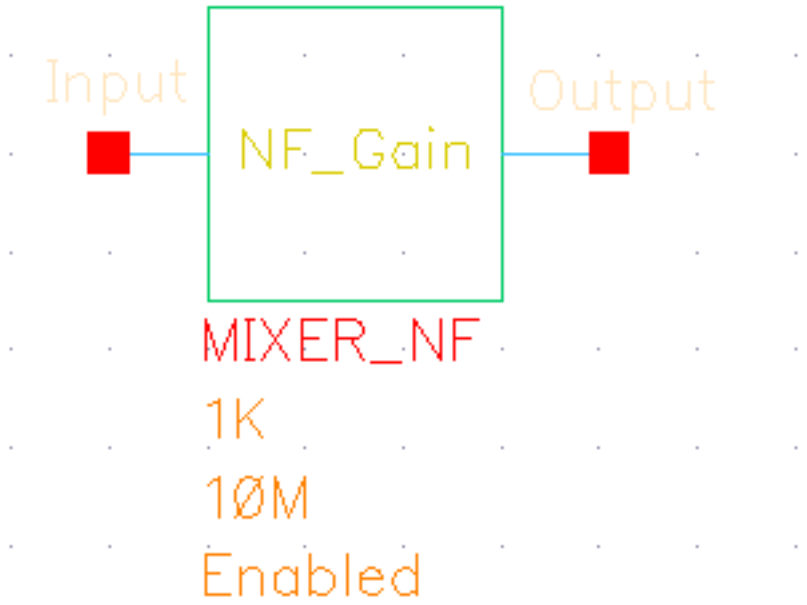
The parameters for this cell are

Parameter	Definition
Sweep param for blocker	Name of the swept parameter for the blocker analysis.
Start	Starting value for the sweep.
Stop	Stopping value for the sweep.

Linear step size	Specifies the size of the step to be used in the sweep.
Number of steps	
Add specific points	
Maxharms (two tones)	Array of the number of harmonics of each fundamental to consider for each fundamental.
Oversample (two tones)	Array of oversample factors for each tone. This parameter overrides oversamplefactor.
Accuracy defaults	Quickly adjusts the simulator accuracy parameters.
Tstab	Specifies the initial stabilization time.
LO frequency divide ratio	Large signal frequency division.
hbnoise start frequency	Starting frequency for the HBnoise analysis.
hbnoise maximum sideband	Specifies the number of frequency conversion terms to take into account.
Refsideband	Identifies the noise generator and the reference sidebands to use for the Pnoise simulation.
Additional parameter for hb	
Additional parameter for hbnoise	
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

mixer_nf_gain

One-tone HB+HBNoise



This measurement cell

1. Treats LO as a large signal and runs a one-tone HB analysis to get the periodic steady state.
2. Uses specified input and output probes and specified reference sidebands, where
   `|f(input)| = |f(out) + refsideband * fund(HB)|`
3. Runs an HBnoise analysis to get the total noise at the output and the NF and conversion-gain from iprobe to oprobe. You can sweep the LO magnitude or the output frequency.

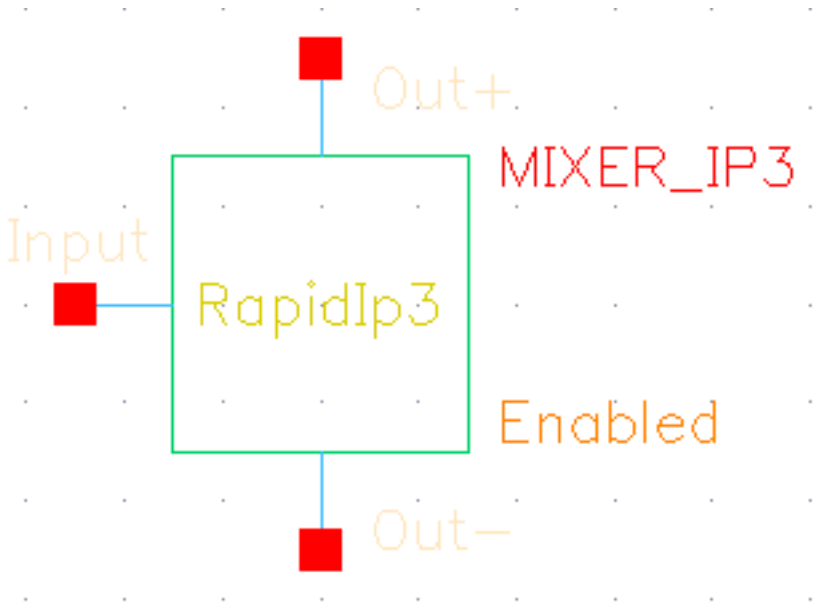
The parameters for this cell are

Parameter	Definition
Maxharms	Array of the number of harmonics of each fundamental to consider for each fundamental.
Oversample	Array of oversample factors for each tone. This parameter overrides oversamplefactor.
Accuracy defaults	Quickly adjusts the simulator accuracy parameters.

Tstab	Specifies the initial stabilization time.
LO frequency divide ratio	Large signal frequency division.
hbnoise start frequency	Starting value for the sweep.
hbnoise stop frequency	Stopping value for the sweep.
Refsideband	Identifies the noise generator and the reference sidebands to use for the Pnoise simulation.
Display output params	Displays the fields used to control output.
Output sideband	Lists the sidebands you requested for the small-signal analysis.
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

mixer_ip3_rapid

One-tone HB + special HBAC (rapid IP2/IP3).



Assuming a down converted mixer, this measurement cell

1. Operates on two small signals, RF1 and RF2, specified for the RF port. The signals have equal amplitude or dbm values.
2. Treats LO as a large signal and runs one-tone HB and HBAC analyses
3. Uses the special HBAC (rapid IP2/IP3) analysis to observe the magnitude changes of the
   - First-order inter-modulation (RF1-LO)
   - Second-order inter-modulation (RF2-RF1) for IP2
   - Third-order inter-modulation (2RF1-RF2-LO) for IP3

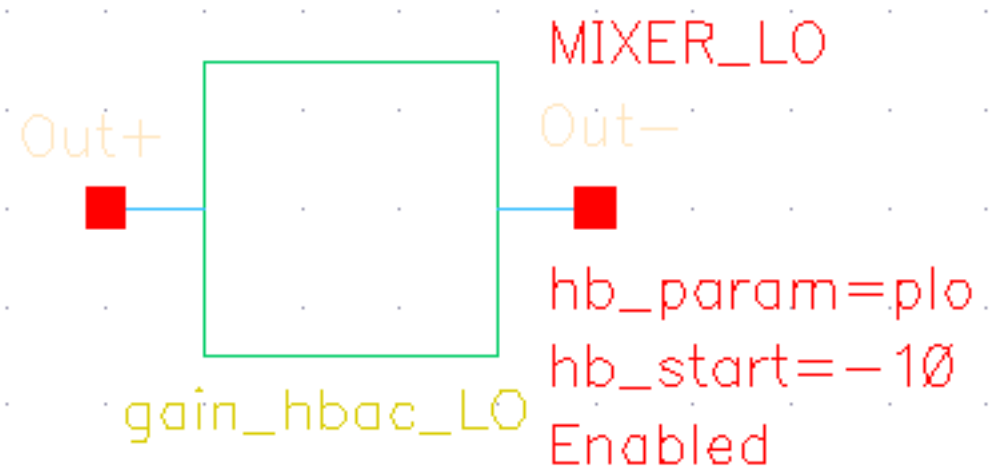
The parameters for this cell are

Parameter	Definition
Maxharms of LO	Array of the number of harmonics of each fundamental to consider for each fundamental.
Oversample	Array of oversample factors for each tone. This parameter overrides oversamplefactor.

Tstab	Specifies the initial stabilization time.
LO Frequency iPar(fund)	The frequency of LO fundamental
RF1 Frequency iPar(f1)	The frequency of the smaller RF fundamental
RF2 Frequency iPar(f2)	The frequency of the larger RF fundamental
Linear output frequency	Specifies the frequency of the linear output signal.
IM output frequency	Specifies the frequency of the IM output signal.
Input power (dBm)	Specifies the RF source power.
Accuracy defaults	Quickly adjusts the simulator accuracy parameters.
LO frequency divide ratio	Large signal frequency division.
Additional parameter for hb	
Additional parameter for hbac	
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

mixer_gain_hbac_LO

One-tone HB+ HBAC sweeping LO magnitude



This measurement cell

1. Treats LO as a large signal and runs a one-tone HB analysis to get the periodic steady state.
2. Uses an input source specified by its pacmag and runs an HBAC analysis to get the conversion gain from the input to any net. You can sweep the LO magnitude or the small signal frequency at the input.

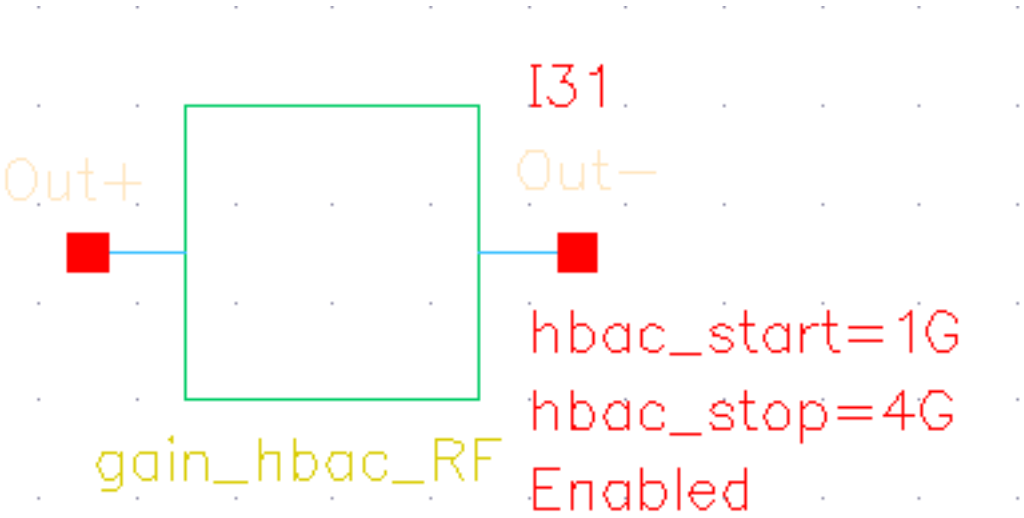
The parameters for this cell are

Parameter	Definition
Param for LO input power	Name of the swept parameter for the LO input.
Start power	Starting value for the sweep.
Stop power	Stopping value for the sweep.
Linear step size	Specifies the size of the step to be used in the sweep.
Number of steps	Array of names of the fundamental frequencies to be used in the analysis.
Maxharms	Array of the number of harmonics of each fundamental to consider for each fundamental.

Oversample	Array of oversample factors for each tone. This parameter overrides oversamplefactor.
Accuracy defaults	Quickly adjusts the simulator accuracy parameters.
Tstab	Specifies the initial stabilization time.
LO frequency divide ratio	Large signal frequency division.
Input frequency	Starting frequency for the HBAC analysis.
Output harmonic	Lists, by number and associated frequency, the available output harmonics.
Additional parameter for hb	
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

mixer_gain_hbac_RF

One-tone HB+ HBAC sweeping RF frequency



This measurement cell

1. Treats LO as a large signal and runs a one-tone HB analysis to get the periodic steady state.
2. Sweep the frequency of the RF signal to run an HBAC analysis and measures how conversion gain varies with the frequency of the stimuli.

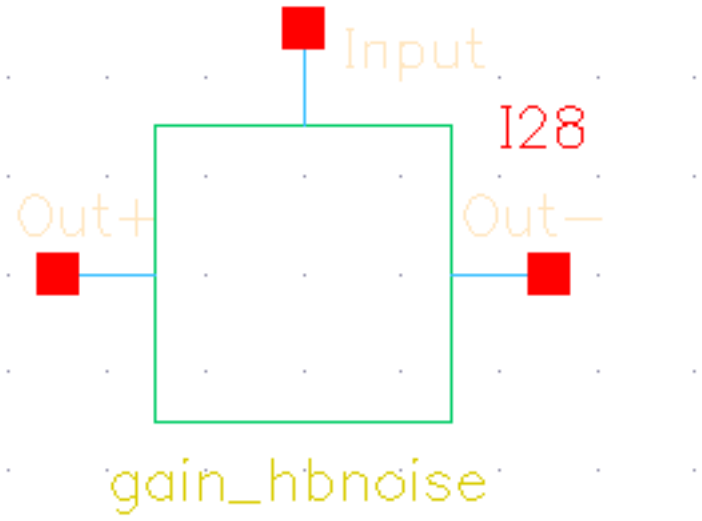
The parameters for this cell are

Parameter	Definition
Maxharms	Array of the number of harmonics of each fundamental to consider for each fundamental.
Oversample	Array of oversample factors for each tone. This parameter overrides oversamplefactor.
Accuracy defaults	Quickly adjusts the simulator accuracy parameters.
Tstab	Specifies the initial stabilization time.
LO frequency divide ratio	Large signal frequency division.
Iput frequency start	Starting value for the sweep.

Input frequency stop	Stopping value for the sweep.
Linear step size	Step size, linear sweep.
Number of linear steps	Number of steps, linear sweep.
Logarithmic step size	Points per decade, log sweep.
Number of logarithmic steps	Number of steps, log sweep.
Add specific points	Array of sweep values.
Output sideband	Lists the sidebands you requested for the small-signal analysis.
Additional parameter for hb	
Additional parameter for hbac	
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

mixer_gain_hbnoise

Conversion gain (XF)



This measurement cell

1. Treats LO as a large signal and runs a one-tone HB analyses to get the periodic steady state.
2. Uses a specified output as an observation probe and runs an HBnoise analysis with `xfonly=yes` to get the conversion gain from any source to the probe. You can sweep the LO magnitude or the observation frequency at the output.

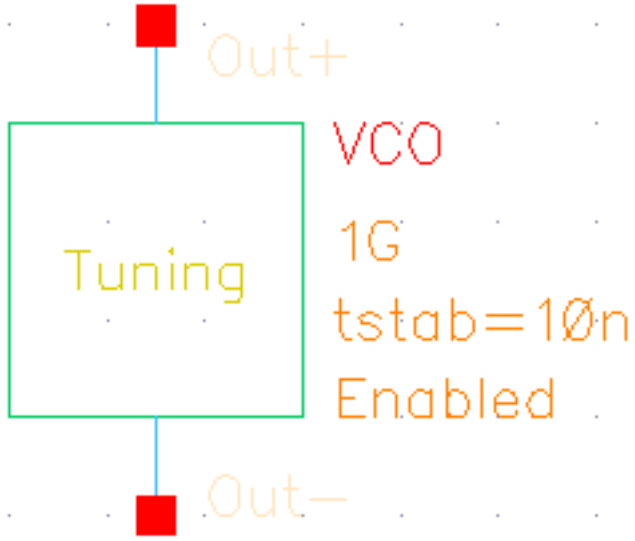
The parameters for this cell are

Parameter	Definition
Maxharms	Array of the number of harmonics of each fundamental to consider for each fundamental.
Oversample	Array of oversample factors for each tone. This parameter overrides oversamplefactor.
Accuracy defaults	Quickly adjusts the simulator accuracy parameters.
Tstab	Specifies the initial stabilization time.
LO frequency divide ratio	Large signal frequency division.

hbnoise start	Starting value for the sweep.
hbnoise stop	Stopping value for the sweep.
Output sideband	Lists the sidebands you requested for the small-signal analysis.
Additional parameter for hb	
Additional parameter for hbnoise	
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

VCO_tuning

Autonomous HB sweep



An HB sweep is used to simulate the tuning range and power consumption of a VCO. Before running the HB analysis, you must use the analog design environment (ADE) to specify the two nodes of the VCO. To do that, in the ADE window, choose *Outputs – To Be Saved* and select the nodes.

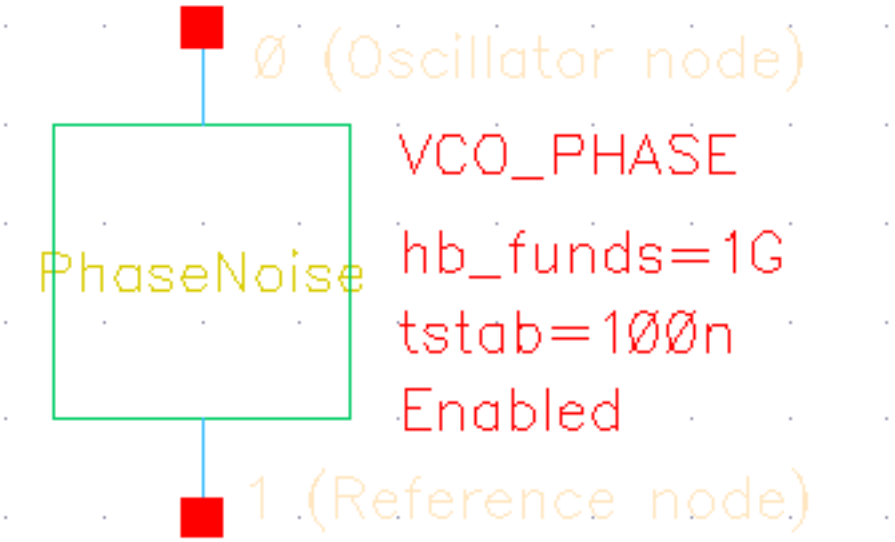
The parameters for this cell are

Parameter	Definition
Sweep control voltage	VCO control voltage to be swept.
Start	Starting value for the sweep.
Stop	Stopping value for the sweep.
Linear step size	
Number of steps	Specifies the number of steps to be used for the sweep.
Guess frequency	The guess frequency for the VCO.
Maxharms	Array of the number of harmonics of each fundamental to consider for each fundamental.
Oversample	Array of oversample factors for each tone. This parameter overrides oversamplefactor.

Accuracy defaults	Quickly adjusts the simulator accuracy parameters.
Tstab	Specifies the initial stabilization time.
Additional parameter for hb	
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

VCO_phasenoise

Autonomous HB + HBNOISE



This measurement cell

1. Uses an autonomous HB analysis to get the period steady state.
2. Uses an HBnoise analysis to get the phase noise.

   Phase noise is the ratio of the output power divided by the noise power at a specified value. Phase noise is expressed in dBc/Hz.

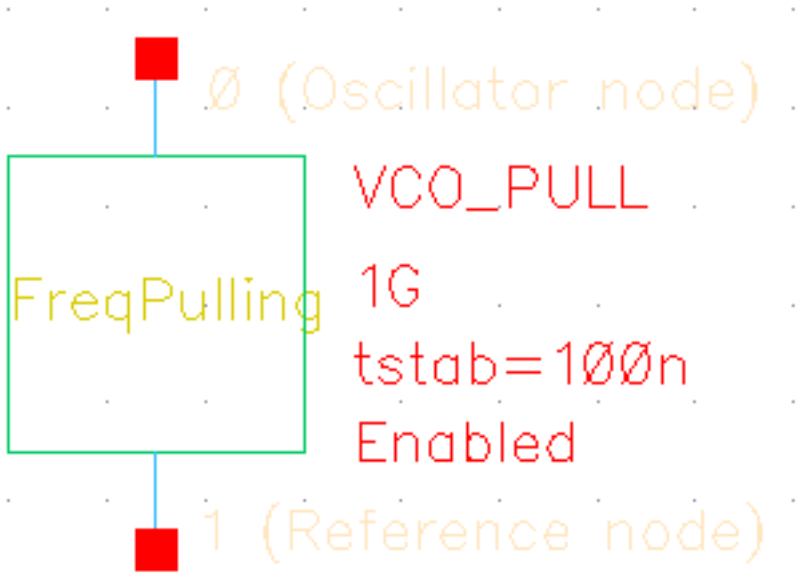
The parameters for this cell are

Parameter	Definition
Guess frequency	The guess frequency for the VCO.
Maxharms	Array of the number of harmonics of each fundamental to consider for each fundamental.
Oversample	Array of oversample factors for each tone. This parameter overrides oversamplefactor.
Accuracy defaults	Quickly adjusts the simulator accuracy parameters.
Tstab	Specifies the initial stabilization time.

Relative harmonic	Specifies the sideband to which a relative frequency sweep should be referenced.
hbnoise start frequency	Starting value for the sweep.
hbnoise stop frequency	Stopping value for the sweep.
Logarithmic step size	
Number of logarithmic steps	Specifies the number of steps to be used for the logarithmic sweep.
Additional parameter for hb	
Additional parameter for hbnoise	
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

VCO_freq_pulling

Autonomous HB sweep



This measurement cell

1. Connects an instance of a PortAdaptor to the load.
2. Runs an HB analysis sweep of the theta parameter from 0 to 359 degrees.
3. Estimates the frequency pulling.

Frequency pulling is a measure of the frequency change due to a non-ideal load. It is measured by determining the frequency change of a load with a nominal 12dB return loss, with all possible phases.

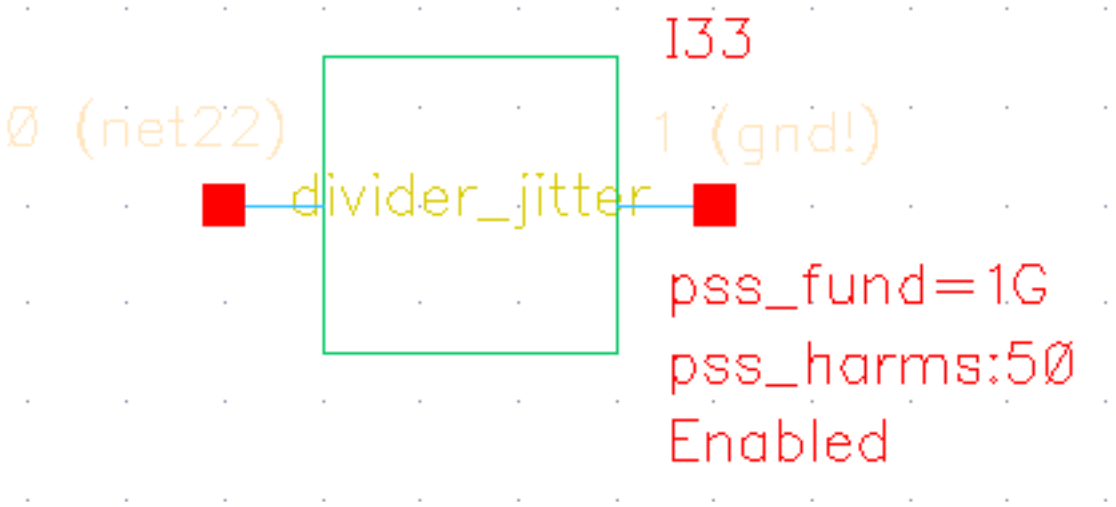
The parameters for this cell are

Parameter	Definition
Param for phase of Gamma	
Start phase	Starting value for the sweep.
Stop phase	Stopping value for the sweep.
Number of steps	Specifies the number of steps to be used for the sweep.

Guess frequency	The guess frequency of the LO
Maxharms	Array of the number of harmonics of each fundamental to consider for each fundamental.
Oversample	Array of oversample factors for each tone. This parameter overrides oversamplefactor.
Accuracy defaults	Quickly adjusts the simulator accuracy parameters.
Tstab	Specifies the initial stabilization time.
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

divider_jitter

Shooting PSS + PNOISE



This measurement cell

1. Runs a shooting PSS analysis and a Pnoise analysis (with `type=pmijitter`) to estimate the jitter spectrum and jitter at the transition point.

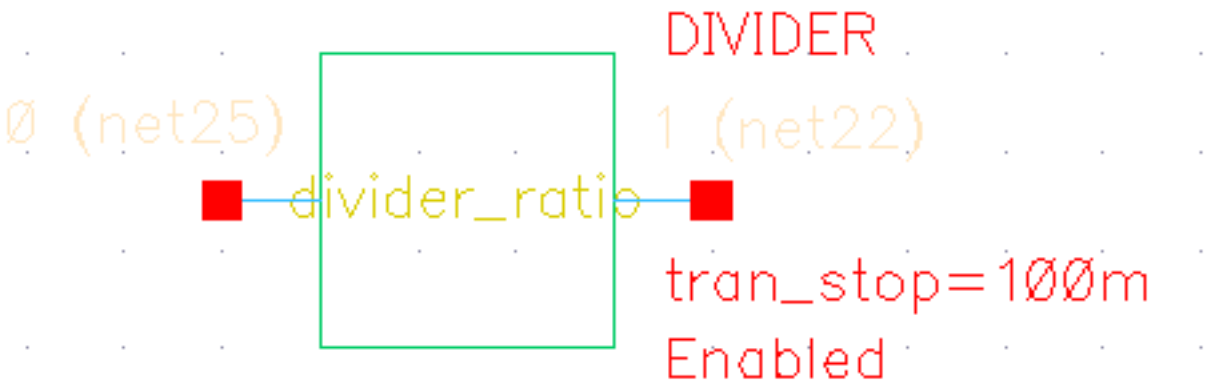
The parameters for this cell are

Parameter	Definition
Beat frequency	The fundamental of the circuit.
Number of harmonics	Specifying 3 harmonics is sufficient for this measurement.
Accuracy defaults	Quickly adjusts the simulator accuracy parameters.
Tstab	For the shooting engine, specifies an amount of additional time to allow for the circuit to settle.
Start freq of pnoise	Starting frequency for the analysis.
Stop freq of pnoise	Ending frequency for the analysis.
Number of steps, log sweep	Specifies the number of steps to be used for the sweep, which is logarithmic.
Maximum sideband	Specifies the number of frequency conversion terms to take into account.

Refsideband	Identifies the noise generator and the reference sidebands to use for the Pnoise simulation.
Crossing direction	Specifies the direction of the transition where jitter is measured.
Threshold value	Specifies the threshold value at which PM jitter is measured.
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

divider_ratio

Transient analysis



This measurement cell

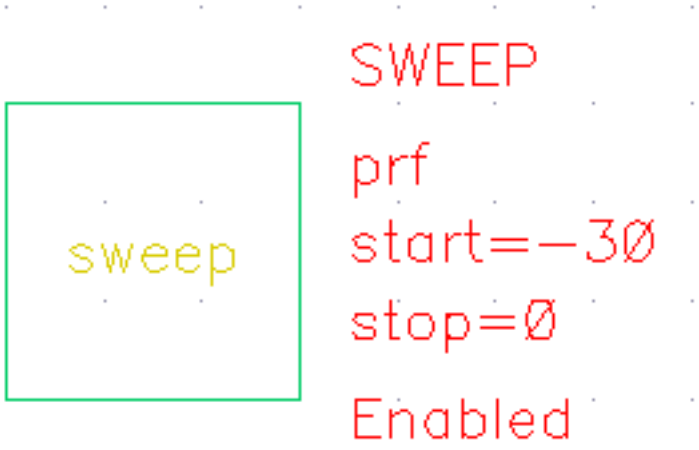
1. Runs a transient analysis to estimate the output period.

The parameters for this cell are

Parameter	Definition
Stop time	Specifies when the transient analysis is to end.
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

gen_sweep

User-specified linear sweep.



This measurement cell

1. Allows you to specify a parametric sweep to be used with another measurement cell. To use this cell, you instantiate it in the schematic and specify the sweep parameters. Then you instantiate the other measurement cell and specify the name of the `gen_sweep` cell as a parent_id.

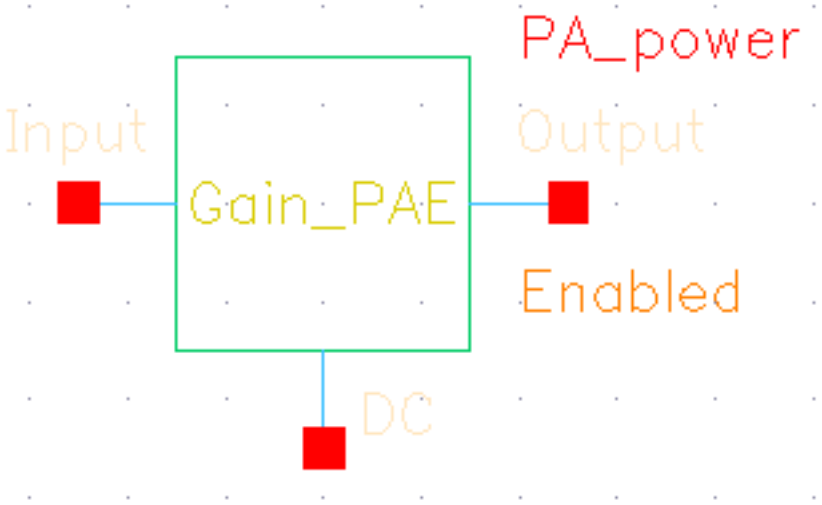
The parameters for this cell are

Parameter	Definition
Sweep parameter	Name of the parameter you want to sweep.
Start	Starting value for the sweep.
Stop	Stopping value for the sweep.
Linear step size	Step size for the sweep.
Number of steps	Specifies the number of steps to be used for the sweep.
Logarithmic step size	
Number of logarithmic steps	

Number of additional points	Specifies the number of additional sweep points for the analysis.
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

PA_power

HB sweep



This measurement cell

1. Treats the input as a large signal and runs an HB analysis. The cell sweeps the input amplitude if necessary.

The parameters for this cell are

Parameter	Definition
Parameter of source power	Swept input power.
Start power	Starting value for the sweep.
Stop power	Stopping value for the sweep.
Linear step size	Specifies the size of the step to be used in the sweep.
Number of steps	
Add specific points	
Maxharms	Array of the number of harmonics of each fundamental to consider for each fundamental.
Oversample	Array of oversample factors for each tone. This parameter overrides oversamplefactor.

Accuracy defaults	Quickly adjusts the simulator accuracy parameters.
Tstab	Specifies the initial stabilization time.
Additional parameter for hb	Saves the initial results of the transient analysis.
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

PA_linearity

HB sweep

This measurement cell

1. Sweeps the input power level and runs a one-tone HB analysis to determine how the gain changes as the input power varies.
2. Treats RF2 as a small signal and uses an HBAC analysis to get the third-order inter-modulation.

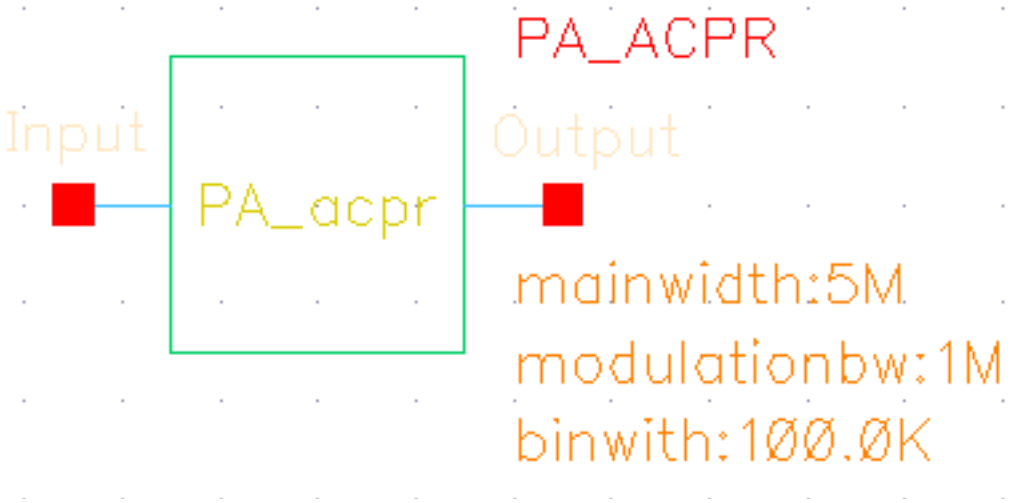
The parameters for this cell are

Parameter	Definition
Sweep parameter for RF	Swept parameter for the RF analysis.
Start	Starting value for the sweep.
Stop	Stopping value for the sweep.
Step size	Specifies the size of the step to be used in the sweep.
Maxharms	
Oversample	
Accuracy defaults	Quickly adjusts the simulator accuracy parameters.
Tstab	Specifies the initial stabilization time.
Small signal frequency	RF2 frequency.

3rd Order Harmonic	Harmonic for the RF1 frequency, which is used to calculate IP3.
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

PA_acpr

Envelope analysis



1. Determines the adjacent channel power ratio (ACPR), which is the ratio of the total power of the adjacent channels (the intermodulation signal) to the power in the main channel (the useful signal).

The parameters for this cell are:

Parameter	Definition
Main channel width	Bandwidth of the main channel.
Modulation band width	Bandwidth of the adjacent channel.
Frequency bin width	Frequency resolution.
Accuracy defaults	Quickly adjusts the simulator accuracy parameters.
Enabled/Disabled	When set to *Enabled*, includes the measurement in future simulations.

3

rfTlineLib Library

The elements contained in the RF transmission line library, `rfTlineLib`, are:

- Microstrip Components
  - microbend2 on page 287
  - microbend90 on page 289
  - microopenend on page 291
  - microstep on page 293
  - microstrip on page 295
- Stripline Components
  - stripbend90 on page 298
  - stripline on page 300

These components can be placed in a schematic to approximate the effect that the layout has on the operation of the circuit. When the schematic is netlisted, the tool generates an equivalent RLCG subcircuit for each `rfTlineLib` component and instantiates the new subcircuits in the design. This additional detail improves the accuracy of the simulation.

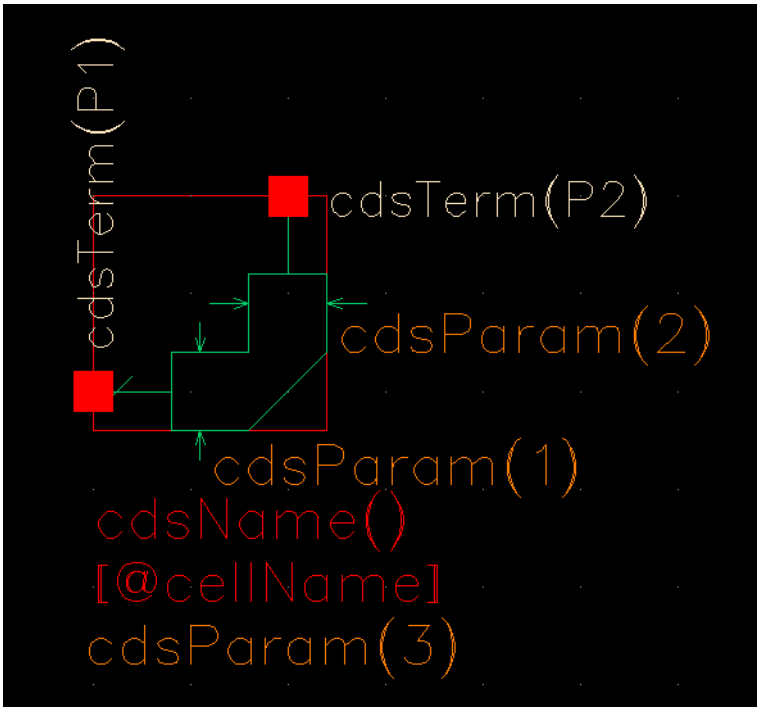
Placing the `rfTlineLib` components in the schematic requires you to provide information such as the widths and thicknesses of strips and the heights of substrates. The process of adding one of these components to the schematic opens a form where you provide the required information as CDF values.

Microstrip Components

A microstrip consists of a conducting strip that is separated from a ground plate by a substrate. The dielectric on the other side of the conducting strip is typically air.

microbend2

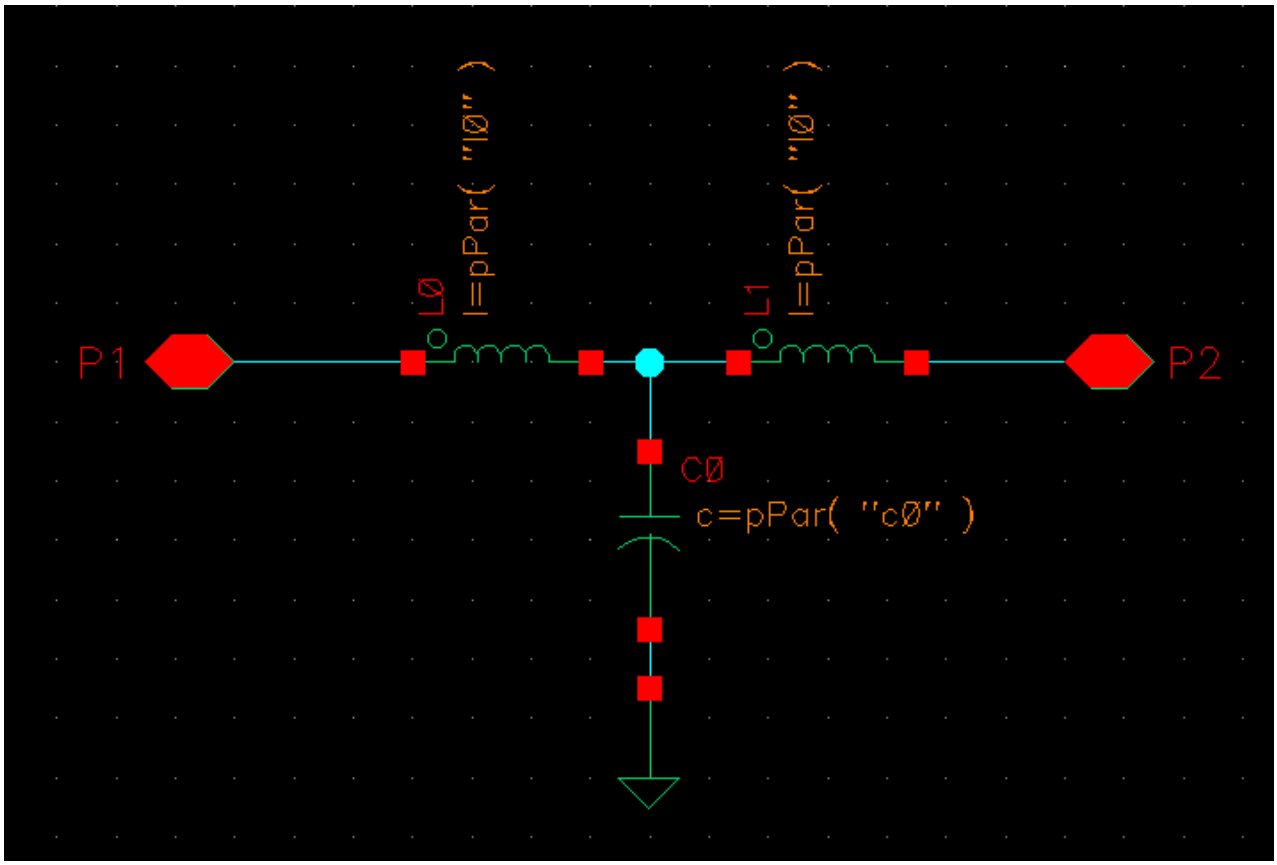
This is a microstrip chamfered right-angle bend.



The parameters are:

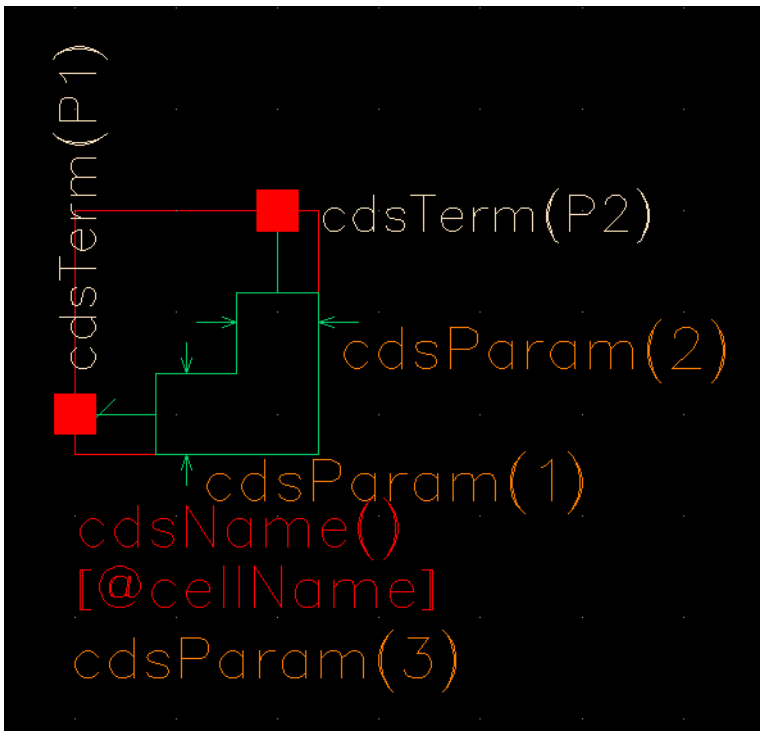
`Dielectric constant of subs`	Relative dielectric constant of the substrate.
`Height of substrate`	Height of the substrate, in meters.
`Width of strip`	Width of the microstrip, in meters. The width is the same on both sides of the bend and, in the symbol, is labeled as both `cdsParam(1)` and `cdsParam(2)`.

The equivalent circuit is this.



microbend90

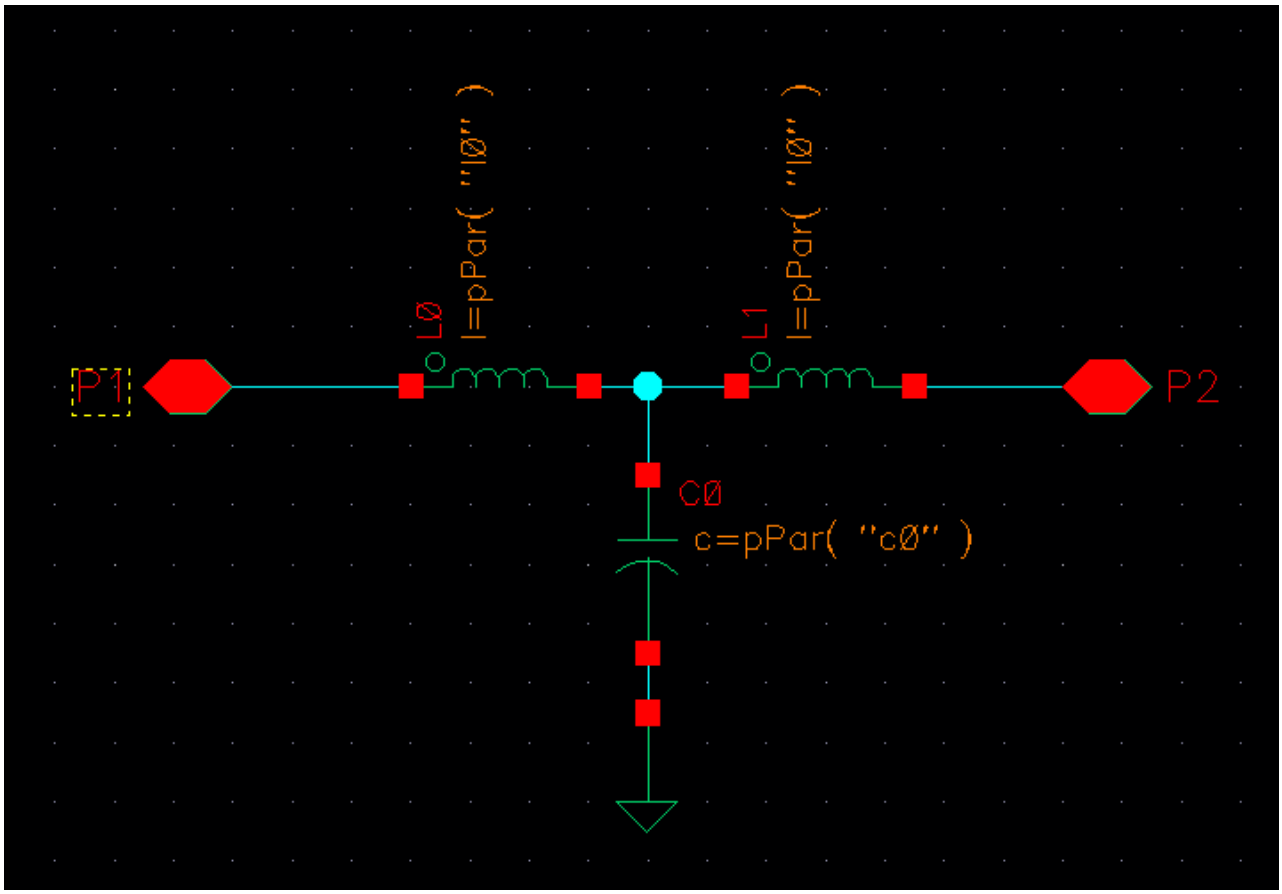
This is a microstrip right-angle bend.



The parameters are:

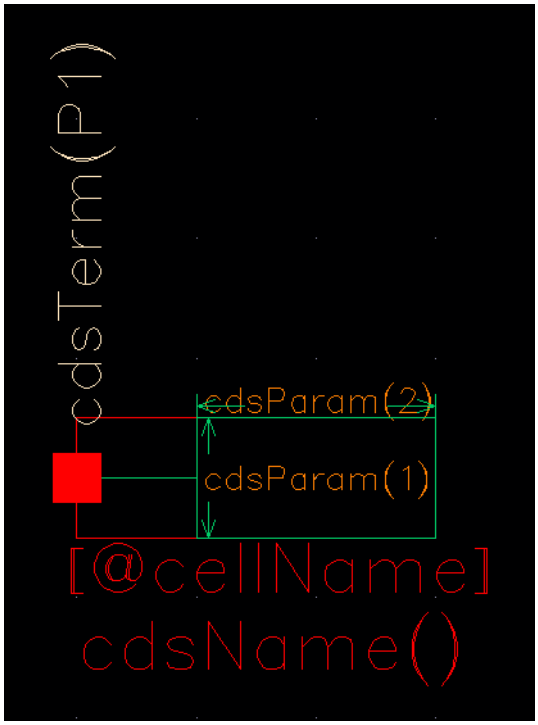
`Dielectric constant of subs`	Relative dielectric constant of the substrate.
`Height of substrate`	Height of the substrate, in meters.
`Width of strip`	Width of the microstrip, in meters. The width is the same on both sides of the bend and, in the symbol, is labeled as both `cdsParam(1)` and `cdsParam(2)`.

The equivalent circuit is this.



microopenend

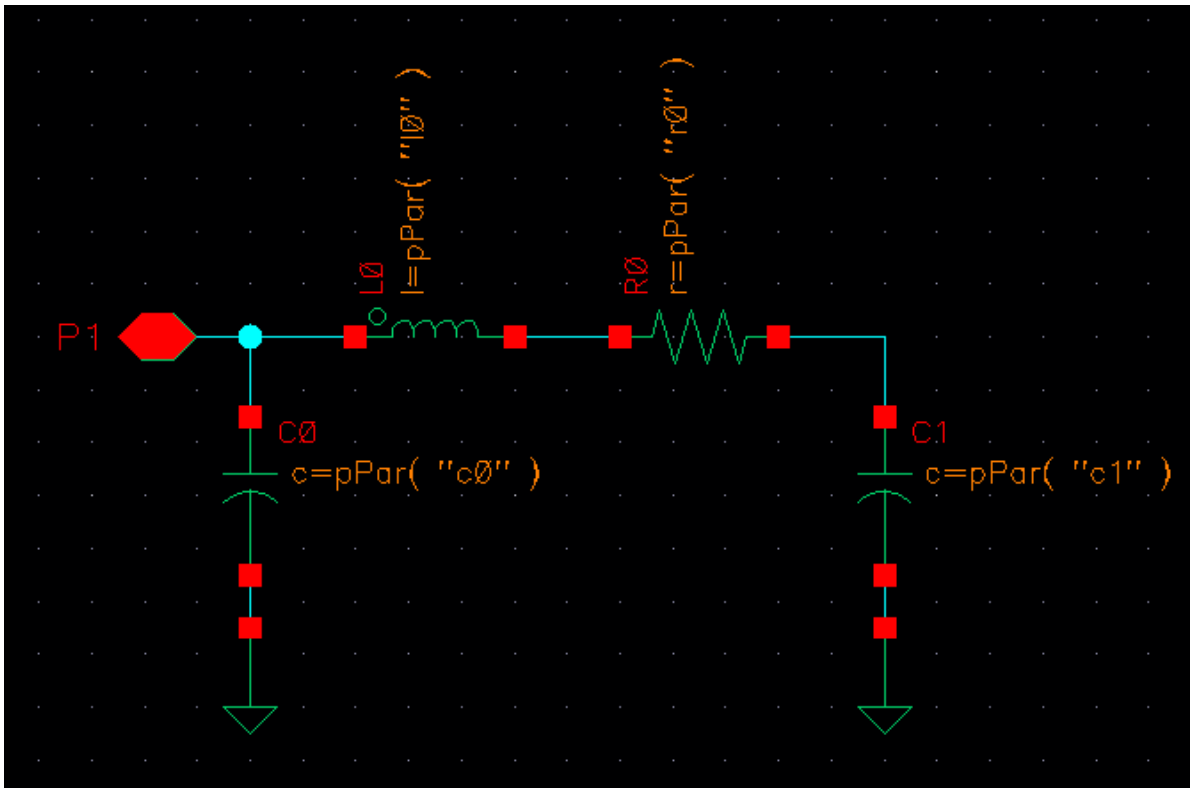
This is a microstrip open end line.



The parameters are:

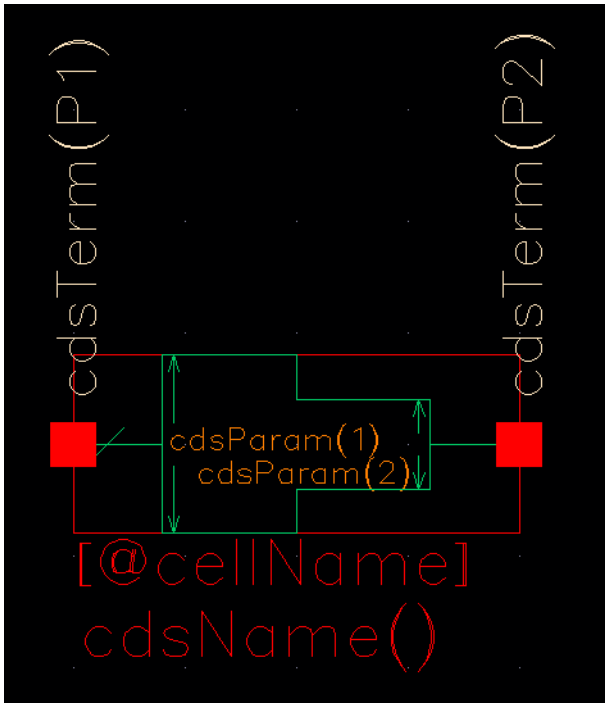
`Electrical conductivity`	Electrical conductivity of the microstrip.
`Thickness of strip`	Thickness of the microstrip, in meters.
`Dielectric content of subs`	Relative dielectric constant of the substrate.
`Height of substrate`	Height of the substrate.
`Width of strip`	Width of the microstrip, in meters. In the symbol, this is labeled `cdsParam(1)`.
`Length of strip`	Length of the microstrip, in meters. In the symbol, this is labeled `cdsParam(2)`.
`Operating frequency`	Operating frequency.

The equivalent circuit is this.



microstep

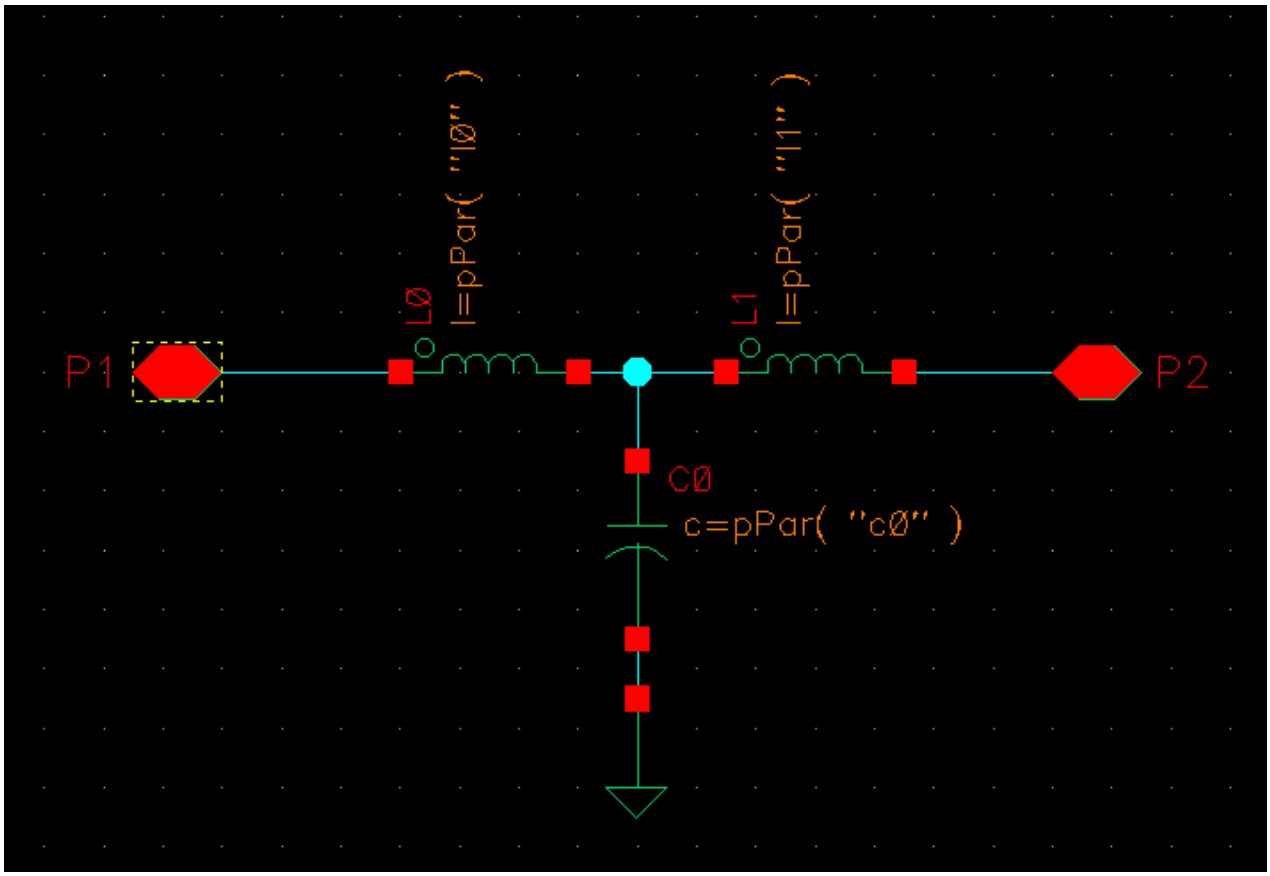
This component steps up or down the width of a microstrip.



The parameters are:

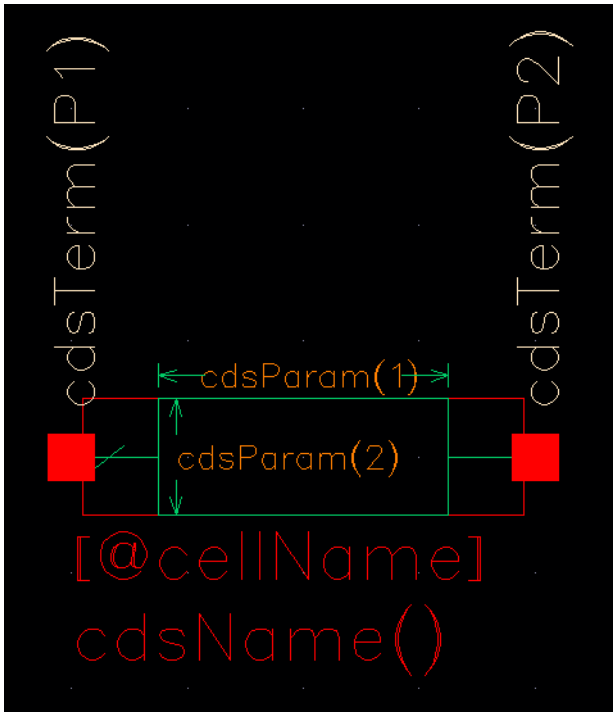
`Dielectric constant of subs`	Relative dielectric constant of the substrate.
`Width of strip (W1)`	Width of one side of the microstrip, in meters. In the symbol, this is labeled `cdsParam(1)`.
`Width of strip (W2)`	Width of the other side of the microstrip, in meters. In the symbol, this is labeled `cdsParam(2)`.
`Height of substrate`	Height of the substrate.
`Operating frequency`	Operating frequency.
`Thickness of strip`	Thickness of the microstrip.

The equivalent circuit is this.



microstrip

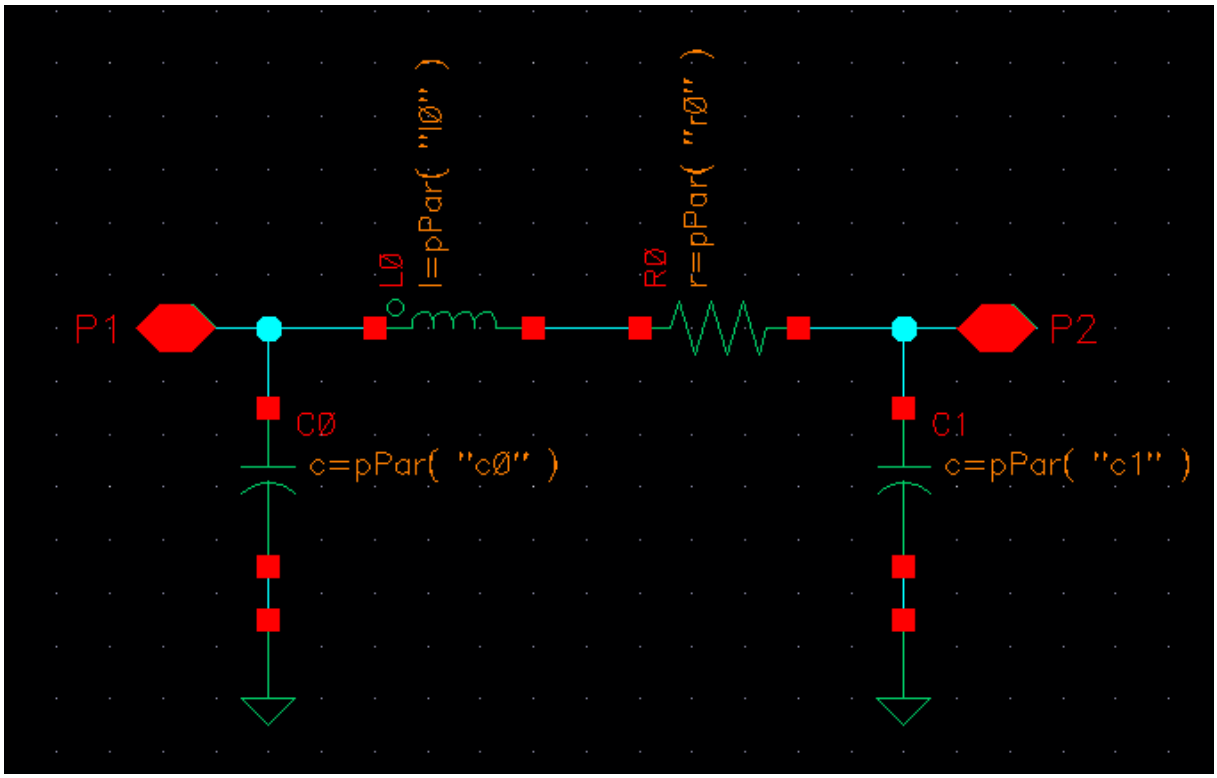
This is the basic microstrip.



The parameters are:

`Electrical conductivity`	Electrical conductivity of the microstrip.
`Thickness of strip`	Thickness of the microstrip, in meters.
`Dielectric constant of subs`	Relative dielectric constant of substrate.
`Height of substrate`	Height of the substrate.
`Width of strip`	Width of the microstrip, in meters. In the symbol, this is labeled `cdsParam(2)`.
`Length of strip`	Length of the microstrip, in meters. In the symbol, this is labeled `cdsParam(1)`.
`Operating frequency`	Operating frequency.

The equivalent circuit is this.

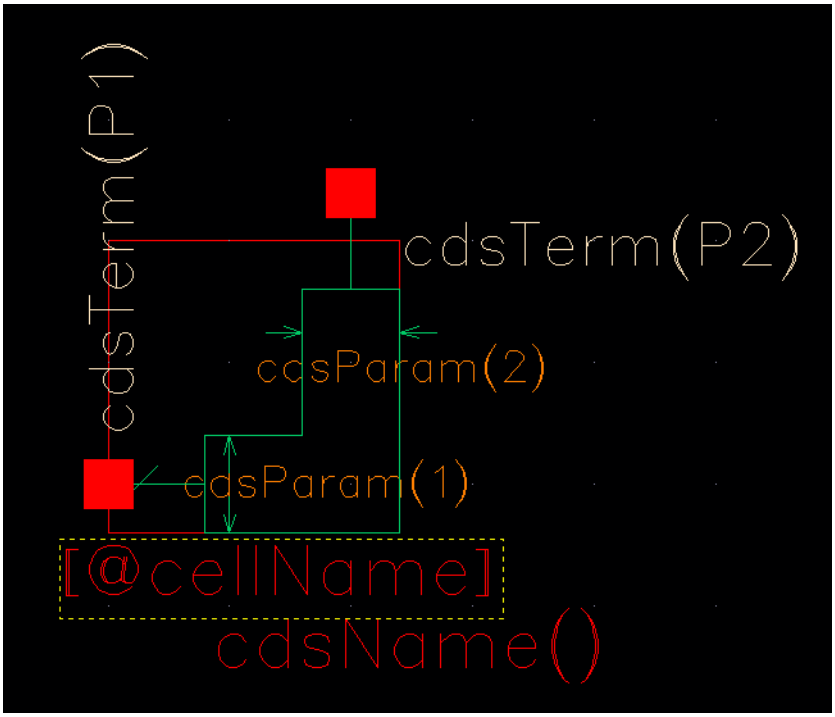


Stripline Components

A stripline consists of a conducting strip that is sandwiched between two ground planes but separated from each of them by a substrate layer.

stripbend90

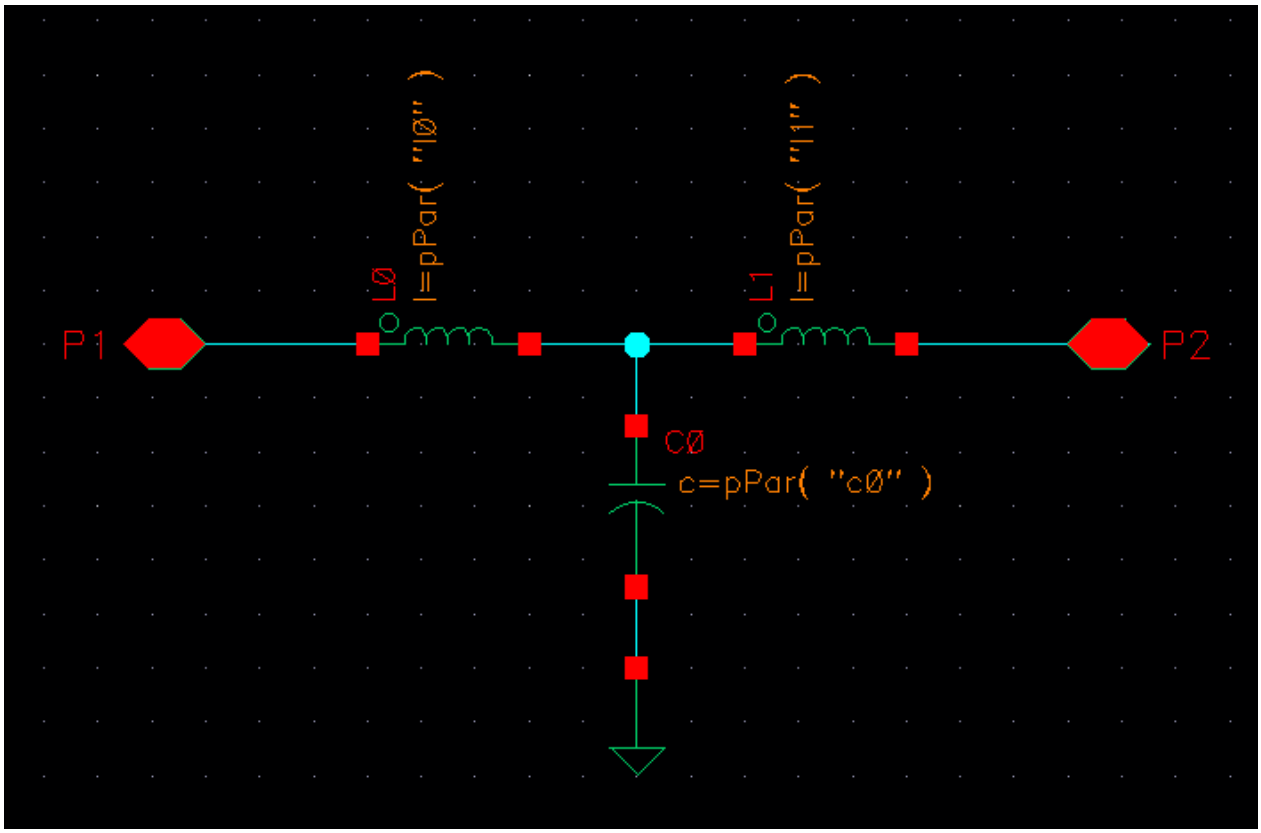
This is a stripline right-angle bend.



The parameters are:

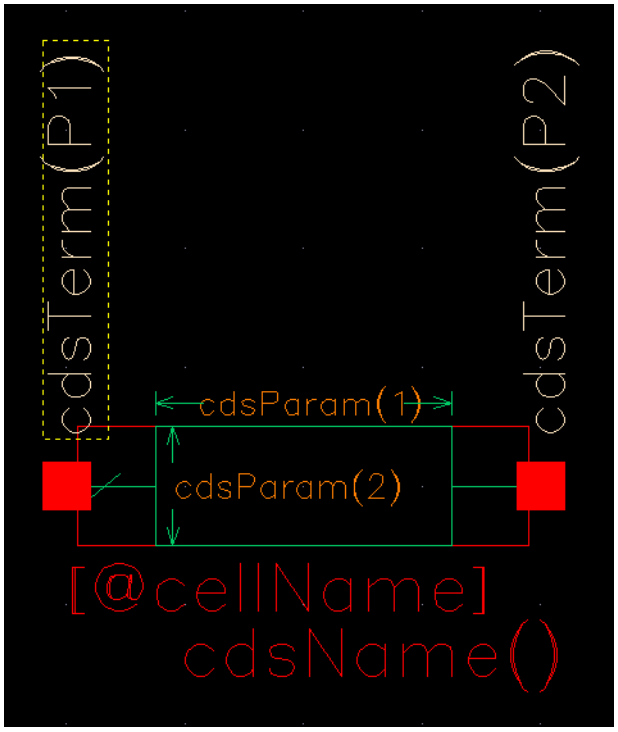
`Dielectric constant of subs`	Relative dielectric constant of the substrate.
`Height of substrate`	Height of the substrate, in meters.
`Thickness of strip`	Thickness of the stripline, in meters.
`Width of strip`	Width of the stripline, in meters. The width is the same on both sides of the bend and, in the symbol, is labeled as both `cdsParam(1)` and `cdsParam(2)`.
`Operating frequency`	Operating frequency.

The equivalent circuit is this.



stripline

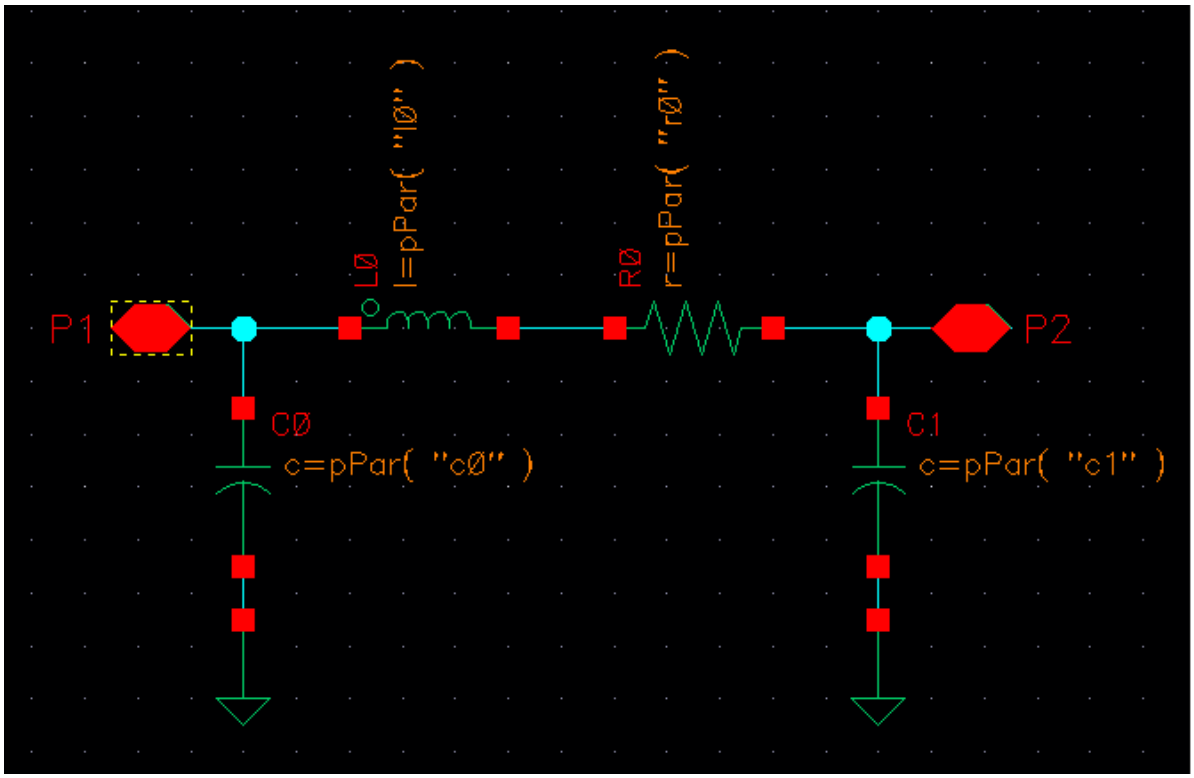
This is the basic stripline.



The parameters are:

`Electrical conductivity`	Electrical conductivity of the stripline.
`Thickness of strip`	Thickness of the stripline, in meters.
`Dielectric constant of subs`	Relative dielectric constant of the substrate.
`Height of substrate`	Height of the substrate, in meters.
`Width of strip`	Width of the stripline, in meters. In the symbol, this is labeled `cdsParam(2)`.
`Length of strip`	Length of the stripline, in meters. In the symbol, this is labeled `cdsParam(1)`.
`Operating frequency`	Operating frequency.

The equivalent circuit is this.

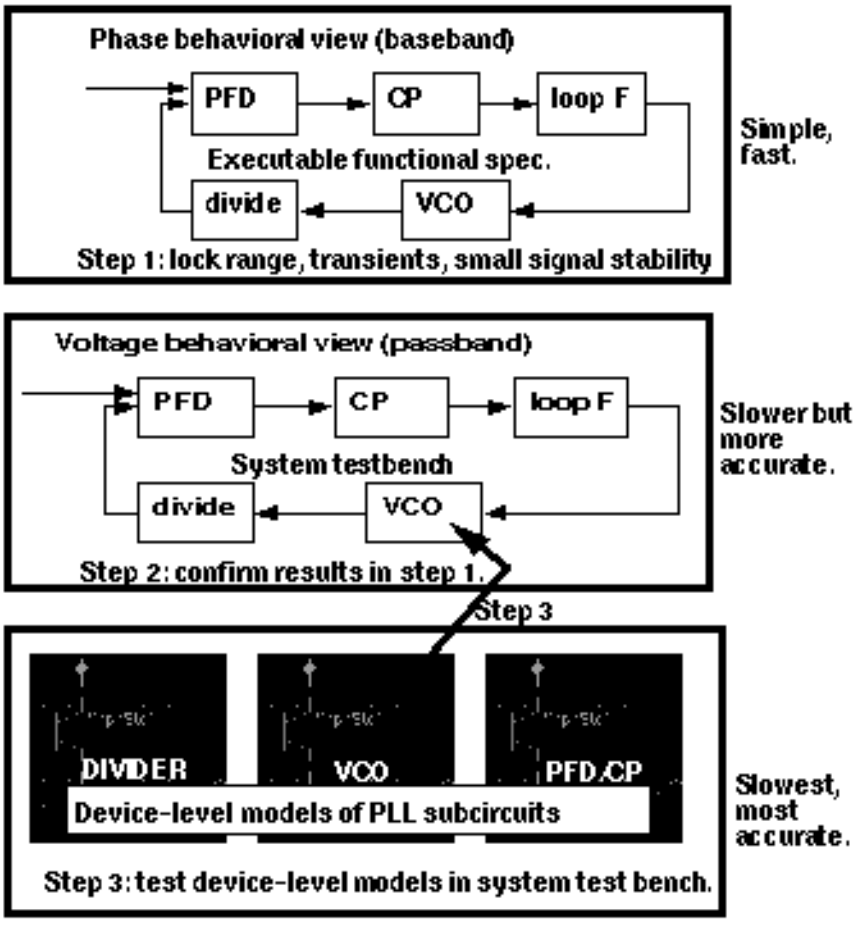


4

Introduction to the PLL library

The models in the phase lock loop (PLL) library support top-down design of PLLs. Figure 4-1 4-1 shows the three steps of the design flow. This appendix describes the first step in detail; all three steps are described briefly.

Figure 4-1 PLL Top-Down Design Flow



1. The first step in Figure 4-1 on page 303 is to develop an executable specification. The executable specification is an arrangement of fast behavioral models that permits fast architectural studies to separate specification and implementation issues. The executable specification contains baseband models [1,2,3,4,5,6,7,8,9]. (Reference [1] uses the terms “baseband” and “bandpass” explicitly.)

   These baseband models suppress clocks and RF/IF carriers. Some literature refers to PLL baseband models as “relative phase” or “phase-domain” models [2]. This appendix uses the latter term. Phase-domain PLL models are exceptionally fast, capture the important non-linear mechanisms, and can be linearized directly for AC analysis.

2. The second step in Figure 4-1 on page 303 is to translate the executable specification into a system testbench. The system testbench, unlike the executable specification, is composed of passband models [1]. This Appendix refers to passband models as voltage-domain models because they simulate voltages you can observe in a laboratory.

3. Comparing voltage- and phase-domain voltage-controlled oscillator (VCO) models highlights the difference between the two models. The output of a voltage-domain VCO model is a clock voltage, a periodic signal. The output of a phase-domain VCO model is a voltage numerically equal to phase. If you unwrap the VCO phase, in steady state, it ramps up indefinitely. Unwrapped phase is not periodic. Voltage-domain models describe non-linear effects related to the shapes of the actual RF waveforms. Such waveform effects include spurs and harmonic locking. Harmonic locking occurs when the PLL locks on to a harmonic of the reference.

4. Phase-domain models do not simulate waveform effects. The system testbench is more accurate than the executable specification, but it is still behavioral. Equipped only with behavioral voltage-domain models, the testbench does not simulate device-level effects associated with specific implementations. Examples of such implementation effects are interstage loading, improper bias, and device parasitics.

5. The last step in Figure 4-1 on page 303 is to gradually replace the behavioral models in the system testbench with device-level models, one or two blocks at a time. Device-level models check for the previously mentioned implementation problems. The entire PLL is simulated at the device-level only as a final verification step because such simulations are very lengthy.

Models in the PLL library

The PLL library includes the following phase-domain models:

- Analog multiplier phase detector
- XOR phase detector with bipolar output

The XOR phase detector is not explicitly discussed here because it is very similar to the analog multiplier phase detector. The only difference is that the duty cycle-phase error transfer curve is triangular instead of sinusoidal.

- Three-state digital phase frequency detector (PFD)
- Charge pump (current source version)
- VCO tuning curve (analytic and tabular versions)
- Frequency divider
- Lock indicator

Introduction to the PLL Library Documentation

The primary system-level specifications captured by this first set of phase-domain models are acquisition time, lock and capture ranges [12], and phase margin. The PFD model also simulates backlash[8]. Backlash is sometimes called “deadband” effect. It is a limit cycle caused by the phase-frequency detector’s inability to linearly reduce its output pulse width to zero as phase error goes to zero.

The remainder of this appendix is divided into two main sections.

The first section introduces phase-domain modeling, describes a feature included to prevent DC convergence problems, and then shows you some examples of using phase-domain models.

The second section explains how to assemble a more complex PLL and discusses an example. The examples are introductory and are not a comprehensive discussion of all applications of phase-domain models.

Phase-Domain Model of a Simple PLL

Description

This PLL example, which is built around the simplest phase detector in the library, introduces the fundamentals of phase-domain modeling. Figure 4-2 on page 306 shows a voltage-domain model of the example and also some selected waveforms. The phase detector in this case is an ideal analog multiplier.

Figure 4-2 Voltage Domain Model

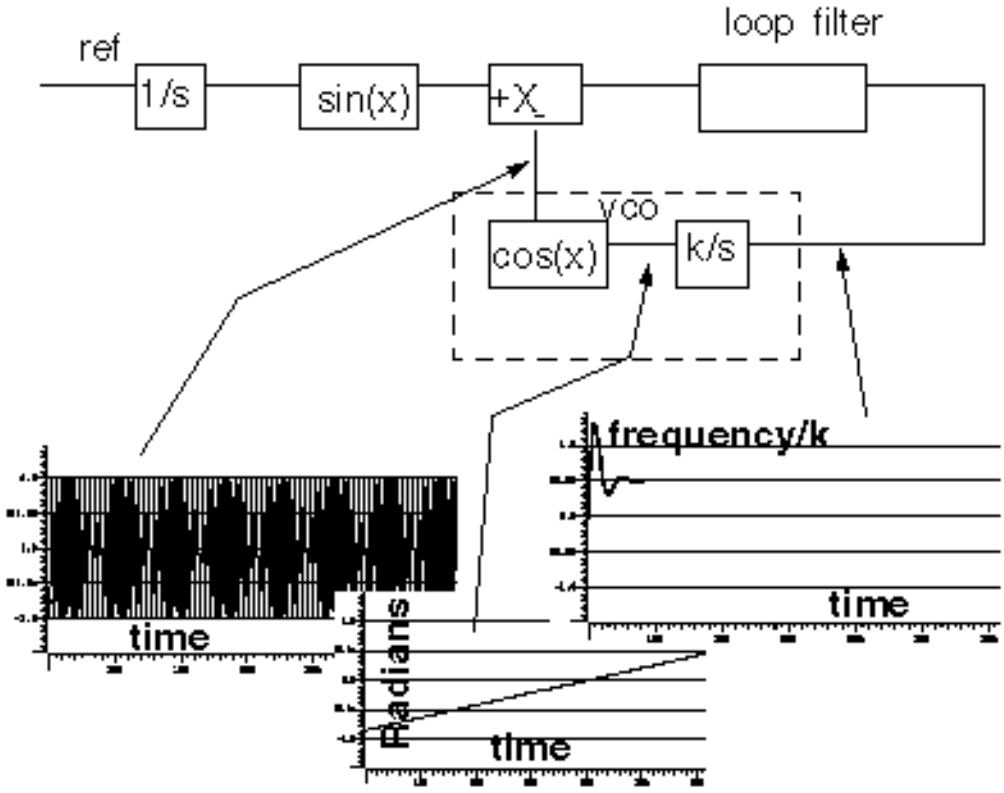


Figure 4-3 on page 307 shows the equivalent phase-domain model. The phase-domain model is based on the following trigonometric identity:

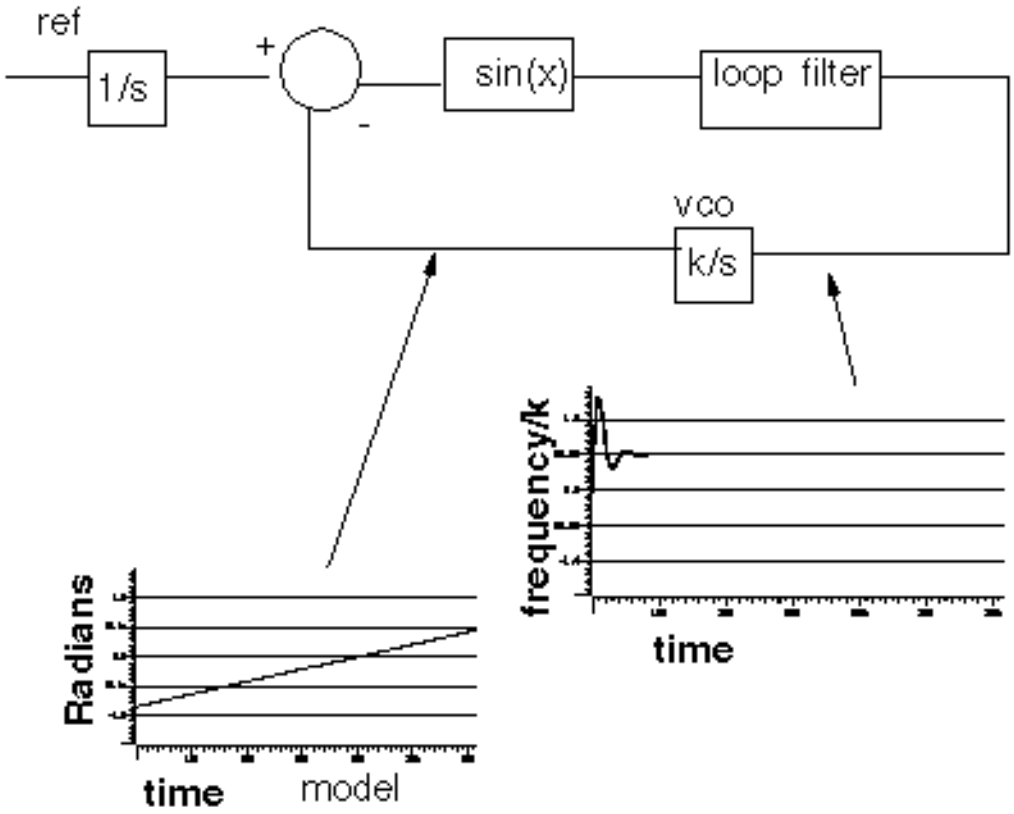
$\sin(\theta_1)*\cos(\theta_2) = (1/2)*[\sin(\theta_1+\theta_2)+\sin(\theta_1-\theta_2)]$

which, after filtering is approximately

$(1/2)*\sin(\theta_1-\theta_2)$

Note: $q_1+q_2=(w_1+w_2)*t$ and w_1+w_2 usually lie far beyond the filter's corner frequency.

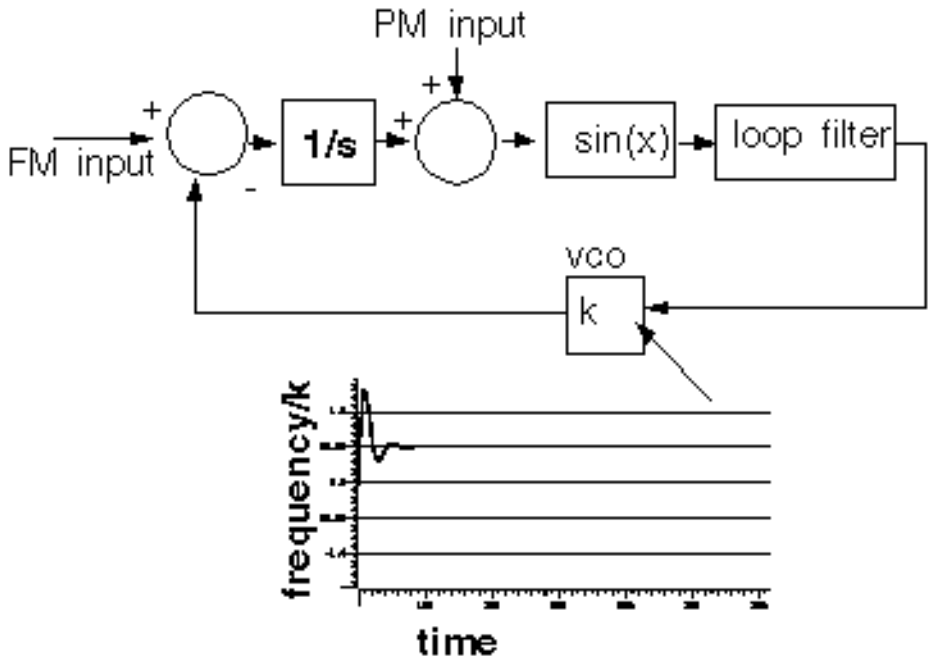
Figure 4-3 Phase Domain Model



The phase and frequency waveforms from the phase-domain model match their voltage-domain counterparts, but the simulation runs faster because the oscillatory waveform is not explicitly simulated.

Combining the integrators, as shown in Figure 4-4 on page 308, eliminates the integrator outside the feedback loop which might cause a problem if you forget to specify an initial condition. Combining the integrators is also necessary if you build phase-domain models of phase-frequency detectors because, in this case, the non-linearity has memory (hysteresis).

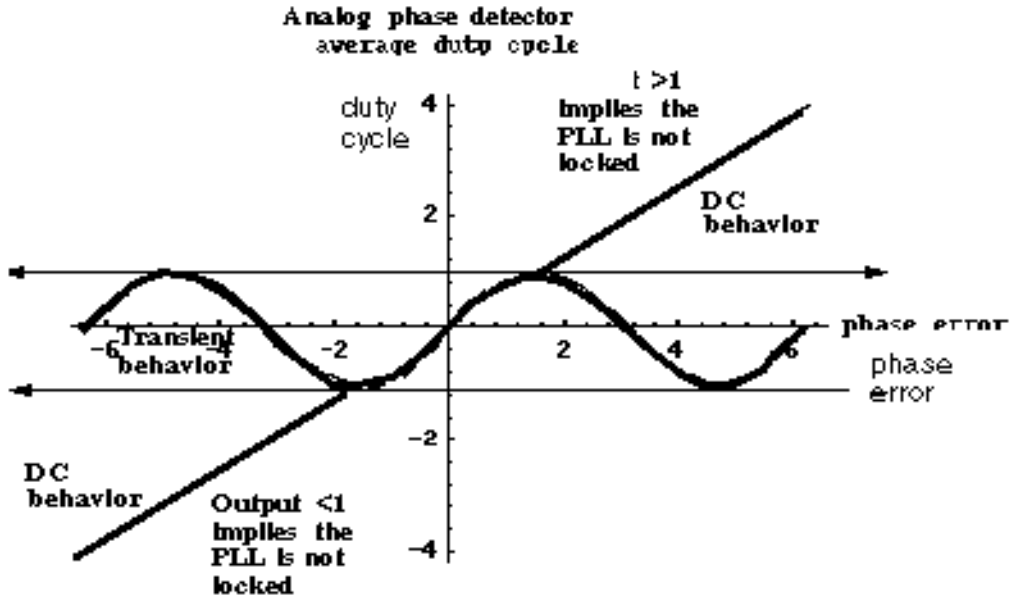
Figure 4-4 A More Practical Phase Domain Model



The models in both Figure 4-3 and Figure 4-4 can fail to achieve DC convergence because the phase detector model either has no DC operating point or because it has an infinite number of operating points.

The sinusoidal function in Figure 4-5 on page 309 is the phase detector transfer curve. The phase detector is the only non-linear element in this PLL model. For reasons associated with phase-frequency detectors, the phase detector output is called the *duty cycle*.

Figure 4-5 Phase Detector Transfer Curves



If the required duty cycle lies outside [-1,1], the loop is not locked in steady state. If the required duty cycle lies within [-1,1], there are an infinite number of possible phase errors. In either case, a Virtuoso® Spectre® circuit simulator RF analysis (Spectre RF) simulation might not converge. The ability of Verilog-A© to perform different tasks for different analyses provides an elegant solution to the DC convergence problems and a quick way to map out lock range. Lock range is the range of input frequencies for which the PLL can maintain lock. (Some literature refers to lock range as hold-in range [8].)

The phase detector model uses the monotonic transfer curve for DC analysis and the true periodic transfer curve for transient analysis. The two transfer curves coincide when the phase error lies in the interval [-π/2, π/2]. If the required duty cycle lies within [-1,1], the monotonic transfer curve forces the steady-state phase error to the interval [-π/2, π/2], where the two curves coincide. The equilibrium point is *open-loop-stable*, meaning that at DC the loop gain is a positive real number. This is true because the slope of the transfer curve is positive over [-π/2, π/2]. The Nyquist stability criterion is therefore easier to apply. The DC analysis is general enough because only the phase error modulo 2π is of interest, and you usually care only about the open-loop-stable operating points. When the loop is not locked, the DC analysis computes a duty cycle with a magnitude greater than one. A duty cycle greater than one is clearly incorrect, but it is much easier to interpret than a convergence error. DC duty cycle is a lock indicator which can be used in a parametric DC analysis to sweep out lock range.

Example 1: Dynamic Test for Capture Range and Lock Range

The circuit used to dynamically test for capture range and lock is range is *example_analog_PD* in the *pllLib* library. *Capture range* is the range of input frequencies that the PLL can acquire from an unlocked state. Because acquisition of frequencies near the edge of the capture range involves a pull-in mechanism [1-12], measuring the capture range requires a transient analysis. You can measure capture and lock ranges by slowly sweeping the input frequency and observing the frequency at which the duty cycle begins and ends a long ramp [12]. You must skip the DC analysis to observe the capture limits. Figure 4-6 on page 310 plots the VCO control voltage against the input frequency voltage. The input frequency first ramps up and then down. A buffered auxiliary circuit responds to the duty cycle and adds 2.5 volts when the input frequency changes direction. This technique makes the plot easier to read because the forward and reverse sweeps occupy different parts of the vertical scale. In this example, lock range is from 1.36Khz to 3.4Khz, and the capture range is from 1.8Khz to 3Khz.

Figure 4-6 Lock Range and Capture Range

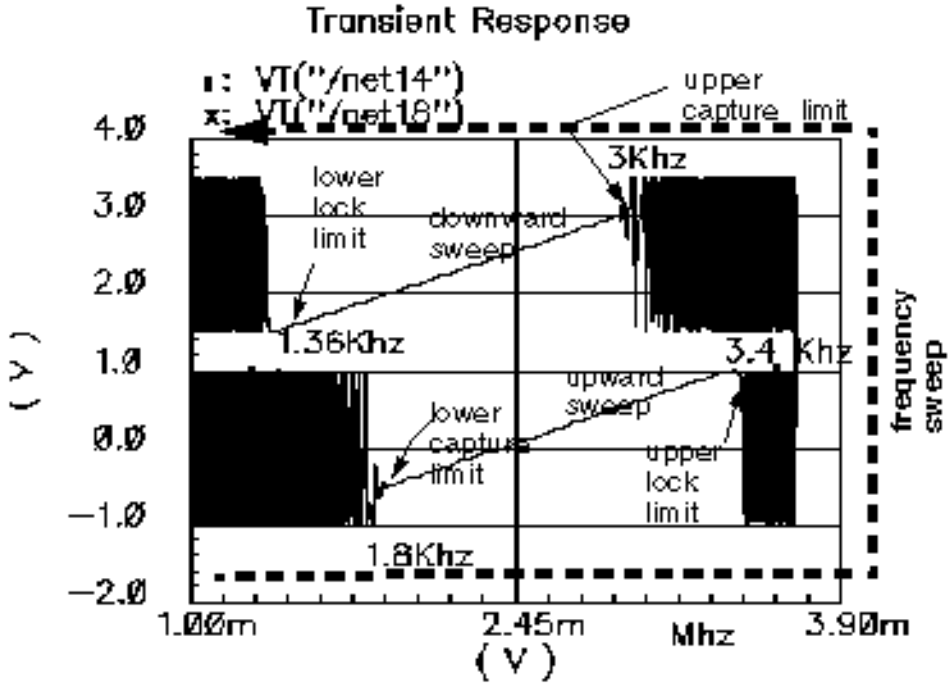
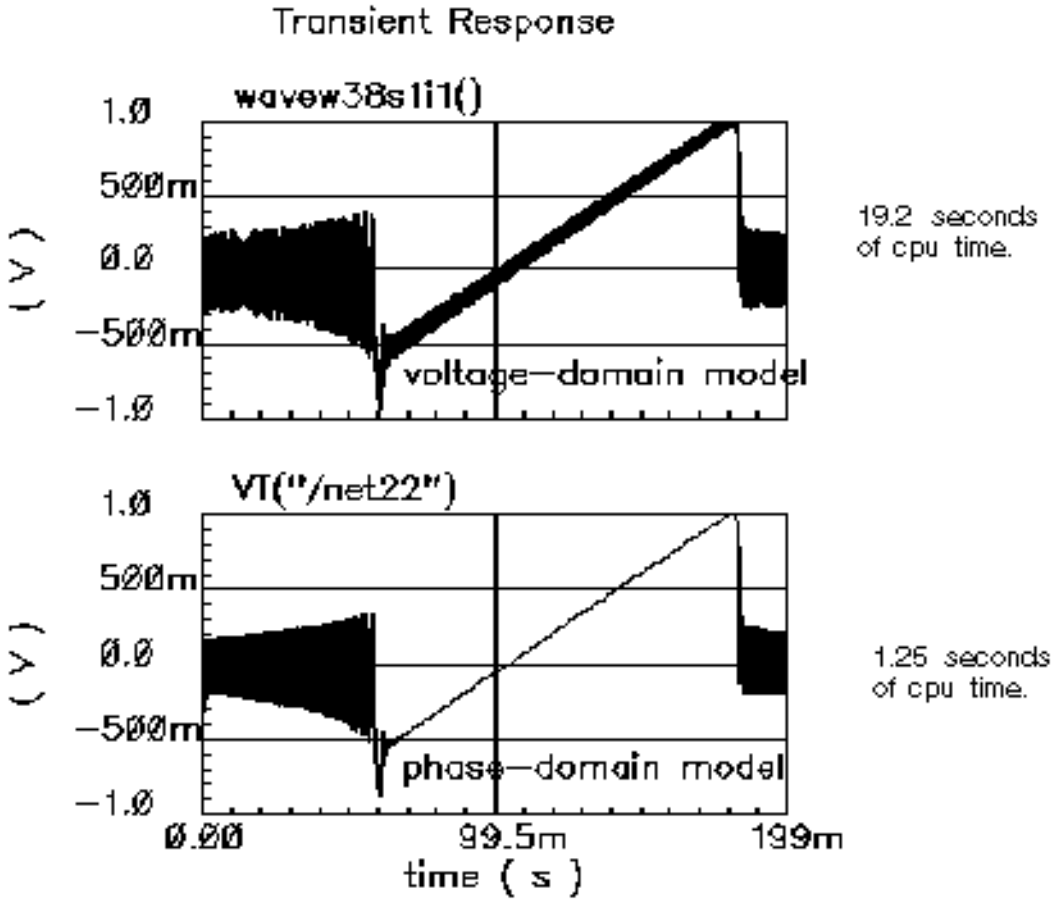


Figure 4-7 on page 311 compares VCO control voltages in the forward sweep when computed with voltage- or phase-domain models. The models produce similar results. In this example (2.5Khz center frequency), the phase-domain model is only about 20 times faster than the voltage-domain model.

Figure 4-7 VCO Control Voltages Computed with Voltage- and Phase-Domain Models

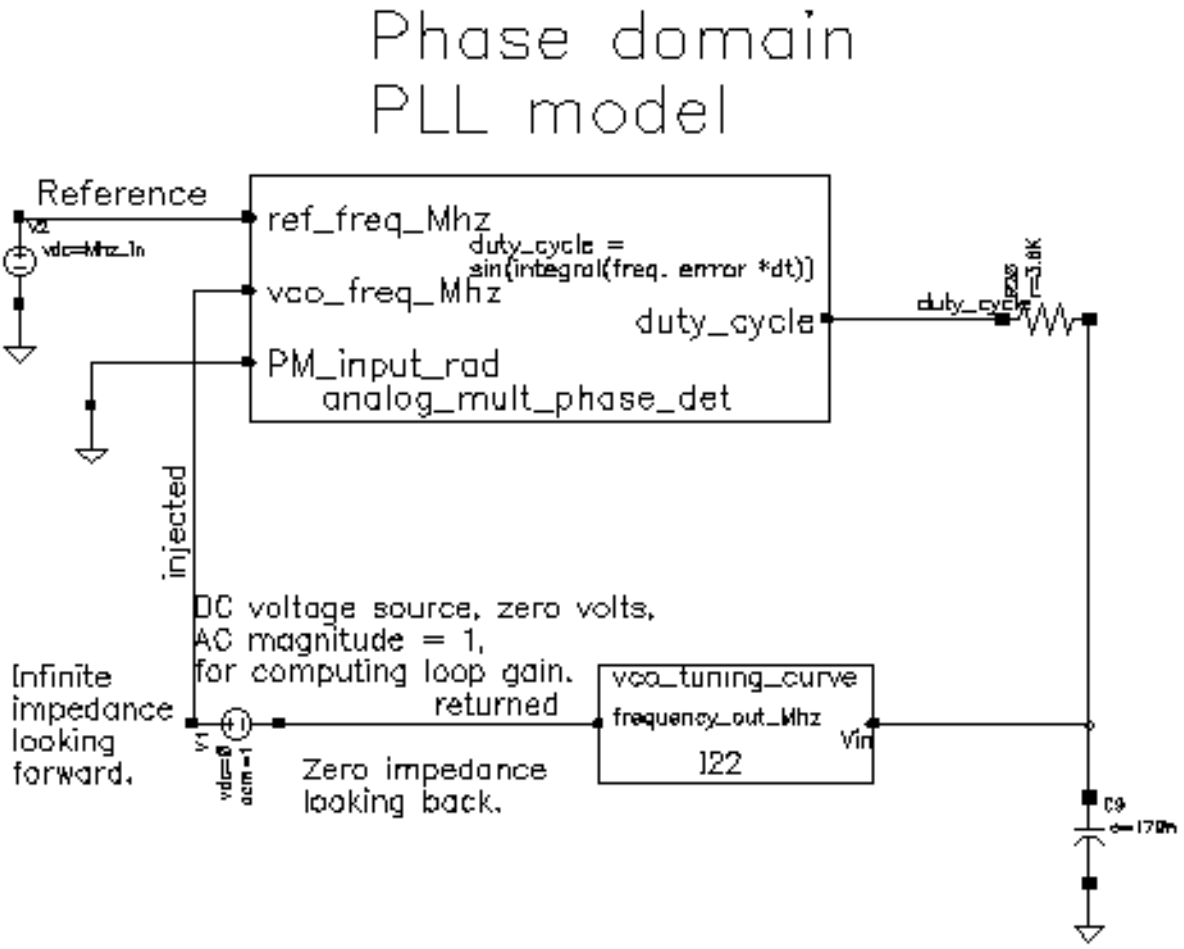


Example 2: Loop Gain Measurement

Spectre RF cannot perform a useful AC analysis on a voltage-domain model because by design, a voltage-domain PLL model has no DC operating point. However, because Spectre RF linearizes phase-domain models about phase error, and phase error is a meaningful DC quantity, subsequent AC analyses are valid.

This example describes how to compute loop gain with a phase-domain model. Figure 4-8 on page 312 shows an analog design environment version of the model shown in Figure 4-4 on page 308. The circuit used to measure loop gain is *example_loop_gain* in the *pllLib* library.

Figure 4-8 Set Up for Loop Gain Measurement



The phase-domain model in Figure 4-8, *example_loop_gain*, includes a voltage source inserted after the VCO. The DC voltage is zero volts, and the AC magnitude is 1 volt. The new voltage source inserts a test signal without changing the DC operating point. You must insert this source at a point where the impedance looking back is much smaller than the impedance looking forward. The accuracy of the resulting loop gain computation depends on how well this condition is met.

Use the following procedure to compute the loop gain.

Open the example_loop_gain Schematic

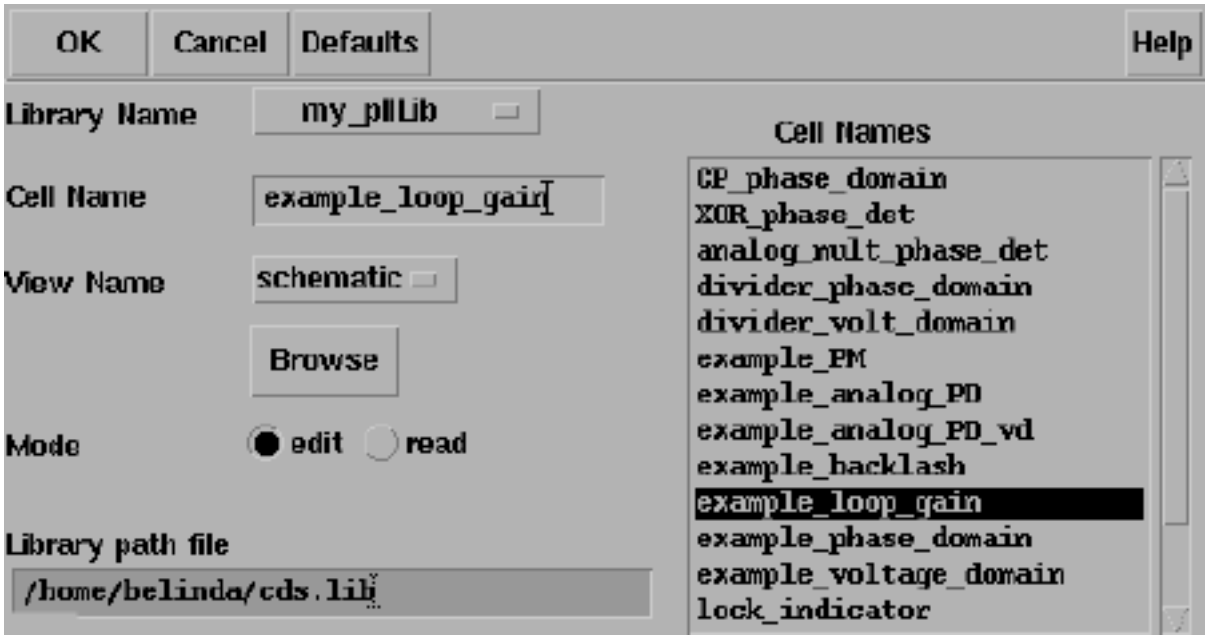
1. In the CIW, choose *File – Open*.

   The Open File form appears.

2. In the Open File form, choose *my_pllLib* in the *Library Name* cyclic field. Choose the editable copy of the pllLib library you created. (You can create an editable copy of the *pllLib* in the same way as is described for the *rfExamples* library in Chapter 3, "Setting Up for the Examples.")

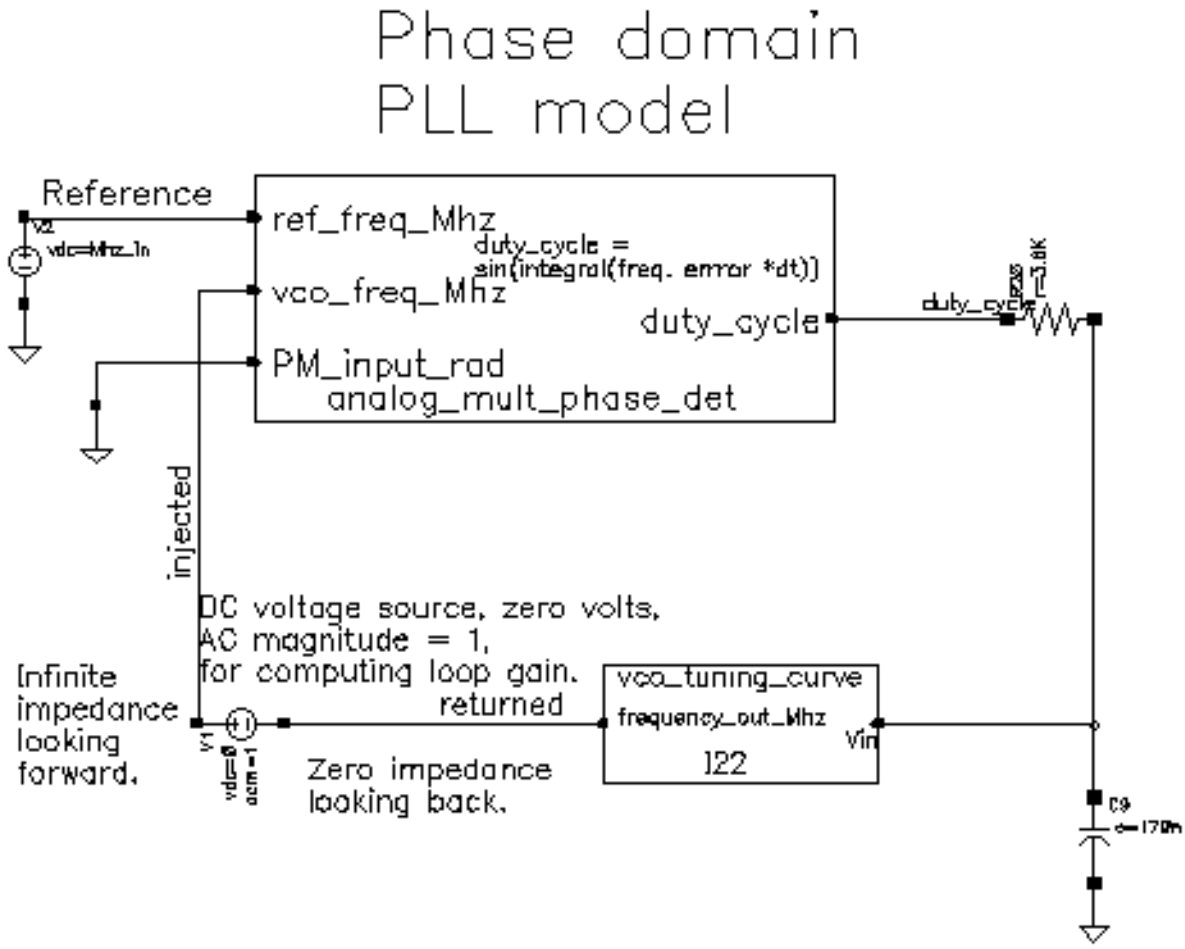
3. Choose *example_loop_gain* in the *Cell Names* list box.

   The completed Open File form appears like the one below.



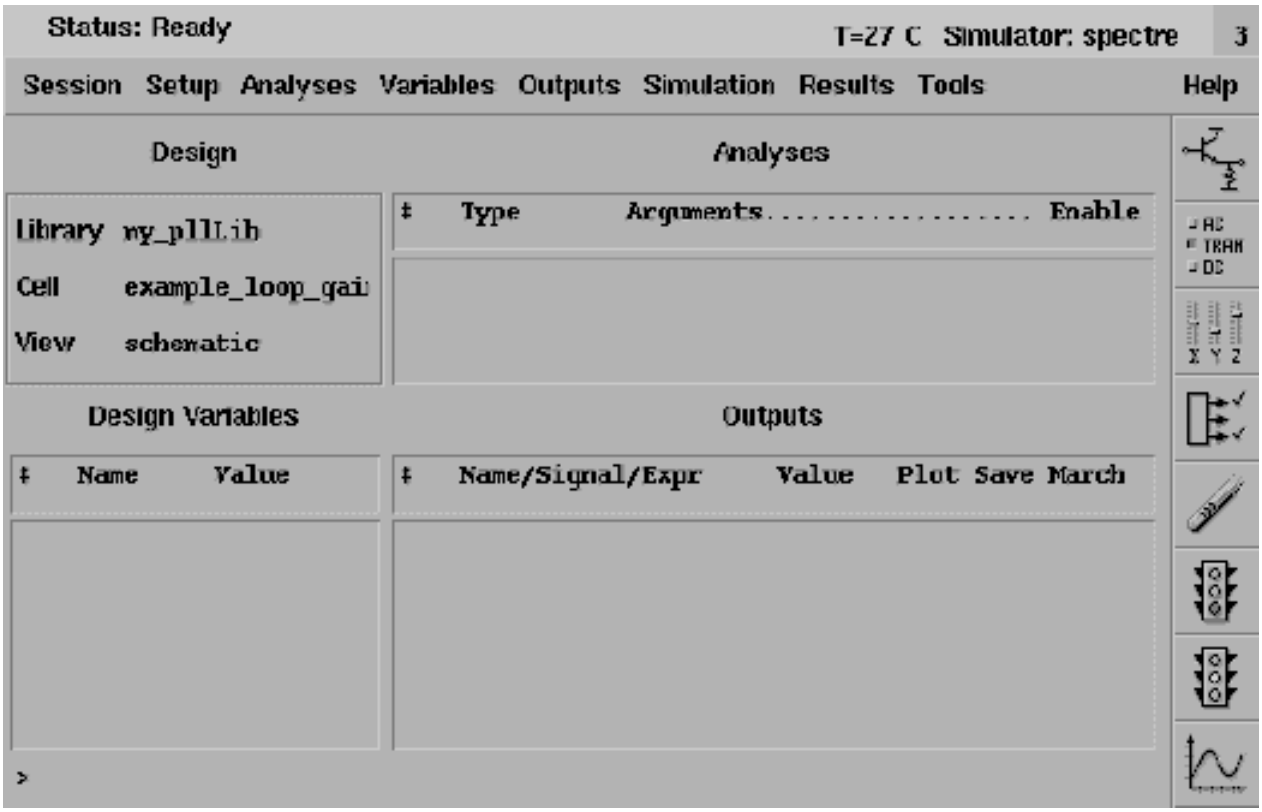
4. Click *OK*.

The Schematic window for the *example_loop_gain* circuit appears.



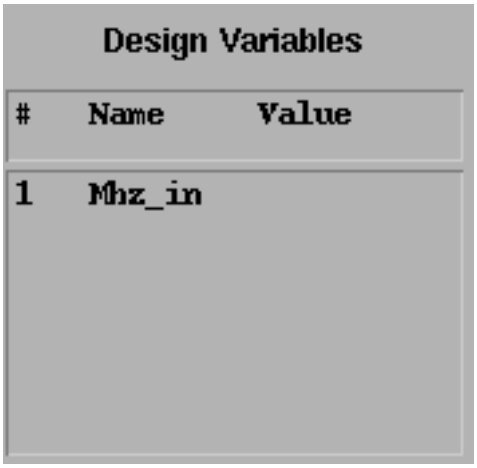
5. In the Schematic window, choose *Tools– Analog Environment*.

The ADE window opens. This window is also called the Cadence® Analog Circuit Design Environment.

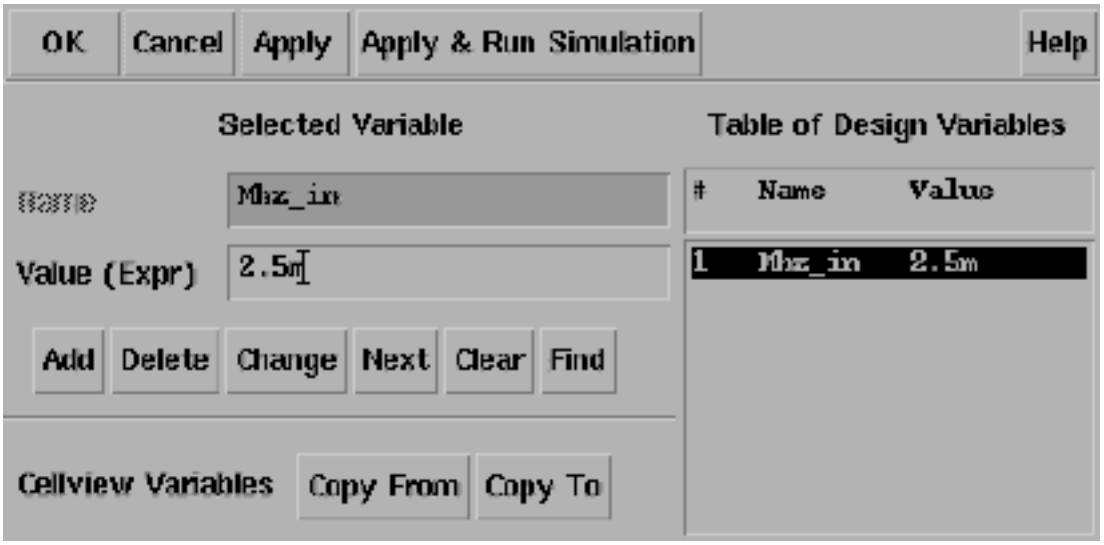


Setting Up the Design Variables

1. In the ADE window, choose *Variables – Copy From Cellview* to copy variables from the schematic to the ADE window. *Mhz_in* displays in the *Design Variables* area of the ADE window.

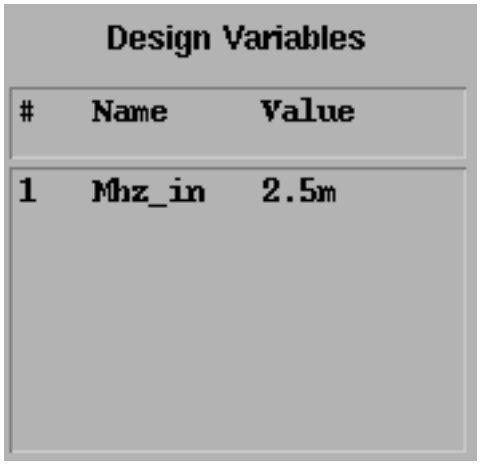


2. In the ADE window, choose *Variables – Edit*, to provide a value for the *Mhz_in* variable.
3. The Editing Design Variables form appears.



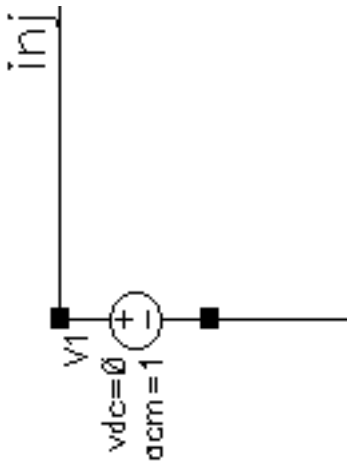
4. In the *Value (Expr)* field, type `2.5m` for the value of `Mhz_in` and click *Change*.
5. In the Editing Design Variables form, click *OK*.

6. The new value for *Mhz_in* displays in the *Design Variables* area of the ADE window.



Setting Up the AC and DC Simulations

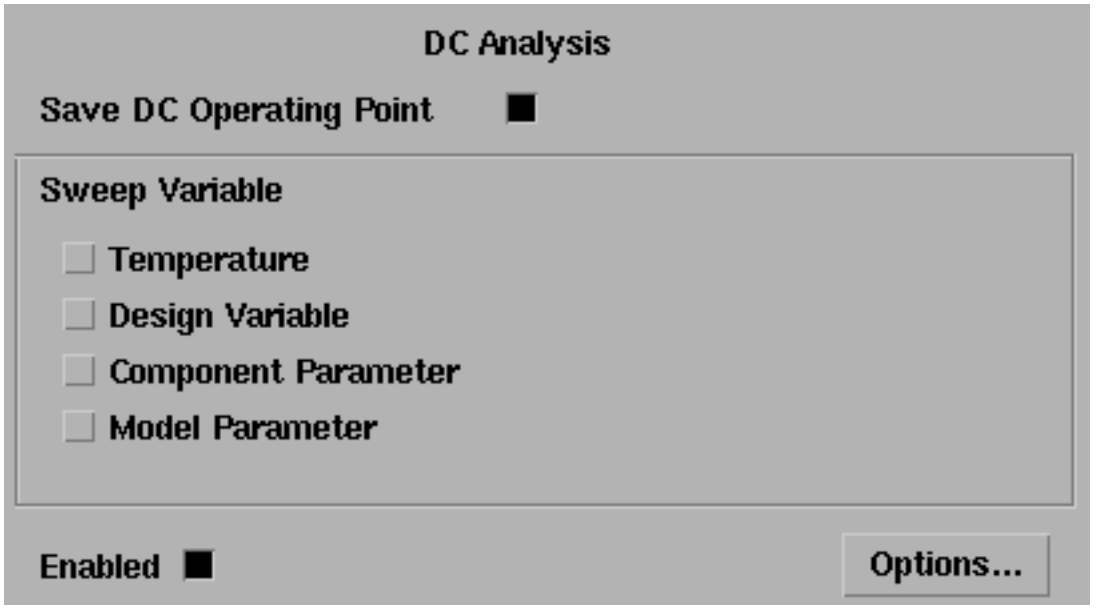
Set up both AC and DC analyses. The zero-voltage *vdc* source must be the only source with a non-zero AC magnitude.



When you set up the DC analysis, save the DC data so you can annotate the schematic with DC node voltages.

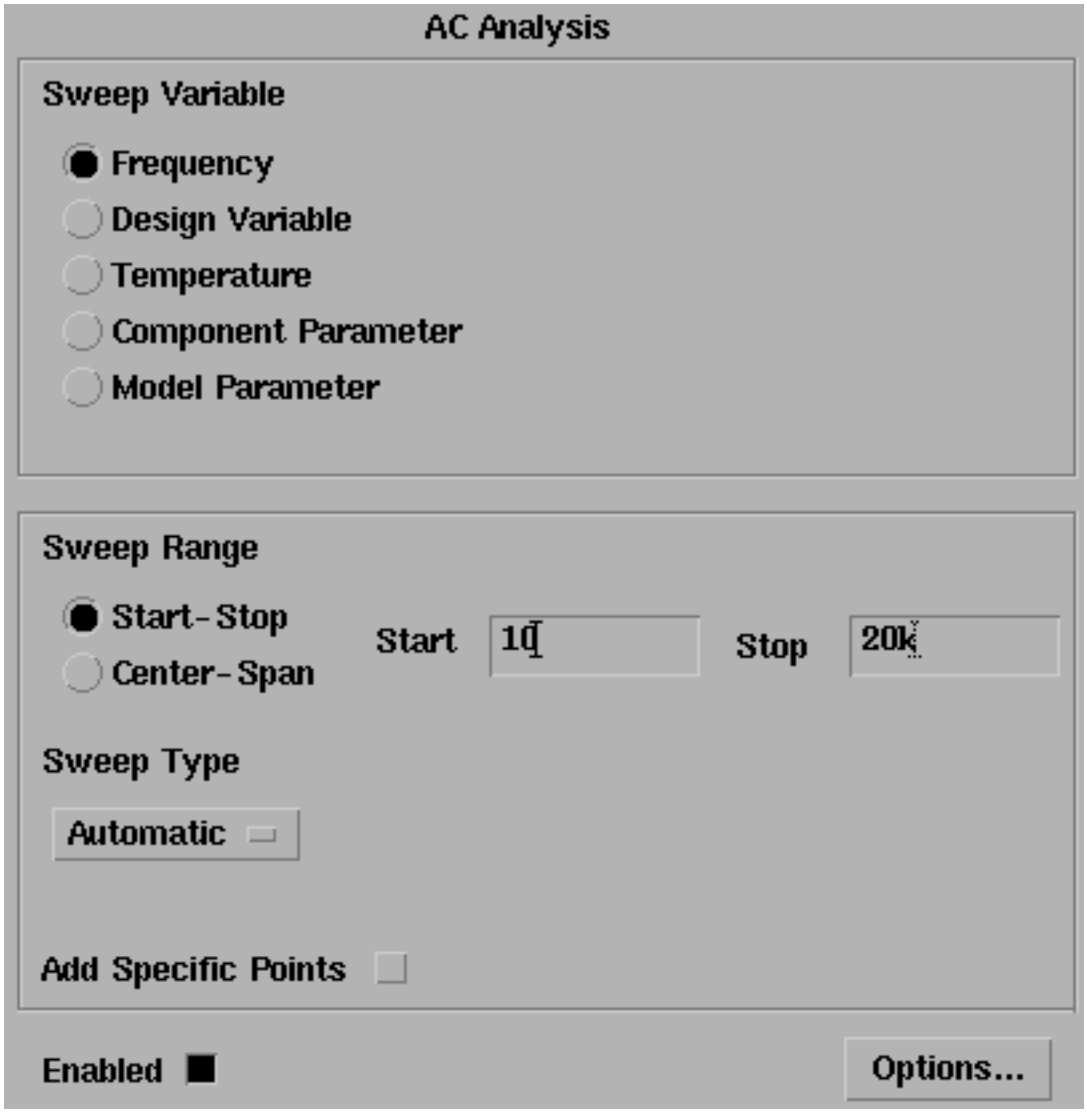
1. In the ADE window, choose *Analysis - Disable* to disable any analyses you ran previously. (Check the ADE window to verify whether or not any analysis is enabled.)
2. In the ADE window, choose *Analysis - Choose* to display the Choosing Analyses form.

3. Click *dc* to set up the DC analysis.
4. In the DC Analysis area
   a. Highlight *Save DC Operating Point*.
   b. Highlight *Enabled*.



5. Click *ac* to set up the AC analysis.
6. In the AC Analysis area
   c. Highlight *Frequency* for *Sweep Variable.*
   d. Highlight *Start - Stop* for *Sweep Range*. Type `10` in the *Start* field and `20k` in the *Stop* field.
   e. Select *Automatic* in the *Sweep Type* cyclic field.

f. Highlight *Enabled*.



7. Click *OK* in the Choosing Analyses form.

8. Both analyses are displayed in the ADE window.

Analyses

#	Type	Arguments.....				Enable
1	dc	t				yes
2	ac	10	20K	Auto..	Star..	yes

Run the Analyses

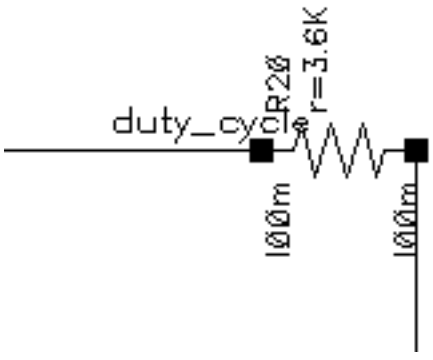
1. To run the analyses, choose *Simulation – Netlist and Run* in the ADE window.

The output log file appears and displays information about the simulation as it runs.

Look in the CIW for a message that says the simulation completed successfully.

Displaying the DC Voltages on the Schematic Nodes

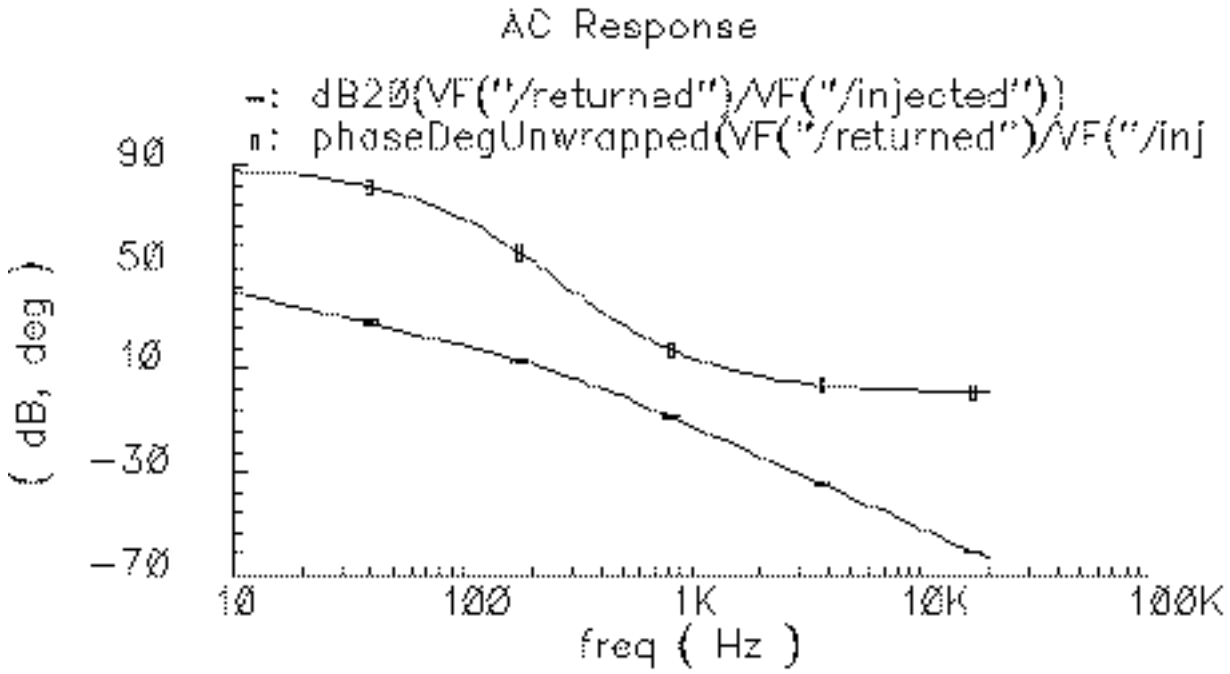
➤ In the ADE window, choose *Results – Annotate – DC Node Voltages* to display the node voltages on the schematic. The DC operating point for the net called *duty_cycle* must remain between `-1` volt and `1` volt.



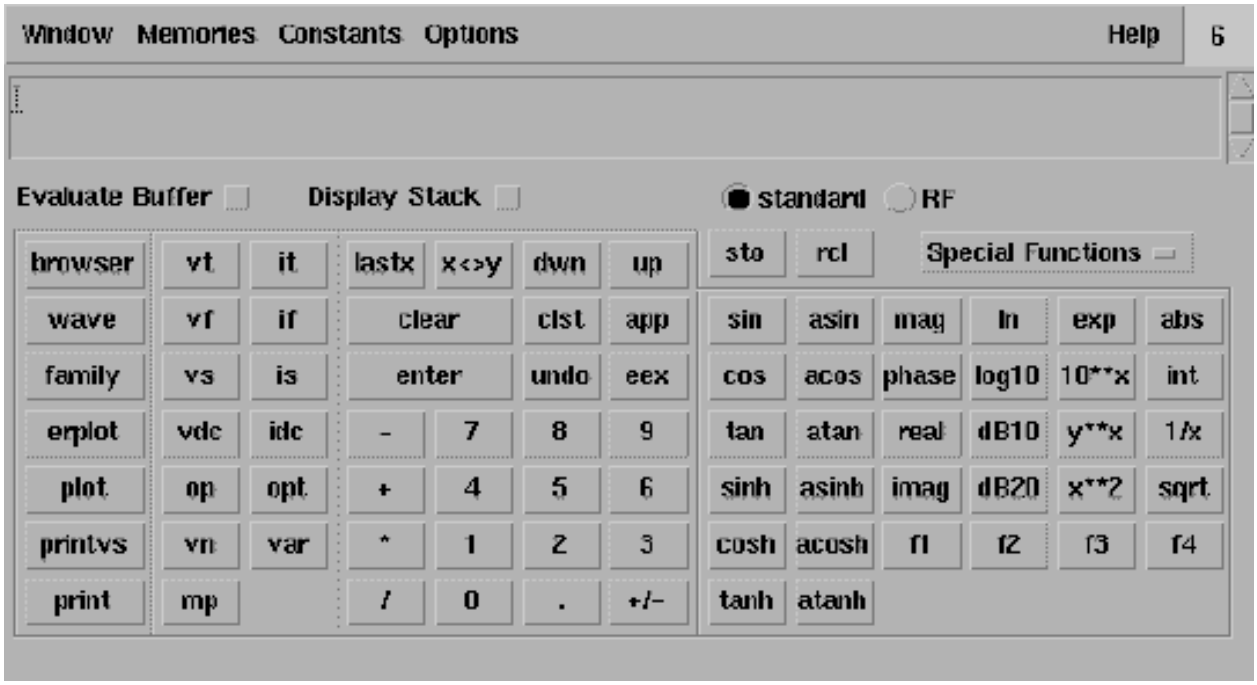
If the operating point falls outside the interval [-1, 1] volt, the loop is not locked and the AC analysis is invalid.

AC Response as Gain and Phase

1. In the ADE window, choose *Results – Direct Plot – AC Gain & Phase* and follow the prompts at the bottom of the Schematic window.
2. *Select first point*—Select the net labeled *returned* in the schematic.
3. *Select second point*—Select the net labeled *injected* in the schematic.
4. The waveform window displays two curves.
5. The top curve plots phase.
6. The bottom curve plots gain.



7. In the waveform window, choose *Tools – Calculator* to open the calculator.



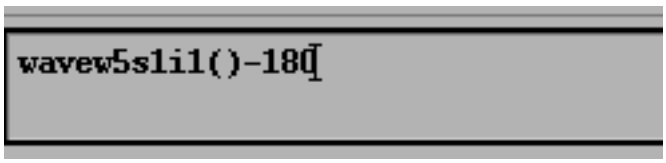
8. In the calculator, click the *wave* button (on the left).

9. Then, in the waveform window, select the phase curve (on the top).

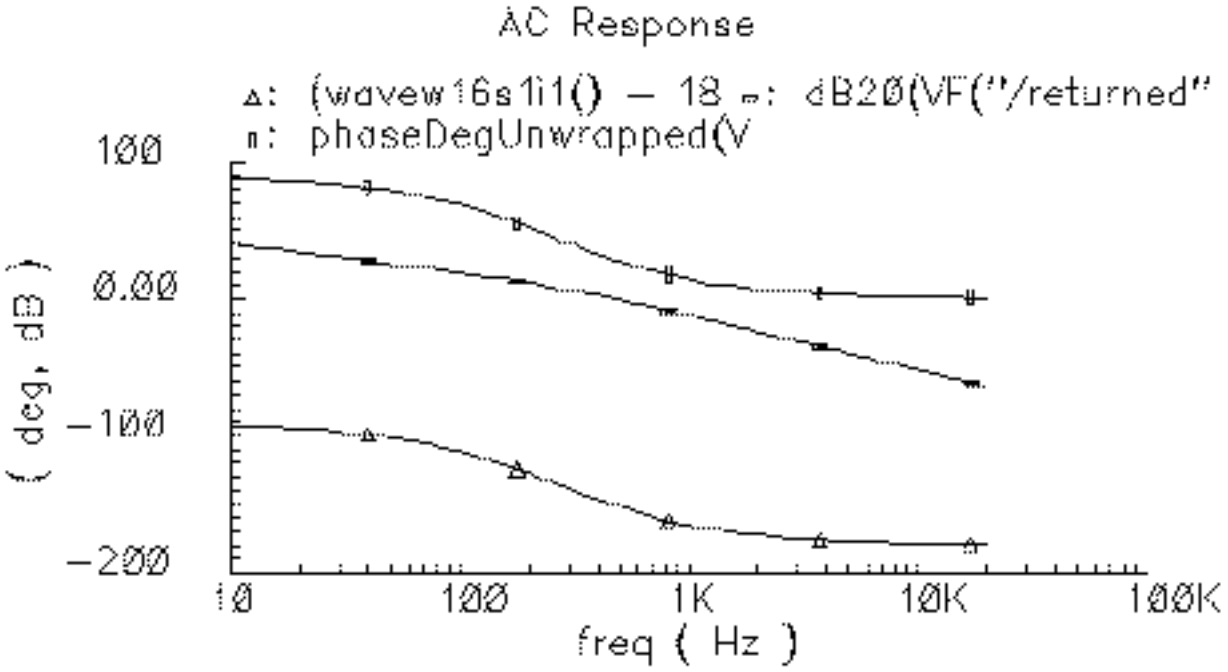
10. In the calculator, to perform the calculations algebraically, choose *Options – Set Algebraic*.

11. Subtract 180 from the phase waveform—In the calculator, click the subtraction symbol (-) followed by the numbers *180*.

12. The calculator buffer should look similar to the following



13. Click the *plot* button to plot the calculated waveform.



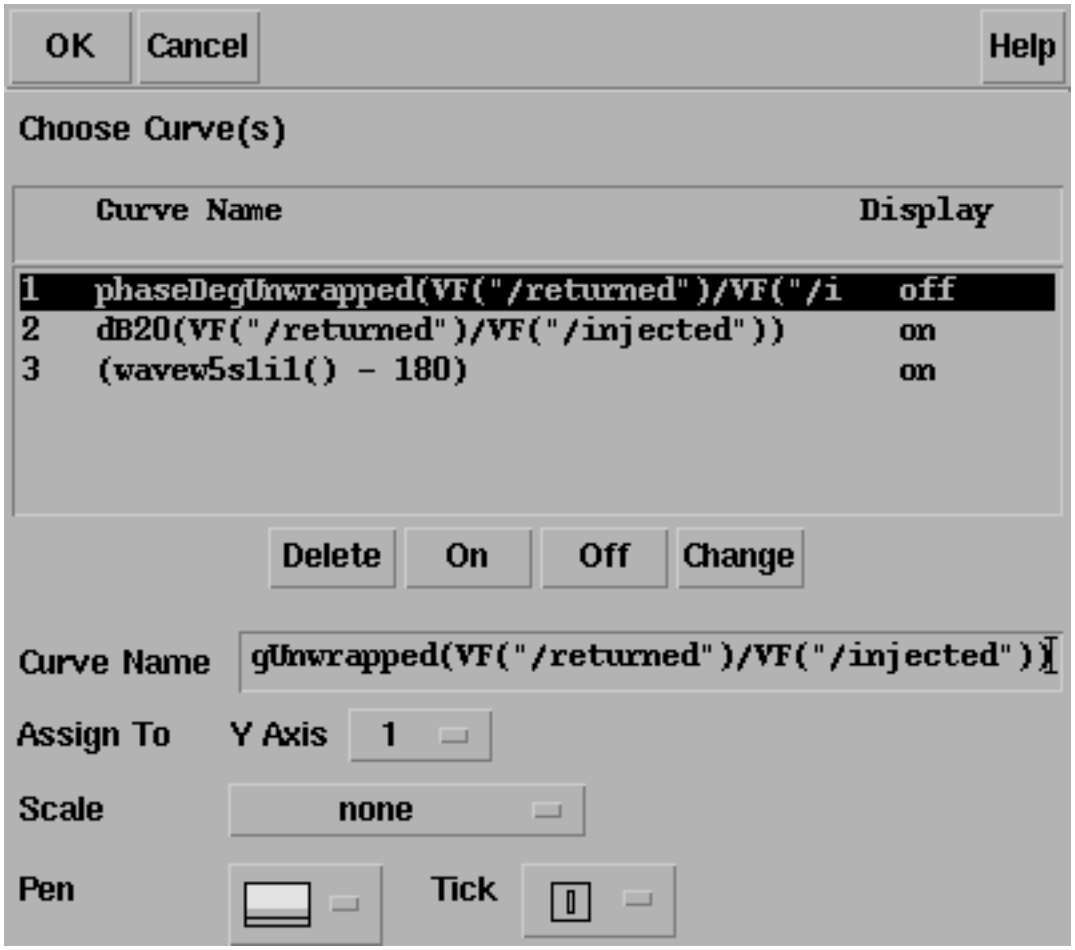
14. To remove the original phase curve from the waveform window, in the waveform window

a. Choose *Curves – Edit* to display the Curves form.

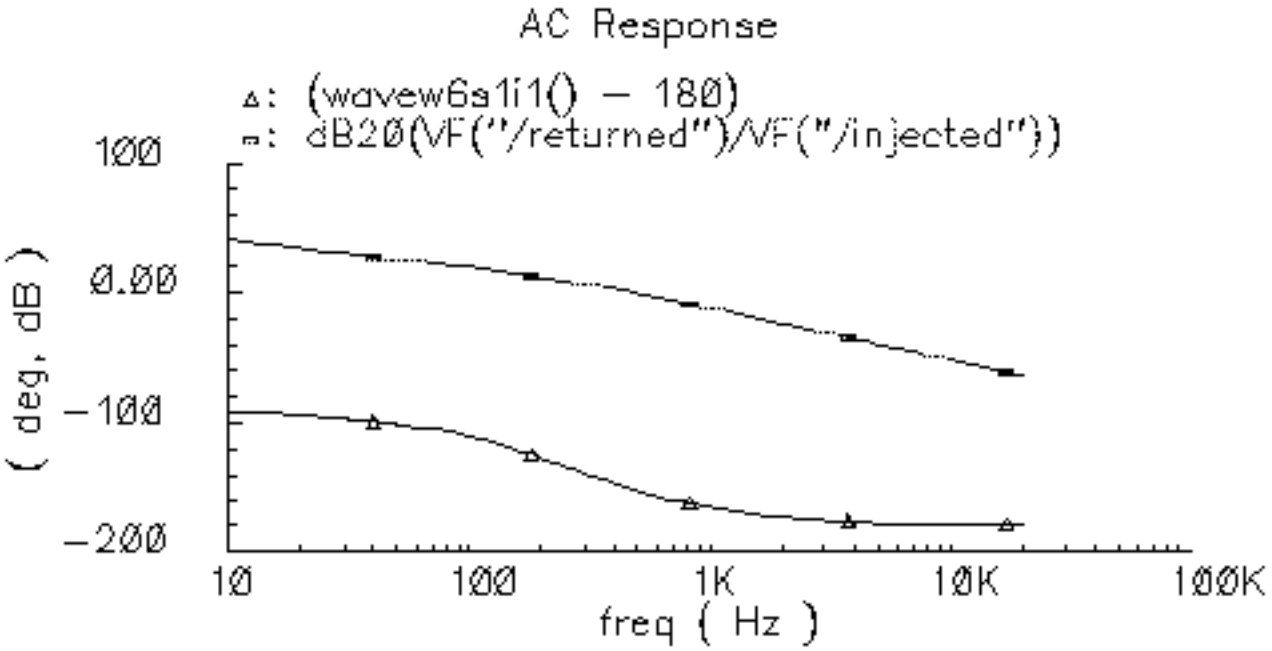
b. In the Choose Curves list box, highlight the original phase curve.

c. Click *Off*.

d. The Curves form looks similar to the following.



The original phase waveform is no longer displayed in the waveform window.



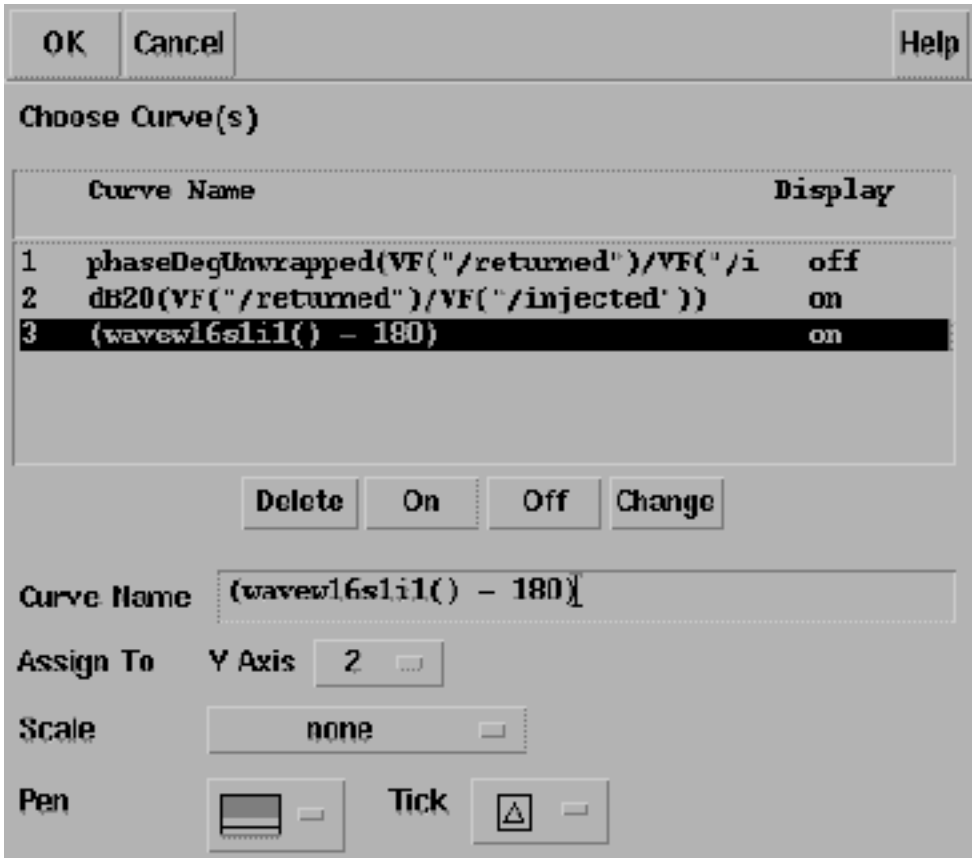
15. To create the Bode plot

a. If necessary, in the waveform window, choose *Curves – Edit* to display the Curves form.

b. In the Curves form, select the shifted (by 180 deg) phase curve

c. In the Curves form *Assign to Y Axis* cyclic field, select *2*.

d. The Curves form looks similar to the following.

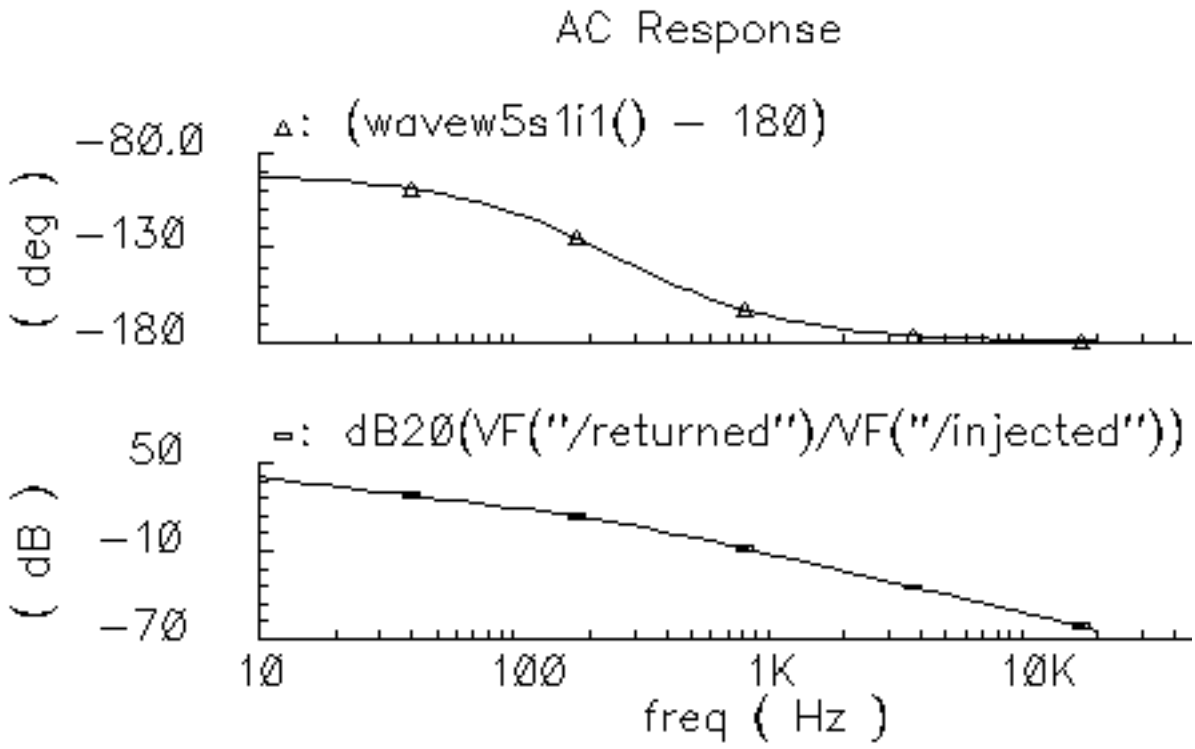


16. Click *OK* in the Edit Curves form.

17. In the waveform window, choose *Axes – To Strip* to change the display as follows.

This produces a Bode plot (magnitude and phase) of the loop gain in the waveform window Shown in Figure 4-9.

Figure 4-9 Bode Plots, Magnitude and Phase of Loop Gain

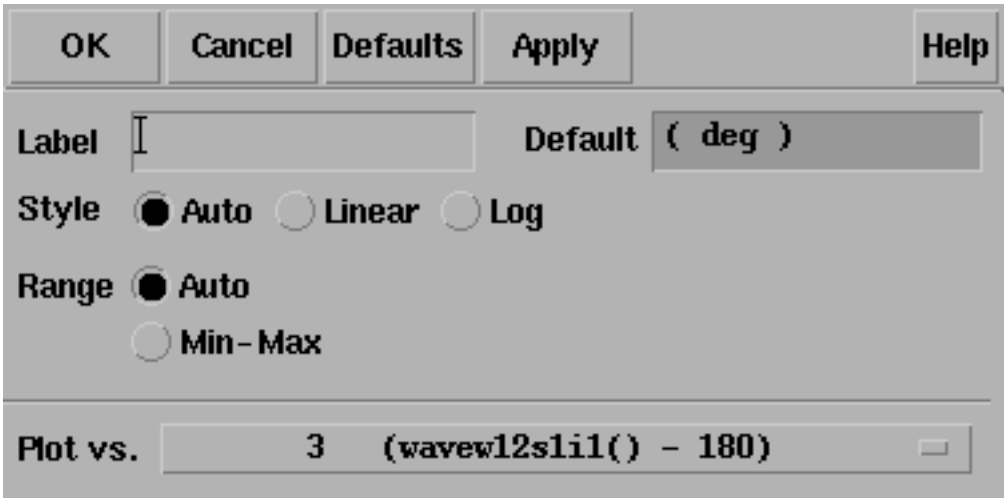


18. To generate a Nichols chart (dB versus degrees) from which you can pick off phase and gain margins,

a. In the waveform window, choose *Axes – X Axes*

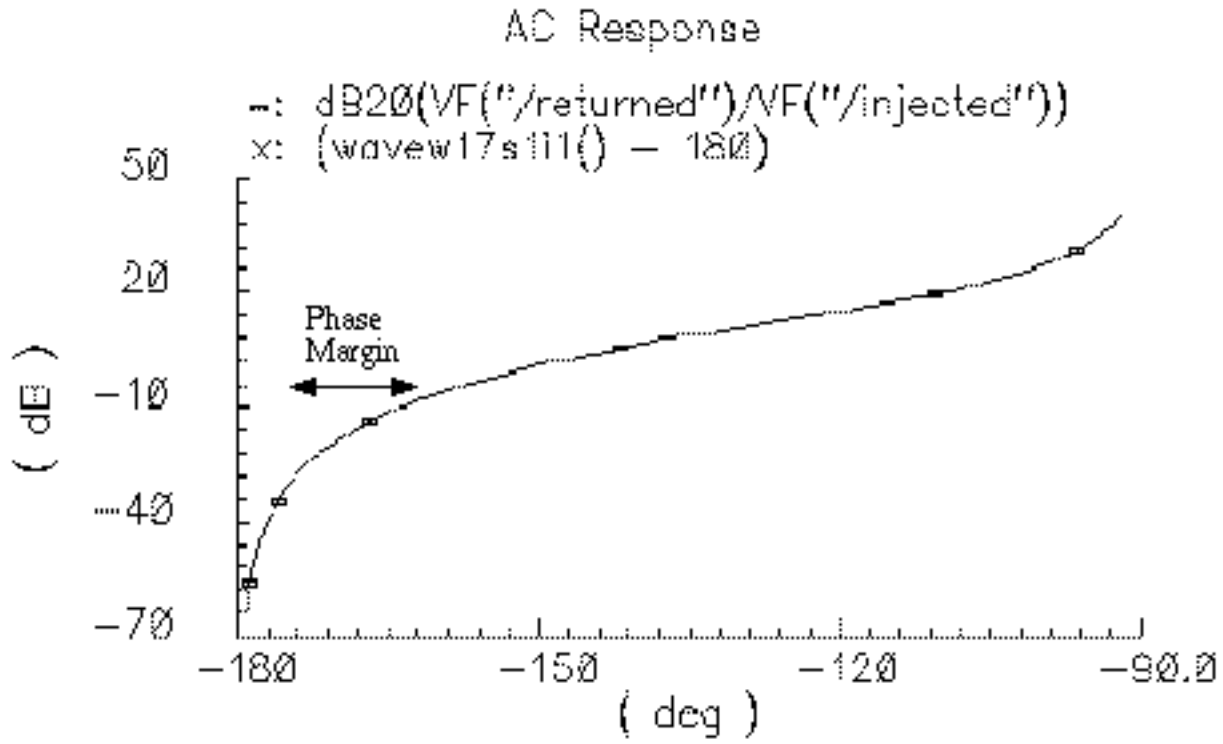
b. In the *Plot vs. cyclic* field select the phase curve you created that is 180 degrees out of phase.

c. Click *OK*.



You now have a Nichols chart like the one in Figure 4-10. The phase margin is 30 degrees.

Figure 4-10 Nichols Chart of Loop Gain



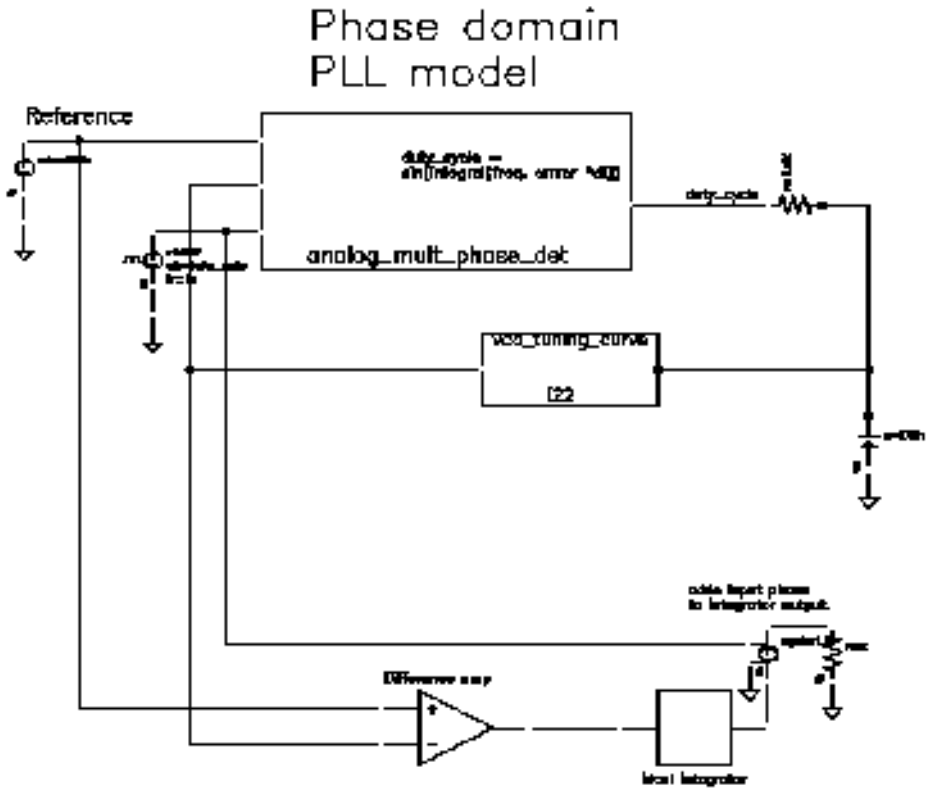
To compute phase margin directly do the following:

1. In the calculator, click *vf*.
2. In the Schematic window, click first on the return node and then on the injected node.
3. In the calculator, click the *divide* button.
4. In the calculator Special Functions menu, choose *phase margin* followed by *print*.
5. Add 180 degrees to the expression in the calculator then choose *print* from the Special Functions menu.
6. The Results Display Window displays the phase margin.

Example 3: PM Input

The circuit used to test for PM input is *example_PM* in the *pllLib* library. The PM (phase modulation) input pin is useful if the PLL is used as a modulator or demodulator, but it also provides a convenient place to perturb the PLL to assess large signal stability. Figure 4-11 on page 330 shows a test circuit for such a stability check.

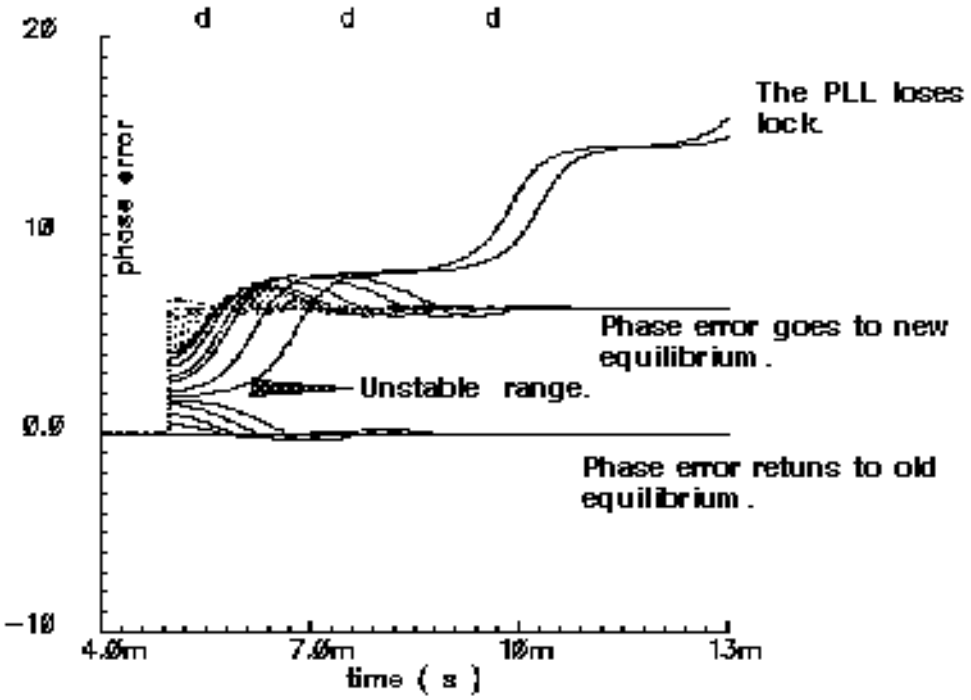
Figure 4-11 Stability Check Using the PM Input



This PLL is very simple but yet different from the previous example. It was modified to produce more interesting results. The lower circuitry requires explanation. The difference amplifier computes frequency error and converts it from Mhz to rad/sec. The integrator computes VCO and reference contributions to phase error. The voltage-controlled-voltage-source at the end adds the input phase stimulus to compute total phase error. The difference amplifier and integrator are from the *ahdLib*.

The input phase is a delayed step. The delay makes the initial phase error easy to read. A parametric analysis on the phase error's step response with respect to the size of the input phase step reveals some interesting behavior. Figure 4-12 on page 331 shows the family of phase error step responses produced by the parametric analysis. The external integrator is intentionally not a circular integrator like the one inside the phase detector model. For large and small steps in input phase, the PLL settles into equilibrium, possibly a new one. However, a narrow intermediate range of steps puts the PLL into an unstable mode. The references examine this behavior in mathematical detail [1,4,6,10]. This example shows one way to assess large-signal stability and to demonstrate that the phase-domain models capture the major non-linear mechanisms.

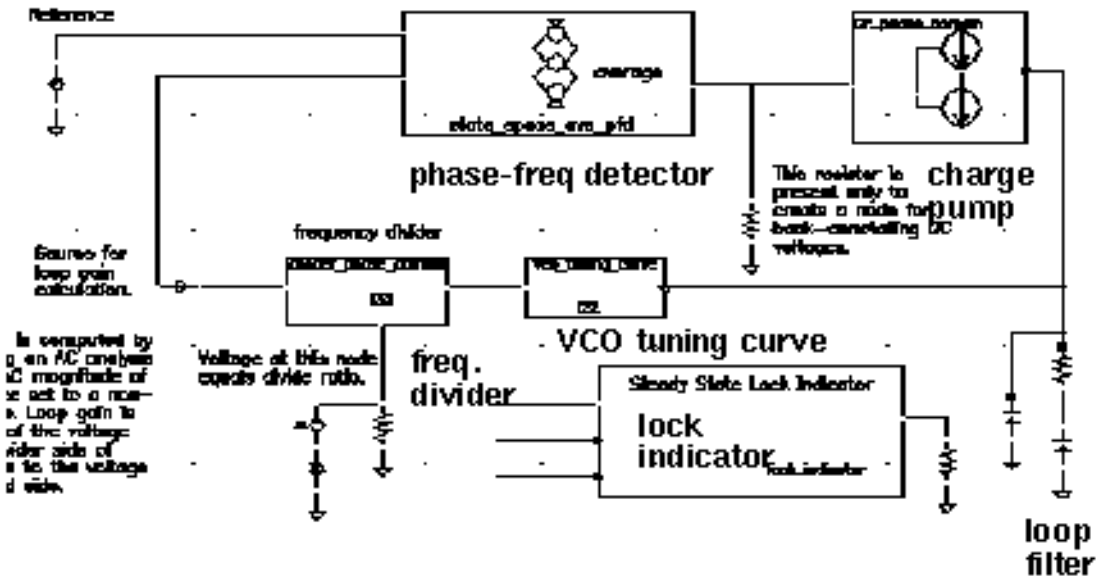
Figure 4-12 Phase Error Response to Step PM Input



Modeling a PFD-Based PLL

Figure 4-13 on page 332 shows a block diagram of a typical PLL with a phase-frequency detector. This section describes how to specify each component in Figure 4-13 and briefly explains what each model does.

Figure 4-13 PLL Block Diagram



VCO

The VCO is modeled by its tuning curve. The tuning curve characterizes the relationship between the input voltage and the output frequency. The input to the VCO model is the loop filter output voltage, also called the VCO control voltage. The VCO output is a voltage representing the VCO's instantaneous frequency in Mhz. Therefore, when the VCO operates at 2 Mhz, the model output is 2 Volts.

The VCO tuning curve is generally nonlinear and can be specified in one of two ways:

- With the coefficients of a fourth order polynomial
- With a look-up table

Polynomial tuning curve: The input voltage is internally clamped to the nearest end point if it moves outside the interval [*min-vco-input-voltage*, *max-vco-input-voltage*]. Although the input voltage may fall outside the interval, the output behaves as though the input voltage value is at the end points. Within the interval, the output is a fourth order polynomial in the quantity, V_{input} minus the free running voltage. When the input voltage equals the free running voltage, the output frequency equals the free running frequency. The scale factor scales the entire polynomial and has a default value of 1. The scale factor is useful in converting data in Khz/volt, for example, to the required Mhz/volt. The parameters are the coefficients of the polynomial.

Table look-up tuning curve: The two parameters are the *scale factor* and the *path to the look-up data*. The look-up model linearly interpolates between data points and linearly extrapolates outside the data interval. The data format is two columns of data delimited by spaces. There is no header, and there are no extra lines at the end. The first column is input voltage. The second column is output frequency. The path to the data can be absolute or relative to the netlist. The netlist is usually stored at

```
<home>/simulation/ckt_name/spectre/schematic/netlist/input.scs
```

but you can choose a different location.

If the data is at

```
<home>/data/table
```

and the netlist is as shown above, the relative path is

```
../../../../../data/table.
Frequency Divider
```

The frequency divider is essentially a simple gain element. It takes an input voltage that represents frequency in Mhz, and then scales it by the divide ratio to generate an output voltage that represents the divided frequency. The divide ratio is numerically equal to the voltage on the control pin. If the divide ratio drops below 0.001, the model assigns it to 0.001 and issues a warning. This assignment prevents division by zero during simulation.

Charge Pump

The charge pump transforms the duty cycle into the expected average current sourced or sunk by the charge pump. You define the maximum source and sink currents, and they can be different from each other. If the charge pump output voltage exceeds the rails you define, the output voltage is clamped to the rail through a 0.001 Ohm resistance. The other parameters are the leakage resistance and open circuit voltage. These last two parameters specify the Thevenin equivalent circuit of a leakage path. The leakage path can source or sink current depending on the open circuit voltage.

Loop Filter

The loop filter is entered component-for-component.

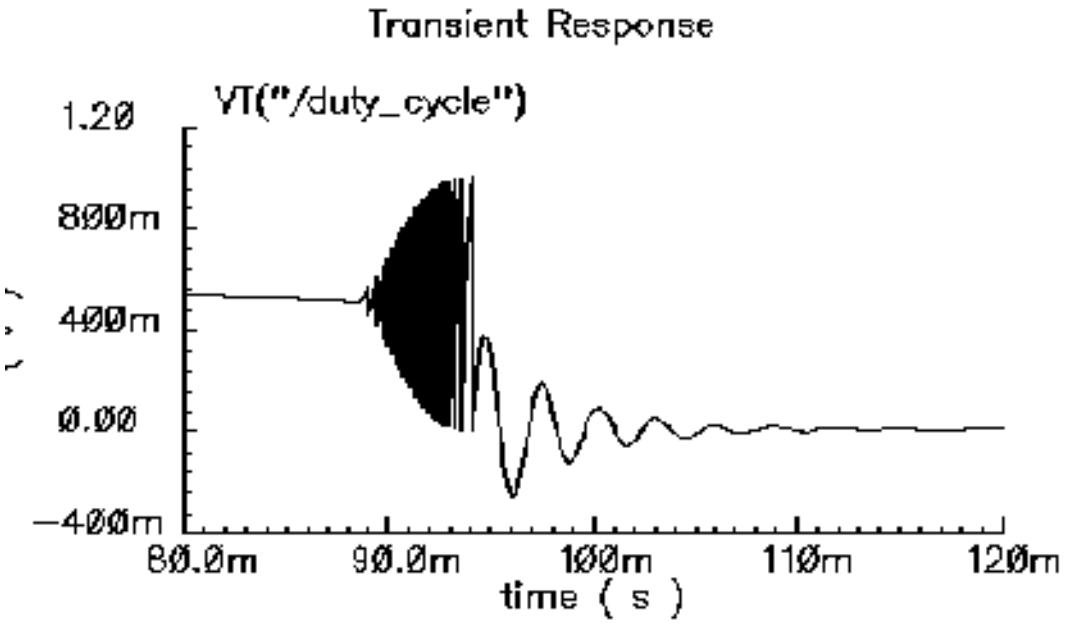
State-Space Averaged PFD (Phase-Domain Phase-Frequency Detector Model)

The phase-frequency detector (PFD) model approximates average behavior of a digital, three-state, phase frequency detector. This is the most complicated model in the PLL library. The term *state-space averaged* is borrowed from the power electronics field [14]. The charge pump currents for the three PFD states are averaged together with a duty cycle much like voltages are averaged together with a duty cycle in a switch-mode power supply model. The PFD inputs are voltages representing the reference and divider output frequencies in Mhz. The output is a voltage that when multiplied by the maximum charge pump current, numerically equals the average charge-pump output current. The PFD output is a duty cycle. When the frequency error is large, the duty cycle is a smooth waveform directly related to the normalized frequency error [1,4]. When the frequency error is small, an integrator inside the PFD model converts frequency error to phase error, and the duty cycle is proportional to the phase error. The duty cycle starts jumping to zero (or resets) as it changes from frequency-mode to phase-mode.

As phase error enters a deadband determined by the minimum-on-time parameter and reference frequency, the model computes a duty cycle pulse with magnitude one and duration equal to the minimum-pulse-width. After the pulse expires, the duty cycle drops to zero until the phase error exits the deadband. As the phase error exits the deadband, the duty cycle increases to a non-zero value. The deadband and fixed-width unity pulse simulate what some texts call *backlash* [8].

The PFD model has two parameters. The first is a numerical option that controls the trade-off between execution speed and accuracy. The *speed_vs_accuracy* parameter controls the number of times the internal integrator is reset during the transition from frequency-mode to phase-mode. Too few resets can cause error. Too many resets can needlessly slow run time. The default value of this parameter is 50k. To reduce the number of resets in a slow PLL, and thereby reduce run time, increase the *speed_vs_accuracy* parameter to 70k or 100k. To increase the number of resets in fast PLLs, and thereby increase the accuracy, reduce the *speed_vs_accuracy* parameter to 10k or 20k. A reasonable setting for the *speed_vs_accuracy* parameter produces a duty cycle step response that resets approximately to zero at least 3-10 times before entering the final transient. Figure 4-14 on page 335 shows reasonable duty cycle step response.

Figure 4-14 Reasonable Duty Cycle Waveform



The other parameter is the *minimum_on_time* which controls the backlash. This is the minimum pulse width the PFD can generate. As the phase error decreases, the pulse width drops discontinuously from the minimum pulse width to zero. This effect creates a deadband in the duty cycle versus phase error curve.

Figure 4-12 on page 331 was generated with the default *minimum_on_time* parameter value of zero µs. The default value of the *minimum_on_time* parameter produces no deadband and no unity pulses. The default deactivates the backlash mechanism.

Figure 4-15 on page 336 was generated with a *minimum_on_time* parameter value of 0.2 µs. Figure 15a illustrates that the pulses only occur as the phase error enters the deadband. Figure 15b shows the limit cycle created by the backlash. The limit cycle is primarily determined by leakage on the loop filter and the minimum pulse width. Some references suggest biasing the duty cycle away from the deadband or loading the filter down to force the limit cycle frequency to a value in which the loop filter attenuates it. The phase-frequency detector model can help quantify the problem and check the solution.

A pulse is not kicked out upon exiting the deadband because that behavior causes convergence problems for Spectre RF. If phase error is entering the deadband, a pulse at that moment pushes phase error in the same direction it was going, into the deadband. If a pulse occurs as phase error exits the deadband, the pulse drives phase error back into the deadband and Spectre RF has trouble figuring out whether phase error should leave the deadband at all. Fortunately, no significant error is introduced by implementing the pulse only when phase error enters the deadband. In a backlash limit cycle, the feedback loop quickly

drives phase error back into the deadband and a pulse occurs on the way in. The error is in the time the feedback loop takes to return phase error to the edge of the deadband and that is usually small when PLL is in a backlash limit cycle.

(The PLL model that generated Figure 4-15 had a center frequency of about 1Mhz and the simulation ran out to 130ms. A voltage-domain model might easily simulate 10 points per carrier cycle. The voltage-domain model would require 1.3 million points to simulate the same amount of action. I did not attempt it. The phase-domain simulation that generated Figure 4-15 ran in a matter of seconds!)

Figure 4-15 Duty Cycle Waveforms With Pulses and Backlash

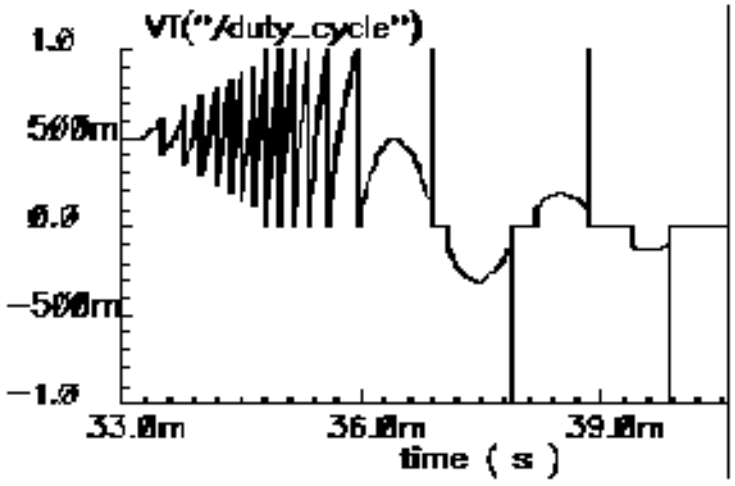


Figure 15a
Duty cycle between
33 ms and 40 ms

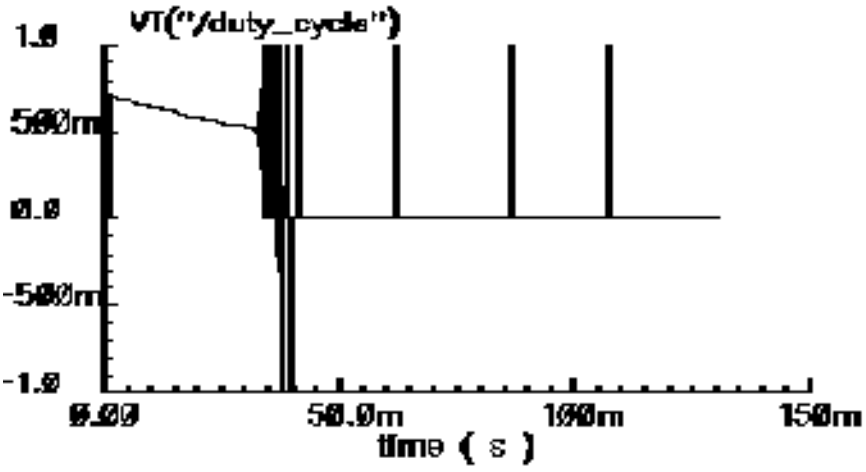


Figure 15b
Duty cycle to
130 ms

Lock Indicator

All real components have limited outputs and the limits of any one component can keep the PLL from locking. Just like the simple phase detector model, all of the phase-domain models operate one way for DC analysis and another for transient analysis to prevent DC convergence errors. The lock indicator monitors three signals. In the example, the lock indicator monitors the phase detector output, the VCO control, and the charge pump output. If any of those signals exceeds its limit, the lock indicator output is zero, signifying that the loop is not locked in steady state. If all signals are within their limits, the output is 1 volt, specifying that the loop is locked. The lock indicator is only valid for DC analysis. Use node names to tie the lock indicator inputs to the right nodes and use variables for the component limits. You must specify the units manually twice, once for each component and once for the lock indicator. With variables, the lock indicator parameters are linked to the proper component parameters, and you specify changes in only one place.

Example 4: Modeling Acquisition Transients

The circuit used to model acquisition transients is the *example_phase_domain* in the *pllLib* library. Figure 4-16 on page 338 shows the duty cycle and VCO frequency response to a momentary change in the divider ratio. When the divider ratio changes, the PFD enters the frequency-mode and slews the VCO frequency toward the new value. As the VCO frequency approaches the final value, the PFD model gradually changes from frequency-mode to phase-mode. When the frequency error is small, but still large enough to slew phase error, the duty cycle waveform looks like a sawtooth waveform. The model gradually increases the amplitude of the sawtooth component of the duty cycle, and always maintains the correct average, as frequency error reduces to zero. The final duty cycle transient is the sawtooth that depends mostly on phase error. Figure 4-12 on page 331, modifies the x-axis of the graph to show the duty cycle in the first transition.Figure 4-17 on page 338 modifies it further to show the sawtooth waveform.

Figure 4-16 Response to Momentary Change in Divider Ratio

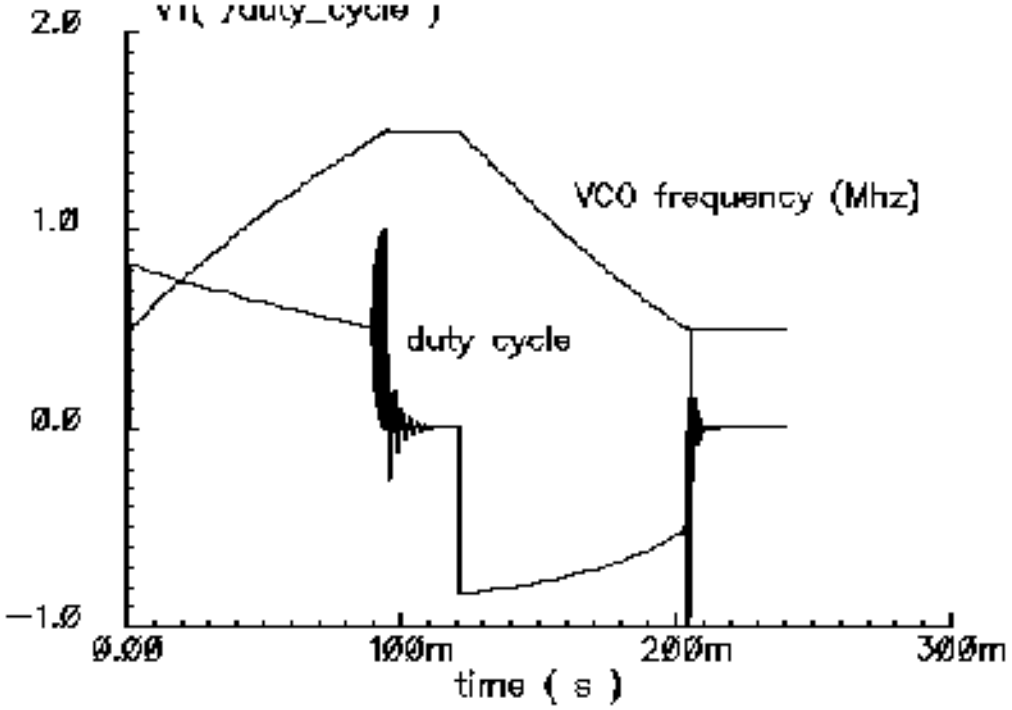
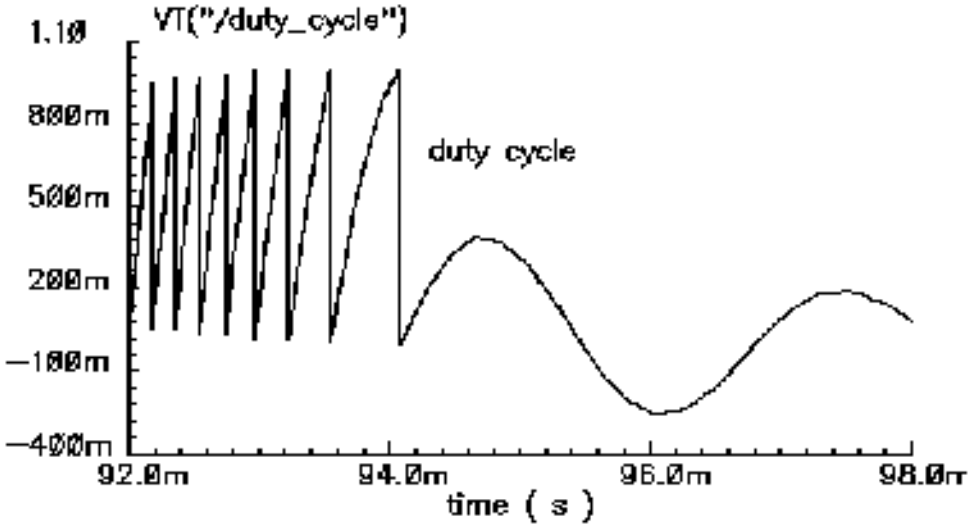


Figure 4-17 Duty Cycle During Transition From Frequency to Phase Mode

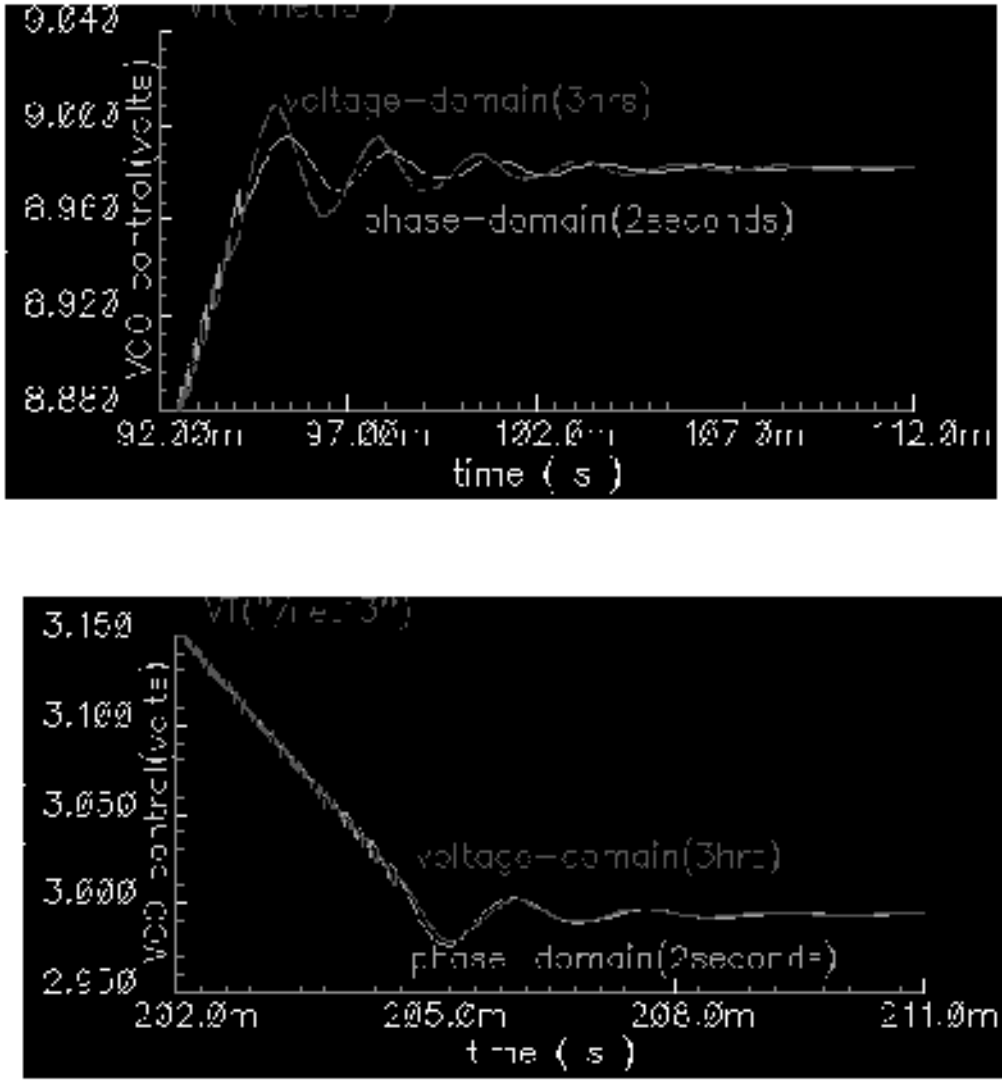


Example 5: Comparison With a Voltage-Domain Model

Use the *example_voltage_domain* model in the *pllLib* library with the only node that can be directly compared between phase- and voltage-domain models, the VCO control node. At full scale, the difference between the two models is not visible. The differences occur at the transitions. Figure 4-18 on page 340 compares the two models at the transitions. In this example and on the same machine, the voltage-domain model simulates in three hours while the phase-domain model simulates in two seconds.

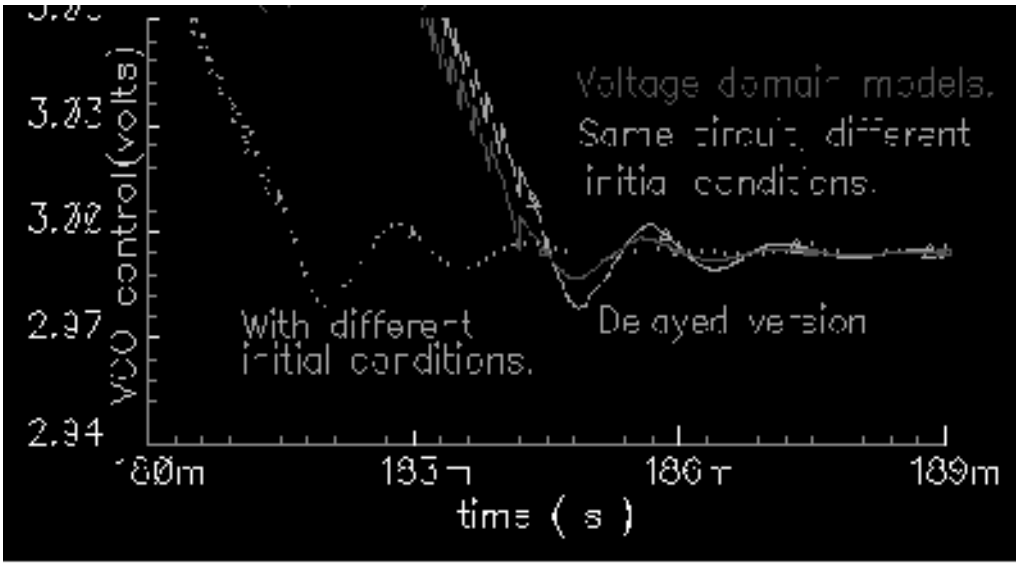
The error between the two models does not appear to be consistent. It is larger in the first transition. Furthermore, decreasing the *speed_vs_accuracy* parameter does not always increase the similarity of the waveforms. This is because the final transient, the one driven primarily by phase error, depends on the residual frequency error at the time the phase error last crossed 2π. The frequency error at that moment, especially after a long frequency slewing period, is sensitive to, among other effects, initial conditions.

Figure 4-18 Comparison with Voltage Domain Model



The voltage-domain model therefore shows the same level of variation for small differences in the initial conditions preceding the transition to the new equilibrium. Figure 4-19 on page 341 compares two voltage-domain simulations of the VCO control signal during the second transition. One of the voltage-domain simulations (the dotted waveform) used different initial conditions. The solid waveform is a delayed version of the dotted waveform. The delay overlays the two simulations for direct comparison. The error is comparable to the error between voltage-domain and phase-domain simulations.

Figure 4-19 Effect of Initial Conditions



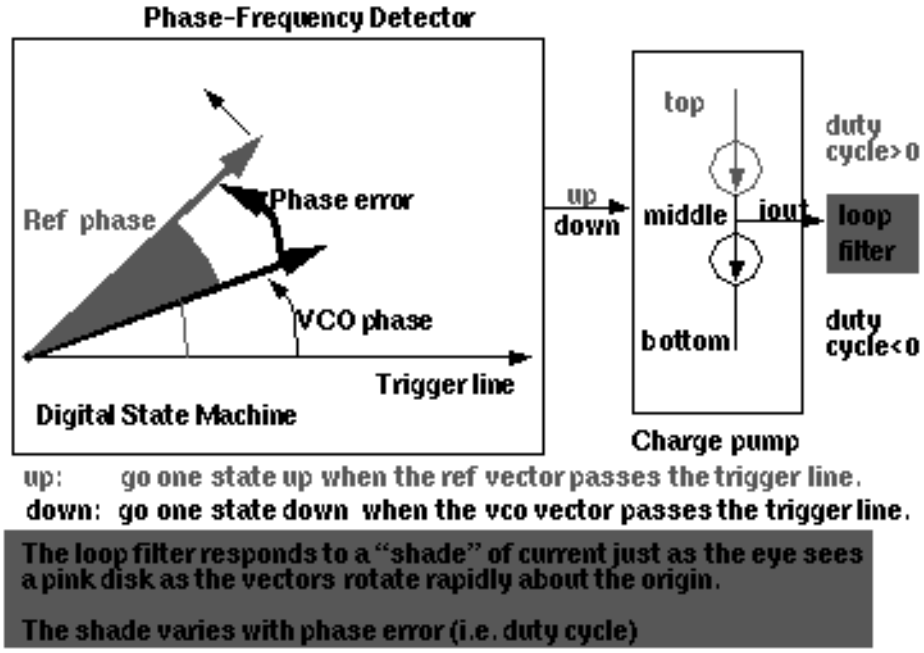
How the PFD Model Works

The heart of the PFD is a digital state machine [8,9].The model is for PFDs with three digital states. The output stage is usually a charge pump or pair of switches. It is convenient to model the PFD in two pieces. The first piece models the state machine and computes a duty cycle that is independent of the output stage. The second piece models the output stage. The charge pump (CP) is used here as an example.

How the PDF/CP Pump Works

Let the three PFD states be stacked. In the top state, the PFD commands the CP to source current. In the middle state, the pump is off. In the bottom state, the CP sinks current. The PFD is edge triggered. Figure 4-20 on page 342 shows vectorial representations of the reference and VCO clocks [1,4]. Both vectors rotate counter-clockwise around the origin. The angle between the hands equals phase error. Phase error lies between +-2π. Whenever the reference passes a trigger line, like 3 o'clock, the state jumps to the next state up. If the PFD is already in the top state, the state does not change. Whenever the VCO passes the trigger line, the state jumps to the next state down. If it is already in the bottom state, it again does not change. For a fixed phase error, the state toggles as the hands rotate. State toggles between the middle and top, or between the middle and bottom states. The percentage of time spent in the top, or bottom, state is the duty cycle. The duty cycle is positive for top to middle toggling and negative for middle to bottom toggling.

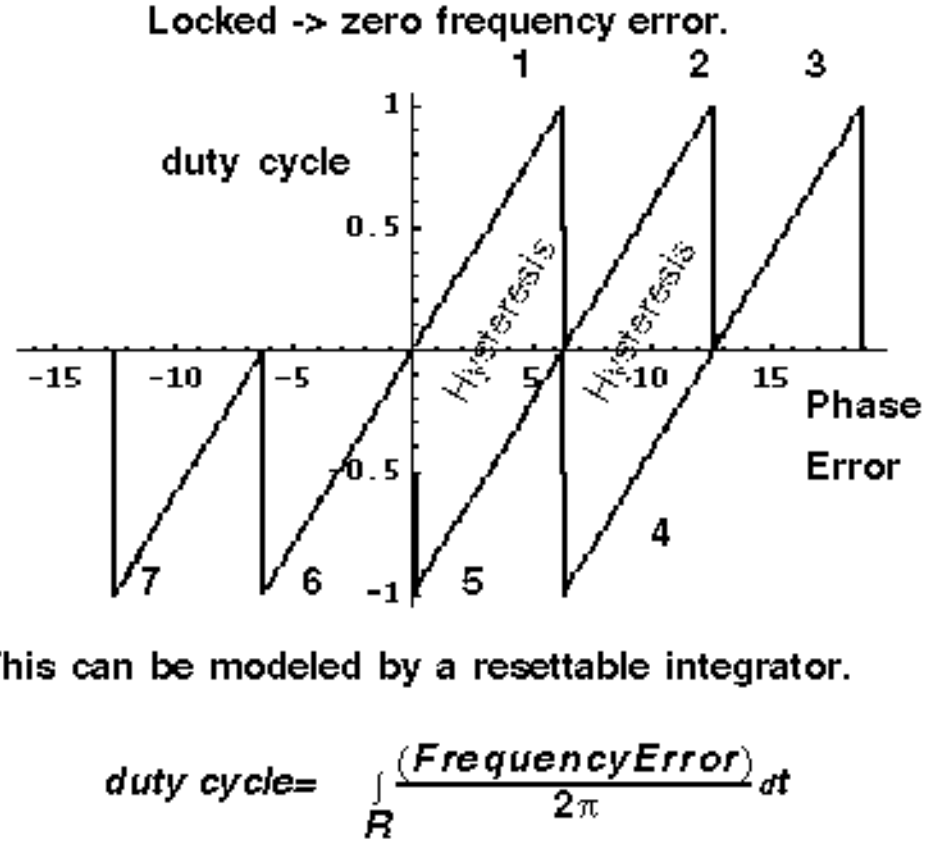
Figure 4-20 PFD Operation



If the reference and VCO frequencies are identical, the vectors in Figure 4-20 rotate together. Let the sector defined by phase error be red if the reference leads and blue if it lags. If the hands rotate once per minute and the reference leads, you see two colors, white and red. At two million revolutions per second, you see pink. The shade of pink depends linearly on the phase error. Although PFD output current toggles between two values, the loop filter and VCO respond mainly to the "shade" of current. The shade is proportional to the duty cycle. With zero frequency error, duty cycle equals phase error divided by 2π. Existing literature uses one function to describe the *phase-error-to-duty-cycle* relationship and a different function to describe the *frequency-error-to-duty-cycle* relationship. These two functions are the *locked duty cycle function* and *averaged unlocked duty cycle function*, respectively. The new model combines these two functions into one practical model.

The locked duty cycle function is a multivalued sawtooth. For monotonic movements away from the origin of steady-state phase error, the duty cycle is a sawtooth in the upper-half plane. The duty cycle lies in the lower-half plane for negative movements. If a movement starts off positive, then changes direction, the duty cycle crosses zero and becomes a sawtooth in the lower half plane. The duty-cycle-phase-error trajectory encloses a nonzero area as shown by the {1-2-3-4-5-6-7} sequence of peaks in Figure 4-21 on page 343. This is a good reason for putting the integrator next to the non-linearity—hysteresis involves memory and the integral supplies it.

Figure 4-21 Duty Cycle Versus Phase Error With Zero Frequency Error



This can be modeled by a resettable integrator.

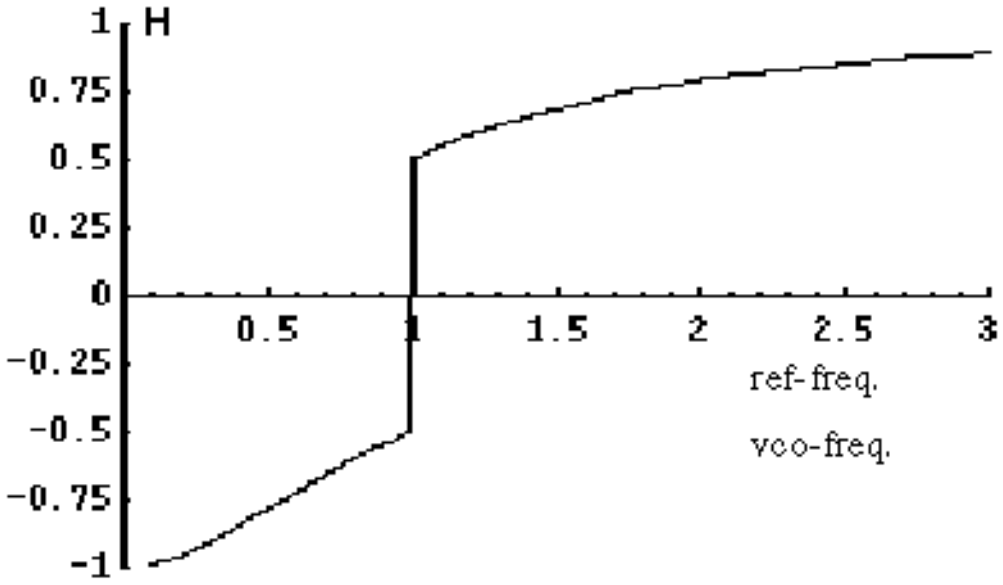
$$duty\ cycle = \int_R \frac{(FrequencyError)}{2\pi} dt$$

The new model operates only on frequency error. For small-frequency errors, the duty cycle is indeed proportional to the phase error. The phase error is the integral, with respect to time, of the frequency error. The duty cycle therefore equals the integral of the frequency error divided by 2π. Resetting the integral whenever it hits +-2π produces the multivalued sawtooth described above. If the frequency error changes sign, the resetting integrator ramps to zero, passes through zero, and generates a sawtooth in the lower-half plane. The phase-error-duty-cycle trajectory is precisely the multivalued sawtooth described above. The resettable integrator (RI) merges the integrator of a phase-domain model with the locked duty-cycle function.

For a sustained frequency error, the RI model predicts an average duty cycle of +-1/2 regardless of error size. This is correct only for small-frequency errors. The true duty cycle goes to +-1 for large-frequency errors. Let the reference frequency far exceed VCO frequency. Whenever the VCO passes the trigger line, the reference frequency passes shortly thereafter. The reference frequency might pass the trigger line several more times before the VCO passes again. In this case, the phase error is still a sawtooth, but the average duty cycle is nearly 1. This behavior lets the PLL acquire input signals faster. The function H, in Figure 4-

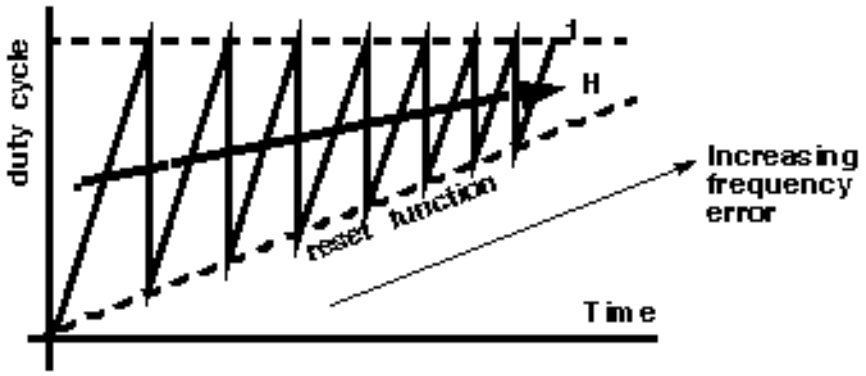
22 on page 344, shows the averaged unlocked duty cycle. This cycle depends on the ratio of the reference to VCO frequencies and is discontinuous where frequency error is zero.

Figure 4-22 Averaged Unlocked Duty Cycle



The RI is modified to include the frequency effect. For small-frequency errors, the predicted average duty cycle equals 1/2. This is true because the RI runs from the reset point (=0) to the reset threshold (=2π). It is not necessary to reset the integrator to zero. Resetting the integrator to a “reset” function gives the correct average duty cycle (Figure 4-23). As the frequency ratio goes to +-infinity, the reset point changes to +-2π. Because the reset threshold is still +-2π, the predicted average duty cycle changes to +-1. As the frequency ratio approaches unity, the reset point returns to zero, and the predicted average duty cycle returns to 1/2.

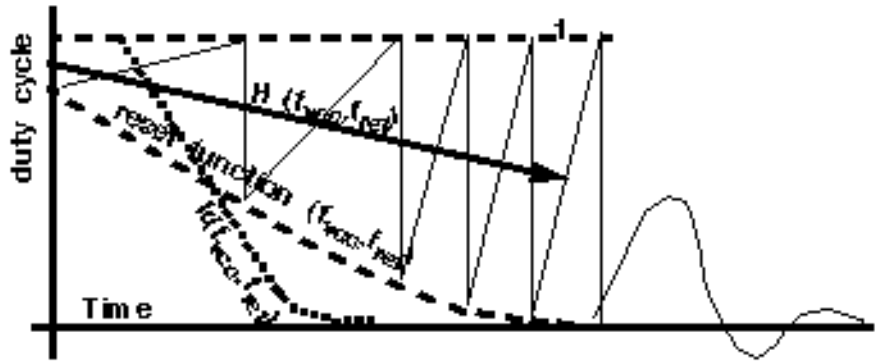
Figure 4-23 Combining Averaged Locked and Unlocked Duty Cycles



$$duty\ cycle = \int_{R(frequncyerror)} \frac{(FrequencyError)}{2\pi} dt$$

The state space averaged PFD model requires one more addition to be practical. As the reset point nears +-2π, the integrator resets very frequently and execution stalls. The integrator must be deactivated for large-frequency errors. The new PFD model uses the weighted sum of a RI and the H. The weighting factors are *k* and (1-*k*). *k* is a function of the ratio of the two input frequencies. *k* approaches 1 for large-frequency errors and approaches 0 as the frequency ratio approaches unity. A factor of (1-*k*) under the integral deactivates the integral for large-frequency errors. The resulting PFD model looks like H for large-frequency errors, and it looks like the reactivated RI for small-frequency errors. *k* determines how fast the RI reactivates and how gradually the model changes from H to the RI. A *speed_vs_accuracy* parameter controls *k*. If the model does not reset a few times before reaching frequency lock, you can improve the results by decreasing the *speed_vs_accuracy* parameter. If the model resets so often that the simulation is too slow, you can speed execution by increasing the parameter. The default setting of 50000 covers a wide range of loop speeds. Figure 4-24 on page 346 shows the transition from frequency-mode to phase-mode.

Figure 4-24 Complete Model



$$duty\ cycle = k*H + (1-k)\int_{R}(1-k)\frac{(FrequencyError)}{2\pi}dt$$

References

[1“Loops, Theory and Application,“ J. L. Stensby.

[2]“Relative -phase modeling speeds Spice simulation of modulated systems,” John Kesterson, November 11, 1993 issue of EDN

[3]“Simulation of Communication Systems,” M. Jeruchim, P. Balaban and K. Shanmugan. Plenum, 1992.

[4]“Synchronization in Digital Communications,” Heinrich Meyr and Gerd Ascheid, Published by John Wiley and Sons.

[5]“Phase-Locked and Frequency-Feedback Systems,” Klapper and Frankle, Published by Academic Press

[6]“Phaselock Techniques,” Floyd Gardner, Published by John Wiley and Sons

[7]“Non-Linear Relative Phase Models of Phaselock Loops,” Jess Chen, Cadence Technical Conference, 1996.

[8]“Phase-Locked Loops: Theory, Design, and Applications,” Roland Best, Published by McGraw-Hill

[9]“Phase-Locked Loop Circuit Design,” Dan Wolaver, Published by Prentice Hall

[10]“Phase-Lock Basics,” William F. Egan. Published by John Wiley and Sons.

[11]“Non-Linear State Space Averaged Modeling of a 3-State Digital Phase-Frequency Detector,” Jess Chen, Cadence Technical Conference, 1997

[12] “Analog and Mixed-Signal Hardware Description Languages,” Edited by A. Vachoux, J. Berge, O. Levia, and J. Rouillard. Kluwer Academic Publishers.

[13] “Macromodeling with SPICE,” J.A. Connelly and P. Choi. Prentice Hall. Pages 168-169.

[14] S. Cuk, California Institute of Technology, Ph.D. Thesis, “Modelling, Analysis, and Design of Switching Converters.” 1977