Application Notes on Direct Time-Domain Noise Analysis using Virtuoso Spectre

Purpose

This document discusses the theoretical background on direct time-domain noise modeling, and presents a practical approach that generates random noise signals with given power spectral density in the time domain. This enables the direct time-domain noise analysis, or the so-called transient noise analysis, of a linear or nonlinear system.

This document further describes the implementation and use model of transient noise analysis in Virtuoso Spectre, and its application on direct time-domain noise analysis of linear and nonlinear noisy systems.

Introduction

To assess the impact of device noise on circuit behavior, small signal approximation is most commonly used in commercial noise analysis tools. It assumes device noise contributions in a design are small enough that it does not alter the operating point or periodic state of interest. As a result, noise analysis becomes a linear problem, and noise can be separated from signals by means of the superposition principle. Noise properties of a particular design can be solved in the frequency domain by linearizing the circuit equation around the operating point or periodic state of interest.

Although it is widely adopted in various design practices, the limitation of this traditional approach is obvious. It produces inaccurate results when noise is large, and/or when the circuit is highly nonlinear. Furthermore, there is no simple way to verify the validity of the small-signal assumption that the signal and noise are indeed independent.

Theoretically, direct noise analysis involves solving stochastic differential equations. When random forces are added to a circuit, all circuit unknowns become random variables. The task of circuit simulation becomes solving the probability distributions of circuit unknowns. Certain simplification has to be made to make this approach doable. For example, all random variables can be assumed Gaussian probability distribution that allows analytic averaging, or the circuit equation can be formulated and approximated using lower order distribution moments. However even with these simplifications, the resulting nonlinear problem is too complicated to solve, and it has not found any commercialized application.

When ultimate accuracy is needed, direct noise analysis in the time domain is the only viable solution. It enables the designer to examine how a nonlinear system responds to noise in cases where noise to

signal ratio is large, and operating point, or periodic state of interest may be affected by the noise. This is a very costly approach. Device noise models need to be evaluated and random noise sources need to be generated at each times step of transient analysis. Time step is also forced to be uniformly small as noise bandwidth is typically larger than circuit bandwidth. In addition, a large number of repetitive simulations need to be performed, or the simulation has to span large number of periodic cycles in order to draw meaningful statistical characteristics.

The necessity of direct noise analysis in time domain has been recognized by various researchers and commercial EDA companies. The key difference is how to model noise sources in time domain that can be easily plugged into transient analysis. Various approaches have been published so far. Bolcato, at al [1] models each noise source as a sum of sinusoids over the frequency range of interest, with random phase, and with amplitude equals to the given noise power spectral density. Recently from [2], it was shown that the magnitude of noise signal in frequency domain is also a random number with Rayleigh probability distribution around given power spectral density, thus random noise spectrum can be properly generated in the frequency domain. However, both noise modeling approaches are CPU intensive -- as the time domain noise signals are obtained using inverse Fourier transformation, and memory intensive -- as all time domain noise sources have to be pre-calculated and stored over the full duration of transient analysis.

In the following sections, we will first summarize the typical device noise models -- thermal, shot, flicker, and noise data table. Then we will describe a different way of providing time-domain white noise sources and its extension to handle frequency dependent noise models. Our approach satisfies stricter performance and capacity requirements. As a result, CPU time used for noise signal generation is much less than that for device evaluation, additional allocated memory is much less than that used for devices and matrix solver. Finally and very importantly, device noise code developed in existing Spectre CMI devices and Verilog-A modules is re-used for time-domain noise modeling, no model code change is ever needed.

Device Noise Models

Device noise models typically have a number of independent noise sources connected to the device terminals or internal nodes. Each noise source is characterized by its noise spectral density, which is a function of bias voltage and/or frequency. Most often used device noise models are:

Thermal noise of a resistor with constant spectral density

$$\overline{n^2} = \frac{4kT}{R} \quad ,$$

bias dependent shot noise

$$\overline{n^2} = 2q(v) \cdot I(v) \, ,$$

bias and frequency dependent flicker noise

$$\overline{n^2} = k\frac{I(v)}{f}.$$

In addition, noise file can be associated to an independent source to model arbitrary frequency dependent noise profile, obtained from measurement or behavioral modeling

$$\overline{n^2} = \overline{n^2}(f).$$

The thermal noise and shot noise are both considered white noise processes, which implies their spectral density is flat with frequency and the noise is not self correlated. The frequency dependent flicker noise is also referenced as pink, or colored noise in many literature. When noise source is embedded in a circuit, the circuit transfer function can modify noise spectral density adding more color to the output noise. It is important to note that all physical circuits have finite bandwidth, and output noise power in the frequency range above the circuit bandwidth is negligible. Therefore, it is sufficient to model only band-limited noise sources

The actual device models have much more complicated noise equations with far more parameters. The Verilog-A language provides special functions to describe frequency independent (`white_noise()`) and frequency dependent flicker noise sources (`flicker_noise()`).

All of these device noise models are supported in Spectre's transient noise analysis. However, be aware of the following two restrictions:

- For an independent source, the spectral density function given in the noise file should be decreasing with frequency.
- The multi-port nport device noise model, internally or externally, is given as a frequency dependent noise correlation matrix, that is currently not supported in Spectre's transient noise analysis.

Frequency- and Time-Domain Noise Analysis

Traditionally, noise analysis is performed in a small-signal fashion. It is assumed that device noises be so small that their presence do not alter the operating point or the periodic state of interest. As a result of this simplification, noise analysis becomes a linear problem, performed by linearizing the circuit at its operating point or the periodic state of interest. This small-signal noise assumption is valid for many applications and has been widely adopted in IC design practices.

The small-signal noise analysis computes the individual contribution of every noise source in the circuit to the output noise spectral density. It also computes the composite noise spectral density. All noise sources are assumed to be independent Gaussian noise processes with a given spectral density

and random phase. As a result, the composite output noise is computed as the mean-square sum of the contributions from each noise source individually.

The limitation of small-signal noise analysis is also obvious. It becomes inaccurate when noise is large, and/or the system is highly nonlinear so that signal and linear can not be easily separated. Furthermore, there is no simple way to verify the validity of the small-signal noise assumption, for a given design.

Thus there is a need to directly solve the large-signal noise analysis problem in the time domain. Time-domain, or transient noise analysis, behaves just like a regular transient analysis, except that all noise sources in the circuit inject random noise signals into the simulation during each time step.

Understandably, this is a computation-intensive approach, for several reasons. Device noise models need to be evaluated and random noise sources need to be generated at each time step of transient analysis. The minimum time step the simulation can take is now limited by the noise bandwidth, that typically is somewhat larger than circuit bandwidth. Finally, a large number of repetitive simulations need to be performed, or the simulation has to span large number of periodic cycles, in order to draw meaningful statistical characteristics.

The key technology that enables time-domain noise analysis is the ability to properly inject random device noise at each time step. Next we will discuss how that can be done for white noise signals, and how to extend it to handle frequency dependent colored noise signals.

Generating White Noise Signals in Time Domain

In the time domain, a white noise signal $n(t)$ with bandwidth F_{max} can be approximated as

$$n(t) = \sigma \cdot \eta(t, \Delta t)$$

where $\eta(t, \Delta t)$is a random number with Gaussian probability distribution updated with time interval Δt. The noise signal amplitude and update time interval are

$$\sigma = \sqrt{\overline{n^2} \cdot F_{max}}$$

$$\Delta t = \frac{1}{2F_{max}}$$

where $\overline{n^2}$ is noise power spectral density of ideal white noise source. Autocorrelation function for this random signal can be found directly by integration over time:

$$n^2(t) = \lim_{T \to \infty} \frac{1}{T} \int_T n(\tau) n(t-\tau) d\tau = \sigma^2 \Lambda\left(\frac{t}{\Delta t}\right)$$

where Λ is a triangular pulse function of width Δt. In order to evaluate this integral we used the following properties of our random signal:

$$\lim_{T \to \infty} \frac{1}{T} \int_{T} n(\tau) n(\tau) d\tau = \sigma^2$$

$$\lim_{T \to \infty} \frac{1}{T} \int_{T} n(\tau) n(\Delta t - \tau) d\tau = 0$$

The power spectrum of *n(t)* can be calculated as a Fourier transform of auto correlation function:

$$n^2(f) = \sigma^2 \cdot \Delta t \cdot \sin c^2(f \cdot \Delta t)$$

At zero frequency the spectral density of *n(t)* is equal to that of ideal white noise source:

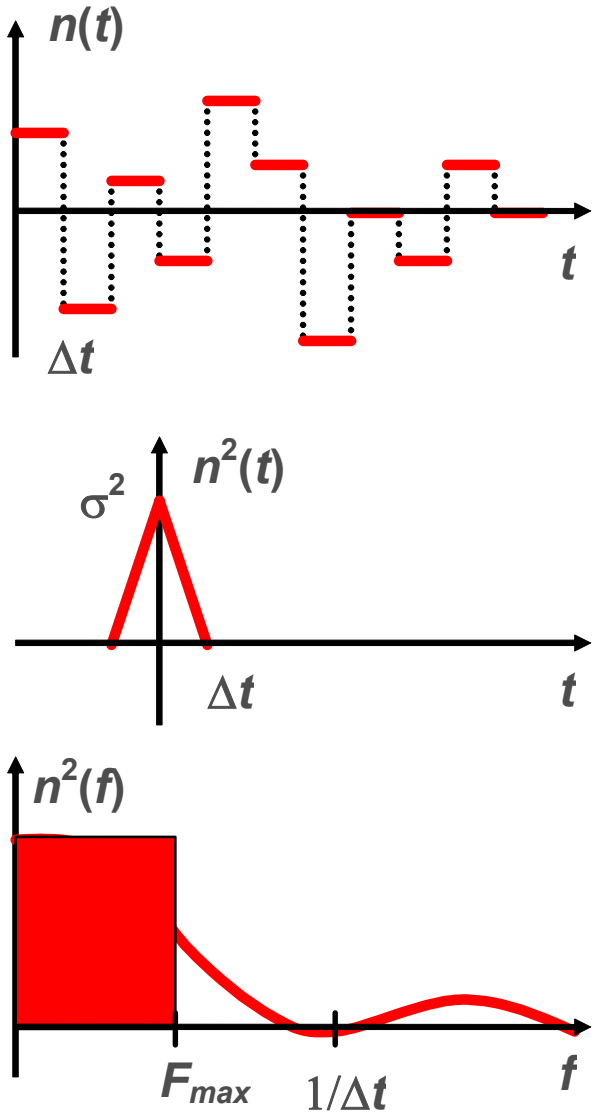
$$n^2(0) = \sigma^2 \cdot \Delta t \cdot \sin c^2(0) = \overline{n^2}$$

Total noise power can be obtained by integrating over the frequency:

$$\int_0^\infty n^2(f) = \overline{n^2} \cdot F_{max}$$

The noise signal, its auto correlation function, and spectral density are shown in Figure 1 on page 6. Filled rectangle indicates spectral density function of ideal white noise source with band limit *Fmax*. This is the algorithm we use to generate white noise signals in time domain.

Figure 1



Generating Frequency Dependent Noise Signals in Time Domain

We extended the white noise approach to handle more generic frequency dependent noise sources, by approximating the frequency dependency using a set of step functions. First we divide the frequency range of interest from *Fmin* to *Fmax* in octaves (steps).

$$f_0 = F_{max}, f_k = 2^{-k} F_{max}, f_N = F_{min}$$

and create white noise sources, one for each octave. The total noise is a sum of individual octave noise sources:

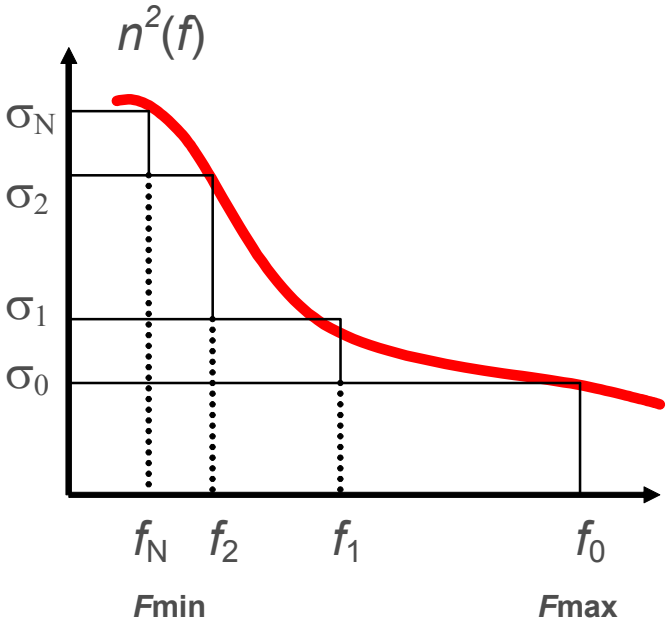
$$n(t) = \sum_k \sigma_k \cdot \eta\left(t, \frac{1}{2f_k}\right)$$

where F_{max} is the noise bandwidth. The amplitudes of octave noise sources are computed in a way to match frequency-dependent noise spectral density given in the device model

$$\sigma_k = \sqrt{\left(n^2(f_k) - \overline{n^2}(f_{k-1})\right) \cdot 2f_k}$$

This is graphically illustrated in Figure 2 on page 7. A Similar approach, also known as Voss - McCartney algorithm, was first proposed for the pink noise generation in audio applications [3].

Figure 2



The computational effort required to generate a noise source with arbitrary spectral density is only twice of that for the single white noise source. Figure 3 on page 8 shows the update times for octave noise sources. As you can see, update time interval doubles for each consecutive octave. We update sources in a staggered fashion, in order to reduce discontinuities in noise signal.

Figure 3

The only assumption made in our modeling approach is that the noise spectral density is a monotonous function, decreasing over frequency. All physical device noise models satisfy this condition.

Transient Noise Analysis

As mentioned before, transient noise analysis behaves just like a regular transient analysis, except that random device noise signals are injected at each time step. However, the injection of noise signals does affect transient analysis time step control and simulation convergence.

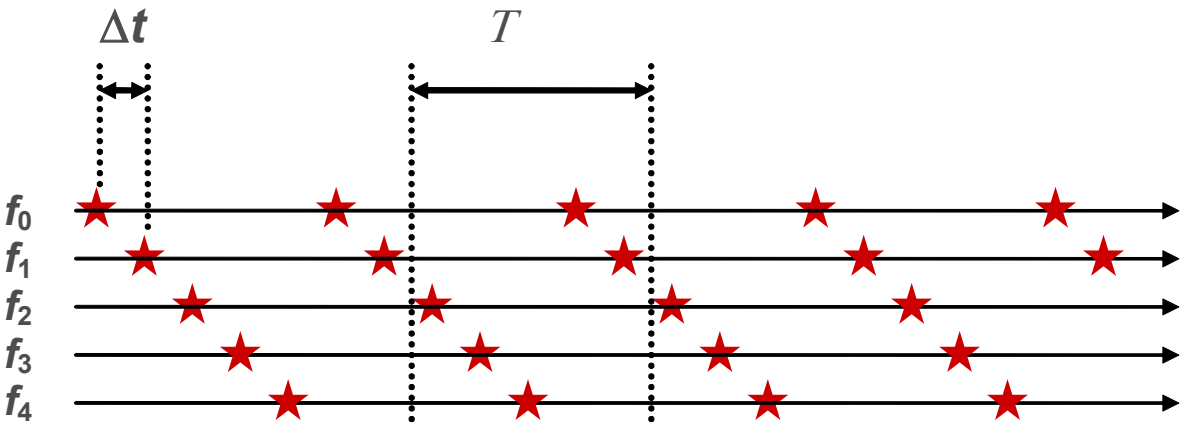
In transient analysis, lower limit of the time step is implicitly influenced by the circuit bandwidth. When the circuit is latent, or signals are changing slowly, the simulator takes larger steps to go faster without compromising accuracy. During transient noise analysis, the time step is limited by the noise bandwidth. Since noise sources are always active and are updated with constant time interval, time step becomes uniform. Therefore, transient noise analysis for circuits with long periods of latency will be significantly slower than normal transient analysis. For performance reasons, it does not make sense to set noise bandwidth much higher than the circuit bandwidth.

Furthermore, one undesired property of the random noise source is its discontinuity in time domain as it is updated with constant frequency. Discontinuity could potentially cause convergence problems during simulation. When this occurs, one solution is to update noise sources with frequency N times higher than noise bandwidth, and put the result through a moving average filter with time window equal to the inverse of the bandwidth.

$$\eta(t) = \frac{1}{N}\sum_{i=0}^{N-1}\eta(t - i\Delta t, t\Delta), \Delta t = \frac{1}{2NF_{max}}$$

In this way, noise signal discontinuities can be reduced below a given threshold, although at the cost of further reducing the transient time step. This is illustrated in Figure 4 on page 9. All sources have equal update time interval. *T* is the time window of the moving average filter. This technique should only be used when there is indeed convergence problem caused by noise injection.

Figure 4



Transient Noise Analysis Parameters in Virtuoso Spectre

Transient noise analysis has been implemented in Virtuoso Spectre circuit simulator as a new option in transient analysis. In order to activate device noise sources, the minimum user interaction is to set the noise bandwidth `noisefmax` parameter to a reasonable value, usually somewhat higher than the circuit bandwidth. The default of the `noisefmax` is zero, thus device noise sources are turned off during regular transient analysis.

The parameter `noisefmin` sets the lowest noise frequency of interest. For frequency dependent noise sources, spectral density will be zero above `noisefmax`, constant below `noisefmin`, and will follow the device noise equation in between. The value of `noisefmin` is adjusted automatically so that the ratio of `noisefmax`/`noisefmin` is an integer power of 2. The time update interval for the lowest frequency octave source is
1/`noisefmin`. This source will be never updated if the transient simulation stop time step is set smaller than that. Set stop time sufficiently large in order to sample low frequency noise correctly. The default of the `noisefmin` is `noisefmax`, thus only white noise sources are included by default.

The parameter `noisetmin` should be only used when the noise is so large that discontinuities in the noise signal cause convergence problems. Its value is adjusted so that `noisetmin*noisefmax` is an integer. It will reduce time step, but noise signals will become smoother.

`noisescale` is a multiplier factor on the actual device noise contribution. It is particularly useful to inflate the device noise contribution to be above the simulation numerical floor.

`noiseseed` is an important parameter if you want to reproduce a previous transient noise analysis run. Every time when you run transient noise analysis, different `noiseseed` will be chosen by Spectre. However, to reproduce the result of a previous transient noise analysis run, you can set the set the same `noiseseed` parameter.

Detailed description of these transient noise analysis parameters can be found in Spectre help page `spectre -h tran`, as shown below.

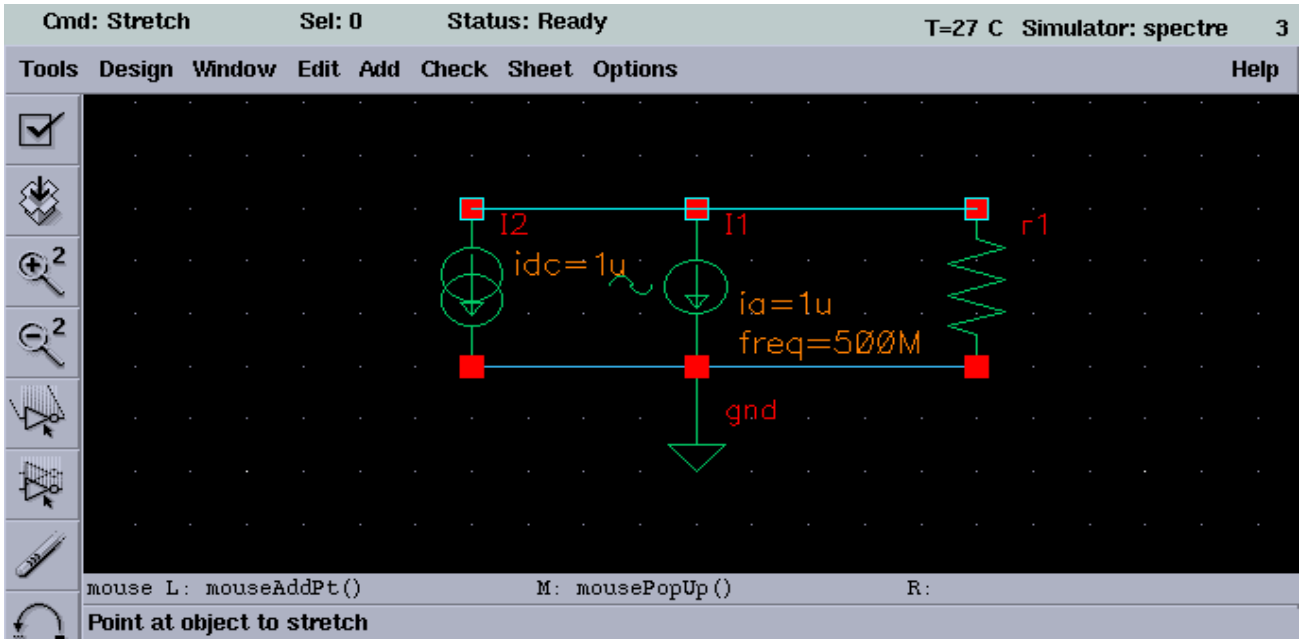
Transient noise parameters

49	`noisefmax=0 Hz`	The bandwidth of pseudorandom noise sources. A valid (nonzero) noisefmax turns on the noise sources during transient analysis. The maximum time step of the transient analysis is limited to 1/noisefmax.
50	`noisescale=1`	Noise scale factor applied to all generated noise. Can be used to artificially inflate the small noise to make it visible above transient analysis numerical noise floor, but it should be small enough to maintain the nonlinear operation of the circuit.
51	`noiseseed`	Seed for the random number generator. Should be positive integer. Specifying the same seed allows you to reproduce a previous experiment.
52	`noisefmin (Hz)`	If specified, the power spectral density of the noise sources will depend on frequency in the interval from noisefmin to noisefmax. Below noisefmin the noise power density is constant. The default value is noisefmax, so that only white noise is included by default, and noise sources are evaluated only at noisefmax for all models. 1/noisefmin cannot exceed the requested time duration of transient analysis.
53	`noisetmin (s)`	Minimum time interval between noise source updates. Default is 1/ noisefmax. Smaller values will produce smoother noise signals, but will reduce time integration step.

A Linear Resistor Circuit

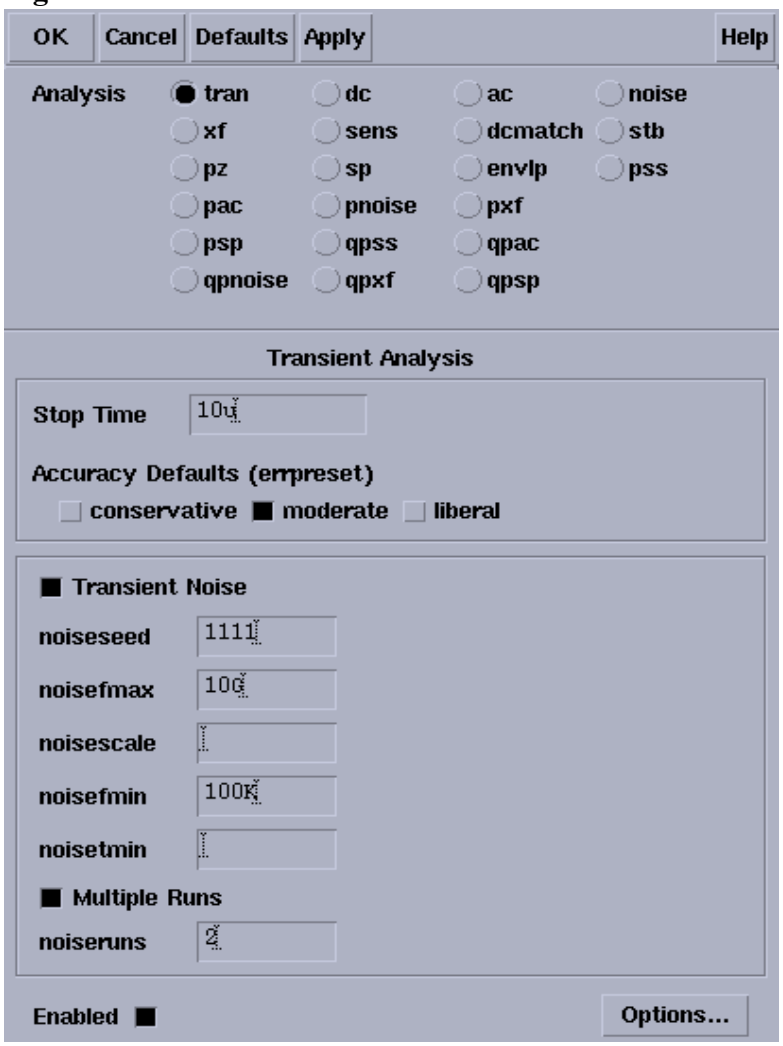
Here a simple linear resistor circuit is used to illustrate the basic concept of transient noise analysis and the Artist UI support inside ADE. The circuit contains a resistor, a 1uA dc source, and a sine source with frequency 500MHz and amplitude 1uA.

Figure 5



The resistor model includes both thermal and flicker noise. Model parameters are `rsh=1k` and `kf=10e-13`. Resistor instance parameters are `l=100u` and `w=1u`. You can specify the transient noise options in Virtuoso® Analog Design Environment in the Transient Analysis Setup window:

Figure 6

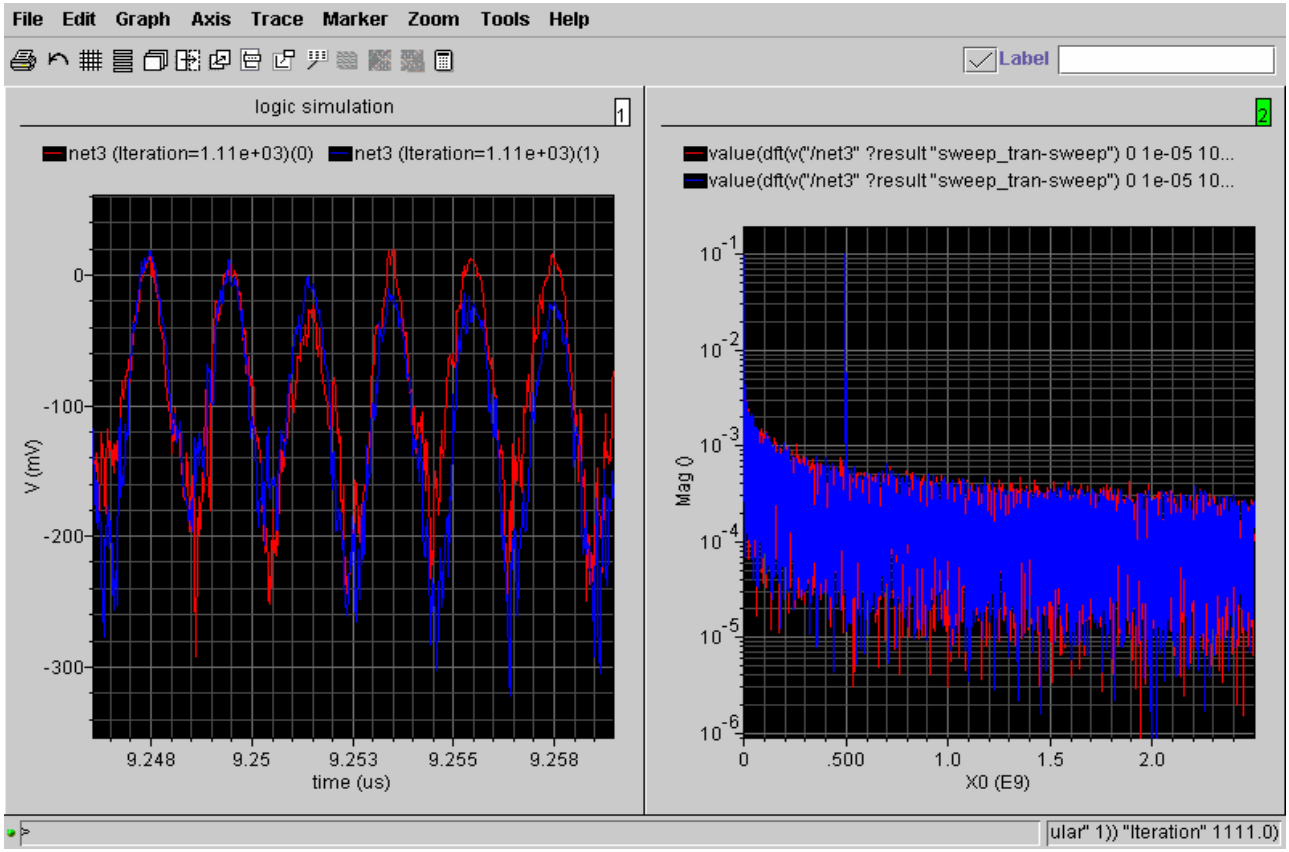


The netlist for this example is given below:

```
//flicker noise test

I1 (0 1) isource type=sine freq=500M ampl=1u
I2 (1 0) isource type=dc dc=1u
R1 (1 0) flikres l=100u w=1u
model flikres resistor rsh=1k kf=10e-13

parameters s=1111
sweep sweep param=s start=1111 step=1 stop=1112 {
    trnoi tran stop=10u noisefmax=10G noisefmin=100K noiseseed=s

}
```

The output of a transient noise analysis for a linear circuit is shown in Figure 7 on page 13. Red and blue curves are waveforms and their Fourier transforms obtained with different values of random seeds (s=1111, 1112).

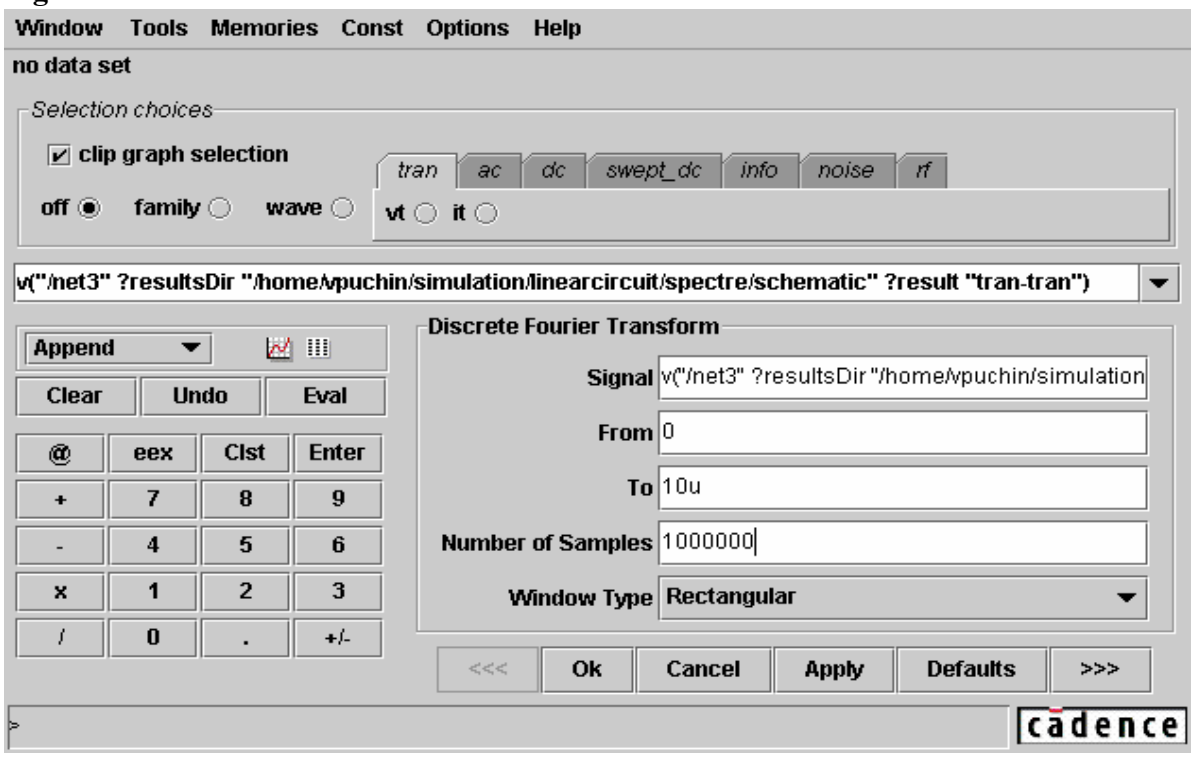
Figure 7



In the frequency spectrum, there is a signal peak at 500MHz, due to the sine source. Noise spectral density is decreasing as 1/f. Since this circuit is linear, the output waveform is a superposition of signal and noise. The transient noise analysis results can be easily verified using traditional small-signal analysis, and they do match very well.

The Fourier transform was done using the WaveScan calculator `dft` function with the following parameters:

Figure 8

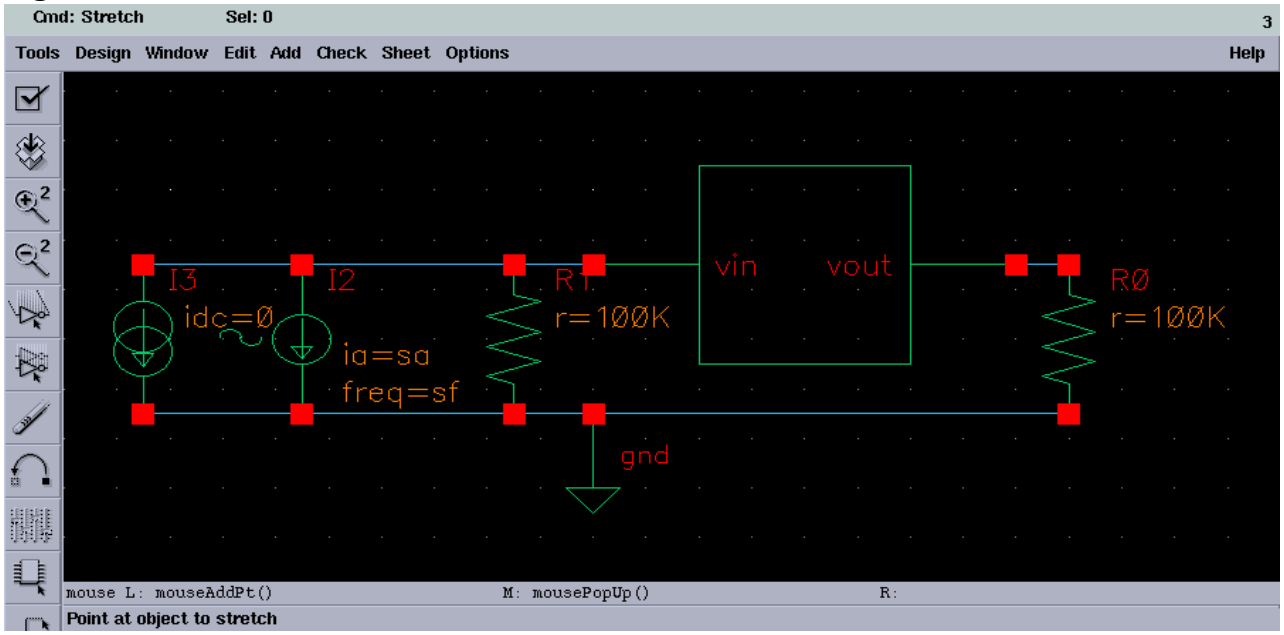


A Nonlinear Oscillator Circuit

The following testcase consists of a VCO with two noise sources connected to the input. This testcase is used to demonstrate how transient noise analysis can be used to measure period jitter in the time domain.

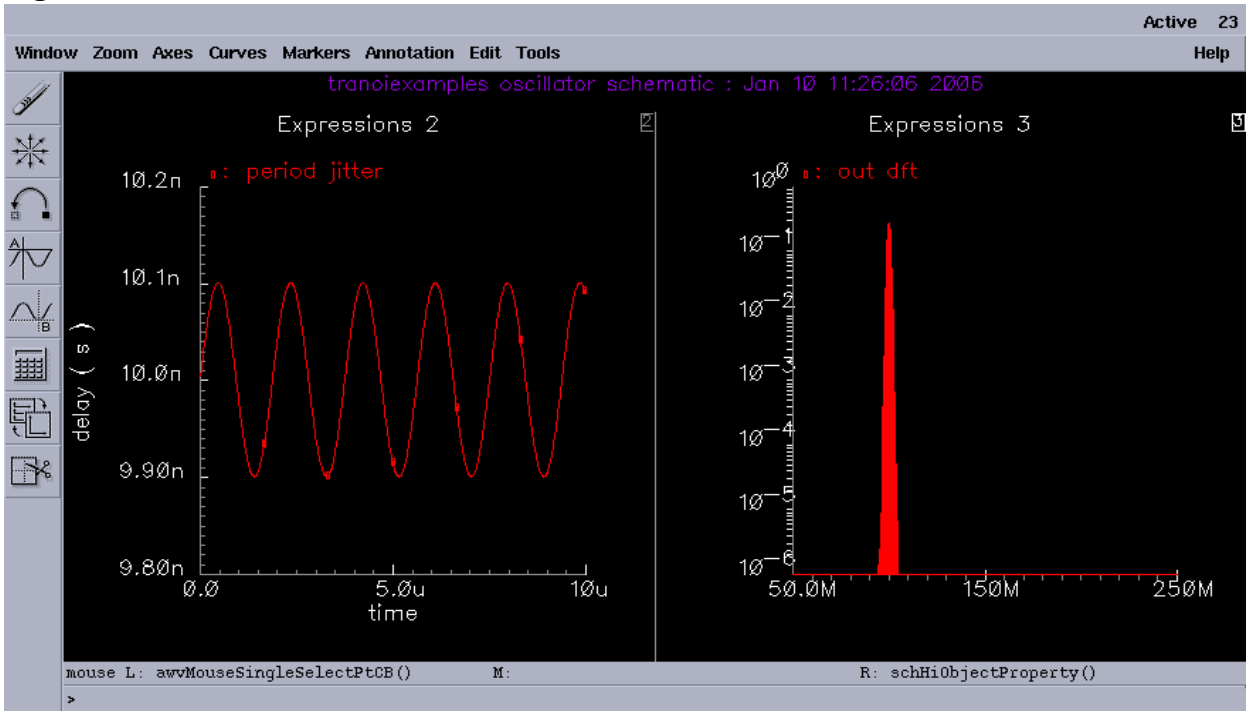
The VCO is a standard cell from `ahdlLib`, with amplitude `1V`, center frequency `100MHz`, and gain `10MHz/V`.

Figure 9



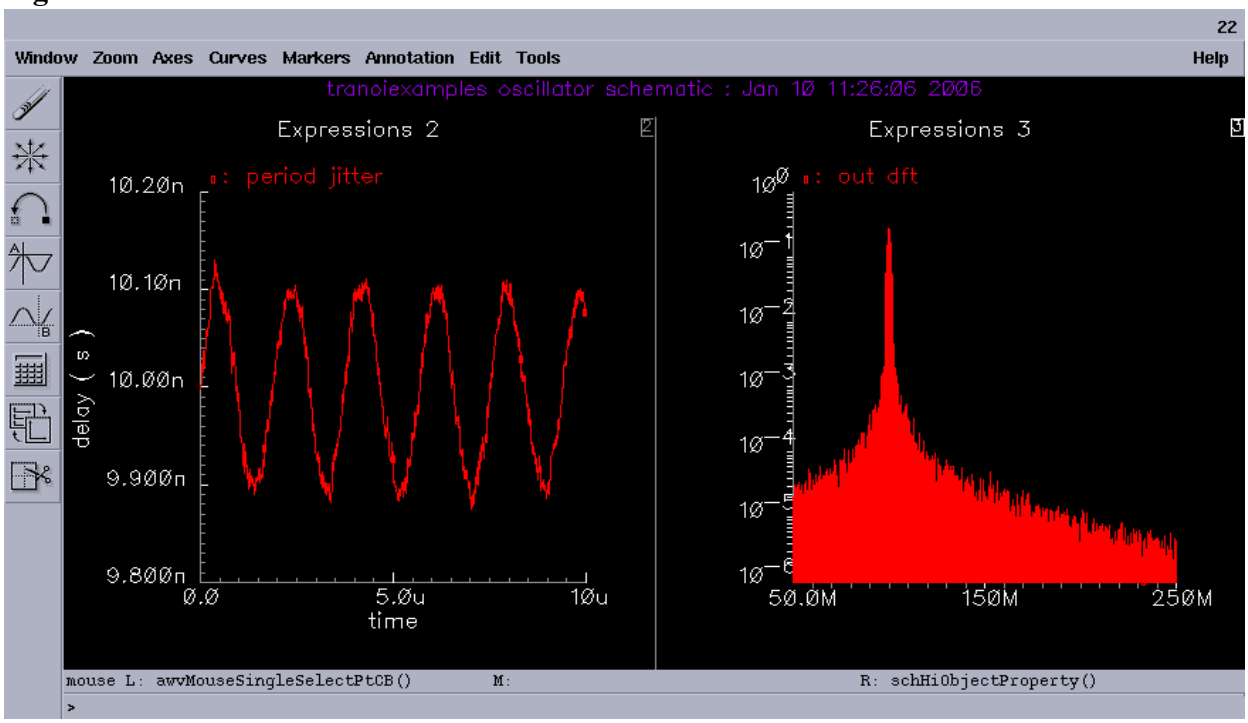
Sine source `I2` with frequency `533kHz`, and amplitude `1uA` models deterministic power supply noise. Flicker noise is modeled by zero current source `I3`, which has noise spectral density table: [100K 1e-20 1M 1e-21 10M 1e-22 100M 1e-23 1G 1e-24 10G 1e-25]. The output waveform is a frequency modulated sinewave. The figure shows period jitter plot and Fourier transform of output signal. When only deterministic noise source `I2` is on, period jitter is a sine function of time and spectrum has a single narrow band centered at `100MHz`.

Figure 10



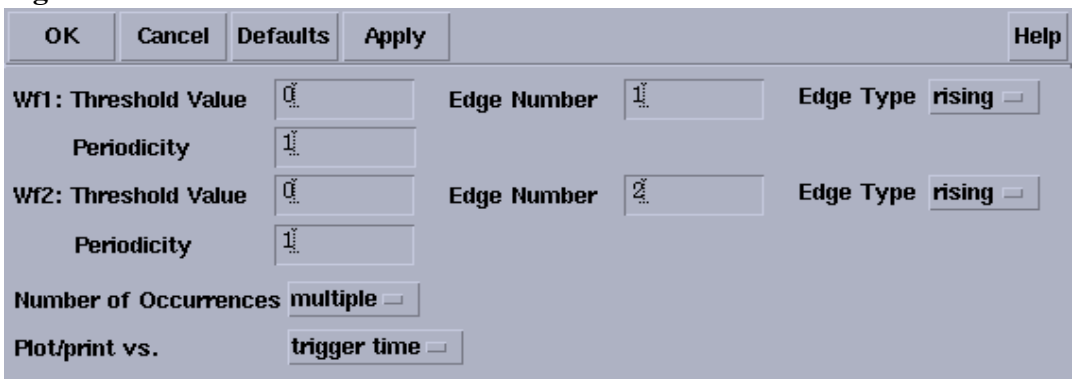
When both deterministic and random flicker noise sources are active, period jitter has a random component and spectrum has additional 1/f bands on both sides of the main peak.

Figure 11



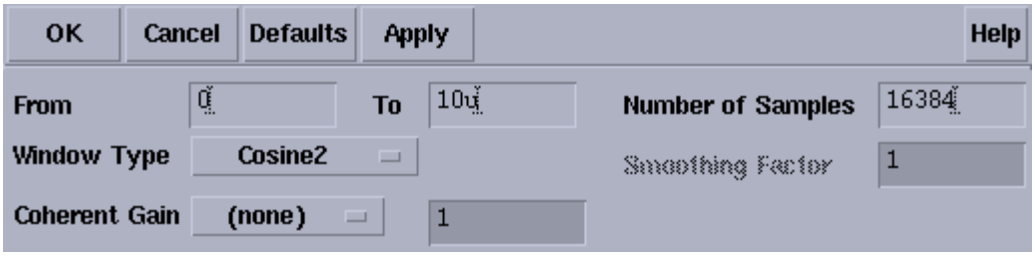
Period jitter was plotted using the WaveScan calculator `delay` function with the following parameters:

Figure 12



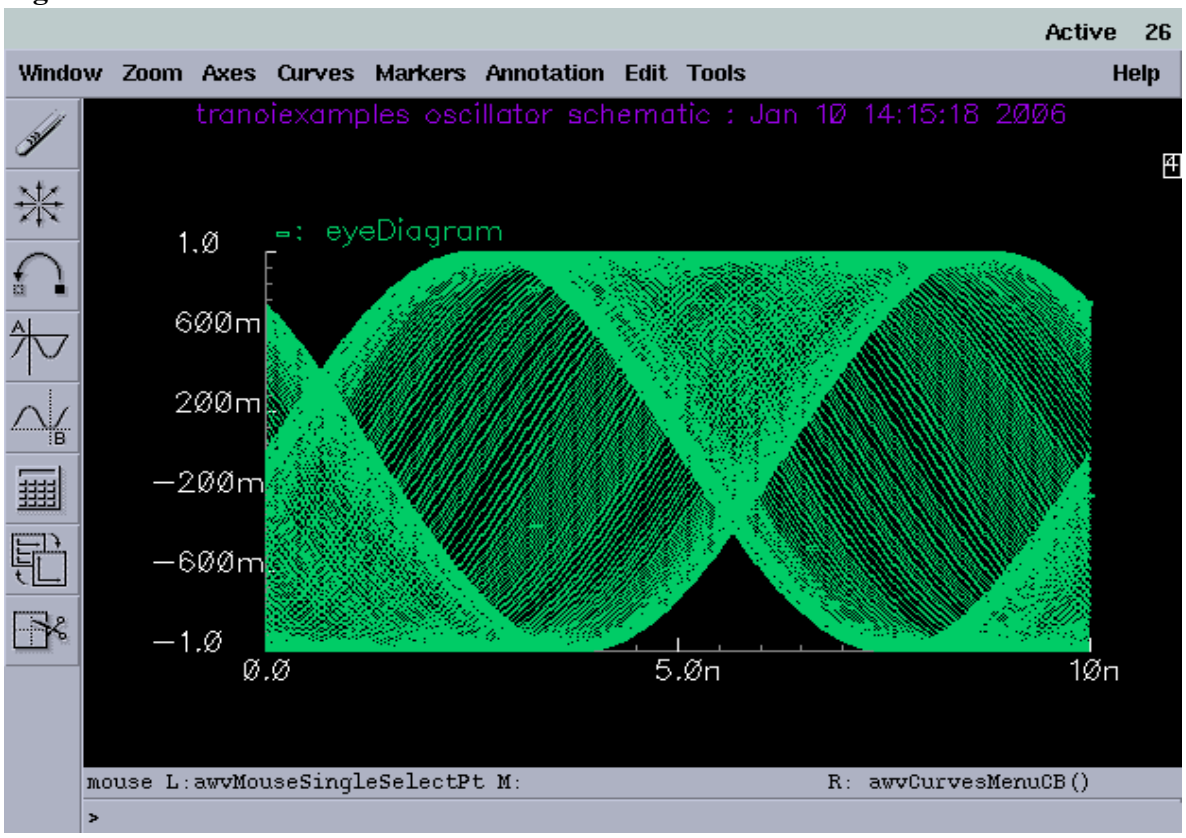
The `DFT` function with the following settings was used to obtain the Fourier transform:

Figure 13



The following figure shows an eye diagram of the output signal with period of `10ns`.

Figure 14



References

[1] P.Bolcato, R.Poujois, “A new approach for noise simulation in transient analysis”, Proc. IEEE Int. Symp. Circuits Syst., May 1992, pp. 887-890

[2] Joon-Jea Sung, Guen-Soon Kang, Suki Kim, “A transient noise model for frequency-dependent noise sources”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, V.22, No.8, 2003, p.1097.

[3] R.F.Voss, J.Clarke, "1/f noise in music: music from 1/f noise", Journal of Acoustic Society of America, v.63, No.1, 1978,