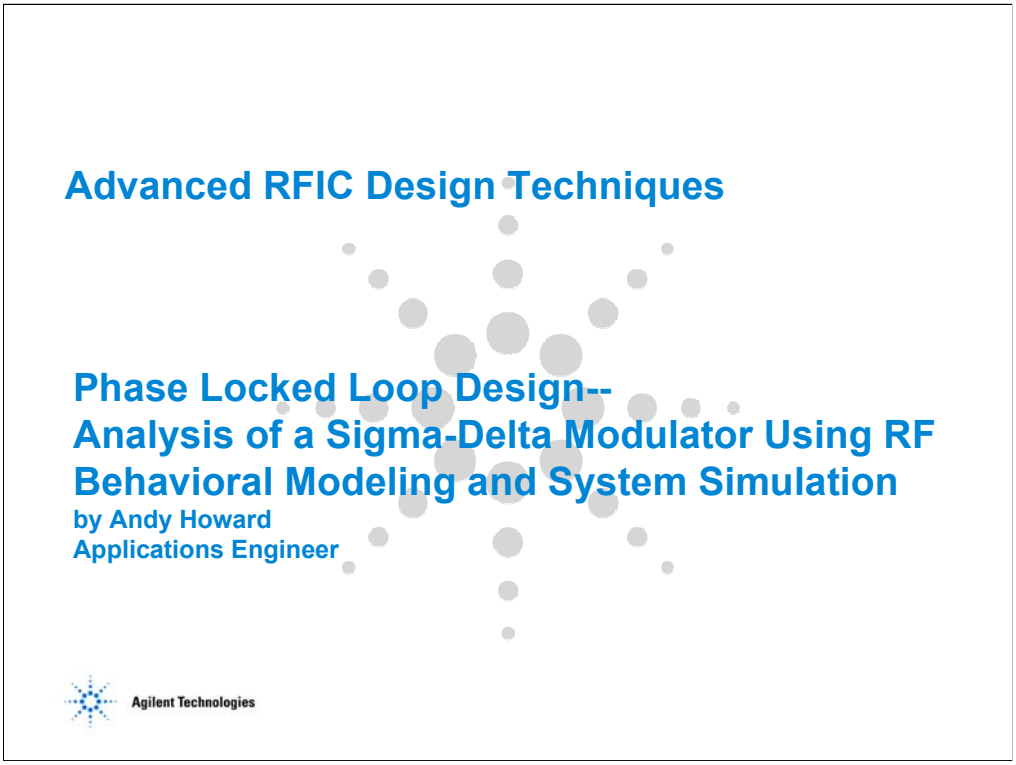


Welcome to this presentation. It is designed to give you an understanding of:

- Advanced Design System (ADS) and RF Design Environment (RFDE) Phase-locked loop (PLL) simulation capabilities.
- PLL component behavioral modeling in Agilent ADS and RFDE
- Agilent ADS and RFDE post-processing capabilities

Agilent Technologies

Introduction

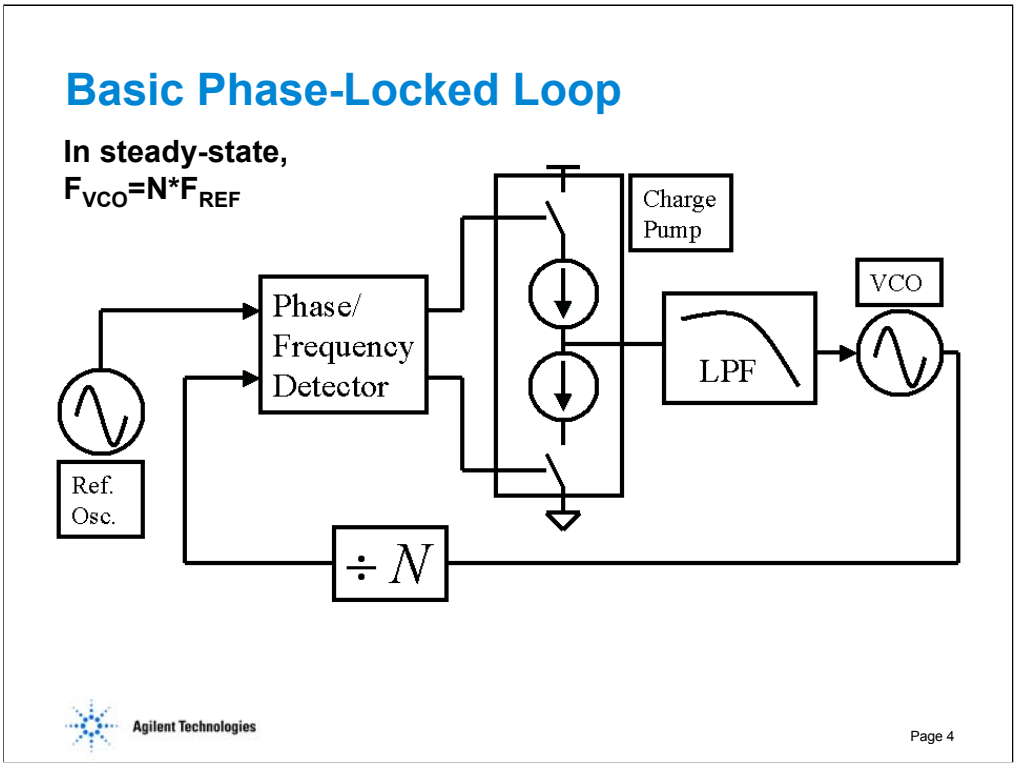
- **Agilent ADS and RFDE capabilities for simulating PLLs**
- **PLL component behavioral modeling**
- **Agilent ADS and RFDE post-processing capabilities**

Agilent Technologies

Page 2

Outline

- Basic phase-locked loop (PLL) operation
- Fractional-N PLL operation and simulation
- Behavioral modeling of a phase/frequency detector
- Simulating jitter in a transistor-level PFD and charge pump
- Behavioral modeling of a VCO/divide-by-N
- Modeling an accumulator with Ptolemy
- Sigma-delta modulators
- Simulating a PLL with a sigma-delta modulator
- Adding phase noise to the VCO



This phase-locked loop consists of a reference source, phase/frequency detector, charge pump, loop filter, VCO, and divider. If the divide ratio is a constant, then the loop will operate to force the VCO signal frequency to be exactly N times the reference signal frequency. The phase/frequency detector and charge pump act to output either positive or negative charge "pulses" depending on whether the reference signal phase leads or lags the divided VCO signal phase. These charge pulses are integrated by the loop filter to generate a tuning voltage. The tuning voltage forces the VCO frequency up or down, such that the reference signal and divided signal phases are synchronized.

Phase-locked loops are used as frequency synthesizers in many applications, where it is necessary to generate a precise signal frequency with low spurs and good phase noise. A VCO's signal frequency can be changed by varying the reference signal frequency or the divide ratio. Often, the reference signal is a very stable oscillator whose frequency cannot be varied. So the divide ratio is changed in integer steps to change the VCO frequency.

Problem with Basic Phase-Locked Loop

N can only have integer values, so

F_{VCO} =…, (N-2)* F_{REF} , (N-1)* F_{REF} , N* F_{REF} , (N+1)* F_{REF} , …

The smallest frequency change in F_{VCO} that can be made is 1*F_{REF}

What if you need finer frequency resolution?

Agilent Technologies

Page 5

One limitation with this type of phase-locked loop is that the VCO frequency cannot be varied in steps any smaller than the reference frequency. (Although you could put a 1/M divider between the reference signal and the phase/frequency detector, in which case the VCO output frequency would be N*Fref/M.) Due to mismatches in the PLL's charge pump and other factors such as the non-ideal behavior of phase/frequency detectors, even when the loop is locked, the charge pump still outputs small charge pulses which cause sidebands or spurs to appear in the VCO output spectrum, at offset frequencies equal to the reference frequency. So, for fine frequency resolution, you want a small reference frequency. But this will cause spurs to be generated at a smaller offset frequency from the VCO, meaning they will require that a narrower loop filter bandwidth be used to filter them. PLLs with narrower loop bandwidths have longer transient settling times (the time required to transition from one frequency to another) and such loops may not operate at the required speed. Reference 1 has a discussion of PLL settling time requirements. Also, the narrower the PLL's loop bandwidth, the less the VCO's phase noise is supressed.

Fractional-N Synthesizer Overcomes Frequency Resolution Problem

Periodically change divide ratio between N and N+1:

$$\text{Average } F_{VCO} = [T_N * N * F_{REF} + T_{N+1} * (N+1) * F_{REF}] / (T_N + T_{N+1})$$

$$= [N + (T_{N+1}) / (T_N + T_{N+1})] * F_{REF}$$

$$= [N + \text{Fraction}] * F_{REF}$$

Divide ratio versus time:

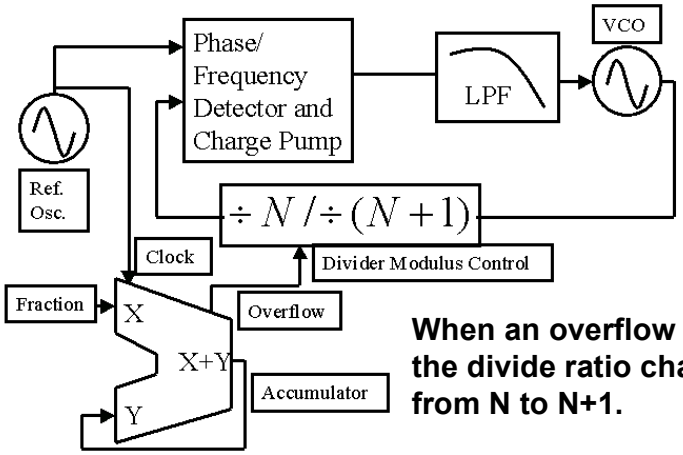
Agilent Technologies

Page 6

An approach to achieving finer frequency resolution than the reference frequency is fractional-N synthesis. In this approach, the divide ratio is varied periodically between two integer values.

Fractional-N Synthesizer Phase-Locked Loop

In steady-state, average F_{VCO}=(N+Fraction)*F_{REF}



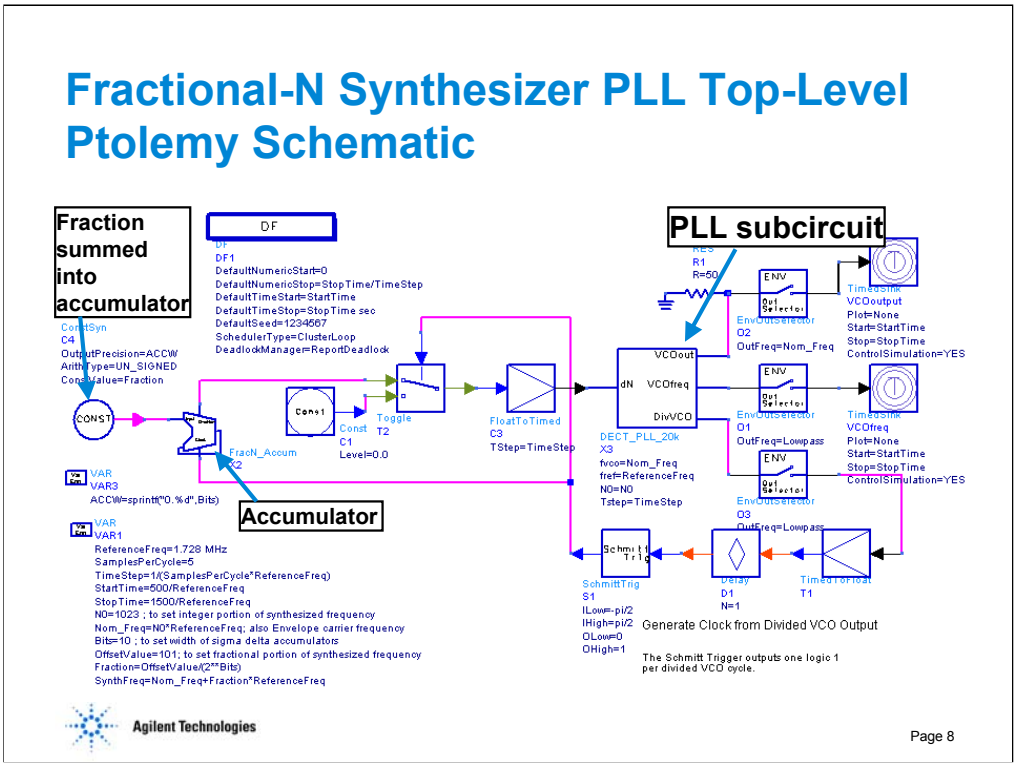
When an overflow occurs, the divide ratio changes from N to N+1.

Agilent Technologies

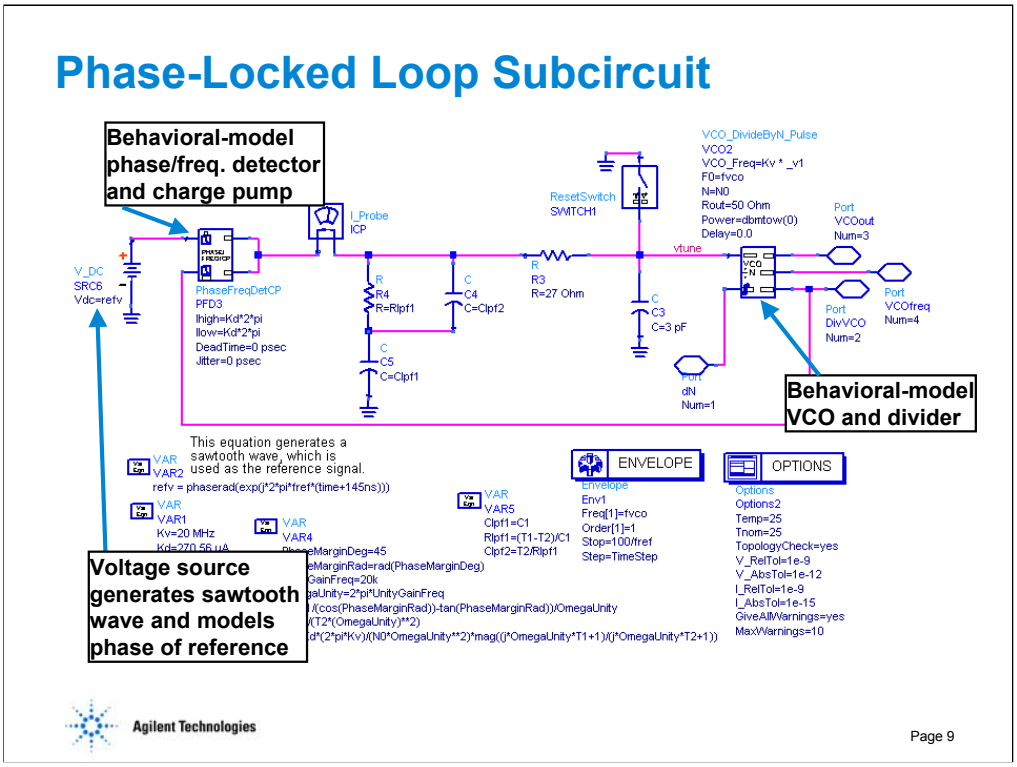
Page 7

This type of PLL can be modeled (as described in reference number 2, at the end of this presentation) using an accumulator that sums the desired fraction to itself each reference clock cycle. While the accumulator is not overflowing (its accumulated sum is less than its capacity) the overflow output is 0, and the divider divides by N. When the accumulator reaches its capacity, it overflows, and the divide ratio is set to N+1. If the desired fraction is about 0.1, then the accumulator will only overflow about once every 10th reference clock cycle. If the desired fraction is about 0.5, then the accumulator will overflow about every other reference clock cycle.

With fractional-N PLLs, the signals at the input to the phase/frequency detector are not at the same frequency. The reference signal is at Fref, and the divided VCO signal is at (1 + fraction/N)*Fref. Referred to the VCO, this frequency difference means that the phase of the VCO advances at a rate of "fraction" radians per reference clock cycle faster than a signal at frequency N*Fref. The accumulator sums this fraction once per reference clock cycle, so it accumulates at the same rate that the VCO phase difference advances. An accumulator overflow occurs at the same time the VCO phase difference (relative to a signal at N*Fref) reaches 2pi radians. When the accumulator overflows, the divide ratio is increased to N+1 for one cycle. This subtracts 2pi/N radians from the divided VCO signal, so the phases of the two signals at the input to the phase/frequency detector are again equal. The phase difference between the two signals at the input to the phase/frequency detector should increase and be reset to zero at the same rate that the accumulator sum increases and overflows.



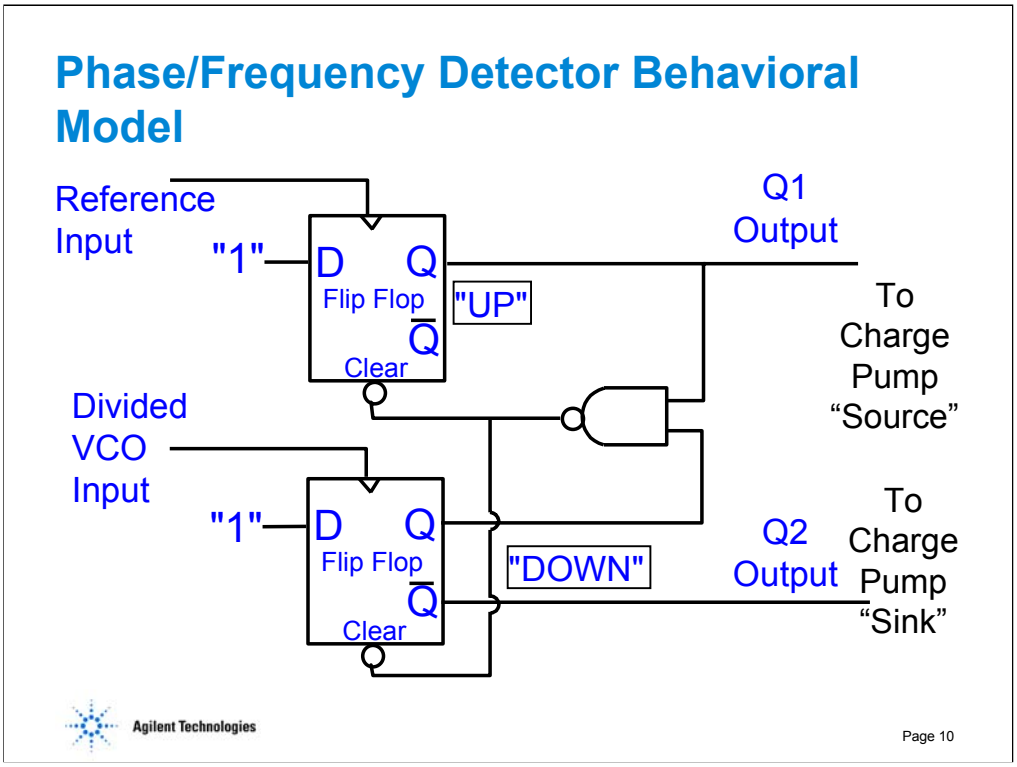
Co-simulation allows numeric processing to be included in simulations along with traditional circuit simulators that solve Kirchoff's Current Law equations. On this top-level schematic, various parameters are set, such as the reference frequency, the simulation time step, the bit width of the accumulator, the nominal divide ratio, and the fraction. The desired fraction is set equal to the constant value that is the input to the accumulator. The five, timed sink components collect data to be displayed after the simulation.



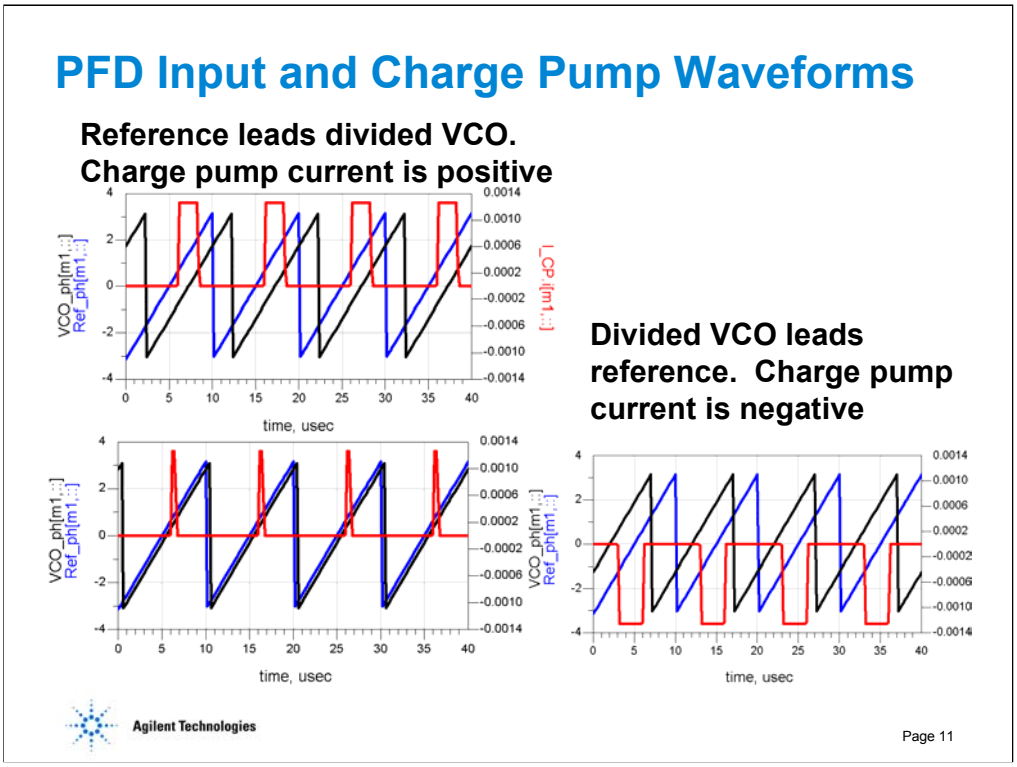
The design of this PLL, including choosing the phase margin, which determines the component values, is described in the Franceschino paper (reference 1). The unity-gain frequency may be set by a single variable from which all the filter component values are computed. For this paper, the unity-gain frequency is set to 20 kHz.

While designers would like to see phase-locked loops modeled at the transistor level, this is usually impractical unless you are willing to tolerate days-long simulation times. PLLs are so difficult to simulate because there are widely-varying frequencies present and logic circuits that require a small simulation time step. Transient responses can be milliseconds long, which when coupled with a small simulation time step can mean millions of time points are required.

To overcome these simulation difficulties, we recommend that users extract behavioral models of various components, such as the VCO, phase/frequency detector, and frequency divider. Then simulate the closed-loop PLL with behavioral models much more quickly and efficiently.

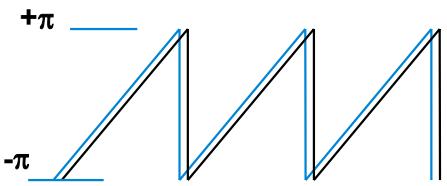


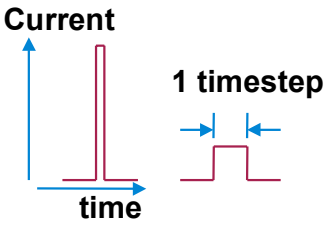
This is the equivalent circuit of the phase/frequency detector behavioral model. It models a PFD made with dual flip flops, and it outputs digital pulses, the duration of which depend on the phase difference between the reference source and the divided VCO signal. In the PhaseFreqDetCP model used in these simulations, a charge pump is included, which converts the digital pulses into either source or sink current pulses.



This figure shows the input reference and divided VCO signal waveforms and the corresponding charge pump current pulses. The polarity of the charge pump current changes, depending on which of the two PFD input signals is leading the other.

Why are Reference and Divided VCO Signals Sawtooth Waves?





Ideally, charge pump current pulse width is equal to time difference between PFD input signals. But in simulation, this width must be a multiple of the timestep. Envelope uses interpolation to get finer resolution than the timestep. Sawtooth waves are easiest to interpolate.

Ideal current pulse (can't be simulated)

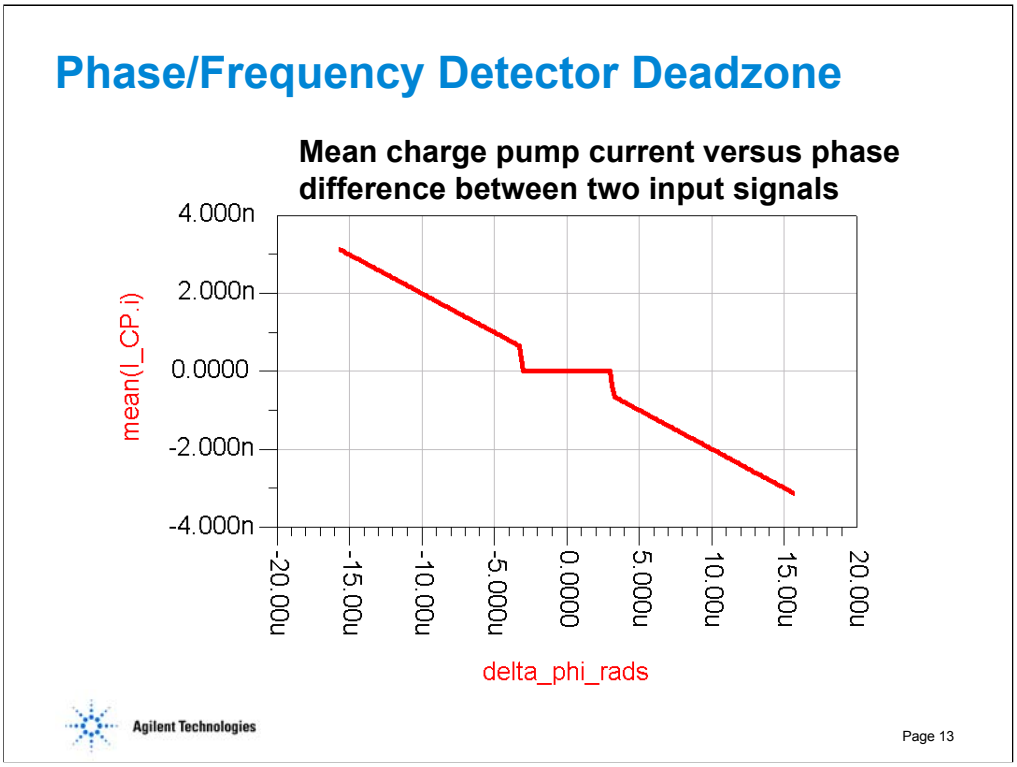
Current pulse, 1 timestep wide, amplitude reduced to give same area as ideal pulse

Agilent Technologies

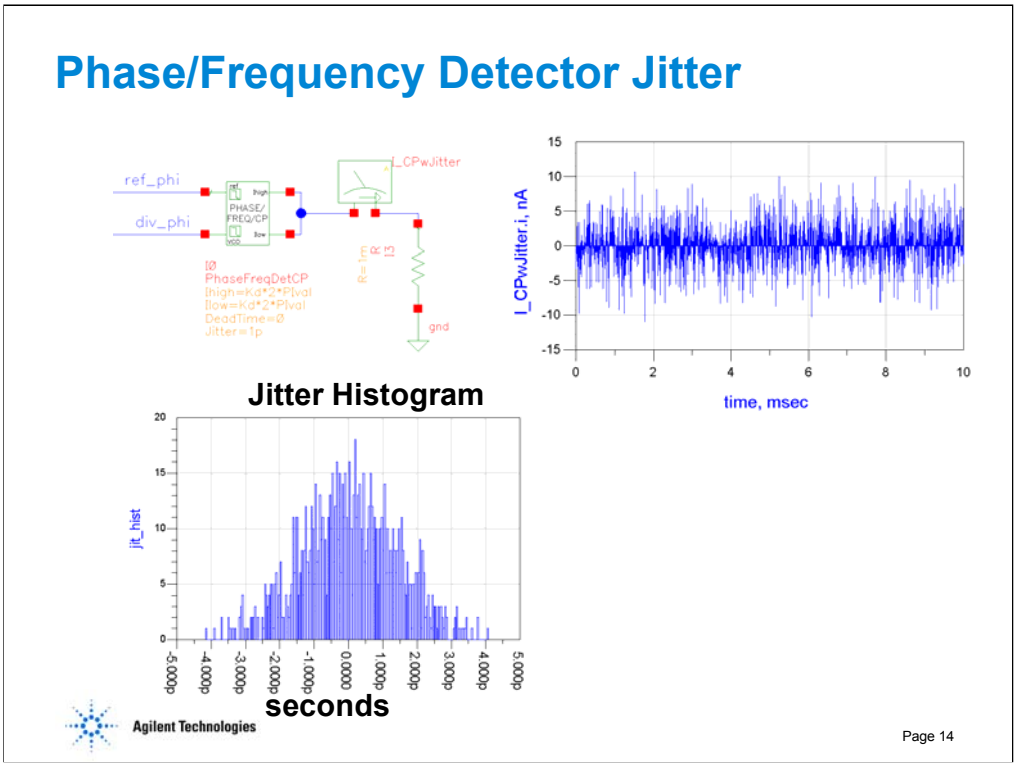
Page 12

Why do we recommend that the input signals to the PFD be sawtooth waves that are the phase, in radians, of these signals, rather than square waves?

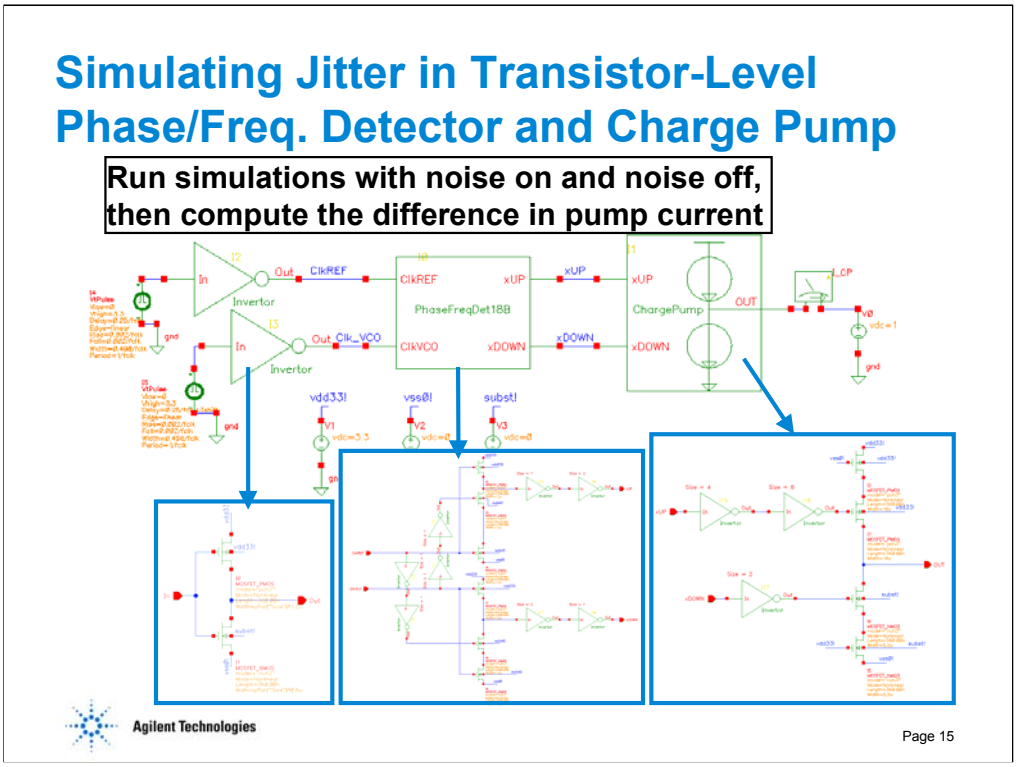
Phase/frequency detectors in real PLLs are asynchronous, meaning that their outputs depend on when the inputs transition through a specified logic threshold, not in response to a periodic clock signal. These circuits are time-consuming to simulate (at the transistor level) because very short clock edges (<100 psec.) must be captured even though the overall simulation time may be milliseconds long. This phase/frequency detector model uses interpolation to determine the input signal transition time points with finer resolution than the time step, and actually modulates the amplitude of the output signal pulses to overcome the fixed time step limitation of the simulator. The phase/frequency detector operation is described in detail in reference 3. To more accurately compute the phase difference between the signals at the input to the phase/frequency detector, these signals are sawtooth waves that model the phase of the reference source and the divided VCO signal. Interpolation also works pretty well if sine waves are used, but square waves are not recommended.



The phase/frequency detector model also has a dead zone, meaning that if the time difference between the two input signals is less than a user-specified dead time, then the charge pump does not output any current. If something is not done to bias the PLL out of the dead zone (such as adding some DC offset current into the loop), then the loop will not be able to attenuate spurious signals that might be introduced into it.



You can add timing jitter to the PhaseFreqDetCP model. This will introduce noise in the charge pump current, which would correspond to a jitter variation in the input signals to the PhaseFreqDetCP model. This jitter parameter can be extracted from a time-domain noise simulation of a transistor-level phase/frequency detector.

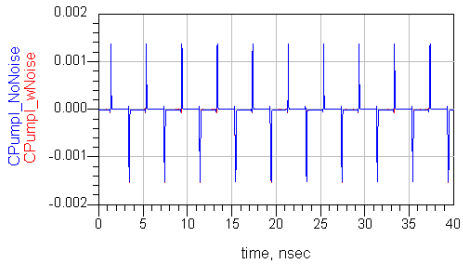


This shows a set-up for simulating the jitter of a transistor-level phase/frequency detector and charge pump. To determine the noise current, two simulations are run over multiple cycles of the reference clock, one with the noise on and the other with it off. The difference between the two currents will be the noise. The phase difference between the two input signals is set close to zero but out of the dead zone.

Reference for this charge pump design:

H. O. Johansson, "A Simple Precharged CMOS Phase Frequency Detector," *IEEE Journal of Solid-State Circuits*, pp 295-299, Feb., 1998.

Charge Pump Noise Current



Charge pump noise current is the difference between current with noise on and off

Eqn NoiseBandwidth=1/(2*MaxTimeStep[0,0])

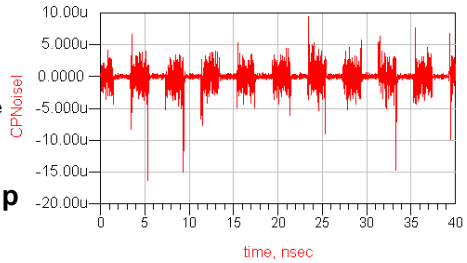
NoiseBandwidth
50.00G

Eqn CPNoiseI=CPumpI_wNoise-CPumpI_NoNoise

Eqn RMS_CPNoiseI=stddev(CPNoiseI)

RMS_CPNoiseI
1.537u

RMS charge pump noise current



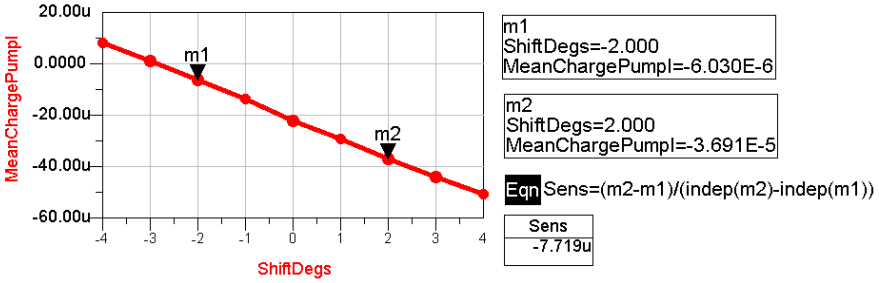
Agilent Technologies

Page 16

The charge pump current pulses with the noise on and off are indiscernible, at the scale shown. However, subtracting one from the other does indicate a difference, which is the noise current. The RMS value of the noise current is computed by taking the standard deviation of this difference. If the noise bandwidth, which is set on the transient simulation controller, is increased by reducing the maximum simulation time step, then the RMS noise current will also increase, unless there is something in the simulation (such as filtering) to limit the bandwidth of the noise.

Simulate Phase/Frequency Detector - Charge Pump Sensitivity

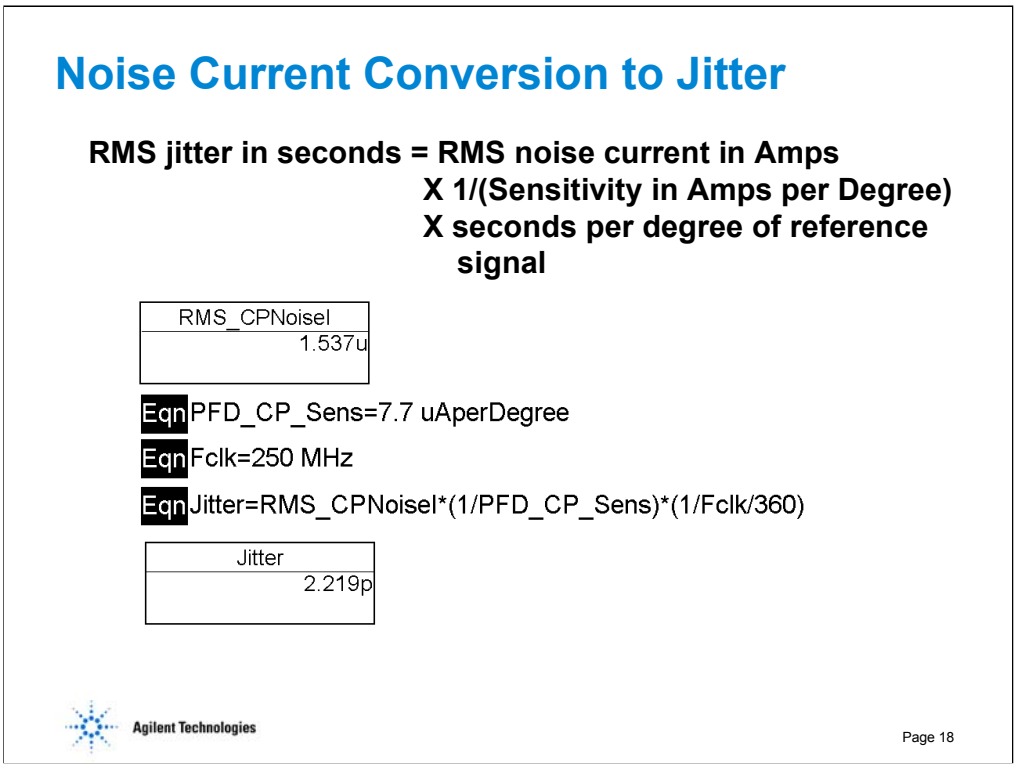
Step phase shift "shiftdeg" between PFD input signals.
Plot mean charge pump current versus this phase shift.
Slope of this plot is sensitivity in Amperes/degree.



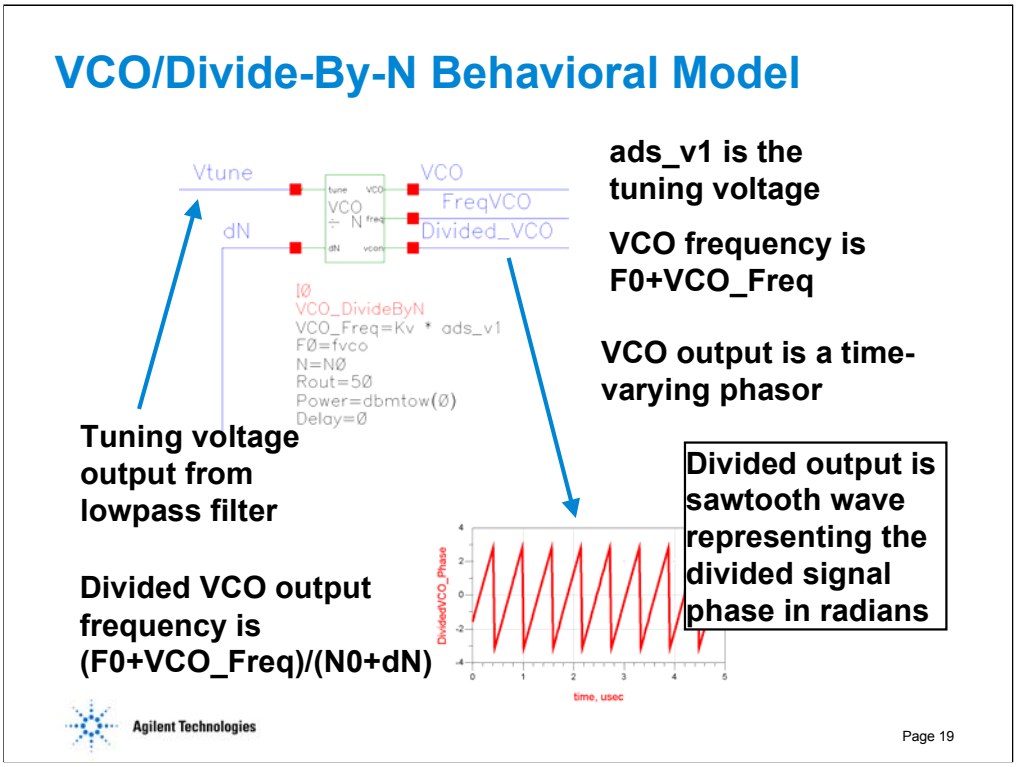
Agilent Technologies

Page 17

To convert the RMS noise current at the charge pump output to a jitter value at the phase/frequency detector input, we have to determine the sensitivity, in Amperes per degree, of the phase/frequency detector and charge pump. To determine this, in a separate simulation, we sweep the phase difference between the two input signals to the PFD and plot the average charge pump current versus this phase difference. The sensitivity is computed as the slope of a straight line drawn between the two markers.

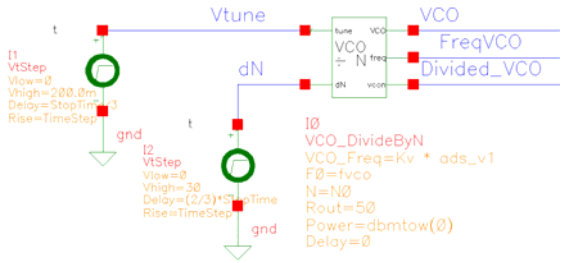


This shows the calculation to get RMS jitter in seconds at the input of the PFD.

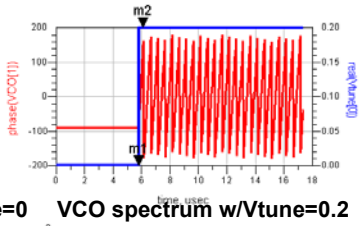


This behavioral model combines the VCO and divide-by-N together. (Separate VCO and divider behavioral models exist, also.) Combining them together makes transient simulations much more efficient, because you only need a time step small enough to adequately sample the tune voltage or the divided VCO signal, which is usually at a frequency orders of magnitude lower than the VCO. The dN input is a voltage versus time, which is the change in divide ratio. The resulting divide ratio is N0 + (the voltage at dN). So if the dN input voltage has increases from 0 to 3 Volts, the divide ratio also increases from N0 to N0+3. In the simulations shown later, the output from the sigma-delta modulator will be applied here.

VCO/Divide-By-N Behavioral Model

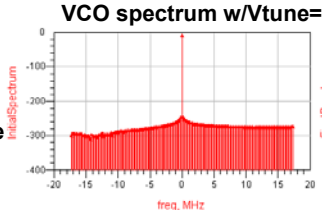


Step in Vtune, and VCO phase indicating 2 MHz increase in frequency

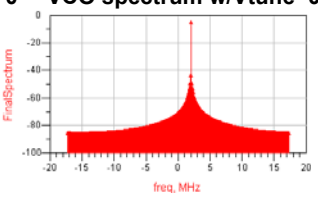


0.2 V step in Vtune forces VCO freq. to increase by 0.2 X 10 MHz

VCO spectrum w/Vtune=0



VCO spectrum w/Vtune=0.2



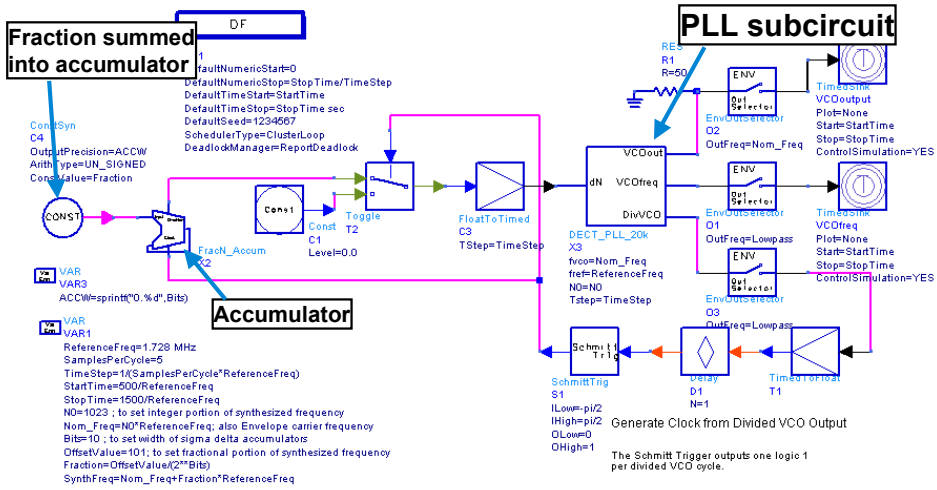
For VCO_DivideByN_Pulse component, dN inputs must be clocked.

Agilent Technologies

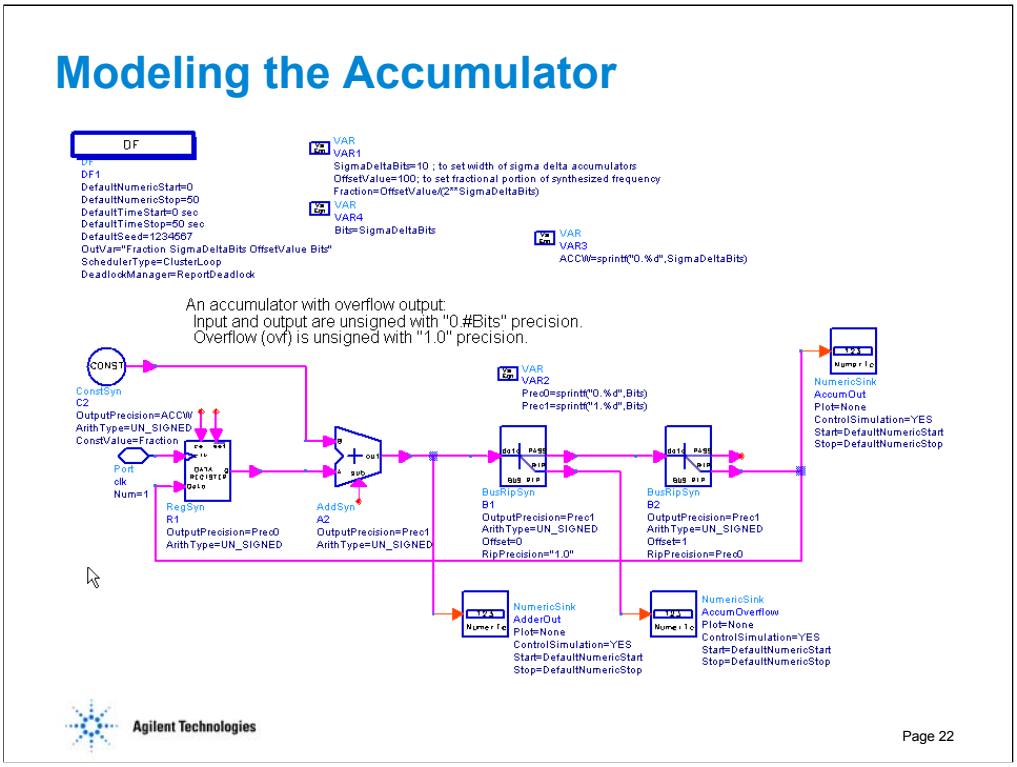
Page 20

This is a simple simulation to show what the VCO/Divide-By-N model does. For the first part of the simulation (10 cycles of the reference clock), the tune voltage and the dN input are 0. So the VCO signal is at fvco, which is N0*ReferenceFreq. Note that 0 Hz on the spectral plots corresponds to the fundamental analysis frequency on the Envelope controller. The phase of the undivided VCO is constant, which is as expected, since its frequency is not changing. After 10 cycles of the reference clock, the Vtune input is stepped from 0 to 0.2 Volts. Because the VCO's Kv is 10 MHz/Volt, theVCO's frequency increases by 2 MHz, as shown in the lower right plot. Also, the phase of the VCO output phasor is increasing at a rate of 360 degrees per 0.5 usec., which is 2 MHz. After 20 cycles of the reference clock, the dN input is increased from 0 to 30 Volts, which increases the divide ratio from 1023 to 1053. But this only affects the divided VCO output.

Back to the Fractional-N Synthesizer PLL Top-Level Ptolemy Schematic



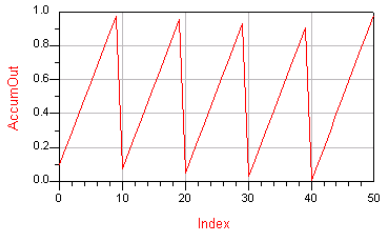
This is a duplicate of the figure on page 8. We have gone over some of the behavioral models in ADS and RFDE, as well as their extraction, and are ready to proceed with the PLL simulation, after a discussion of some of the Ptolemy modeling.

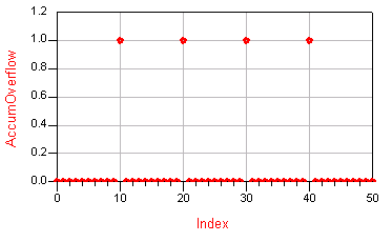


The accumulator consists of an adder, two bus rippers and a data register. The accumulator continuously adds a constant (in this case the fraction) to itself, generating an accumulating sum. The arithmetic precision setting of the output of the first bus ripper is set to 1.0, which means that it only outputs a 1 if the adder output is >1. Otherwise, it outputs a 0. This is the accumulator overflow output. The other bus ripper feeds back to the input of the adder the fractional part of the adder's output. So, for example, if the desired fraction to be summed is 100/(2**10) = 0.097656..., then after 10 summations, the adder output will be 1.0742187, the overflow will be 1, and the amount fed back to the input of the adder will be 0.0742187. With this simulation, you can easily change the number of bits used (precision) of the summation.

Accumulator Simulation Results

Fraction that is summed is 100/(2^{10}), so the accumulator overflows about once every 10 clock cycles





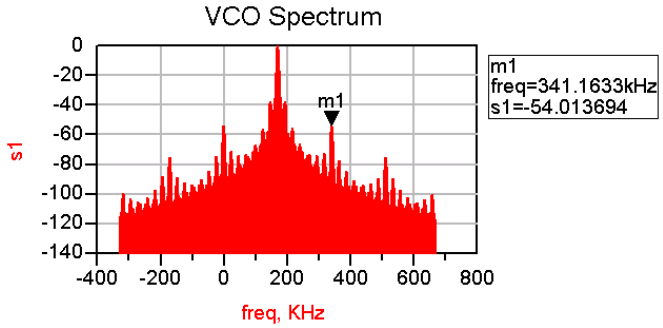
Index	AccumOut	AccumOverflow	AdderOut
0	0.09765625000	0.00000000000	0.09765625000
1	0.19531250000	0.00000000000	0.19531250000
2	0.29296875000	0.00000000000	0.29296875000
3	0.39062500000	0.00000000000	0.39062500000
4	0.48828125000	0.00000000000	0.48828125000
5	0.58593750000	0.00000000000	0.58593750000
6	0.68359375000	0.00000000000	0.68359375000
7	0.78125000000	0.00000000000	0.78125000000
8	0.87890625000	0.00000000000	0.87890625000
9	0.97656250000	0.00000000000	0.97656250000
10	0.07421875000	1.00000000000	1.07421875000
11	0.17187500000	0.00000000000	0.17187500000
12	0.26953125000	0.00000000000	0.26953125000

Agilent Technologies

Page 23

This shows the sawtooth waveform of the accumulator output and the accumulator overflow bit switching from 0 to 1 about once every 10 times a summation occurs.

Fractional-N Simulation Results



0 Hz corresponds to N0*ReferenceFreq = 1023*(1.728 MHz)
= 1.767744 GHz

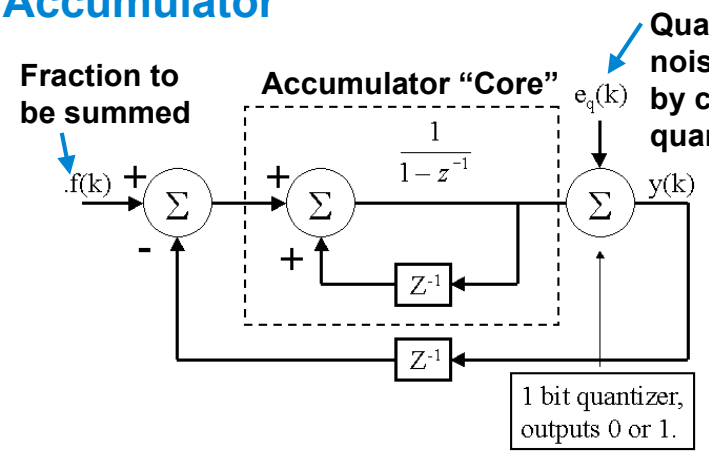
Fraction = 101/(2^{10}) = 0.098633
Synthesized frequency = (N0 + Fraction)*ReferenceFreq
= 1.767744 GHz + 0.098633*1.728 MHz
= 1.767744 GHz + 170.438 kHz

Agilent Technologies

Page 24

The simulation set-up has the reference frequency (Fref) specified at 1.728 MHz, the nominal divide ratio (N0) set to 1023, the number of bits used in the accumulator set to 10, and the fraction is defined to be 101/(2^{10}). This means the VCO frequency should be (on average) = N0.fraction * Fref = 1.767914 GHz. The spurs at fraction*Fref offset from the synthesized signal are clearly visible.

Using a Sigma-Delta Modulator as an Accumulator



When accumulator core overflows, 1-bit quantizer outputs a 1.

Agilent Technologies

Page 25

In the fractional-N PLL, the desired fraction, is converted to a sequence of 1's and 0's (1 when the accumulator overflow bit is set, and 0 when it is clear.) This could be considered a coarse analog-to-digital conversion, using a 1-bit A-to-D converter.

As described in references 2 and 3, an accumulator may be considered a simple sigma-delta modulator.

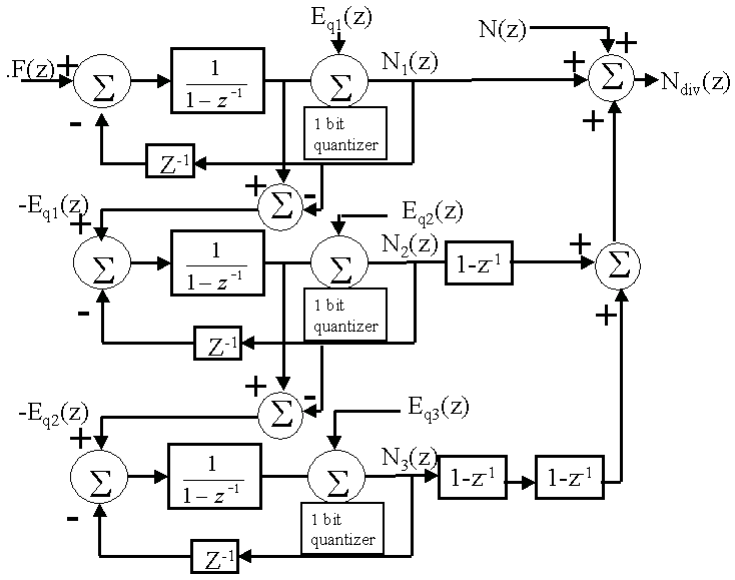
The $1/(1-z^{-1})$ block implements the basic accumulator operation. The 1-bit quantizer outputs a 0 while the accumulator has not overflowed, and a 1 when it overflows. The overflow quantity is subtracted from the input, which in effect just keeps the fractional part of the accumulated sum in the accumulator when it overflows.

Sigma-Delta Modulator Z-Domain Equation

$$Y(z) = .F(z) + (1 - z^{-1})Eq(z)$$

The quantization noise, Eq(z), is high-pass filtered,
(let $z = e^{j\omega}$ then $1 - z^{-1} = 1 - e^{-j\omega} \approx j\omega$
for ω small)
if .F(z) is sufficiently random.
But the fraction is constant, so the quantization noise varies periodically, generating spurs.

Using a 3-Stage Sigma Delta Modulator



Agilent Technologies

Page 27

To attain a divide ratio with quantization noise that has a high-pass-shaped frequency response and that does not suffer from the periodicity (and consequently the spurs) of a single-accumulator sigma-delta modulator, a multi-stage architecture shown here and described in references 2 and 3 was simulated.

Conceptually, the quantization noise from the first-stage sigma delta modulator becomes the input to the second-stage sigma delta modulator. The quantization noise from the second-stage sigma delta modulator becomes the input to the third-stage sigma delta modulator. The differentiators ($1-z^{-1}$ blocks) connected at the outputs of the second- and third-stage sigma delta modulators are necessary for canceling the noise from the previous stages at the final output summation.

The noise from the third stage is not cancelled but it is sufficiently random and it is high-pass filtered by a $(1-z^{-1})^3$ term.

3-Stage Sigma-Delta Modulator Equation

Z-domain equation for frequency:

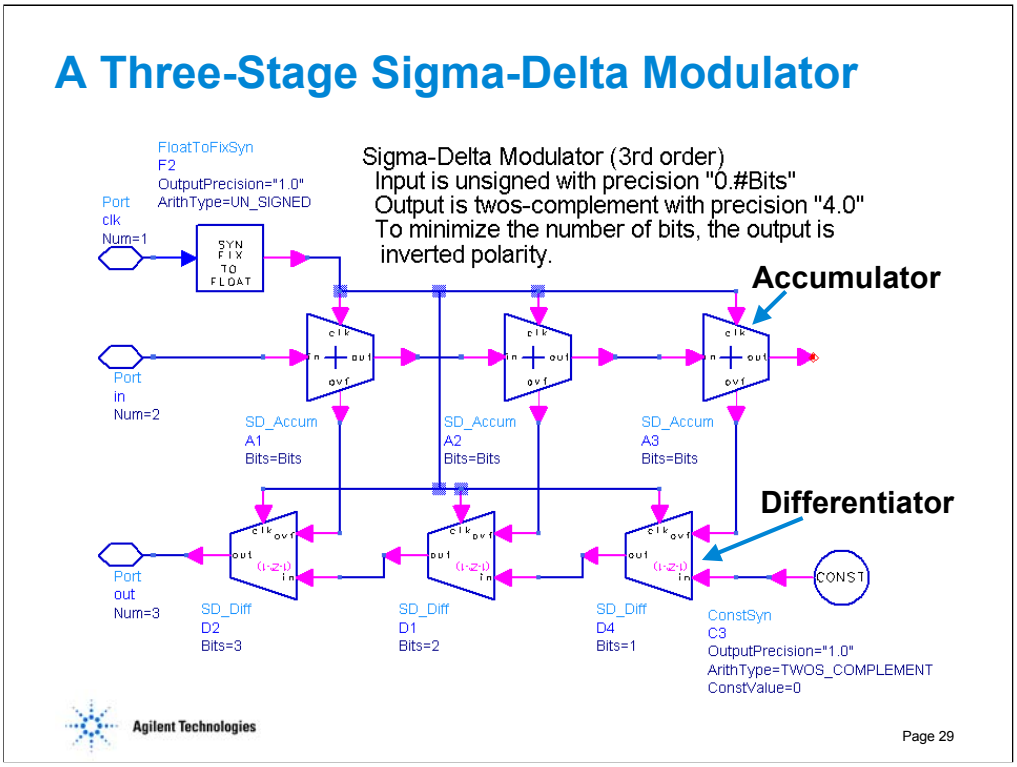
$$F_{out}(z) = N.F(z)F_{ref} + (1 - z^{-1})^3 F_{ref} E_{q3}(z)$$

3rd-stage quantization noise is more random than 1st, and this noise has a more high-pass shape

Agilent Technologies

Page 28

$E_{q3}(z)$ is the quantization noise of the third modulator stage (References 2 and 3) This equation may be derived easily from the block diagram in the previous figure. This equation gives frequency noise, whereas phase noise is of more interest to frequency synthesis applications. An equation for the phase noise as a function of offset frequency (f) and the number of modulator stages, (m), due to the shaping of the quantization noise is equation 12 in reference 2. However, this does not predict the overall noise performance of the PLL, since this will depend on the loop bandwidth, the free-running VCO's phase noise, noise from the phase detector and divider, and so on.

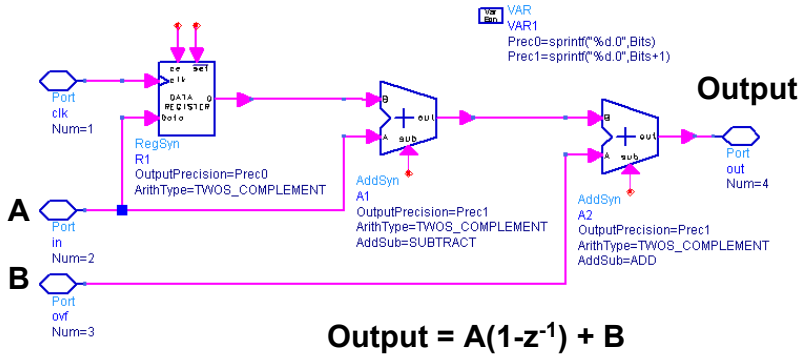


This shows a three-stage sigma-delta modulator, implemented via accumulators, described above, and differentiators.

The accumulators have a clock input, an input that is accumulated, an overflow output (that will be 1 or 0), and an output that is the fractional part of the accumulator's sum.

Differentiator in Ptolemy

A differentiator/summer block for the sigma delta modulator.
ovf is a single input with "1.0" precision, that is treated as [0,-1] twos complement.
Input is twos_complement with "Bits.0" precision.
Output is twos_complement with "Bits+1.0" precision.

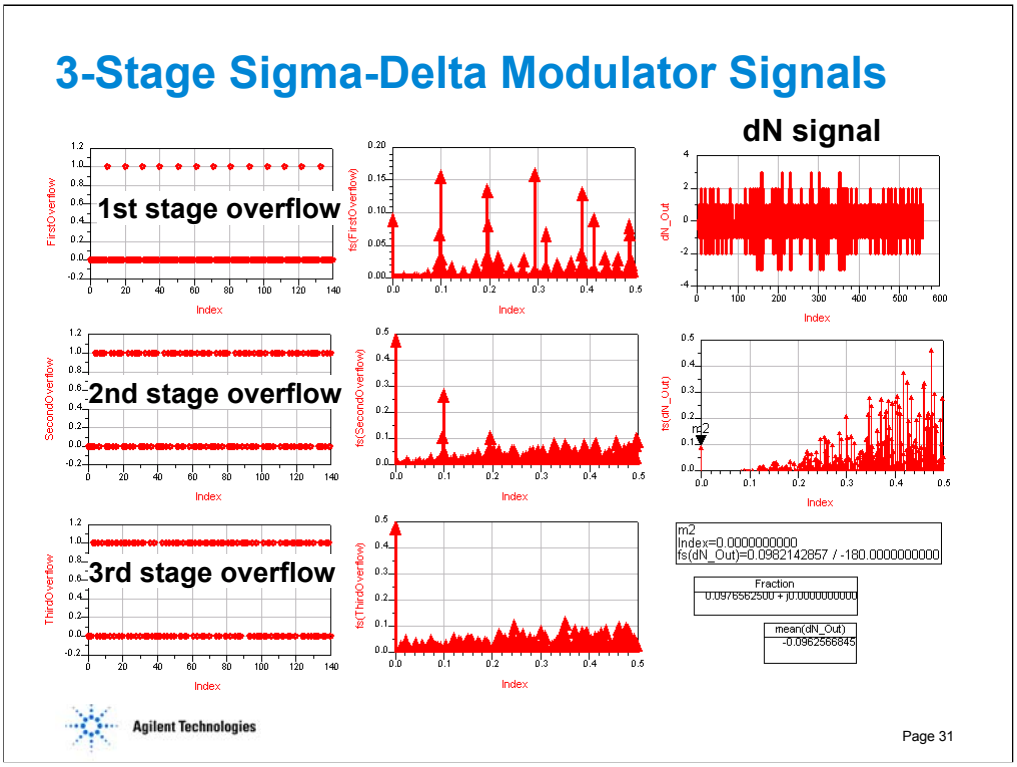


$$\text{Output} = A(1-z^{-1}) + B$$

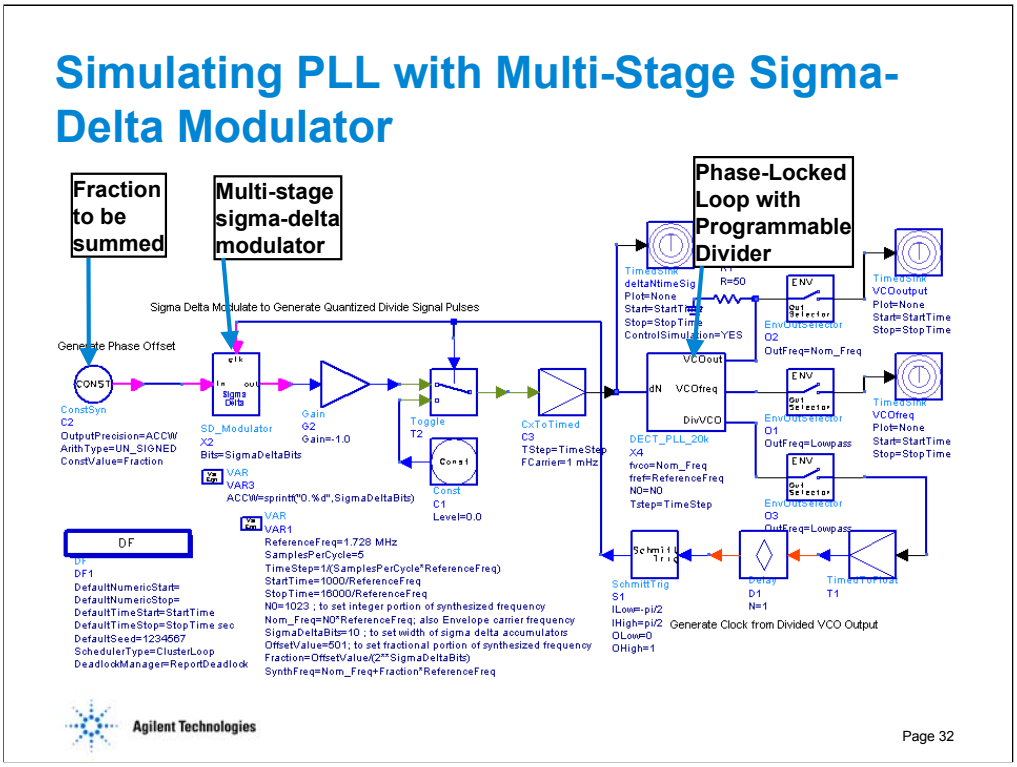
Agilent Technologies

Page 30

The "A" input is differentiated via the data register and the adder, configured to subtract, generating A-B, or the current value of "A" minus the value of "A" one clock sample ago. The result is then added to the "B" input, so this subcircuit really combines a differentiation with an addition.

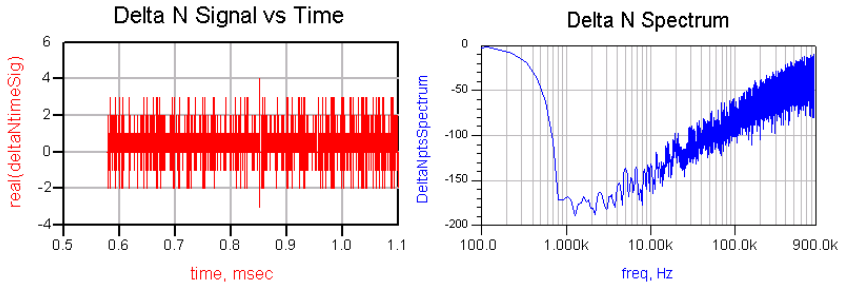


This figure shows how much more random the third-stage overflow signal is than is that of the first-stage. The average value of the dN signal is nearly exactly equal to the desired fraction, as expected. For a longer simulation, it should be exactly equal to the fraction. The average divide ratio will be N0, the nominal divide ratio, plus dN.



This shows the ADS simulation set-up, which is quite similar to the fractional-N simulation set-up (in fact, configuring the sigma-delta modulator to just use a single stage should give the same results.)

Delta N Signal and Spectrum



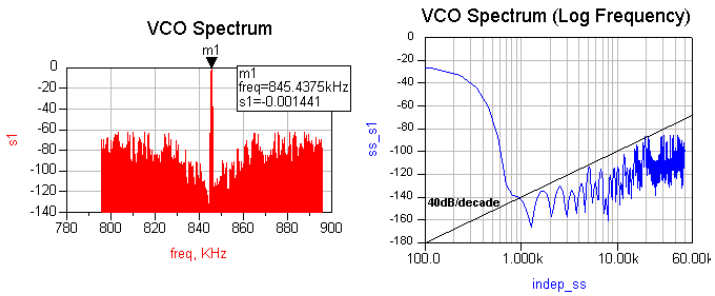
Agilent Technologies

Page 33

Simulated output signal (deviation in divide ratio from the nominal value) from the sigma-delta modulator, both in time and as a spectrum.

This signal, when added to the nominal divide ratio, becomes the instantaneous divide ratio. Note that its spectrum has a high-pass shape, as expected.

Resulting VCO Spectrum



0 Hz corresponds to N0*ReferenceFreq = 1023*(1.728 MHz)
= 1.767744 GHz

Fraction = 501/(2^{10}) = 0.489258
Synthesized frequency = (N0 + Fraction)*ReferenceFreq
= 1.767744 GHz + 0.489258*1.728 MHz
= 1.767744 GHz + 845.438 kHz

Agilent Technologies

Page 34

This shows the simulated output spectrum, and a log-offset spectral plot, showing the close-in spectrum. The loop bandwidth has been set to about 20 kHz, the noise spectrum flattens out above this offset frequency. In the "VCO Spectrum" figure, the X-axis is not the absolute frequency, but the offset from the nominal analysis frequency at N0*ReferenceFreq. The Fraction is set to 501/(2^{10}), and this value times the reference frequency at 1.728 MHz is 845.4375 kHz, which is how far above the nominal analysis frequency the VCO is, as expected.

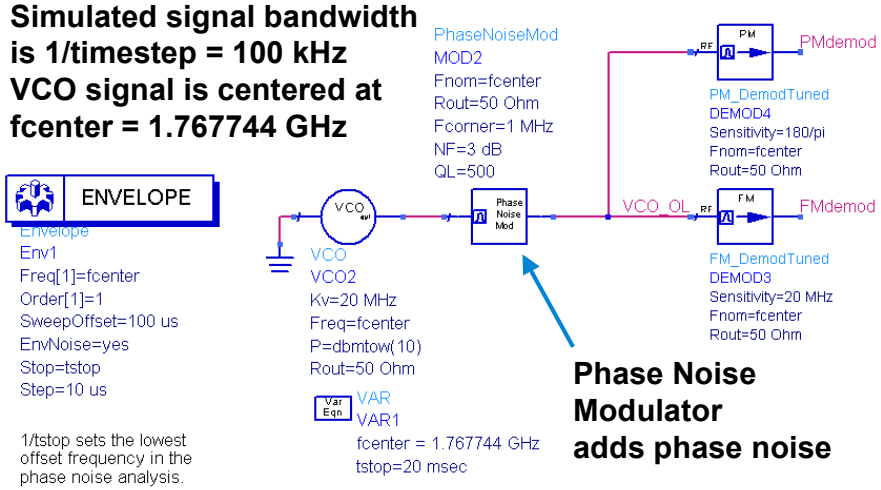
From the equation in the reference that gives the phase noise as a function of offset frequency and number of modulator stages:

$$L(f) = \frac{(2\pi)^2}{12F_{ref}}\left[\frac{f}{F_{ref}/(2\pi)}\right]^{2(m-1)}$$

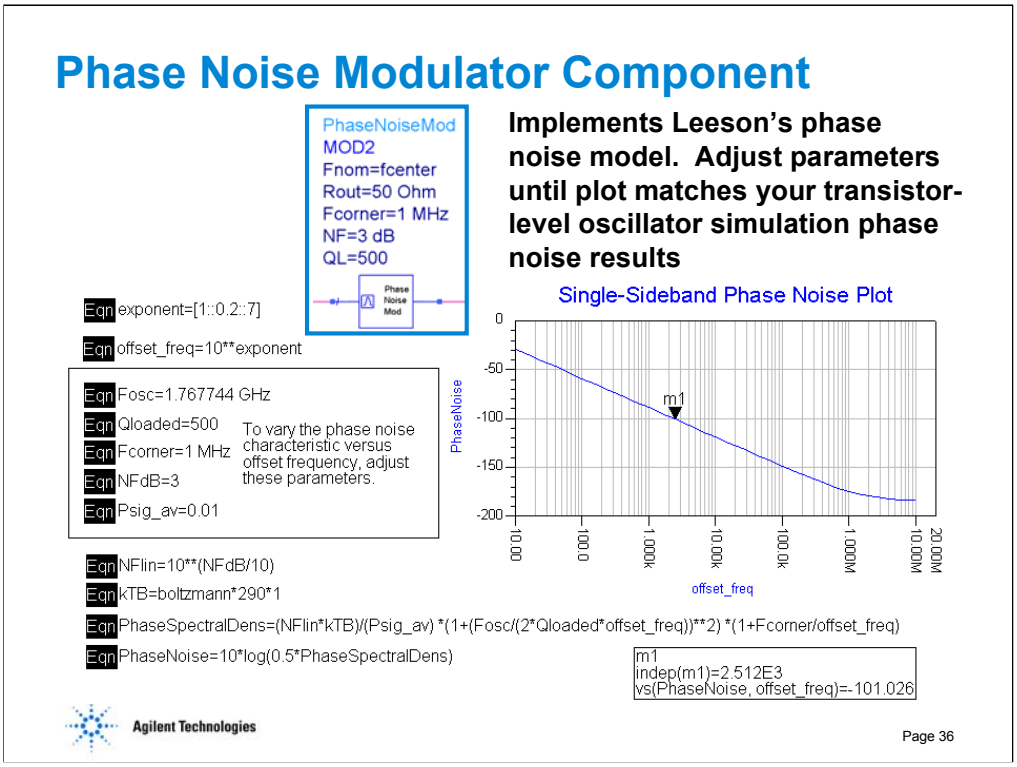
with m=3 stages, the phase noise should increase at a rate of 40 dB/decade increase in offset frequency, f.

Including VCO Phase Noise in the Simulation

Simulated signal bandwidth is 1/timestep = 100 kHz
VCO signal is centered at fcenter = 1.767744 GHz



The previous simulation results did not include any phase noise from the VCO or from other sources such as the phase/frequency detector. This shows how phase noise may be added to a VCO, via the phase noise modulator component.



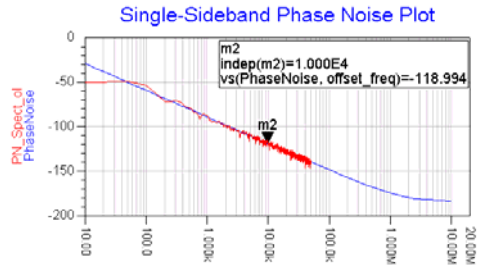
The phase noise modulator component adds phase noise in accordance with Leeson's equation The equation is repeated here as "PhaseSpectralDens." If you have a plot of phase noise versus offset frequency, from a simulation or from measurements, to use this phase noise model you have to adjust the PhaseNoiseMod parameters until you get a phase noise plot that matches what you want to model.

The "Single-Sideband Phase Noise Plot" here is from a frequency-domain noise simulation, in which the noise offset frequency is swept and the noise is simulated as a small-signal perturbation on a large-signal solution. In these simulations with the sigma delta modulator, the noise must be simulated in the time domain.

Reference:

Leeson, D., "A Simple Model of Feedback Oscillator Noise Spectrum," Proceedings of the IEEE, vol. 54, pp. 329-30, February 1966.)

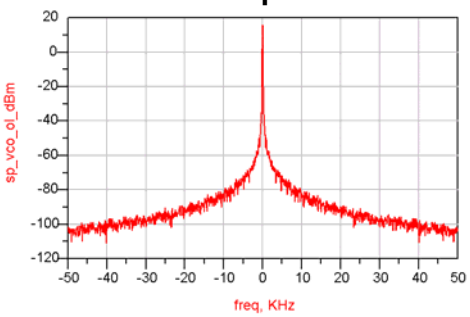
Noisy Open-Loop VCO Spectrum



Spectral bandwidth = 1/(time step) => decrease time step to see noise at higher offset frequencies.

Frequency resolution = 1/(stop time) => increase stop time to see noise at lower offset frequencies.

VCO Spectrum



Agilent Technologies

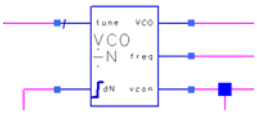
Page 37

These plots show the phase noise spectrum of the VCO, with noise added via the PhaseNoiseMod component and simulated in the time domain. When this noise is simulated in the time domain, random noise voltages and currents are added into the circuit. These noise signals are treated the same way as the large-signal voltages and currents. To see phase noise at large offset frequencies, a small simulation time step must be used. To see phase noise at small offset frequencies, a large stop time must be used. Also, because the signals are random noise, it may be necessary to run multiple simulations and average the results, to get reasonably smooth plots.

But VCO_DivideByN Model Does Not Allow Use of PhaseNoiseMod

Cannot add phase noise modulator between VCO and divider

VCO_DivideByN_Pulse
VCO2
VCO_Freq=Kv * _v1
F0=fvco
N=N0
Rout=50 Ohm
Power=dbmtow(0)
Delay=0.0



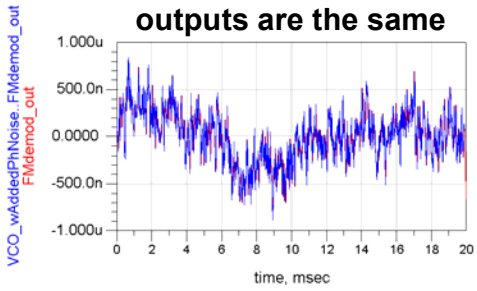
Agilent Technologies

Page 38

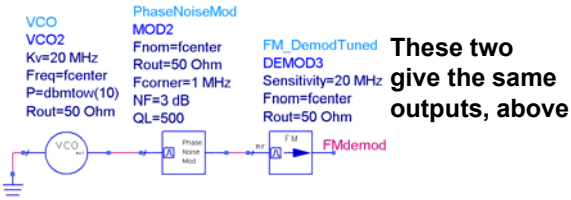
The VCO_DivideByN_Pulse and VCO_DivideByN models include both the VCO and frequency divider together, so it is not possible to insert the PhaseNoiseMod in between them.

Instead, Use VtDataset to Frequency Modulate VCO to Generate Phase Noise

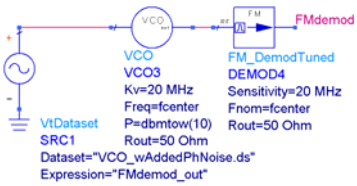
Frequency demodulator outputs are the same



Demodulator's sensitivity of 20 MHz/Volt => 1uV on vertical scale corresponds to 20 Hz frequency deviation from nominal



These two give the same outputs, above

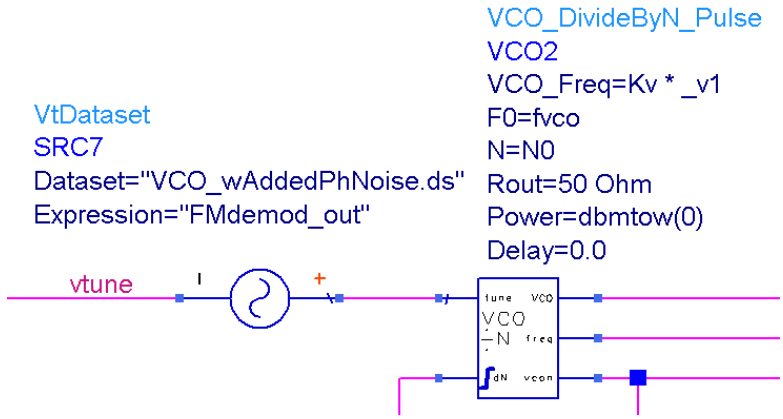


Agilent Technologies

Page 39

To overcome this issue with the VCO/Divide-By-N, we use an FM demodulator to obtain a voltage versus time that, when applied as a frequency modulation signal at the VCO's tune input, will force the VCO to have the same frequency variation as is produced by connecting the PhaseNoiseMod to the output of a VCO with its tune input grounded. This is accomplished via the VtDataset source.

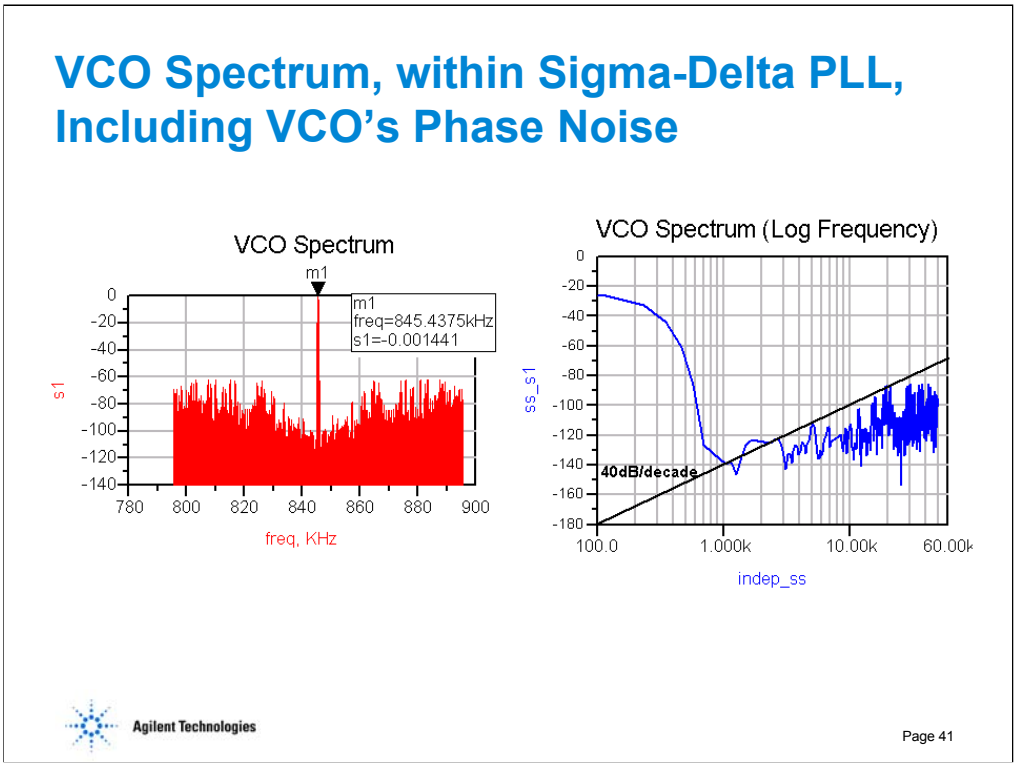
Frequency Modulate VCO/Divide-By-N Source to Add VCO Phase Noise



Agilent Technologies

Page 40

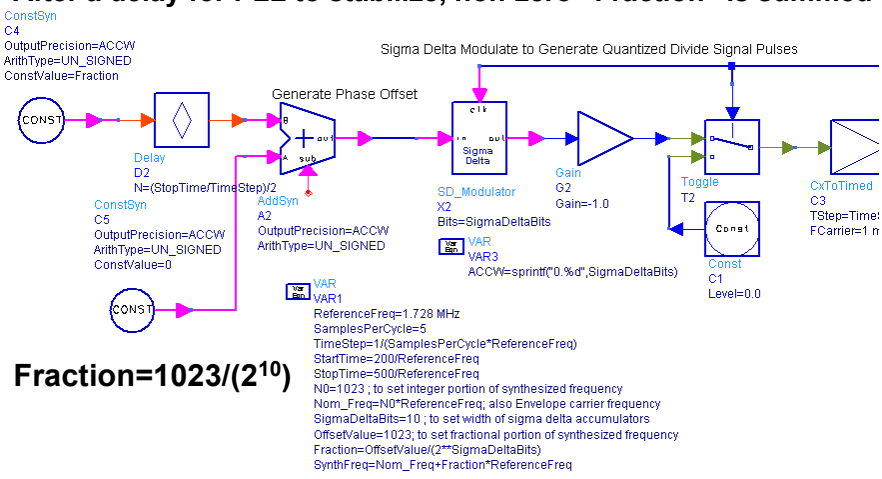
Here, the VtDataset source is used to frequency modulate the VCO in the VCO_Divide-By-N source, and add phase noise.



Comparing these phase noise plots with the earlier ones, the phase noise is higher, most noticeably between 1 and 5 kHz.

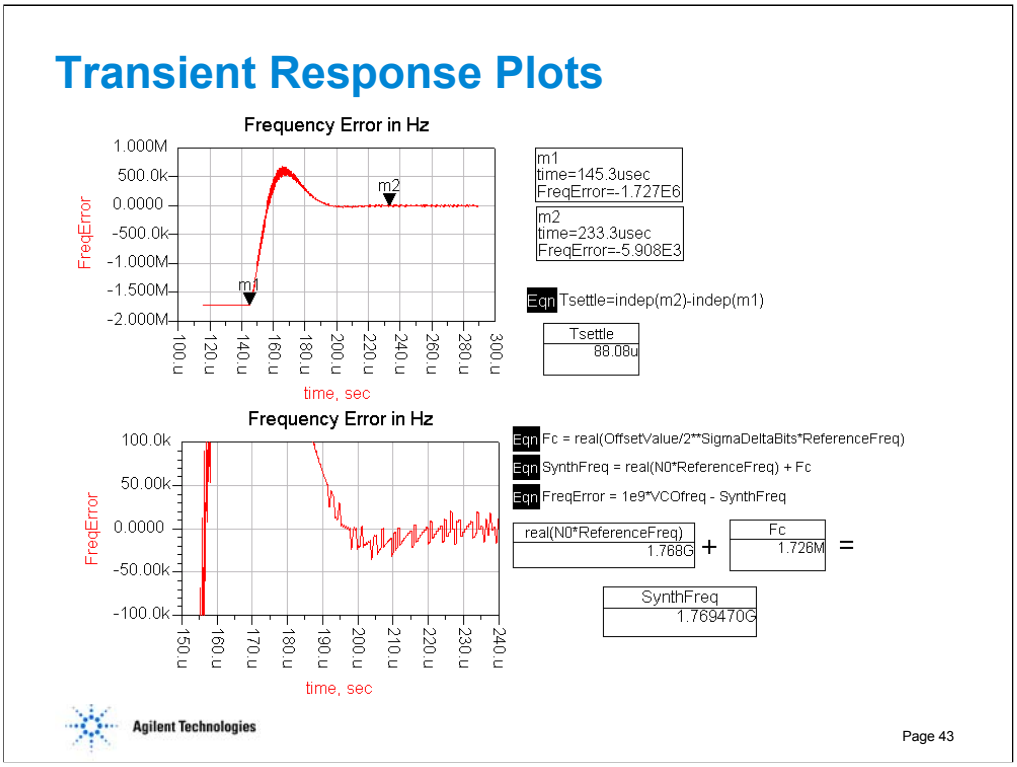
Transient Response Due to Change in Fraction

Fraction summed into sigma-delta modulator is 0 at first.
After a delay for PLL to stabilize, non-zero "Fraction" is summed in.



Fraction=1023/(2^{10})

Agilent Ptolemy and ADS may also be used to simulate the transient response of a phase-locked loop. Here, the fraction that is summed into the sigma-delta modulator is initially 0. After a delay, it is stepped to nearly 1, and the simulation is run until the VCO frequency settles.



This shows the change in VCO frequency versus time, as a frequency error from the final, expected steady-state value.

Conclusion

- **This paper has shown:**
  - **Agilent ADS and RFDE capabilities for simulating PLLs**
  - **PLL component behavioral modeling for including transistor-level effects**
  - **Agilent ADS and RFDE post-processing capabilities**
- **Agilent Ptolemy and ADS are able to simulate a complex phase locked loop using a sigma-delta modulator**
- **RFDE may be used for extracting behavioral models from transistor-level simulations**

References

1. Franceschino, Albert, “Phase-Locked Loop Primer and Application to Digital European Cordless Phone,” *Applied Microwaves and Wireless*, Fall 1994.
2. Miller, Brian, “Technique Enhances the Performance of PLL Synthesizers,” *Microwaves and RF*, January 1993.
3. Miller, Brian and Robert J. Conley, “A Multiple Modulator Fractional Divider,” *IEEE Transactions on Instrumentation and Measurement*, Vol. 40, No. 3, June 1991.
4. Howard, Andy, “Simulating a Phase-Locked Loop Using a Sigma-Delta Modulator to Attain Nearly Arbitrary Frequency Resolution Without Spurs”
http://www.chipcenter.com/networking/technote013.html