An introduction to the PLL library to be released in 4.4.5

Written by

Jess Chen

The procedures described in this application note are deliberately broad and generic. Requirements for your specific design may dictate procedures slightly different from those described here.

Purpose

This application note describes

- where the library fits in a PLL design flow,
- phase-domain models of PLLs,
- how to use the new models.

Audience

The intended audience consists of Analog Artist users and application engineers interested in simulating deterministic behavior of phaselock loops. This application note assumes the reader has a basic knowledge of phaselock loops and knows how a phase-frequency detector works.

Overview

The models in the new phaselock loop (PLL) library are the first installment of a set of models supporting top-down design of PLLs. Figure 1 shows the design flow. This application note focuses on the first step in the flow but the overview briefly describes all three steps for perspective.

The first step in Figure 1 is to develop an executable specification. The executable specification is an arrangement of fast behavioral models that capture desired system performance. The executable specification facilitates fast architectural studies and isolates specification issues from implementation issues. The executable specification is composed of baseband models [1[1],2,3,4,5,6,7,8,9]. Baseband models suppress clocks and RF/IF carriers. Some literature refers to PLL baseband models as "relative phase" or " phase-domain" models [2]. I will use the latter term, "phase-domain" models. Phase-domain PLL models are exceptionally fast, capture the dominant non-linear mechanisms, and are directly linearizable for AC analysis.

1. This reference uses the terms "baseband" and "bandpass" explicity.

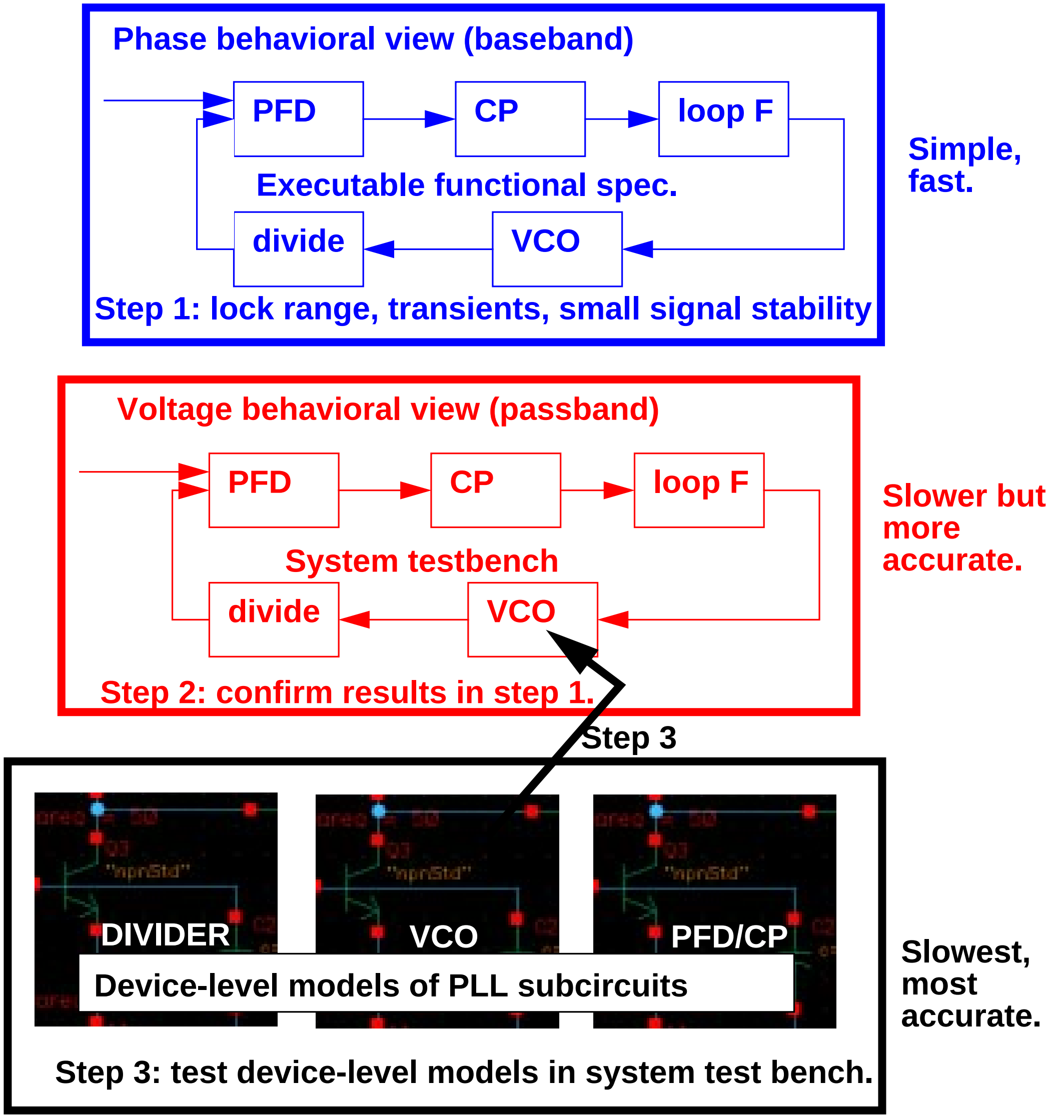

Phase behavioral view (baseband)
PFD
CP
loop F
Executable functional spec.
divide
VCO
Step 1: lock range, transients, small signal stability
Simple, fast.
Voltage behavioral view (passband)
PFD
CP
loop F
System testbench
divide
VCO
Step 2: confirm results in step 1.
Slower but more accurate.
Step 3
DIVIDER
VCO
PFD/CP
Device-level models of PLL subcircuits
Step 3: test device-level models in system test bench.
Slowest, most accurate.

Figure 1. PLL top-down design flow

The next step is to translate the executable specification into a system testbench. The main difference between the executable specification and system testbench is that the testbench is composed of passband models [1]. I will refer to passband models as voltage-domain models because they simulate voltages one can observe in a lab. A comparison of 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 VCO phase is unwrapped, in steady state it ramps up indefinitely. Unwrapped phase is not periodic. Voltage-domain models capture non-linear effects related to the shapes of the actual waveforms. Waveform effects include spurs and harmonic locking[1]. Phase-domain models do not simulate waveform effects. The system testbench is more accurate than the executable specification but it is still behavioral. Armed only with behavioral voltage-domain models, the testbench does not simulate device-level effects associated with specific implementations. Examples of implementation effects are interstage loading, improper bias, and device parasitics.

The last step in Figure 1 is to replace behavioral models in the system testbench with device-level models, one or two blocks at a time. Device-level models check for the aforementioned implementation problems. The entire PLL is simulated at the device-level only as a final verification step because such simulations take too long to run.

1. "Harmonic locking" is when the PLL locks on to a harmonic of the reference.

The remainder of the application note describes the phase-domain models. The new PLL library includes:

- Analog multiplier phase detector
- XOR phase detector with bipolar output[1]
- Three-state digital phase frequency detector (PFD)
- Charge pump (current source version)
- VCO tuning curve (analytic and tabular versions)
- Frequency divider
- Lock indicator
- Examples

1. The XOR phase detector is not explicitly discussed in this application note 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.

Introduction

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[1] [8]. Future enhancements to the PLL library and design flow will address jitter and phase noise. The examples discussed below do not encompass all applications of the phase-domain models. They are only meant to get users started.

The application note contains two main sections. The first section introduces phase-domain modeling, describes a feature included to circumvent DC convergence problems, then discusses a few examples of how to use phase domain models. The second section explains how to assemble a more complex PLL and discusses an example.

Phase-domain model of a simple PLL

Description

This section introduces the main ideas of phase-domain modeling by describing a PLL example built around the simplest phase detector in the new library[2]. Figure 2 shows a voltage-domain model of the example and some selected waveforms.The phase detector in this case is just an ideal analog multiplier. Figure 3 shows the equivalent phase-domain model. The phase-domain model is based on the trigonometric identity below:

1. 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.

$\sin(\theta_1)*\cos(\theta_2) = (1/2)*[\sin(\theta_1+\theta_2)+\sin(\theta_1-\theta_2)]$, which after filtering is approximately

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

Note: $\theta_1+\theta_2 = (\omega_1+\omega_2)*t$, and $\omega_1+\omega_2$ usually lies far beyond the filter corner frequency.

2. Phase-domain models of this simple PLL are by no means new [1,2,3,4,5,6,7,8,9]. However, to my knowledge the modifications I included to circumvent DC convergence problems, and the non-linear PFD model discussed in the next section, are unique to our PLL library.

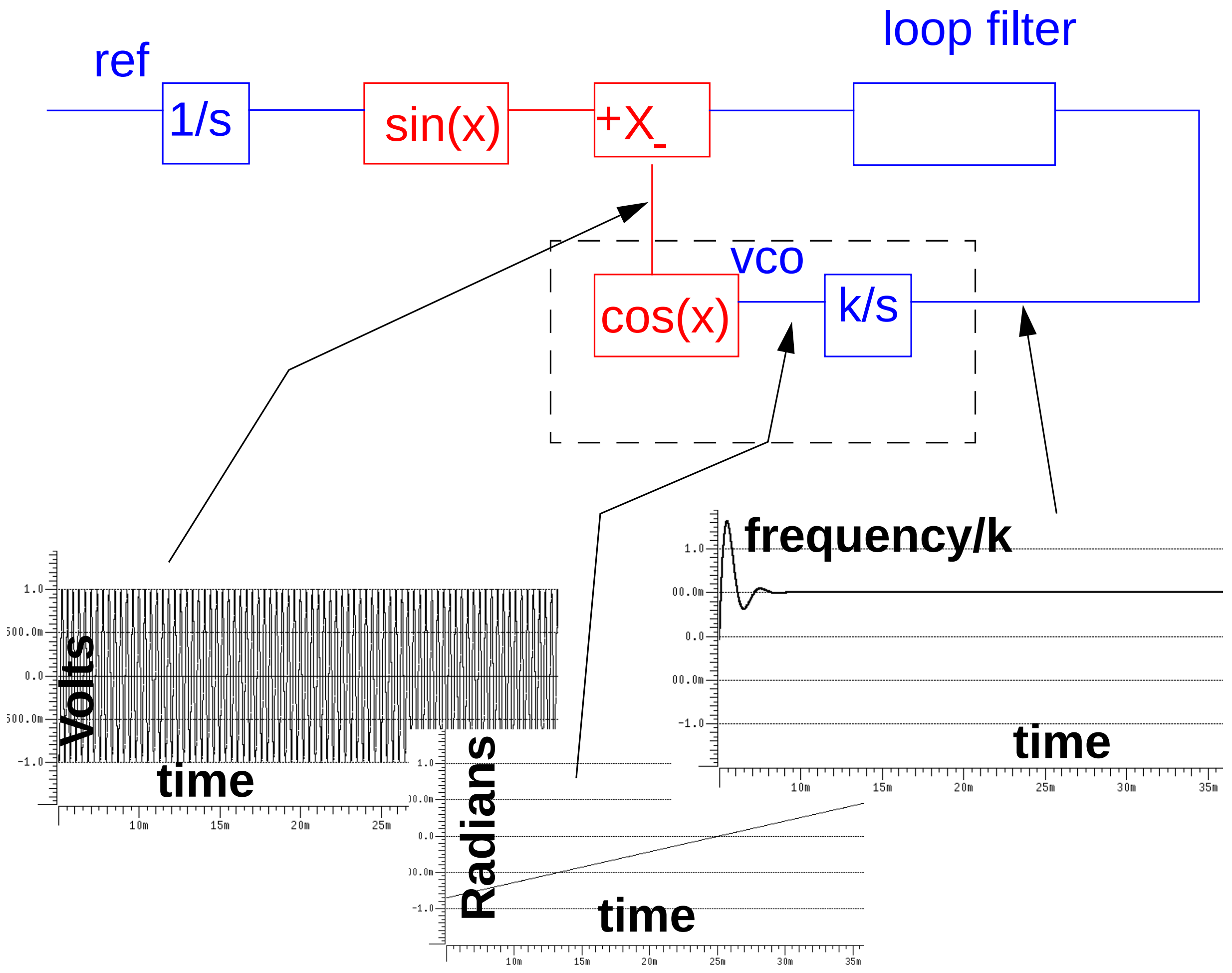


Figure 2, Voltage domain model

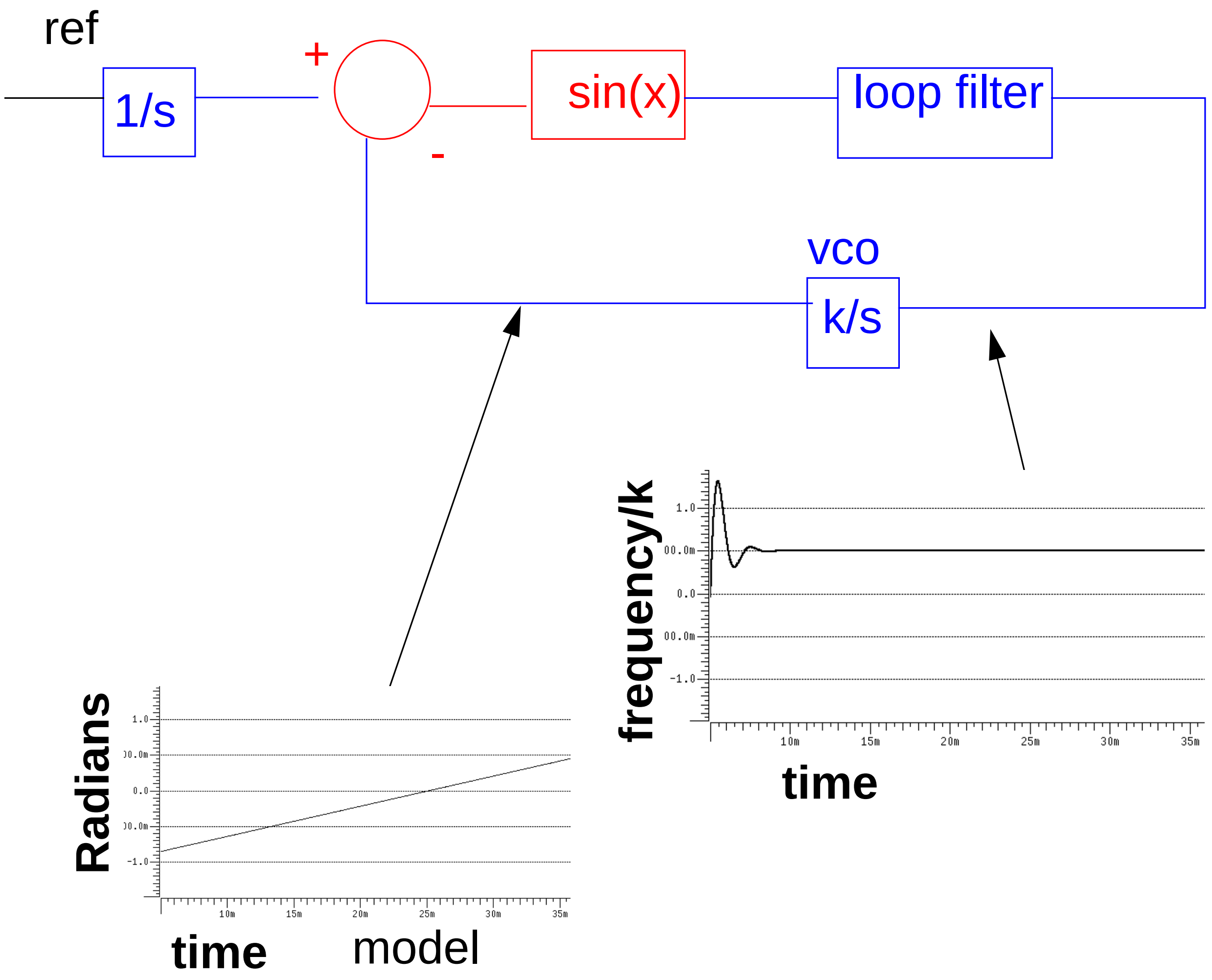


Figure 3, phase-domain model.

The phase and frequency waveforms from the phase-domain model match their voltage-domain counterparts but the simulation runs much faster because the oscillatory waveform is not explicitly simulated. Combining the integrators, as shown in Figure 4, eliminates the integrator outside the feedback loop. Integrators outside of feedback loops cause problems if the user forgets to specify an initial condition. Also, this step is crucial in building phase-domain models of phase-frequency detectors because the non-linearity in that case has memory (hysteresis).

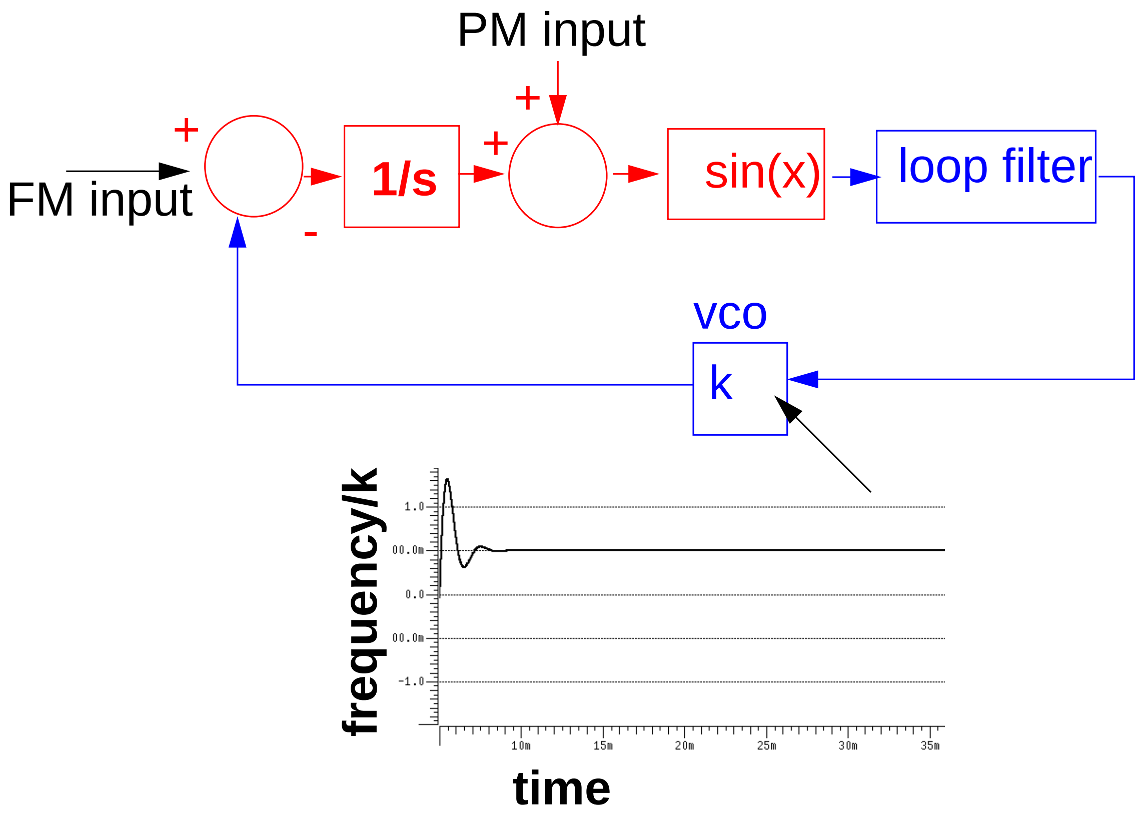


Figure 4, A more practical phase-domain model.

The models in Figures 3 and 4 both suffer potential DC convergence problems because the phase detector model either precludes a DC operating point or produces an infinite number of them. The sinusoidal function in Figure 5 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, I call the phase detector output "duty cycle".

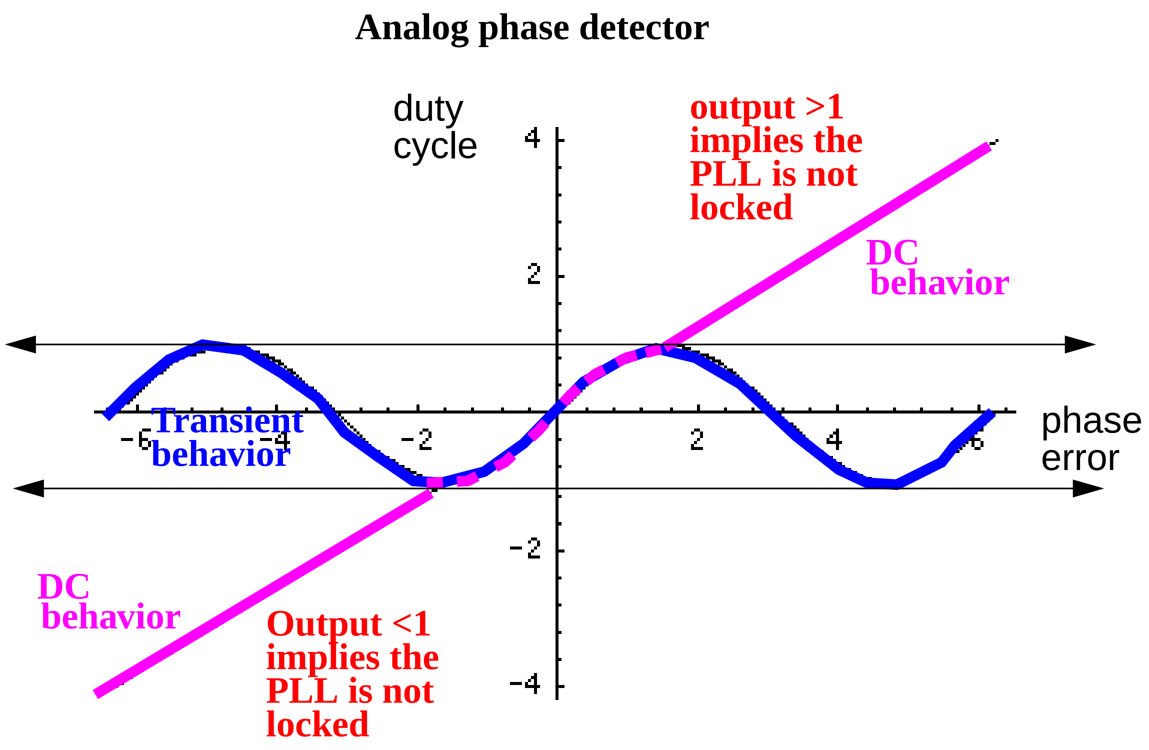


Figure 5, Phase detector transfer curves.

If the required duty cycle lies outside [-1,1], the loop is not locked in steady state and Spectre can have trouble figuring that out. If the required duty cycle lies in [-1,1], there are an infinite number of possible phase errors and Spectre can have trouble choosing one. Verilog-A’s ability to do different things for different analyses provides an elegant solution to the DC convergence problems while providing 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 phase error lies in the interval [$-\pi/2$, $\pi/2$]. If the required duty cycle lies in [-1,1], the monotonic transfer curve forces steady state phase error to the interval [$-\pi/2$, $\pi/2$] where the two curves coincide. The equilibrium point is "open-loop-stable", meaning that at DC the loop gain is a positive real number since the slope of the transfer curve is positive over [$-\pi/2$, $\pi/2$]. This makes the Nyquist stability criterion easier to apply. The DC analysis is general enough because only phase error modulo 2π matters and we usually only care about the "open-loop stable" operating points. When the loop is not locked, the DC analysis gives a duty cycle greater than one in magnitude. A duty cycle greater than one is obviously wrong but it is far easier to interpret than a convergence error. DC duty cycle is lock indicator which can be used in a parametric DC analysis to sweep out lock range.

Example 1: A dynamic test for capture range and lock range (example_analog_PD in pllLib).

Capture range is the range of input frequencies which the PLL can lock on to (i.e. acquire) from an unlocked state. Since acquisition of frequencies near the edge of the capture range involves a pull-in mechanism [1-12], measurement of capture range requires a transient analysis. One way to measure capture and lock ranges is to slowly sweep input frequency and observe the frequency where duty cycle begins and ends a long ramp [12]. The DC analysis must be skipped to observe the capture limits. Figure 6 plots the VCO control voltage against the voltage representing input frequency. The input frequency ramps up then down. A buffered auxiliary circuit senses duty cycle and adds 2.5 volts when the input frequency changes direction. This trick makes the plot easier to read because the forward and reverse sweeps don’t occupy the same part of the vertical scale. In this example, lock range is from 1.36Khz to 3.4Khz and capture range is from 1.8Khz to 3Khz.

Figure 7 compares VCO control voltages in the forward sweep computed with voltage- and phase-domain models. The models agree quite well. For this example (2.5Khz center frequency) the phase domain model is only about 20 times faster than the voltage-domain model.

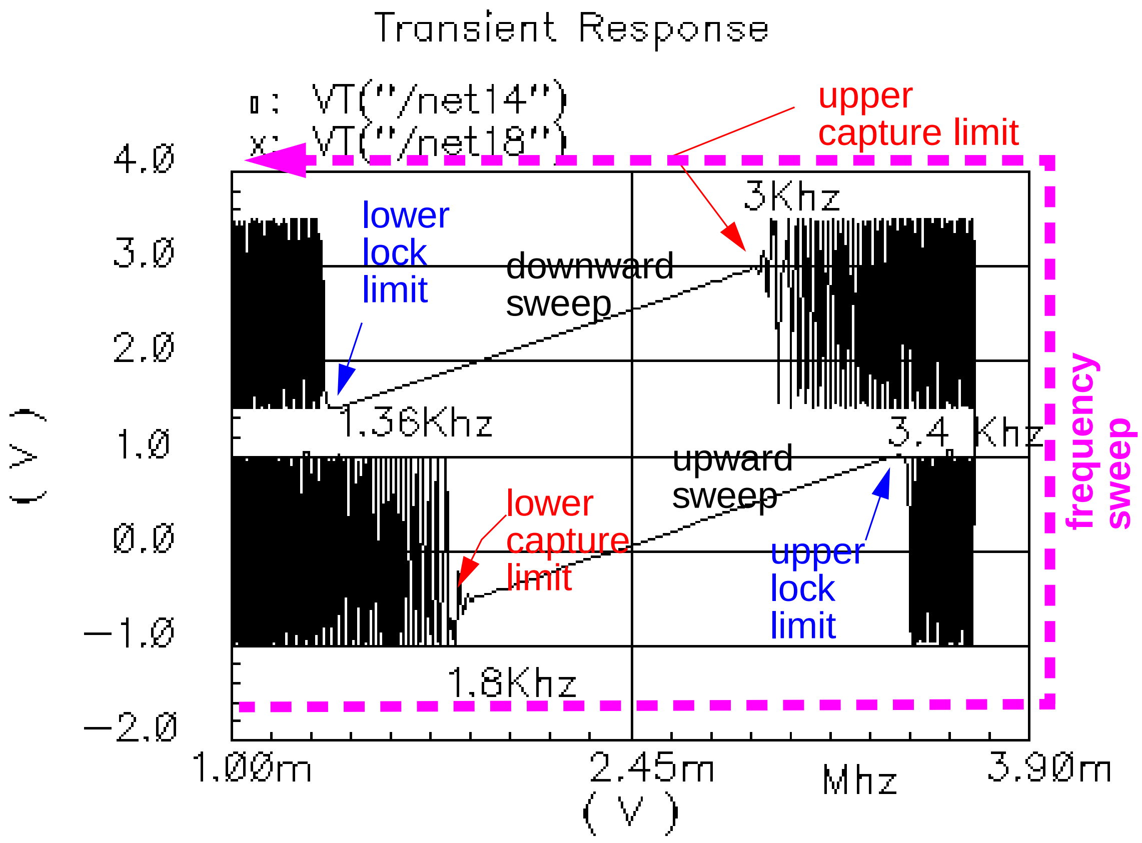
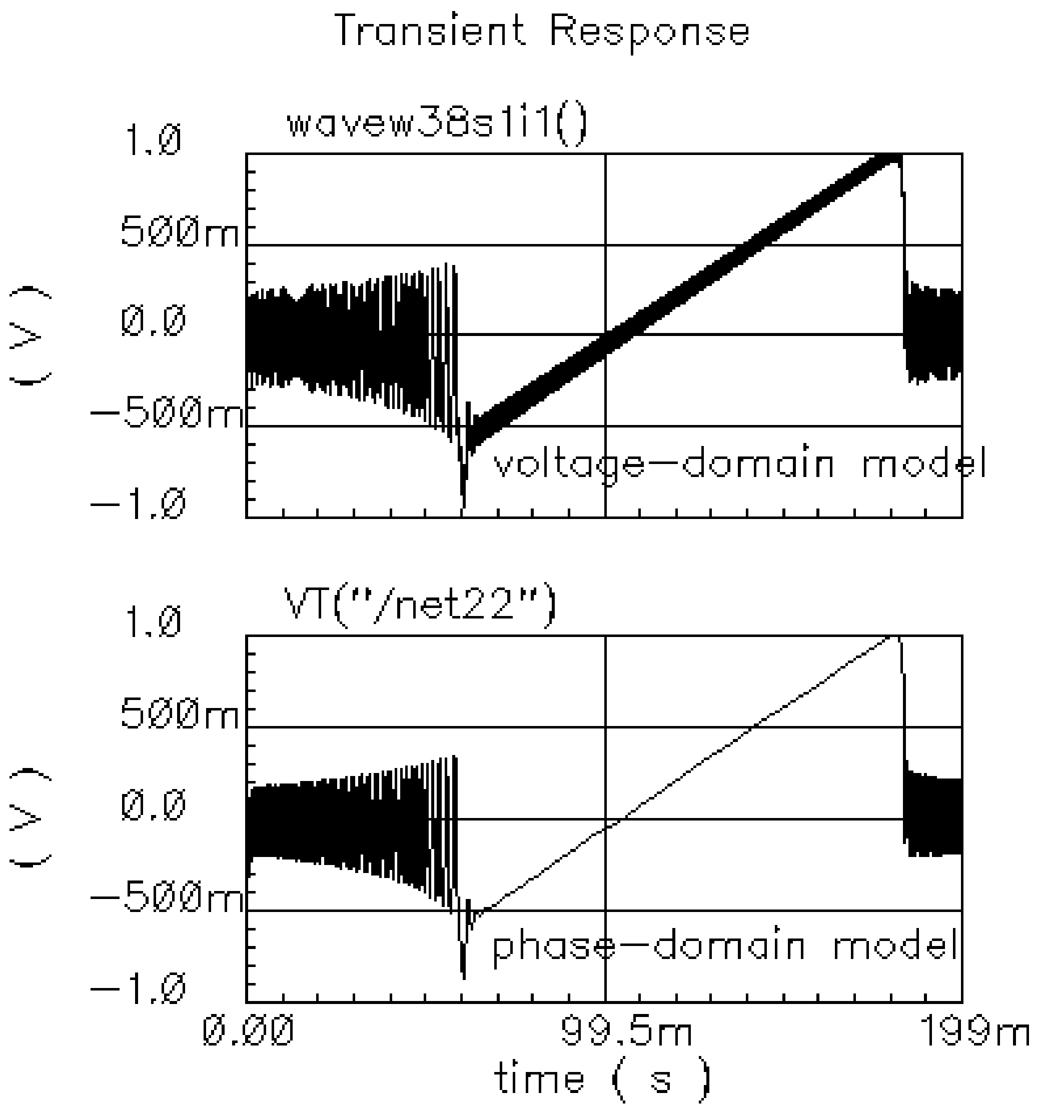


Figure 6, lock range and capture range.



19.2 seconds of cpu time.

1.25 seconds of cpu time.

Figure 7, VCO control voltage response to a ramping PLL reference frequency as computed with voltage- and phase-domain models.

Example 2: Loop gain measurement (example_loop_gain in pllLib).

Spectre cannot perform any meaningful AC analysis on a voltage-domain model because by design, a voltage-domain PLL model has no DC operating point. However, since Spectre 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 8 shows an Analog Artist version of the model in Figure 4. The model in Figure 8 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 injects a test signal without disturbing the DC operating point. This source must be inserted at a point where the impedance looking back is much much smaller than the impedance looking forward. The accuracy of the resulting loop gain depends on how well this condition is met.

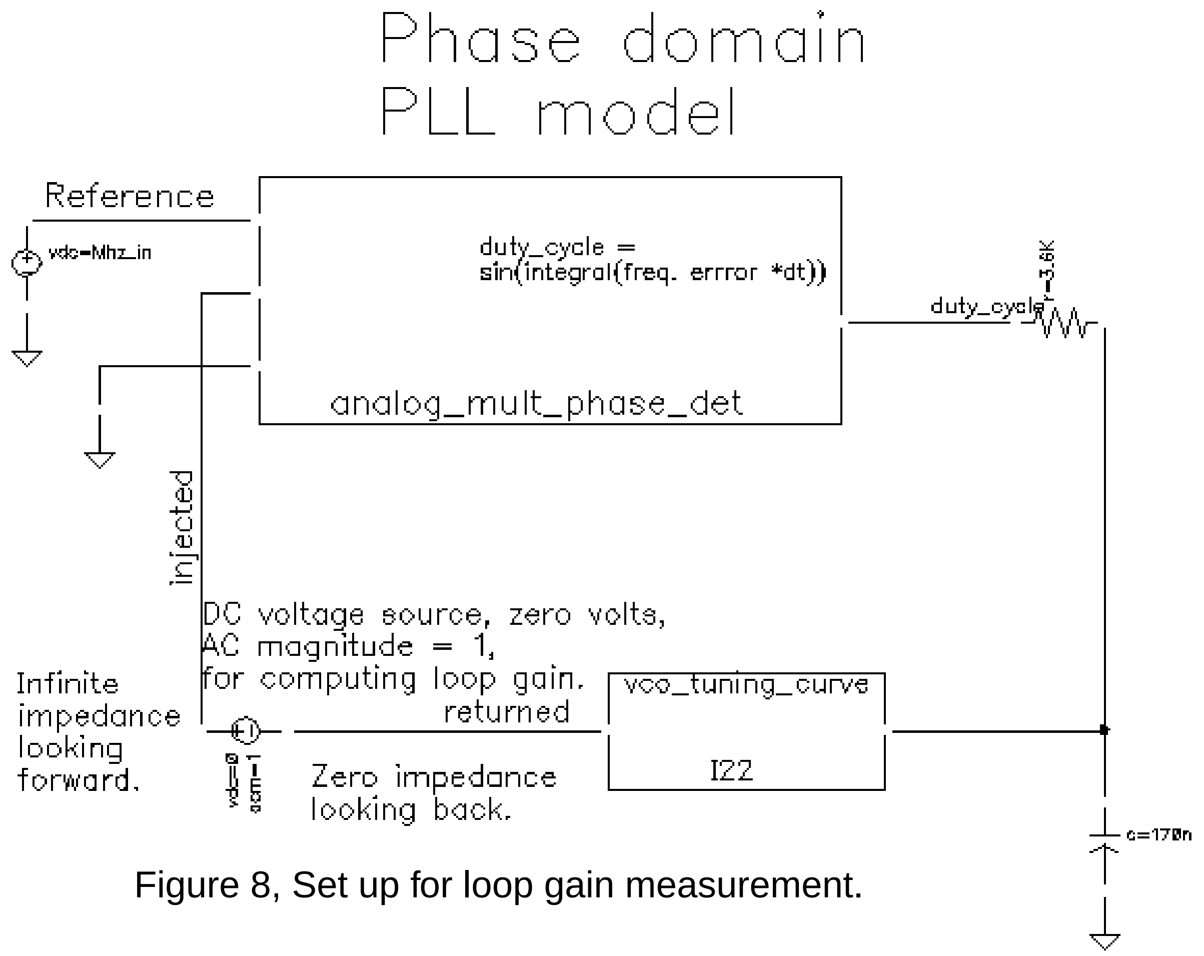


Figure 8, Set up for loop gain measurement.

The steps for displaying loop gain are as follows:

1. Set up the AC and DC analyses. The zero-voltage vdc source must be the only source with a non-zero AC magnitude. Save the DC data so you can annotate the schematic with DC node voltages.
2. Run the analysis.
3. Click results->annotate->DC node voltages to display the node voltages on the schematic. Make sure the DC duty cycle lies in [-1,1]. If it falls outside this interval the loop is not locked and the AC analysis is invalid.
4. Click results->direct plot->AC difference, then select the returned signal then the injected signal (as labeled in Figure 8). If the waveforms do not appear, hit the escape key. The waveform display will show two curves.
5. In the waveform display window hit tools->calculator to bring up the waveform calculator.
6. Hit the "wave" button and select the phase. Subtract 180 from the phase waveform and plot the result. You may want to delete the original phase waveform now.
7. In the waveform display window, click Curves->Edit Curves, then select the shifted (by 180 deg) phase, then change "Y axis" to 2, and OK the form. This should produce Bode plots of the loop gain. You can change to the display to "strip" to produce a display like the one in Figure 9.
8. To generate a Nichols chart (dB versus degrees), from which one can pick off phase and gain margins, click Axes->x-axes then select the phase.

9. Click Curves->Edit then select the magnitude. Then change the Scale to dB20 and OK the form. You should now have a Nichols chart like the one in Figure 10. The phase margin is 30 degrees.
10. To compute phase margin directly: In the waveform calculator, click "vf", click on the return node, click on the injected node, back in the waveform calculator click on the divide button, then under the Special Functions menu hit the phase margin button followed by the print button. Add 180 degrees to the expression in the waveform calculator then hit the print button. The Results Display Window that comes up will display the phase margin.

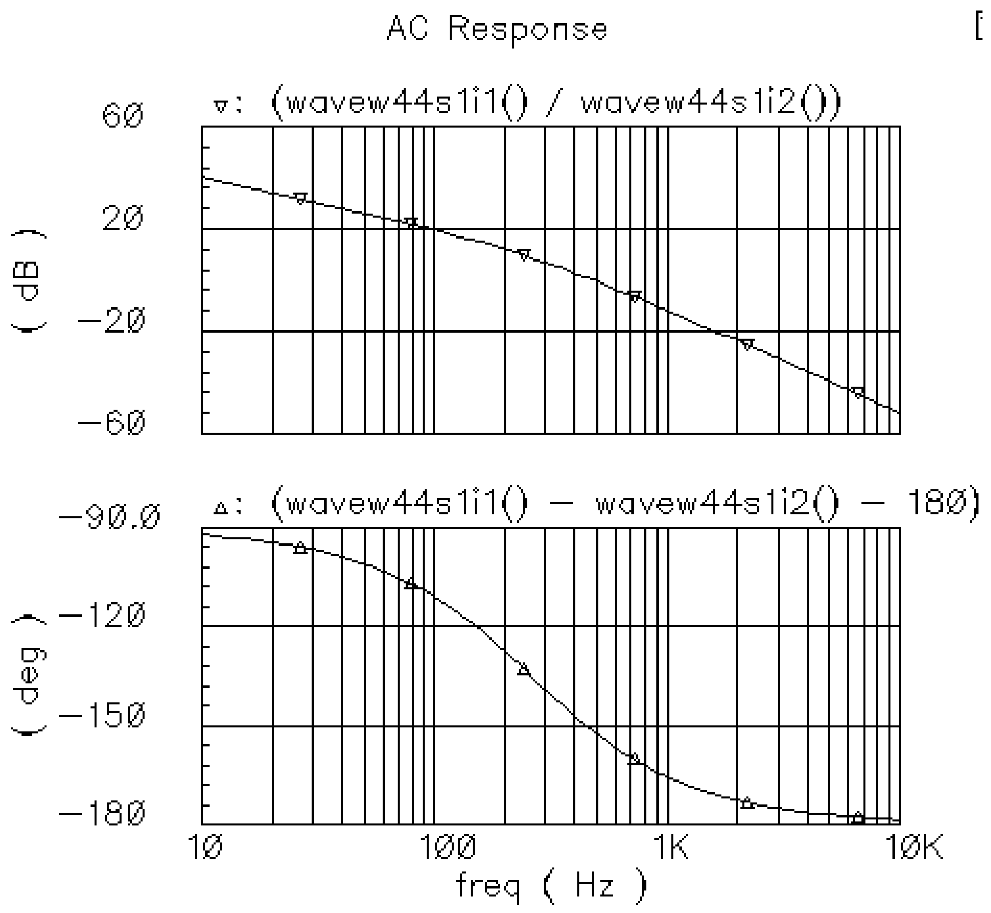


Figure 9, Bode plots, magnitude and phase of the loop gain.

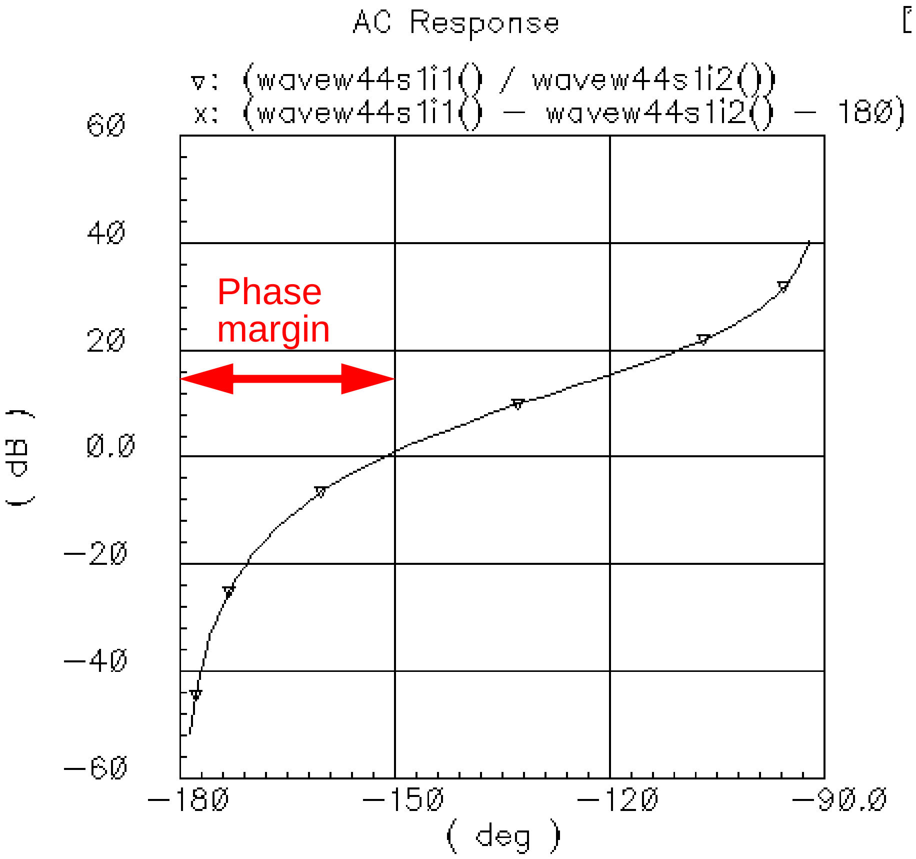


Figure 10, Nichols chart of the loop gain.

Example 3: PM input (example_PM in pllLib).

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 kick the PLL to assess large signal stability. Figure 11 shows a test circuit for such a stability check.

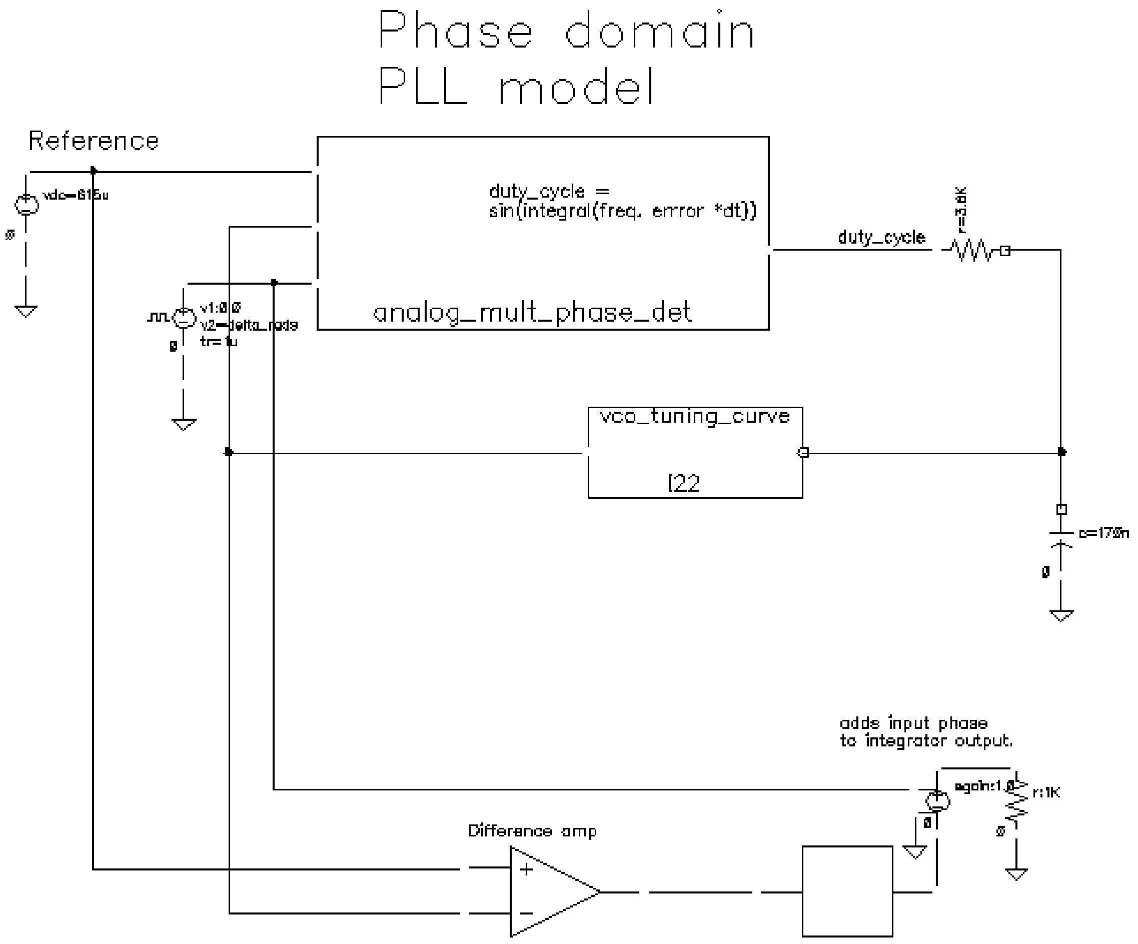


Figure 11, Stability check using the PM input.

The PLL is very simple but different from the previous example. It was modified to produce more interesting results. The lower circuitry needs a explaining. 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 ahdl library.

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 12 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 kicks the PLL into an unstable mode. The references examine this behavior in mathematical detail [1,4,6]. The purpose of this example is to show one way to assess large signal stability and to demonstrate that the phase-domain models capture the dominant non-linear mechanisms.

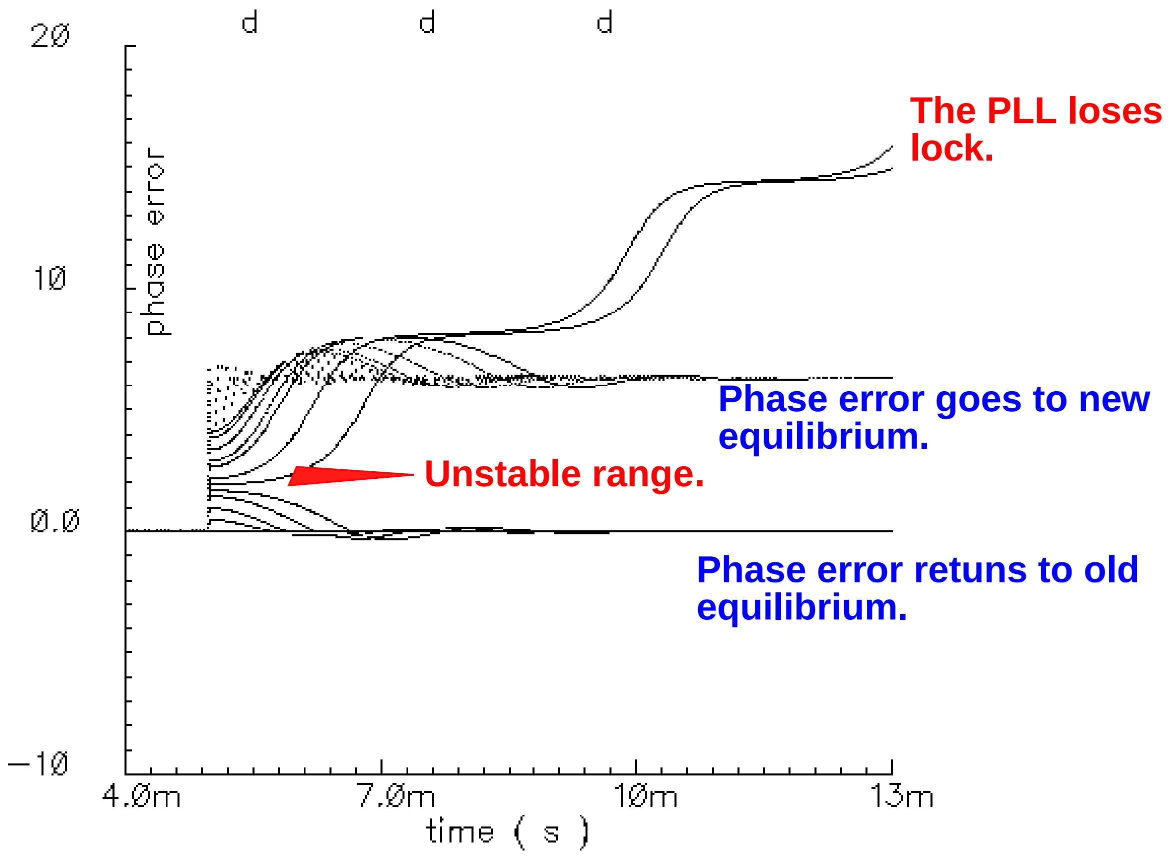


Figure 12, phase error response to step PM input.

Assembling a model of a PFD-based PLL

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

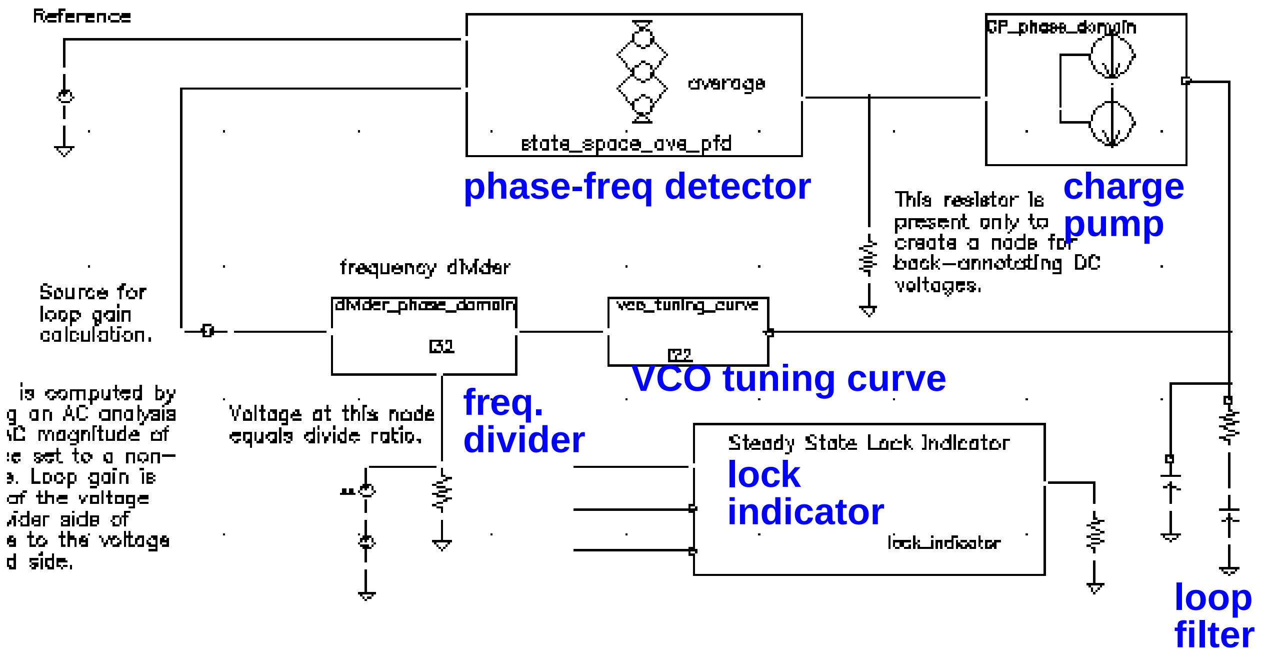


Figure 13, PLL block diagram

VCO

The VCO is modeled by it’s tuning curve. The tuning curve characterizes the relationship between input voltage and 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. Thus, when the VCO operates at 2 Mhz the model output is 2 Volts. The tuning curve is non-linear in general and can be specified either with the coefficients of a fourth order polynomial or a look-up table.

NOTE: In general, to edit or enter parameters, instantiate the model and edit the object properties. Under " CDF Parameter of view", change the view from "Use Tools Filter" to "veriloga". The parameter fields will appear.

Polynomial tuning curve: The input voltage is internally clamped to the nearest end point when it tries to venture 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 is clamped 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 scale factor (same as before) 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 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 the user can override that.

Suppose the data was at <home>/data/table and the netlist was as shown above. The relative path would be

../../../../../data/table.

At the time I wrote this application note, the path had to be enclosed in double quotes.

Frequency divider

The frequency divider is not much more than a simple gain element. It takes an input voltage which represents frequency in Mhz, 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 internally clamps it to 0.001 and issues a warning. The clamp prevents division by zero amid simulation.

Charge pump

The charge pump transforms the duty cycle into the expected average current sourced/sunk by the charge pump. The maximum source and sink currents are user defined and can be different from each other. If the charge pump output voltage tries to exceed the user-defined rails, the output clamps 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.

A future enhancement will address charge pumps that switch voltages instead of controlled currents. For now, please refer to the PLL paper in [11] for the switched-voltage type of charge pump.

Loop filter

The loop filter is entered component-for-component.

State-space averaged PFD (Phase-domain phase-frequency detector model. The appendix explains how it works.)

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. I took the term "state-space averaged" from the power electronics field [13]. The charge pump currents associated with the three PFD states are averaged together with a duty cycle much in the same way voltages are averaged together with 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 frequency error is large, duty cycle is a smooth waveform directly related to normalized frequency error [1,4]. When frequency error is small, an integrator inside the PFD model converts frequency error to phase error and duty cycle becomes proportional to phase error. The duty cycle experiences discontinuous jumps (or resets) towards zero as it transitions from frequency-mode to phase- mode.

As phase error enters a deadband determined by the minimum-on-time parameter and reference frequency, the model kicks out a duty cycle pulse of unity magnitude and duration equal to the minimum-pulse-width. After the pulse expires, the duty cycle drops to zero until phase error exits the deadband. As phase error exits the deadband, duty cycle jumps up 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 classic trade-off between execution speed and accuracy. The speed_vs_accuracy parameter controls the number of resets the internal integrator undergoes as the model transitions from frequency-mode to phase-mode. Too few resets can introduce 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, try increasing the speed_vs_accuracy parameter to 70k or 100k. To increase the number of resets in a fast PLLs, and thereby increase the accuracy, try reducing the speed_vs_accuracy parameter to 10k or 20k. Experience shows that a reasonable setting for the speed_vs_accuracy parameter should produce a duty cycle step response that resets nearly to zero roughly 3-10 times before entering the final transient. Figure 12 shows reasonable duty cycle step response.

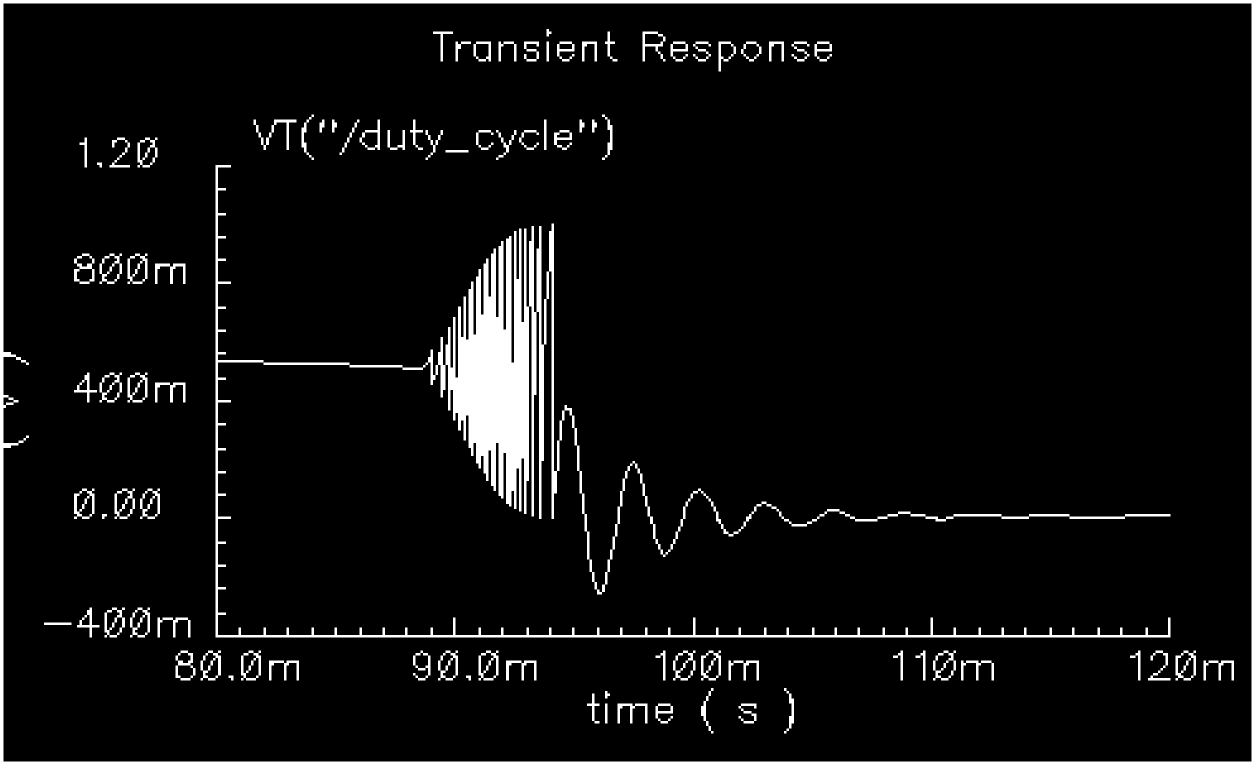


Figure 12, Duty cycle waveform.

The other parameter is the minimum_on_time. It controls backlash. This is the minimum pulse width the PFD can generate. As phase error drops further, 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 12 was generated with the default minimum-on-time of zero us. The default value of this parameter produces no deadband and no unity pulses; The default deactivates the backlash mechanism. Figure 13 was generated with a minimum on time of 0.2 us. Figure 13a illustrates how the pulses only occur as phase error enters the deadband[1]. Figure 13b shows the limit cycle created by backlash. The limit cycle is dominated 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 out to where the loop filter attenuates it. The phase-frequency detector model can help quantify the problem and check the solution.

(The PLL model that generated Figure 13 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 13 ran in a matter of seconds!)

1. A pulse is not kicked out upon exiting the deadband because that behavior causes convergence problems for Spectre. 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 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.

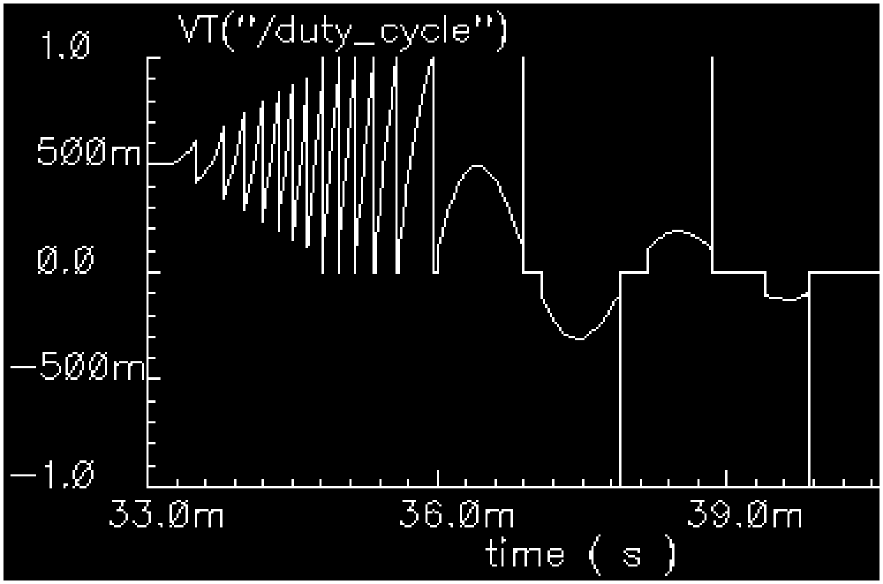


Figure 13a, duty cycle between 33ms and 40ms.

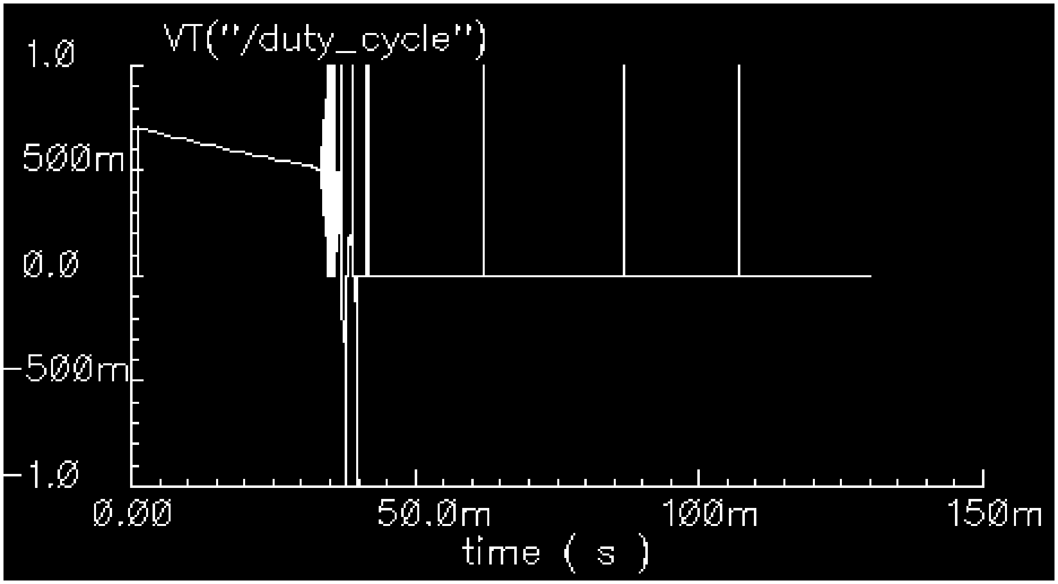


Figure 13b, duty cycle to 130ms

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 do one thing 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 phase detector output, VCO control, and charge pump output. If any of those signals exceeds their limits, 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, indicating that the loop is locked. The lock indicator is only valid for DC analysis. I recommend using node names to tie the lock indicator inputs to the right nodes and using variables for the component limits. The limits must be manually entered twice, once for each component and once for the lock indicator. With variables, the lock indicator parameters are guaranteed to be linked with the proper component parameters and changes only have to be entered in one place.

Example 4: Acquisition transients (example_phase_domain in pllLib).

Figure 14 shows the duty cycle and VCO frequency response to a momentary change in 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 nears the final value, the PFD model gradually transitions from frequency-mode to phase-mode. When frequency error is small, but large enough to slew phase error, the duty cycle waveform should look like a sawtooth waveform. The model gradually increases the amplitude of the sawtooth component of the duty cycle as frequency error goes to zero and always maintains the correct average. The final duty cycle transient is the sawtooth dominated by phase error. Figure 12, above, blows up the duty cycle in the first transition. Figure 15 blows it up further to show the sawtooth waveform.

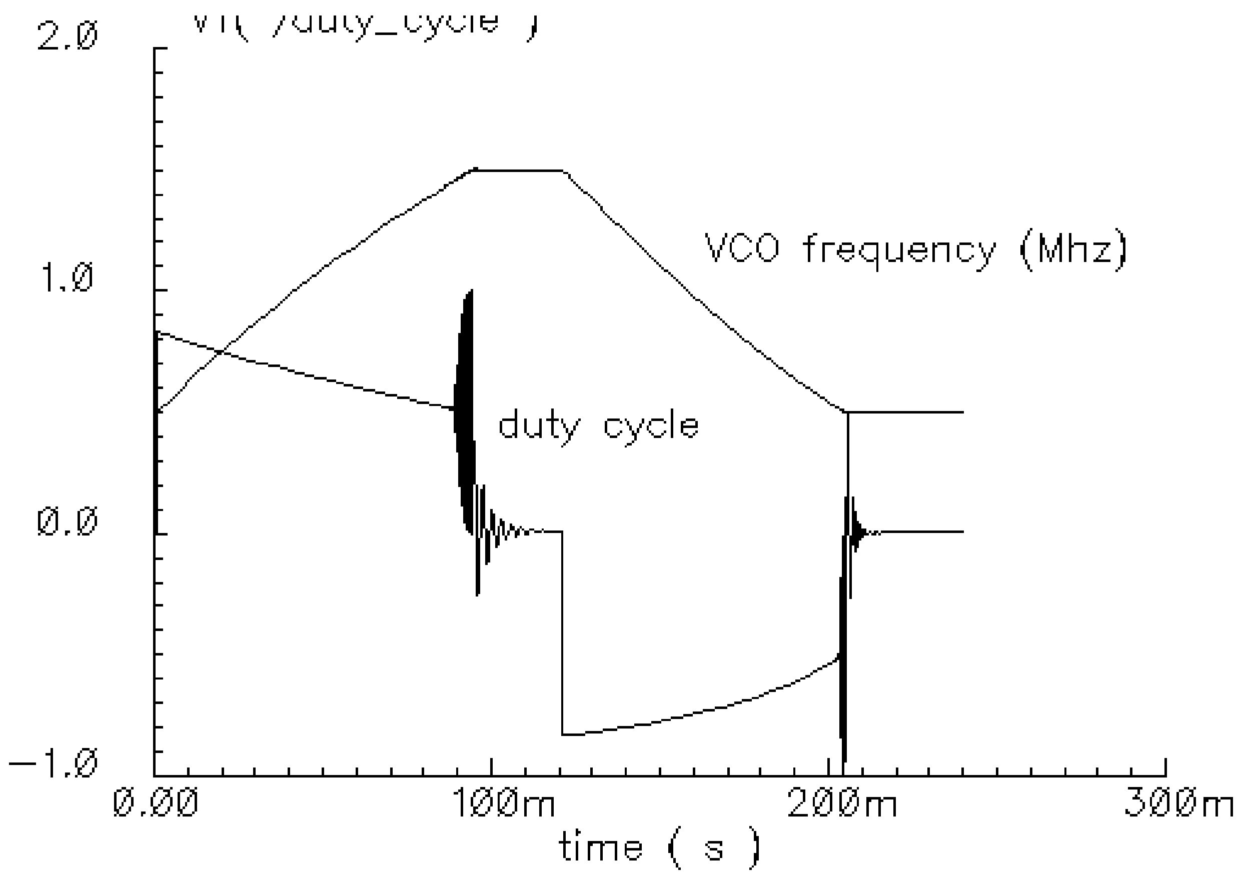


Figure 14, response to momentary change in divider ratio.

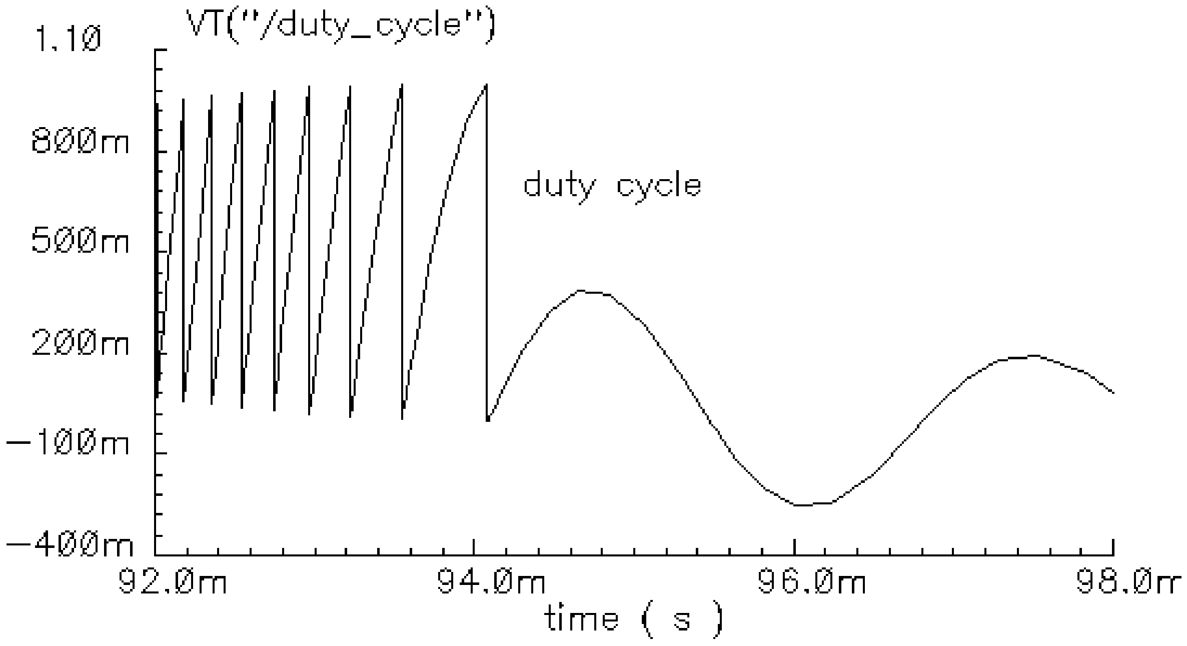


Figure 15, duty cycle during transition from frequency mode to phase mode.

Example 5: Comparison with a voltage-domain model (example_voltage_domain in pllLib).

The only node that can be directly compared between phase- and voltage-domain models is the VCO control node. At full scale the difference between the two models is not visible to the naked eye. The differences occur at the transitions. Figure 16 compares the two models at the transitions. In this example the voltage-domain model runs in 3 hours and the phase-domain model runs in 2 seconds on the same machine.

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 bring these waveforms closer. The reason is that the final transient, the one dominated by phase error, depends on 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 initial conditions among other things. This means the voltage-domain model will show the same level of variation for small differences in the initial conditions preceding the trip to the new equilibrium. Figure 17 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 blue 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.

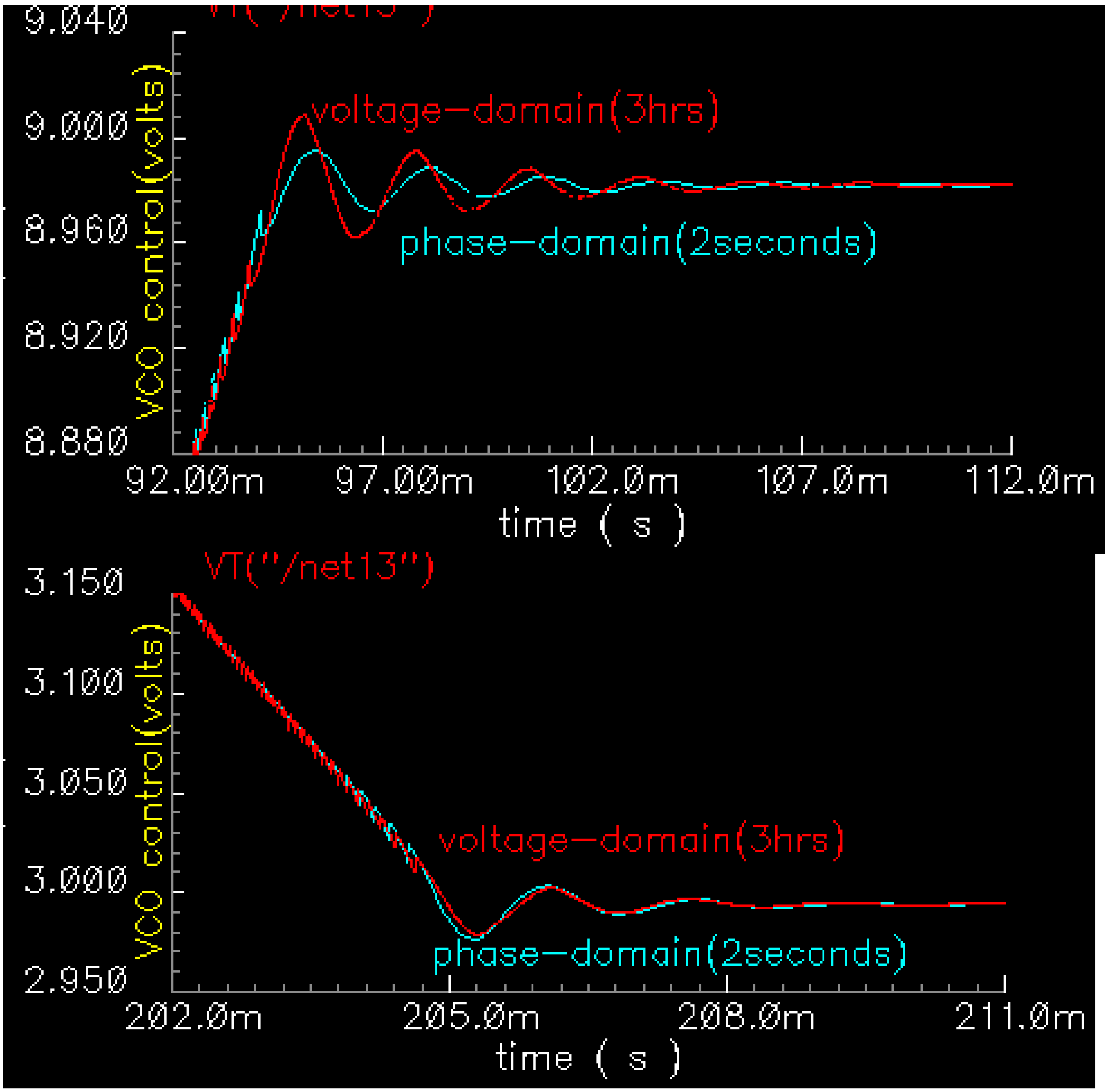


Figure 16, comparison with voltage-domain model

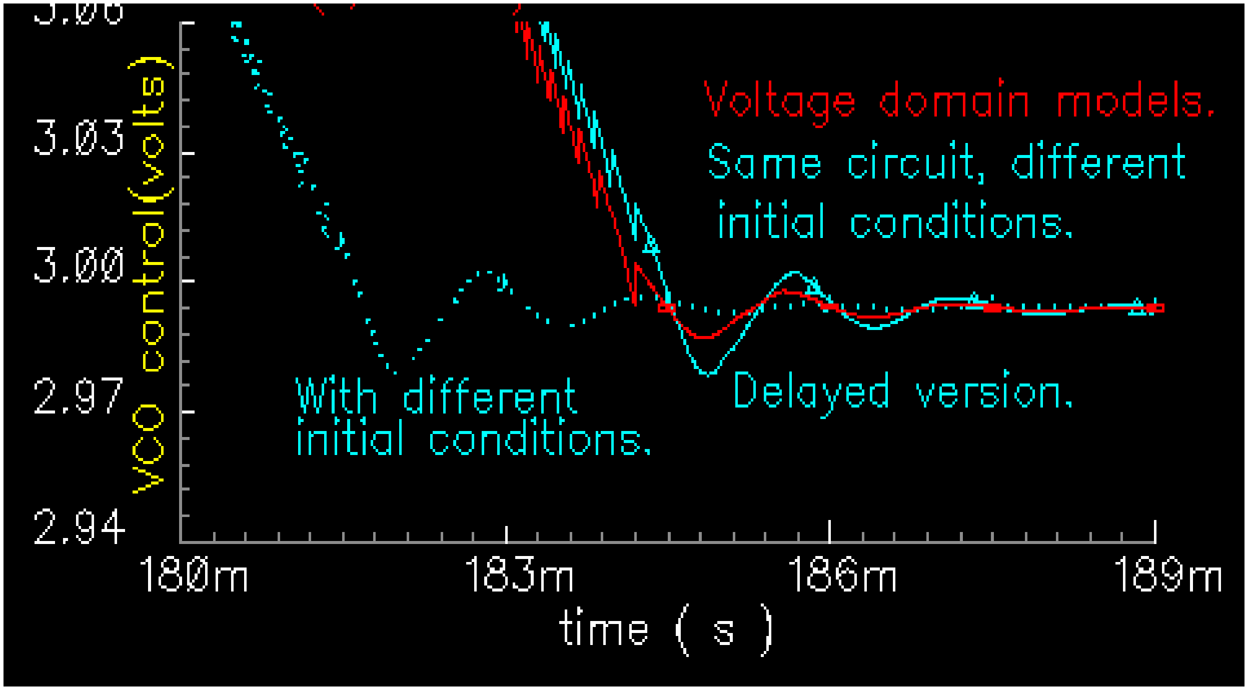
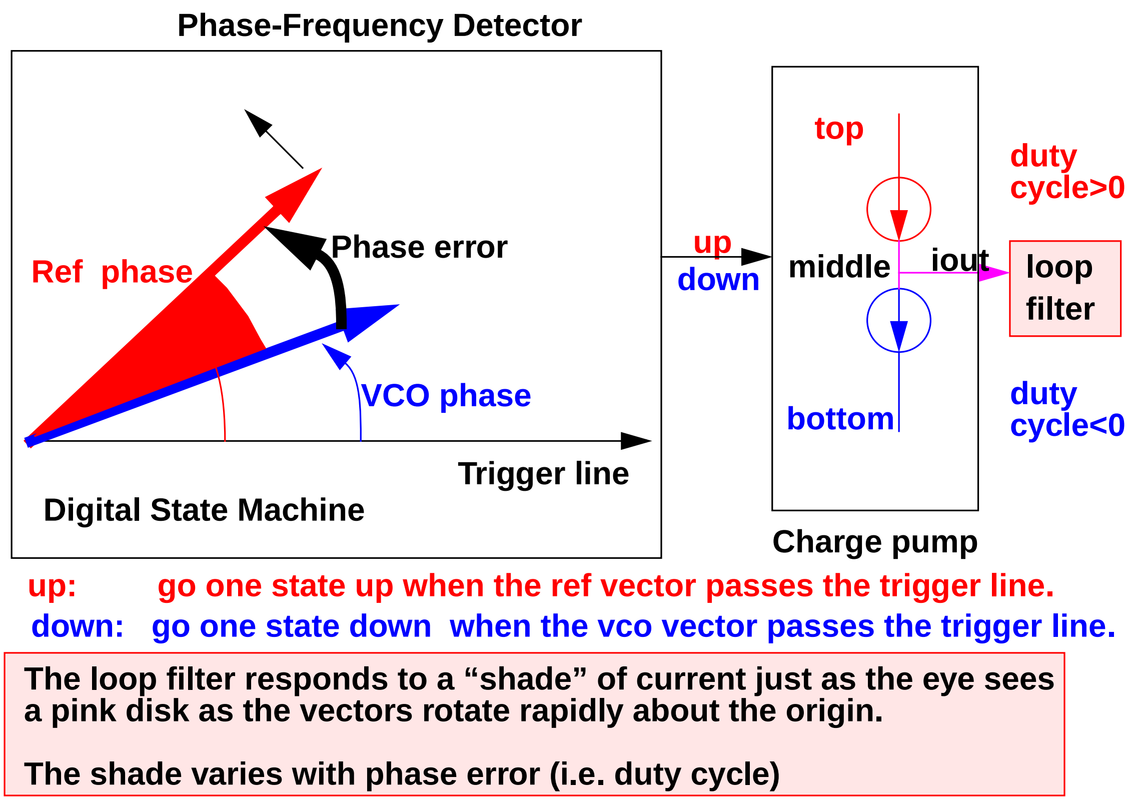


Figure 17, Effect of initial conditions.

Appendix: 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 switches. It is convenient to model the PFD in two pieces. The first piece models the state machine and outputs a duty cycle independent of the output stage. The second piece models the output stage. I use the charge pump (CP) here as an example.

Let’s first review how the PFD/CP 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 A1 shows vectoral representations of the reference and VCO clocks [1,4]. Both vectors rotate counter-clockwise round the origin like hands on a clock running backwards. 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 one up. If the PFD is already in the top state it stays there. Whenever the VCO passes the trigger line the state jumps to the next one down. If it's already in the bottom state it stays there. For a fixed phase error, state toggles as the hands rotate. State toggles between middle and top or between middle and bottom states. The percentage of time spent in the top (or bottom) state is the duty cycle. Duty cycle is positive for top-middle toggling and negative for middle-bottom toggling.

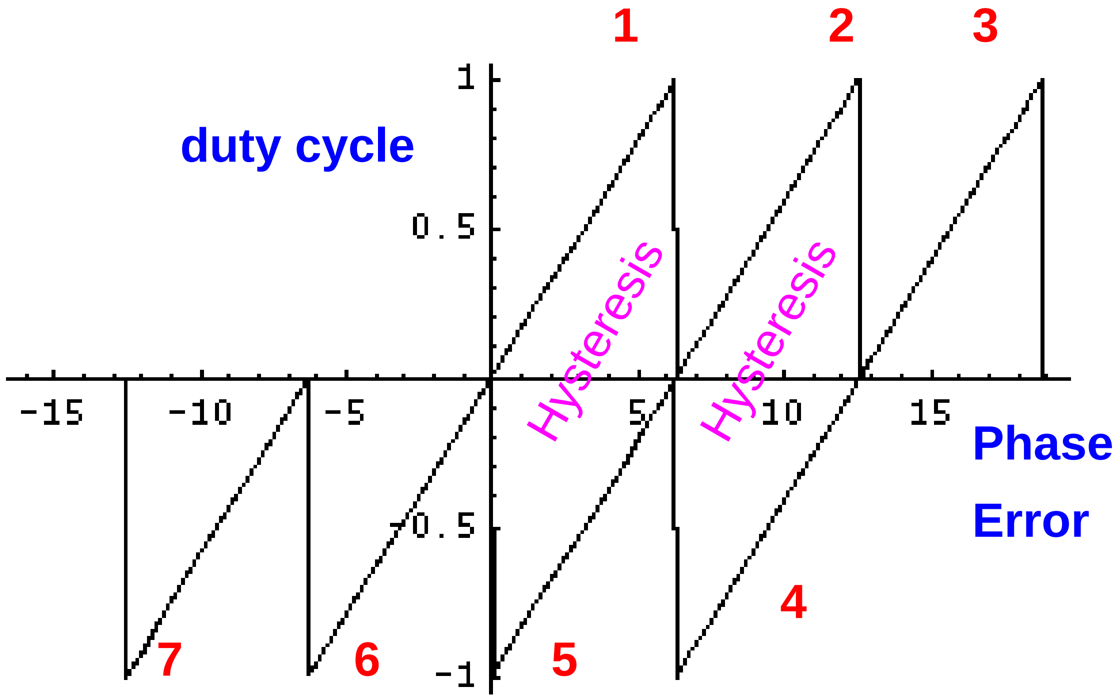

Phase-Frequency Detector
Ref phase
Phase error
VCO phase
Trigger line
Digital State Machine
up
down
top
middle
iout
bottom
Charge pump
duty cycle>0
duty cycle<0
loop filter
up: go one state up when the ref vector passes the trigger line.
down: go one state down when the vco vector passes the trigger line.
The loop filter responds to a "shade" of current just as the eye sees a pink disk as the vectors rotate rapidly about the origin.
The shade varies with phase error (i.e. duty cycle)

Figrue A1, PFD operation

If reference and VCO frequencies are identical the hands in Figure A1 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 we see 2 colors, white and red (if the reference leads). At 2 million revolutions per second we see pink. The shade depends linearly on phase error. Although PFD output current toggles between 2 values, the loop filter and VCO respond mainly to the "shade" of current. The shade is proportional to 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. I refer to these two functions as 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 excursions in steady state phase error away from the origin, the duty cycle is a sawtooth in the upper half plane. It lies in the lower half plane for negative excursions. If an excursion starts off positive, then changes direction after some time, duty cycle crosses zero and becomes a sawtooth in the lower half plane. The (duty cycle)-(phase error) trajectory encloses a non-zero area as shown by the {1-2-3-4-5-6-7} sequence of peaks in Figure A2. This is a good reason for putting the integrator next to the non-linearity: hysteresis involves memory and the integral supplies it.

Locked -> zero frequency error.



This can be modeled by a resettable integrator.

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

Figure A2, duty cycle versus phase error with zero frequency error.

The new model operates only on frequency error. For small frequency errors duty cycle is indeed proportional to phase error. Phase error is the integral with respect to time of frequency error. Duty cycle therefore equals the integral of frequency error divided by 2π. Resetting the integral whenever it hits +-2π produces the multivalued sawtooth described above. If 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 passes shortly after. The reference may pass the trigger line several more times before the VCO passes again. Phase error is still a sawtooth but average duty cycle is nearly 1. This behavior allows the PLL to acquire input signals faster. The function H, in Figure A3 below, shows the averaged unlocked duty cycle. It depends on the ratio of reference to VCO frequencies and is discontinuous where frequency error is zero.

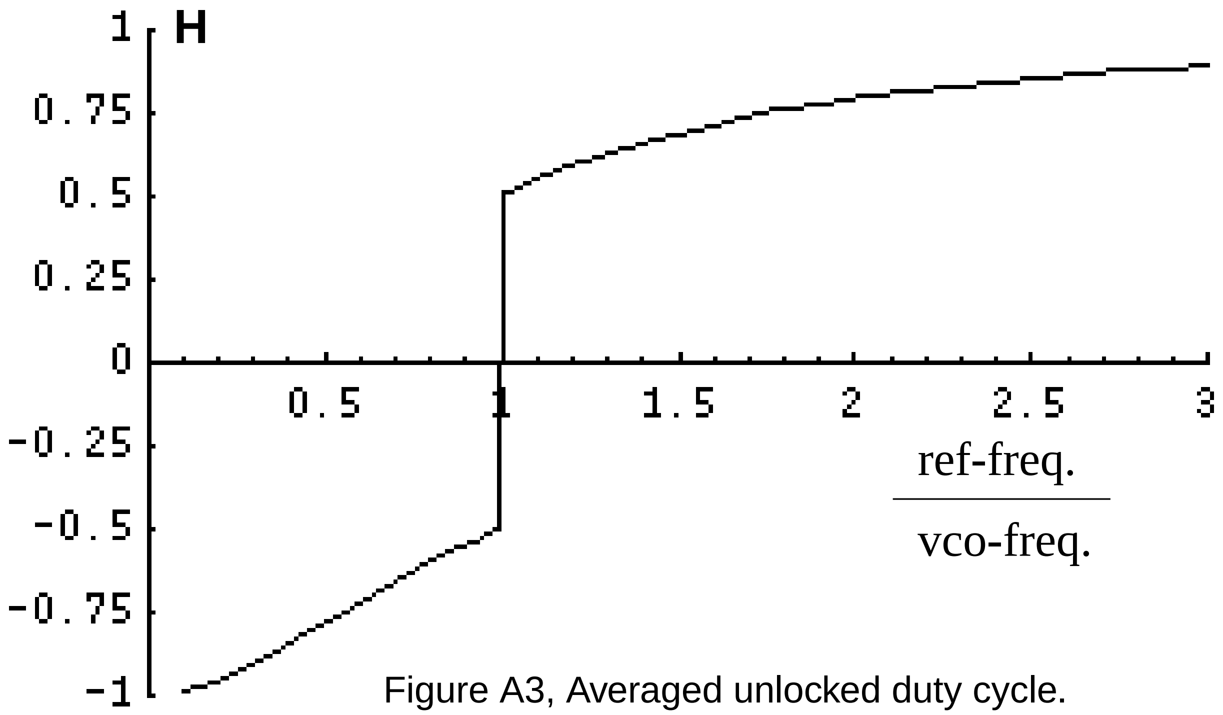
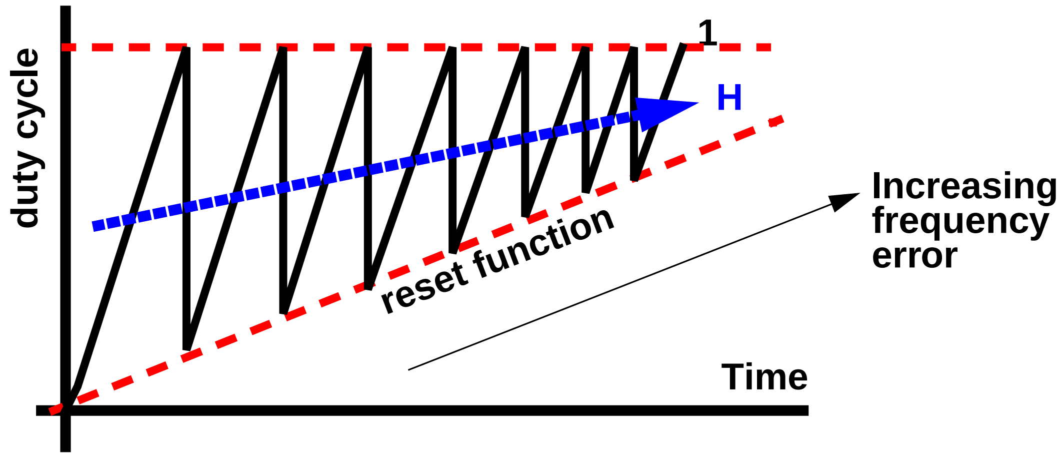


Figure A3, Averaged unlocked duty cycle.

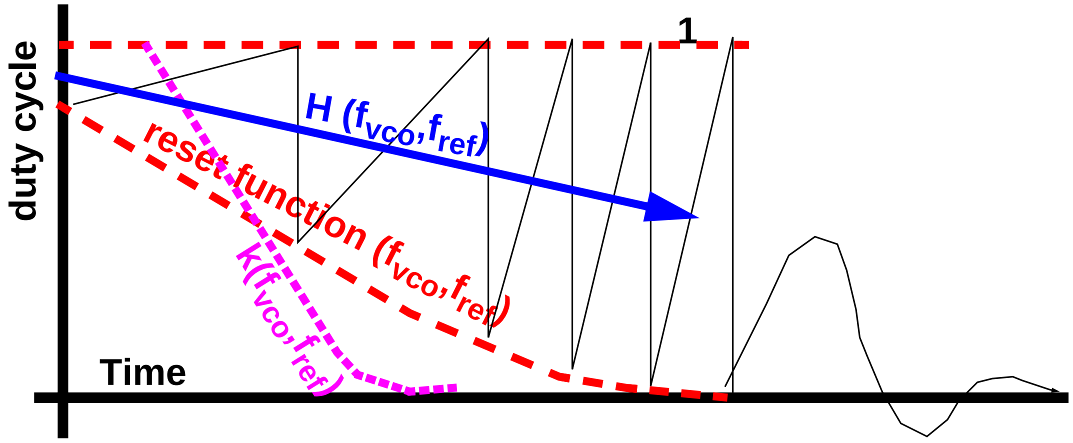
I modified the RI to include the frequency effect. For small frequency errors the predicted average duty cycle equals 1/2 because the RI runs from the reset point (=0) to the reset threshold (=2π). We do not have to reset the integrator to zero. Resetting the integrator to a "reset" function gives the correct average duty cycle (Figure A4). As the frequency ratio goes to +-infinity, the reset point goes to +-2π. Since the reset threshold is still +-2π, the predicted average duty cycle goes to +-1 just as it should. As the frequency ratio nears unity the reset point returns to zero and the predicted average duty cycle returns to 1/2 as it should.



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

Figure A4, Combining averaged locked and unlocked duty cycles.

The state space averaged PFD model is not practical without one more thing. As the reset point nears +-2π the integrator resets almost constantly. Execution stalls. The integrator must be "put to sleep" 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. k approaches 0 as the frequency ratio goes to unity. A factor of (1-k) under the integral puts the integral to sleep for large frequency errors. The resulting PFD model looks like H for large frequency errors and it looks like the "awakened" RI for small frequency errors. k determines how fast the RI awakes and how gradual the model transitions 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 the results could be improved by decreasing the speed_vs_accuracy parameter. If the model resets so many times that it runs too slow, execution can be accelerated by increasing the parameter. The default setting of 50000 should cover a wide range of loop speeds. Figure A5 shows how the transition from frequency-model to phase-mode.



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

Figure A5, Complete model.

References

[1] "Phase-Locked 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]"Non-Linear State Space Averaged Modeling of a 3-State Digital Phase-Frequency Detector", Jess Chen, Cadence Technical Conference, 1997

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

[12] "Macromodeling with SPICE", J.A. Connelly and P. Choi. Prentice Hall. Pagees 168-169.

[13] S. Cuk, Cal. Inst. of Tech. PhD Thesis, "Modelling, Analysis, and Design of Switching Converters". 1977