Finding Your Way Around ADIsimCLK

Let's have a quick look at the gadgets that will help you get around in ADIsimCLK, and to get back to this tutorial.

Tutorial Navigation Buttons

These buttons are located in the lower right-hand corner of this document. They only appear when this tutorial is visible and are used to navigate between tutorial pages. Pressing >> advances to the next page, and << returns to the previous page.

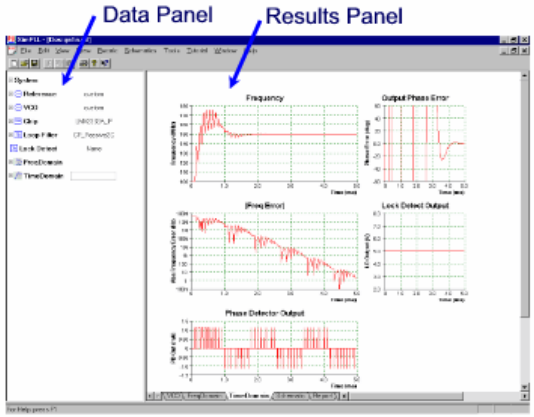
To view the rest of this page you will need to use the scrollbar on the right.

Results Navigation Tabs

The navigation tab bar appears below this tutorial text. It is used to move between pages of results and this Tutorial when it is running. (Normally there is no Tutorial tab.) Click on the other tabs to see the other pages, make sure that you return to this tutorial page when you are ready to continue. If all the other tabs are not visible, use the mouse to drag the splitter bar at the left end of the scrollbar. Alternatively, if you are short of screen space, the arrows at the left end of the tabs can be used to bring the obscured tabs into view.

Main Screen

OK, so you are already finding your way around the ADIsimCLK screen. We just need to name a few parts of the screen. In normal usage the main screen of ADIsimCLK is divided into two parts as shown below:



Data Panel: This is the area of the screen which controls most of the data entry to ADIsimCLK. This is where you alter the loop bandwidth, change the charge pump current, set arm dividers, choose output driver settings and so forth.

Results Panel: This area of the screen is used primarily for displaying pages of results (and also for the tutorial page).

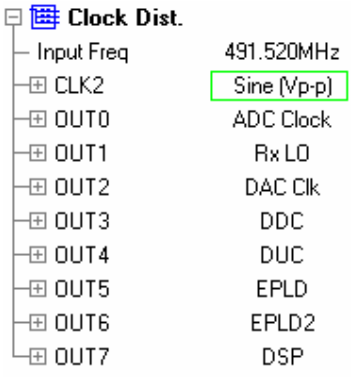
A third area of the screen is the **Message Panel**. This is used for error and warning messages. To conserve screen space this panel appears automatically at the bottom of the screen if there are any messages and hides automatically when there are no messages.

The bars between the various panels can be moved with the mouse if necessary.

Exploring ADIsimCLK - DataPanel

Remember the Data Panel is the area to the left of the screen - it is where you enter data to control ADIsimCLK. Data is organised in a simple tree structure. For example, the clock distribution network configuration is contained in the **Clock Dist** folder.

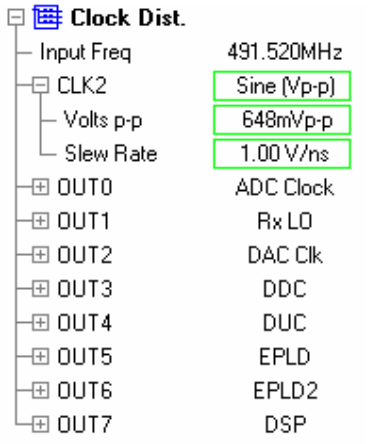
To expand this folder, click on the box with the '+' in it, revealing



Note that the ADIsimCLK convention is that items with a green border around them are ones that you can alter.

For further details on any item in the Data Panel, simply right-click on it and choose "**What's This**" from the menu that appears - try it out!

The input signal **CLK2** has its amplitude specified as a peak-peak voltage. To see further details, expand the row (click on the '+' again), revealing

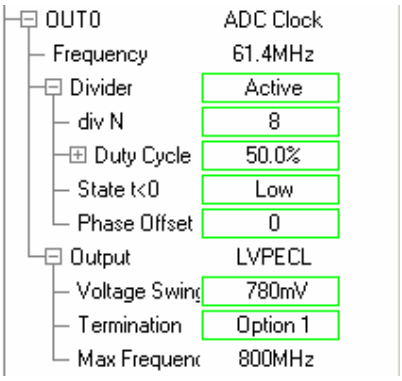


Here we discover that we are specifying the signal at the CLK2 input as a sinusoidal signal, with amplitude of 648mV p-p. ADIsimCLK computes this default amplitude as sufficient to meet the recommended minimum slew rate of 1V/ns at the CLK1/2 inputs. Note that this default amplitude is sufficient to meet the recommended **minimum** slew rate, for extremely low jitter designs higher slew rates are desirable.

The slew rate presented at the CLK1/2 input affects the jitter on the output clocks, so to achieve best accuracy from ADIsimCLK you should ensure that the CLK1/2 input signal specified in ADIsimCLK is representative of the signal in your circuit.

ADIsimCLK has defaulted to specifying the input waveform as the p-p amplitude of a sinusoid, to explore other methods of specifying the input waveform, click on the Sine(Vp-p) box.

Each of the output arms of the Clock Distribution circuit has their own entry in the Clock Dist folder. For example, OUT0 has been given the name 'ADC Clock'. Expanding this gives:



This shows the current configuration of the arm **OUT0.** The divider is dividing by 8 with a 50% duty cycle - click on the 8 and press **Page Up** or **Page Down.**

Similarly, try Page Up/Down on the Duty Cycle, State t<0 and Phase Offset.

Now you have the idea of finding, entering and changing data in the Data Panel.

Exploring ADIsimCLK - Results Panel

The Results Panel is this area - where you are reading this. You navigate between screens here with the Navigation Tabs

Go and explore the Results Pages, note that most items have explanatory notes accessible by right-clicking on them and choosing **What's This.**

Tutorial
This page - remember to come back!

Components
Displays results for various components in the system:
- VCO Phase Noise
- VCO Tuning Law
- VCO Tuning Sensitivity
- Reference / Input Clock Phase Noise
- Clock Filter Response

These results are only displayed if the component is being used in your configuration.

Freq Domain
Results from the frequency domain PLL analysis:
- PLL Loop gain and Phase
- PLL Output Phase Noise
- PLL Leakage Spurs

Time Domain
Results from the time domain PLL analysis:
- Frequency Transient response
- Phase transient response

Schematic
A simplified schematic of the clock distribution circuit, including loop filter details if a PLL is used.

Timing
A Logic Analyser display of the operation of the Clock Distribution circuit.

Report
A summary report of performance parameters.

OUT0 - 7
A page for each output arm with custom results - including phase noise, timing jitter, typical waveforms, SNR etc...

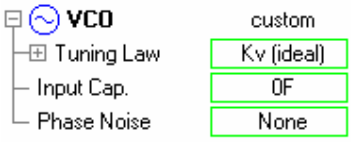
Making the Design Realistic

So far we have set up a Clock Distribution circuit with an idealised VCO (specifying only 25kHz/V - no phase noise) and an idealised Reference Oscillator (30.72MHz - no phase noise). We will now add phase noise to each of these, and then we will be in a position to evaluate and optimise the performance of our circuit.

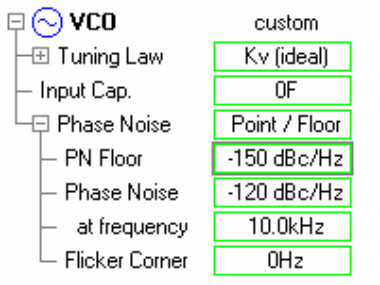
Before we add VCO phase noise, go to the **FreqDomain** page, find the **Phase Noise** graph, right click on it and choose **Save Trace**.

In the Data Panel, open the **VCO** folder and change the **Phase Noise** from **None** to **Point / Floor,** and set the **PN Floor** to -150dBc/Hz and the **Phase Noise** to -120dBc/Hz at 10kHz. Note that you don't have to type the units into the Data Panel, ADIsimCLK will assume that the units are the same as were there before (so to change the **PN Floor**, simply type -150, or press **Page Up** until the correct value appears).

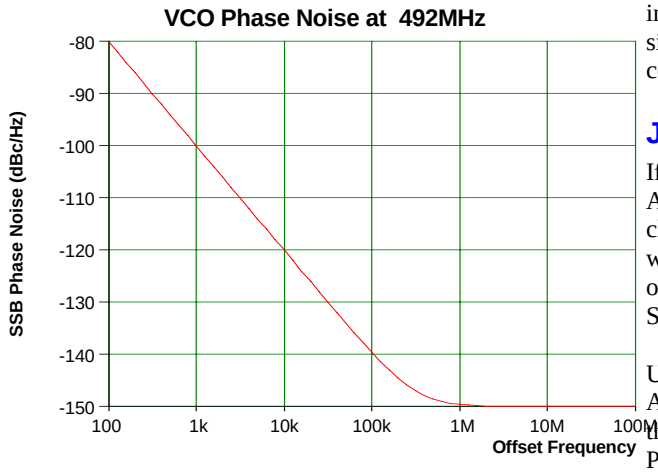
Change:



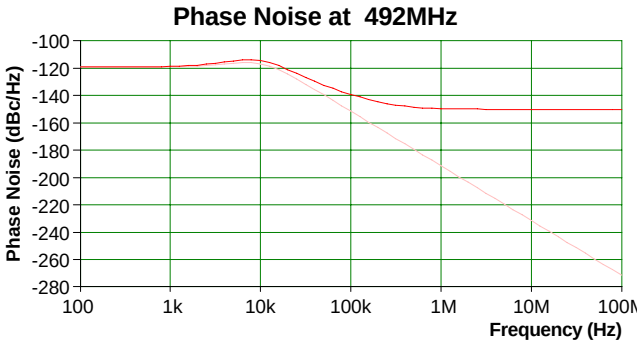
To:



You can see what ADIsimCLK is using as the VCO Phase Noise on the Components Page:

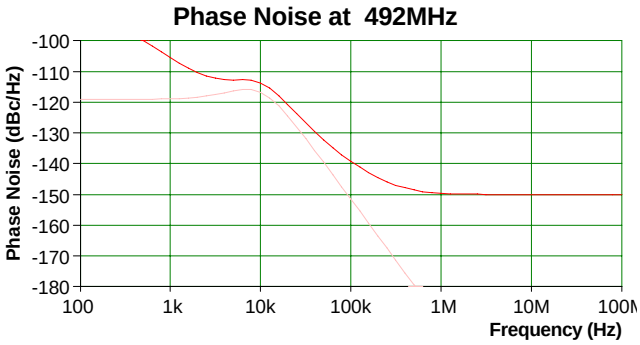


On the FreqDomain page where we saved the Phase Noise trace for the noiseless VCO case, you can see on the Phase Noise plot the current phase noise (with noisy VCO) and the saved phase noise (ideal VCO):



Note that the VCO phase noise dominates outside the PLL loop bandwidth of 10kHz. The bright red trace is the current phase noise (with noisy VCO); the pale red trace is the saved trace. You can turn the saved trace on/off by right clicking on the plot and choosing **view / saved trace.**

In a similar manner, add phase noise to the Reference Oscillator, simply accept the Point / Floor defaults. Now the PLL Phase Noise as plotted in the FreqDomain page shows



Notice that the Reference Phase Noise dominates inside the loop bandwidth. Around 10kHz there is significant contribution from the VCO and the PLL components.

Jitter Performance - OUT0

If you recall in our hypothetical design, OUT0 is the ADC Clock. This is a critical clock for our system, clocking an ADC which is sampling a 100MHz analog waveform. We want to reduce the timing jitter on this output to allow the ADC to achieve in excess of 72dB SNR when sampling a 100MHz sinusoid.

Unfortunately changing pages in this tutorial resets ADIsimCLK to the same state - so you will have to add the phase noise back into the Reference (choose default Point / Floor), and VCO (-120dBc/Hz at 10kHz, floor = -150dBc/Hz);

Click on the Out0 page, you will see a number of plots and a summary:

OUT0: ADC Clock
Frequency: 61.4400MHz
Broadband Timing Jitter = 401fs rms
SNR = 71.97dB ENOB = 12.00bits
at IF Freq = 100MHz
Integrated Phase Noise from 10.0kHz to 20.0kHz
Phase Jitter EVM = 0.0025 %rms
Phase Jitter = 0.001 degrees rms
ACI / ACR = -94.9dBc
Delay from CLK2 to OUT0 is 530ps

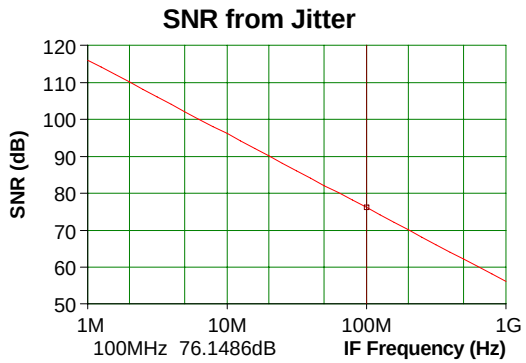
This signal has a broadband timing jitter of 401fs, which contributes a noise floor of -71.97dB to the ADC when sampling a 100MHz analog signal. It is easy to investigate the sources of this jitter in ADIsimCLK; you can turn each of them off:

Disable the reference noise by selecting Reference / Phase Noise = None:
Jitter reduces to 393fs

Leave the reference phase noise disabled and disable the VCO noise by selecting Phase Noise: None
Jitter reduces to 214fs

Thus, the jitter is being dominated by the VCO. Now, is it the 1/f noise or the broadband noise floor? Again, try reducing the floor and you will see that the jitter reduces considerably, reducing the PN Floor to -160dBc/Hz reduces the jitter to 248fs. Clearly obtaining a VCO with a lower phase noise floor will improve the design performance significantly.

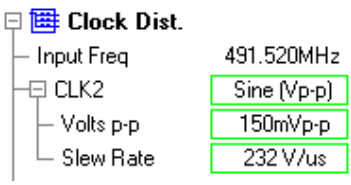
The OUT0 page also shows the plot of SNR for various analog frequencies being sampled by the ADC. For a 248fs timing jitter, the SNR plot is:



For an IF signal of 100MHz, the SNR due to the 248fs of jitter alone is 76dB.

Another potential source of jitter that you can influence is that generated when the clock input signal is digitised by the clock chip at either CLK1 or CLK2.

If there is inadequate slew rate on this signal then significant jitter can be added at this point. To illustrate this, remove the VCO and Reference phase noise so that ADIsimCLK is showing 214fs jitter for OUT0. Go to the **Clock Dist** folder in the Data Panel and adjust the CLK2 input signal level to 150mVp-p (the minimum sensitivity of the AD9510 chip). As the cell shows 648mVp-p you can simply type 150<enter> and the same units (mVp-p) will be retained.



ADIsimCLK automatically computes the slew rate as 232V/us, and the jitter has increased to 490fs. Increase the CLK2 level to 1Vp-p (type 1000<enter> or 1<space><enter> or 1Vp-p<enter>) and the jitter will reduce to 198fs.

Phase Noise Performance - OUT1

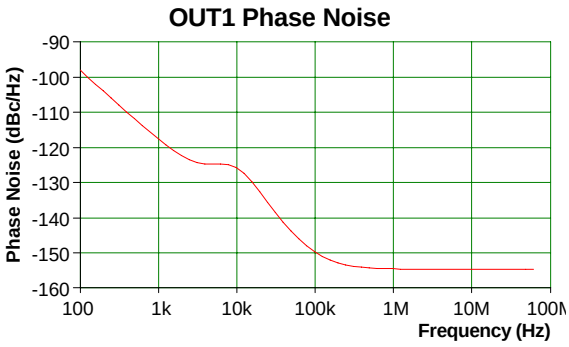
If you recall in our hypothetical design, OUT1 is the Rx LO, and as such there is a requirement on the phase noise to achieve adequate rejection of the adjacent channel.

Unfortunately changing pages in this tutorial has removed the phase noise so please add the phase noise back into the Reference (choose default Point / Floor), and VCO (-120dBc/Hz at 10kHz, floor = -150dBc/Hz);

Click on the OUT1 page, you will see a number of plots and a summary:

OUT1: Rx LO
Frequency: 122.880MHz
Integrated Phase Noise from 100kHz to 300kHz
Phase Jitter EVM = 0.0015 %rms
Phase Jitter = 0.00088 degrees rms
ACI / ACR = -99.3dBc

The OUT1 phase noise plot appears:



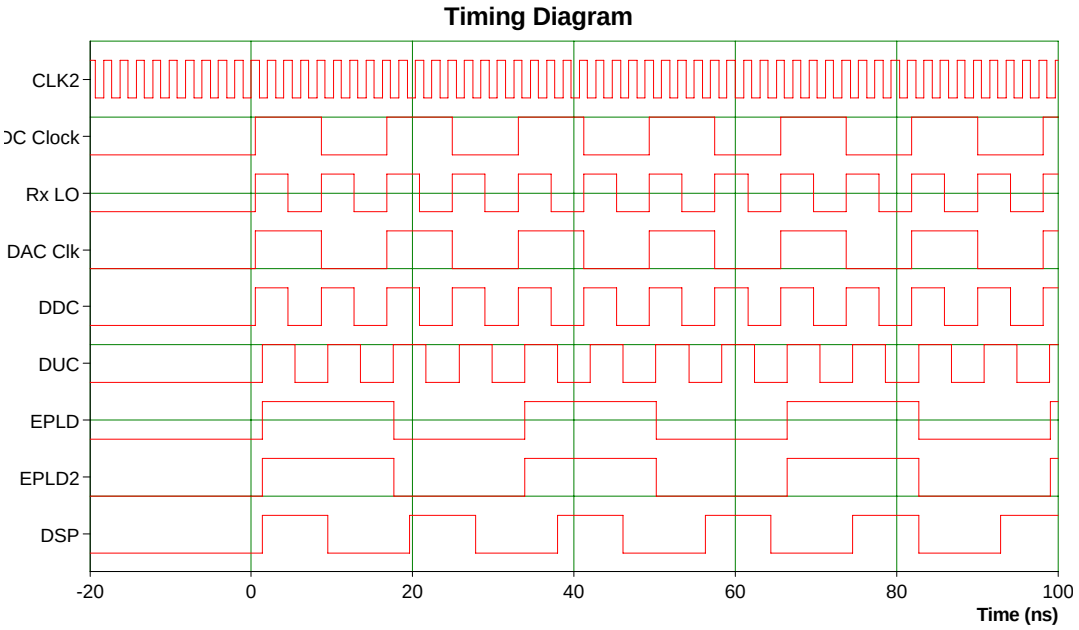
We can see the ultra-low phase noise performance of the AD9510 device, and we also clearly have a low-noise VCXO specified.

The summary result tells that the integrated phase noise power in the 100kHz to 300kHz adjacent channel results in an ACR of 99.3dB.
You can alter any of the parameters in the DataPanel of ADIsimCLK and observe the variation in the Phase Noise and derived parameters on the OUT1 page. For example, try altering the VCO phase noise parameters.

Setting Relative Timing - Timing Analysis

One of the key functions of a clock distribution system is to provide clock signals with definite time relationships to each other. The primary method of observing the timing relationships between the clock output signals from the clock distribution circuit in ADIsimCLK is the Timing Analysis display on the **Timing** page.

Click on the **Timing** page and observe the **Timing Diagram**:



This is a Logic Analyser display of the outputs of the clock distribution circuit.

You can use this as an assistant to configuring each distribution channel. For example, whilst observing the **Timing Diagram**, open the OUT5 folder on the Data Panel and experiment with each of **Duty Cycle**, **State t<0** and **Phase Offset.** Simply click on the cell containing the current value and press **Page Up** or **Page Down** to alter the entry, or type in a new value. For the **Duty Cycle** entry you can expand that line and alter the **High Cycles** and **Low Cycles** directly.

Whilst still in the OUT5 folder, try enabling the **Delay** element, and activate the programmable delay.

The Timing Diagram provides relative timing indication between each clock output. You can alter the range of times analysed in the **Timing An.** folder in the Data Panel. Also by right clicking on the Timing Diagram you can activate a marker, or alter the time scale.

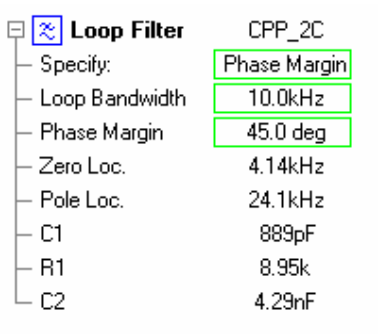
The other useful output to determine relative timings is the delay from clock input to each output that can be displayed for each output. For example on the OUT5 page

OUT5: EPLD
Frequency: 30.7200MHz
Broadband Timing Jitter = 412fs rms
Delay from CLK2 to OUT5 is 1.39ns

This indicates that the edges on OUT5 occur 1.39ns after the rising edge on CLK2.

PLL Loop Bandwidth Optimization

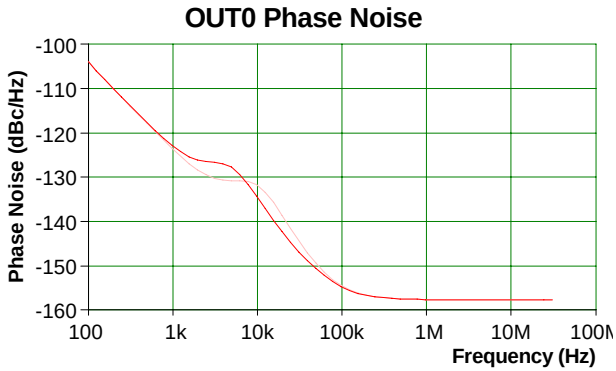
There is no simple golden rule for setting the PLL loop bandwidth. For this example, ADIsimCLK has chosen a default bandwidth of 10kHz and a phase margin of 45 degrees, which can be seen in the **Loop Filter** folder in the Data Panel.



Again, changing pages in this tutorial has removed the phase noise so please add the phase noise back into the Reference (choose default Point / Floor), and VCO (-120dBc/Hz at 10kHz, floor = -150dBc/Hz).

You will find that in this application that with the jitter being dominated by the VCO broadband noise floor, that there is little variation in OUT0 timing jitter as the PLL loop bandwidth is altered. (To change the loop bandwidth, click on the cell containing 10kHz and press **Page Up** or **Page Down**).

If you save the trace for the OUT0 phase noise at 10kHz loop bandwidth, and reduce the bandwidth to 5kHz, you will see a plot like:



The bright red trace corresponds to the current 5kHz bandwidth, the pale red trace is the saved one, corresponding to a loop bandwidth of 10kHz.

The effect of loop bandwidth on the output phase noise / jitter parameter is very dependent on the VCO and Reference phase noise, and the significance of the results is very application dependent. With ADIsimCLK you can see the effect of all these effects immediately.

Of course we should also never lose sight of the fact that we have to build this PLL, so the physical realizability of the loop filter components should also be checked. The values are shown in the Loop Filter folder and also on the Schematic page.

"Building" the PLL

So far we have been designing the PLL to achieve a specified loop bandwidth and phase margin. This has resulted in ADIsimCLK calculating exact component values and performing all analyses with these values. The next phase of the design process is to evaluate the performance of the PLL with real component values and to see how this performance varies with component parameter tolerances.

To 'build' the PLL using nearest real component values in the loop filter select from the main menu **Tools / Build.** This will take the resistor values to the nearest E24 series and the capacitor values to the nearest E12 series. (These options may be altered by selecting **Edit / PLL Options** from the main menu.)

If you look in the loop filter folder now you will find that the component values are user inputs and the Loop Bandwidth and Phase Margin are outputs. The loop bandwidth has changed to 10.3kHz and the phase margin to 47.8degrees.

Note that the **Specify** parameter in the Loop Filter folder has changed from **Phase Margin** to **Components**. We will investigate this more in the next section. The effect of this is that ADIsimCLK stops recalculating the loop filter components every time some parameter changes. Now we have a PLL with C1 = 820pF, R1 = 9.1k and C2 = 4n7. If we alter other parameters such as the VCO Kv or the charge pump current we can see exactly how varying these parameters will affect our "built" PLL.

Simulating existing PLL's

Up until the last page where you "built" the PLL, the PLL you have been investigating has been specified to have a certain loop bandwidth and phase margin. How do we go about modelling an existing PLL where we know its components and not the loop bandwidth and phase margin?

Currently, whenever you changed any loop parameter (Kv, charge pump current etc.), ADIsimCLK recalculated the loop filter component values to achieve the desired bandwidth and phase margin. We think of this as the Loop Filter being specified to ensure that the PLL has a particular bandwidth and phase margin. There are other ways of specifying the loop filter; these are set in the **Loop Filter** folder as the **Specify** parameter, currently set to **Phase Margin.**

There are two other ways to specify the loop filter which you will see if you try to change the current

Phase Margin setting, they are **Pole / Zero** and **Components**.

If you want to:

- hold the component values constant whilst you vary some other parameter (e.g. Kv)
- see the effects of varying a component or
- have a PLL with given component values that you want to model

then you want to specify the Loop Filter by **Components.** This will cause ADIsimCLK to use the component values you specify in its analyses and simulations.

Go into the Loop Filter folder and alter the **Specify** parameter from **Phase Margin** to **Components.** The loop filter is now specified by component values; note that C1, C2 and R1 now are user inputs (the green border). The component values stay unchanged when we change Specify to Components, but now we are now free to change them.

You can now enter a set of component values for an existing PLL if desired. Note that you will usually have to set the frequency requirements (frequency range and channel spacing) as well as the chip and VCO.

The other choice under **Specify** is **Pole / Zero**. Using the **Phase Margin** option which we did for most of the tutorial, the dominant pole and zero are located with geometric symmetry around the unity gain frequency in order to maximize the phase margin at that frequency. This is normally a good compromise; however in some special cases (for example modulation loops) having control over the pole and zero locations enables further optimization. If you select **Pole / Zero** you will be able to position the poles and zero at any location that results in a realizable filter for the specified loop bandwidth.

Working with ADIsimCLK

Now that you have seen how the major features of ADIsimCLK operate we hope that you will explore with your own designs.

Each new design commences with the New Design Wizard and you can access the wizard at any later time to alter the configuration using the main menu (**Edit / Configuration)**, or using the  button on the toolbar.

At any stage of the design process you can save the design file (a .clk file). This saves the workspace, everything in the Data Panel and the locations of the graphs on the results pages, but currently does not save the marker locations, custom graph scales or saved traces in the graphs.

To get a hardcopy from ADIsimCLK, the results pages and the report can be printed. Simply click on the page you wish to print (to ensure it has the focus) and then click on the print icon on the toolbar. Our preferred method of operation is to work with a word processor open (e.g. Microsoft Word, or even WordPad that comes with Windows) to keep progress notes and to paste in the results consisting of graphs, reports and schematics. To copy a graph simply right click on it and select copy. In your word processor select paste. Do the same for schematics, on the report it is necessary to select the text you want to copy - or right click and choose **select all**, then **copy.**

Explore the library files - these are simple text files and you can easily make your own, for example to use the measured tuning characteristics and phase noise of a VCO. They are in the ADIsimCLK\Lib directory. Format details for library files are given in the on line help.

We welcome comments and suggestions; feedback may be posted on our user forums, or sent via our website: **www.radio-labs.com**

What's Next

You have reached the end of the ADIsimCLK tutorial where we have shown you many of the features of the ADIsimCLK package. Further details can be found in the on-line help.

For more information on any of the Analog Devices high performance Clock Distribution products, please visit

www.analog.com/clocks

ADIsimCLK utilises the proven phase noise modelling techniques utilised in SimPLL and ADIsimPLL.

ADIsimPLL is the Analog Devices PLL Design and Simulation software available from

www.analog.com/pll

For technical support, Users' Forum and for further details on the SimPLL package for PLL Design and Simulation visit the Applied Radio Labs website:

www.radio-labs.com

Applied Radio Labs welcome any comments and suggestions on our software. These may be sent by email to feedback@radio-labs.com or via our website above.

ADIsimPLL and ADIsimCLK are trademarks of Analog Devices
SimPLL is a trademark of Applied Radio Labs