CHALMERS

Modelling PLLs used for frequency generation in radio base stations

Master of Science Thesis in Radio and Space Science

MARTIN FAXÉR

Chalmers University of Technology
Department of Signals and Systems
Göteborg, Sweden, May 2011

Modelling PLLs used for frequency generation in radio base stations

Martin O. Faxér

Examiner: Thomas Eriksson

Chalmers University of Technology
Department of Signals and Systems
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Signals and Systems
Göteborg, Sweden May 2011

Prepared (also subject responsible if other)		No.		
EMARFAX				
Approved	Checked	Date	Rev	Reference
		2011-01-17	PA1	

Modelling PLLs used for frequency generation in radio base stations

Martin Faxér

Master thesis at Ericsson AB

Supervisors: Mikael Hörberg, Ronny Hansson (Ericsson AB)

Examiner: Thomas Eriksson (Chalmers)

Acknowledgement

I would to thank everyone who has in any way contributed to the thesis or helped make it happen.

First and foremost I would like to thank my supervisors at Ericsson, namely Mikael Hörberg and Ronny Hansson, for their support and assistance throughout the thesis. Thanks as well to Magnus Molander for letting me work in his team.

I would also like to extend a big thank you to my examiner at Chalmers, Professor Thomas Eriksson, for taking the time to supervise me and giving many insightful comments and suggestions.

Thanks to Oliver for being the opponent at my presentation and a great friend. Thanks to Johan for proofreading the report and giving very many good comments. Thanks to all other classmates and other friends as well, for all the fun we've had.

Last but not least, I would like to thank my family and all loved ones.

Prepared (also subject responsible if other)		No.		
EMARFAX				
Approved	Checked	Date	Rev	Reference
		2011-01-17	PA1	

Abstract

PLLs are vital parts of almost all current communication systems. As such, it is of great importance to be able to accurately simulate them. In this thesis PLL theory is explored and simulation is performed using Agilent ADS. The focus is on frequency generating PLLs and related figures of merit such as frequency error, phase error, phase noise and EVM. Theoretical explanations are made as to how it is possible to calculate some of the above figures of merit given a PLL's phase noise curve. Simulations are performed using behavioural models and are very general in nature. Steady state noise phase noise simulations are performed and results found to agree with theory and other simulation tools. Test benches are provided for transient simulations simulating phenomena such as the appearance of reference spurs, pushing and pulling. The results from the pushing and pulling simulations are not very accurate due to the behavioural components not being able to simulate all imperfections found in a real VCO. In addition to pure PLL simulations, system level simulations are performed on a WCDMA system with a transmitter using a noisy LO (PLL with phase noise) and the resulting EVM is measured. The system simulations show that the PLL's phase noise has most effect on the EVM in a WCDMA system in the 1000 Hz – 2 MHz frequency range.

Contents

1 Introduction

This thesis aims to give some insight into how to model phase-locked loops used for frequency generation in radio base stations, specifically using the simulator software known as Advanced Design System (ADS®) from Agilent Technologies.

A radio base station is a device which is at the other end of a phone call made using a mobile telephone. The mobile telephone communicates with the base station using electromagnetic waves in the radio part of the spectrum[1], hence the name radio base station. What happens next involves a lot of different steps, but in a simplified fashion one might say that after the base station, the call is routed to controlling equipment and then through the so called core network, which generally consists of wired optical links. When the call reaches its destination, it is once again converted to radio waves and sent to the receiving mobile telephone by another radio base station. This is illustrated in Figure 1[2].

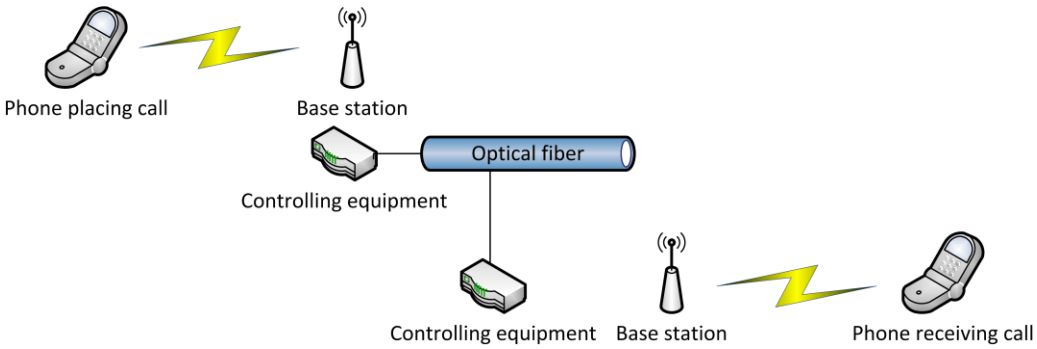


Figure 1 Simplified diagram showing how a mobile telephone call is made.

In order to be able to transmit and receive at the correct frequencies, it is necessary for the radio base station to have a system for frequency generation, responsible for generating electrical signals oscillating at the desired frequencies. This is normally accomplished by the use of a voltage controlled oscillator (VCO) connected in what is known as a phase-locked loop (PLL). The phase-locked loop is a feedback loop in which the oscillator is set to match the phase of a given reference signal. When the two signals are locked in phase they will also be locked in frequency. The output signal from the PLL will thus be an electrical signal oscillating at the reference (or a multiple thereof) frequency.

[1] Radio frequencies are generally defined as lying between 30 kHz and 300 GHz.
[2] The figure and related explanation is of course heavily simplified, for a more detailed explanation one might refer to a book such as [1].

Aside from frequency generation, PLLs may be used for clock recovery (in communication systems where the clock signal is not explicitly known, but has to be extracted from the received data), jitter cleaning (in the case where a reference signal is available but very noisy), demodulation of AM/FM modulated signals etc.

The goal of the thesis has been to investigate PLL theory and how to simulate the workings of a PLL in a modern simulation environment such as ADS. It was desired to have a number of “test benches” available to simplify different types of PLL simulations (phase noise response, lock-in time, spurious frequencies, pushing and pulling). A special focus has been put on the phase noise aspects of a PLL system and how different design trade-offs affect the resulting phase noise profile of the complete PLL. The simulations have been done in general terms and no specific PLL has been designed in the thesis. The simulations involve both circuit-level simulations of a PLL (albeit using behavioural models) and system-level simulations with the phase noise profile of a PLL used to describe the up-mixing local oscillator in a WCDMA[3] system.

In chapter two VCO and PLL theory will be explored. The third chapter details the practical simulation aspects of simulating PLLs in ADS. The fourth chapter presents results and conclusions drawn from simulations. The fifth and final chapter offers some discussion on the results obtained.

[3] Wireless Code Division Multiple Access, a popular standard for 3G mobile telecommunication networks.

2 Theory

This chapter aims to give a theoretical introduction to how a PLL and the components of which it is made up work. Transfer functions are derived to illustrate how noise injected at different places along the loop affect the output of the PLL. An introduction is given to different types of noise and how they affect a PLL system.

2.1 PLL building blocks

A PLL is made up of a number of different components such as a VCO, a phase detector, a loop filter and dividers. It is possible and very common to buy entirely integrated PLL solutions on a chip but most commercial solutions require at least an external VCO. The VCO and PLL are thus two separate physical components, even though the VCO is usually included as part of the PLL on a conceptual level.

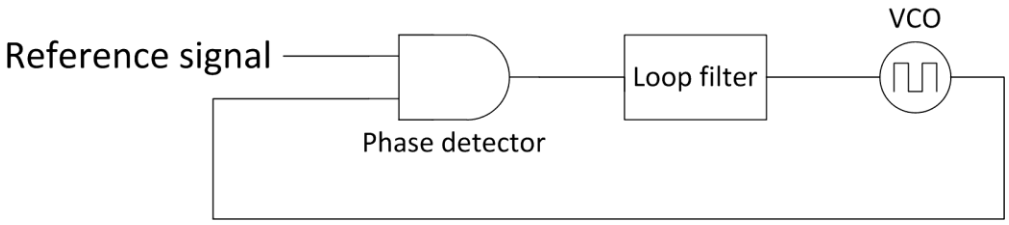


Figure 2 Schematic block diagram of a simple PLL circuit, not including dividers.

A schematic diagram of a basic PLL circuit is shown in Figure 2. The input is given by a reference signal source. The phase detector compares the VCO output signal to the reference signal and generates a voltage proportional to their phase differences. This voltage is then fed through a low-pass loop filter and into the voltage controlled oscillator. The VCO output is then fed back to the phase detector. This results in a feedback loop, where the voltage controlled oscillator will be tuned to match the phase (and frequency) of the input reference signal.

PLLs find use in a variety of different areas such as jitter cleaning (where a noisy reference signal is cleaned of jitter before entering a circuit), frequency synthesis (where one stable reference frequency source is used to create other frequencies that are multiples of the reference) and clock recovery (in cases where a data signal is sent without an explicit clock signal) [3]. Thus, PLLs are a lot more versatile than they might seem at first, when one might assume that they simply make a “copy” of the reference signal.

The theoretical operation of the different PLL components will be explored in more detail in the following sub-chapters.

2.1.1 Voltage controlled oscillator (VCO)

The heart of the PLL is the voltage controlled oscillator (VCO). An oscillator is a component which has no inputs and outputs a signal that oscillates at a certain frequency. Normally, in harmonic oscillators, the frequency of oscillation is decided by a part of the oscillator called the “tank circuit”, which exhibits electrical resonance at the desired oscillation frequency. The tank circuit could for example be made up of a simple LC resonator, as shown in the left side of Figure 3.

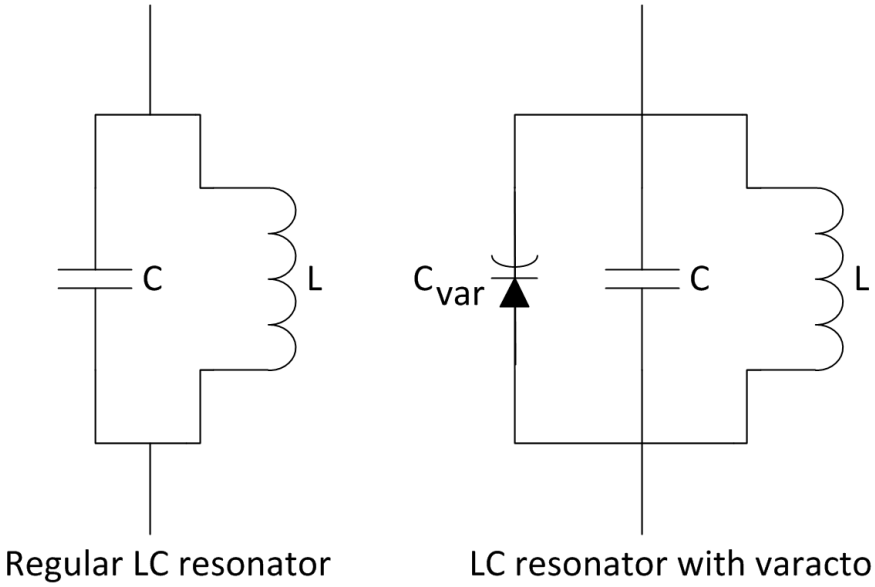


Figure 3 Circuit diagrams illustrating a) a basic LC resonator and b) an LC resonator with an added varactor (variable capacitor) to enable tuning of the oscillation frequency.

In the case of an LC resonator the oscillation frequency is given by [2]

$$f_{\text{osc}} = \frac{1}{2\pi\sqrt{LC}} \tag{1}$$

where L is the inductance and C is the capacitance. As can be seen, the oscillation frequency depends on both the inductance and the capacitance. Thus, if one has the means of varying one of them, it is possible to tune the oscillation frequency.

In a VCO, tuning is usually achieved by using a component which capacitance can be varied (called a varactor, which is a voltage controlled capacitor), as illustrated in the right side of Figure 3. In theory, variable inductors would work as well but they are more complicated to manufacture [2]. The frequency of oscillation can be tuned by supplying a control voltage which fine-tunes the varactor's capacitance. The VCO's output frequency is ideally described by

$$f_{\text{out}} = f_0 + K_{\text{v}} v_{\text{tune}} \tag{2}$$

where f_{out} is the output frequency, K_{v} is the VCO's tuning gain (usually specified in MHz/V) and v_{tune} is the tuning voltage applied to the VCO's input. Note however that this equation would give perfectly linear tuning and an infinite tuning range. In reality (2) holds true over a certain limited range defined by the varactor's voltage tuning range. This defines the VCO's tuning range.

Other important figures of merit for VCOs are their pushing and pulling characteristics. Pushing is defined as the VCO's response to a changed supply voltage. A small voltage ripple on the supply line for example may affect the transistor biasing or couple to the tune line and result in a periodic change of the VCO's output frequency.

Pulling is defined as the VCO's response to a changed load impedance. Ideally the output should be buffered enough to not be affected by a changed load impedance but in practice this is usually not the case. Thus a change in the load impedance will be reflected back and affect the voltage seen at the PFD in addition to changing the tank's reactance (and thus altering the oscillation frequency).

Another very important figure of merit for a VCO is its phase noise. All practical oscillators exhibit a certain phase noise. The output power spectrum contains energy not only at the fundamental oscillation frequency but also at near-lying frequencies. At very far away frequencies the phase noise is given by the thermal noise floor. This is illustrated in Figure 4 (adapted from [4]).

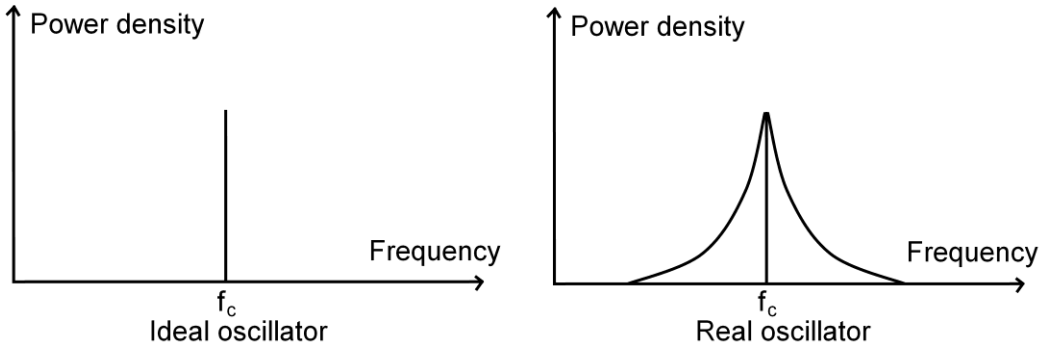


Figure 4 Power density spectrum of an ideal and a real oscillator. The real oscillator will have a significantly broadened spectrum due to phase noise (adapted from [4]).

2.1.2 Phase-Frequency detector (PFD)

The phase-frequency detector is a more advanced version of a component known as the phase detector. A phase detector is a component which compares the phase of its two input signals and then gives an output signal that is proportional to their phase difference, in accordance with

$$v_{\text{out}} = K_{\text{d}}(\Theta_{\text{ref}} - \Theta_{\text{VCO}}) \tag{3}$$

where v_{out} is the output voltage, K_{d} is the phase detector's gain, Θ_{ref} is the phase of the reference signal and Θ_{VCO} is the phase of the VCO's output signal. Thus a large phase difference will give rise to a large output voltage. It will however wrap around at large phase differences (e.g. $\pm 90°$ or $\pm 180°$, depending on the type of phase detector used). Such a phase detector, wrapping at $\pm 180°$ is displayed in Figure 5 (adapted from [5]).

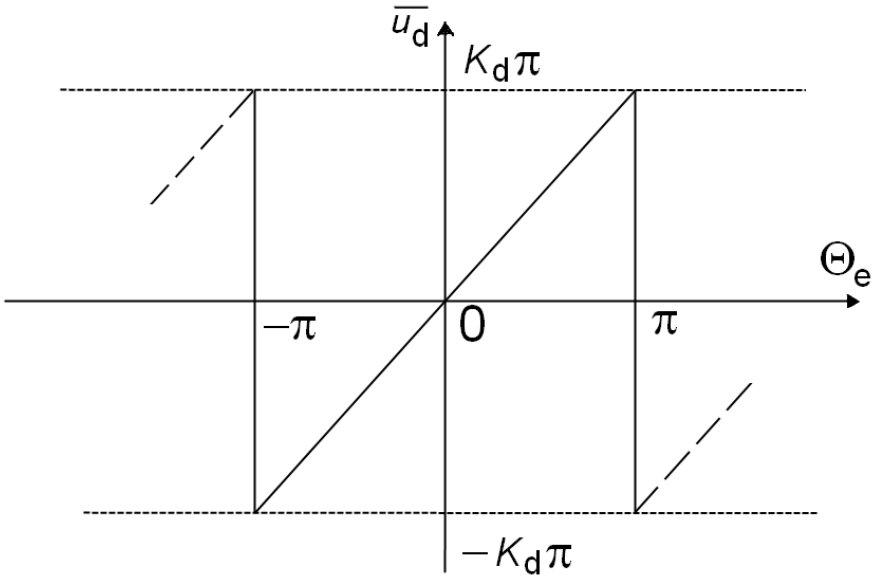


Figure 5 Mean output signal ($\overline{u_d}$) versus phase error (Θ_e) when using a phase detector (PD) implemented using a JK flipflop. The output signal will wrap for phase differences larger than ±180° (adapted from [5]).

A phase detector can for example be implemented by an XOR logic gate or a JK flipflop. A downside to the phase detector is that it has a limited pull-in range. For large frequency differences the pull-in process (the process of reaching a locked state) will be very slow and if the frequency difference is large enough it simply won't happen [5]. The phase-frequency detector (PFD) improves on this by providing an output that's not only proportional to the phase difference but also to the frequency difference between the two input signals. It is usually implemented by a simple state machine triggering on positive flanks seen on its input signals, such as the one shown in Figure 6.

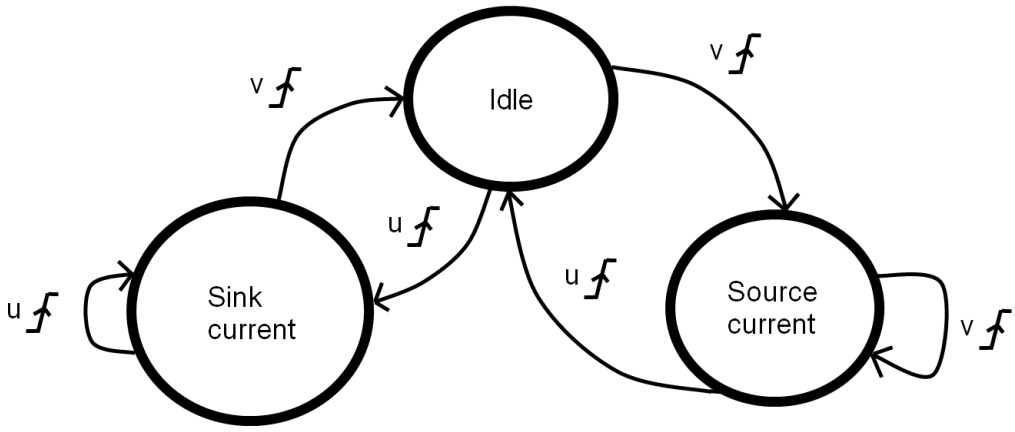


Figure 6 State diagram describing the operation of a phase frequency detector (PFD) with two input signals, u and v. The triggers are positive flanks seen on the signals.

A phase detector built around this principle will spend most of its time in the same state as long as one frequency is much larger than the other, leading to a much faster pull-in process. There are many different types of PFDs. A popular type of PFD is the “charge pump” PFD, which uses a capacitor as an integrator of the phase error. A charge pump PFD can have either voltage or current output. Current output is to be preferred, since it reduces spurious frequencies (“spurs”) seen at the output [5]. A charge pump PFD with current output includes two current sources able to sink (consume) and source (produce) current.

A PFD will not wrap around at phase differences that are larger in magnitude than $180°$. It will continually output an almost full magnitude correction pulse until it reaches the desired frequency. Thus, a PFD provides much faster pull-in times than a phase detector and will be able to obtain a lock for arbitrarily large frequency differences. PFD characteristics are shown in Figure 7 (adapted from [5]), as compared to Figure 5.

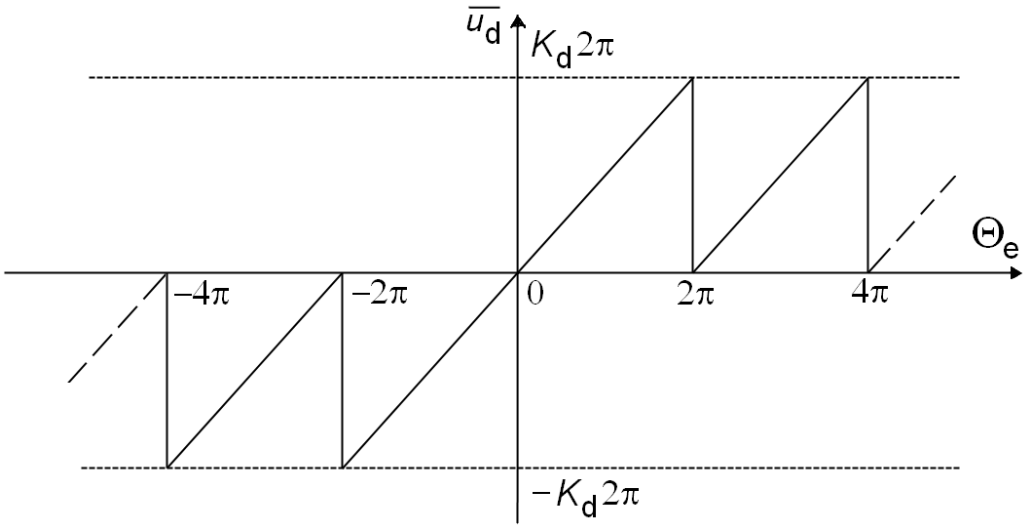


Figure 7 Mean output signal ($\overline{u_d}$) versus phase error (Θ_e) when using a phase frequency detector (PFD) (adapted from [5]).

2.1.3 Loop filter

The so called loop filter is a low-pass filter placed at the output of the PFD. Most phase detectors and phase-frequency detectors output a correction pulse in the time span between the rising edges of the reference and VCO signals. If the two signals are perfectly aligned in phase, there will ideally not be any output at all. If the signals are not in phase, the output pulse from the PFD will be proportional to their phase difference. The PFD output signal thus resembles a square wave signal with a limited duty cycle.

In order to smoothen the PFD output signal (to get it to resemble a DC voltage in case of a voltage output PFD), a low-pass filter is used. The filter may be either passive or active. Active filters include operational amplifiers. From a noise perspective, a passive filter is always to be preferred since operational amplifiers will generate a lot of noise. Active filters might however be needed for example in the case where the VCO requires a tuning voltage larger than what the PFD is able to supply [3].

Loop filters can have different orders, depending on the number of poles present in the filter's transfer function. A second order loop filter would look as shown in Figure 8[4]. Its transfer function may be derived (using Kirchoff's voltage and current laws) to be

$$F(s) = \frac{v_{\text{out}}}{i_{\text{in}}} = \frac{1 + sR_1C_1}{s(sR_1C_1C_2 + C_1 + C_2)} \tag{4}$$

where the component variables are according to Figure 8.

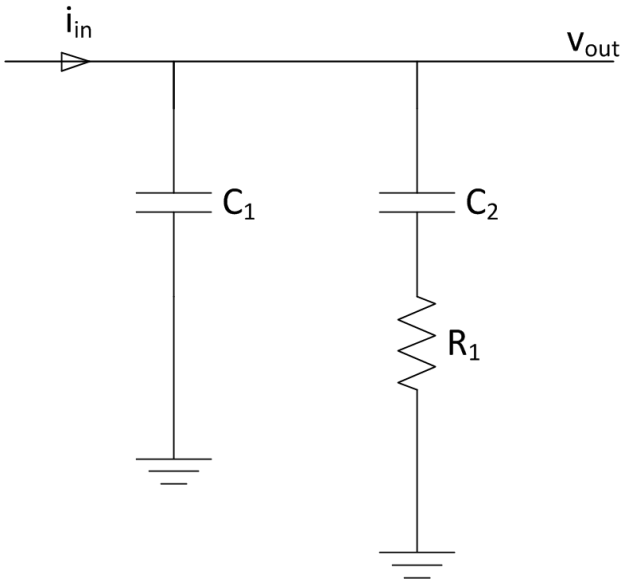


Figure 8 Second order loop filter for use in a PLL with a current output PFD.

The loop filter has a very profound effect on the PLL characteristics. It is one of the major "user-tuneable" parts of a PLL. As will be seen in later chapters, the so called "loop bandwidth" of the loop filter sets a cut-off frequency for the frequency up to which reference noise will be passed on to the output of the PLL. It also has a large effect on the "speed" of the PLL, i.e. its lock-in time (the time it takes to obtain phase lock on a reference signal).

[4] Note that this loop filter is intended for use with a PFD with current output. For a PFD with voltage output a series resistor would have to be added.

In order for a PLL to be unconditionally stable (i.e. always able to maintain phase lock on the reference signal) one needs to ensure that the phase margin of the loop filter is sufficient. In addition to the phase margin the loop bandwidth also has a profound impact on stability if it is too large. As a rule of thumb, one should not use a loop bandwidth that's larger than about one tenth of the comparison frequency[5] in order to avoid stability problems [3]. Theoretically it is impossible to make a second-order loop filter unstable as long as the loop bandwidth requirements are fulfilled.

2.1.4 Divider

An important component in most PLLs is the frequency divider[6]. Frequency dividers are normally used on the PFD inputs, to divide down the reference and VCO frequencies. It is necessary for them to be at the same frequency for comparison purposes. The VCO frequency is generally much higher than the reference frequency and thus needs to be divided down.

Frequency dividers are usually implemented in digital logic by using a cascade of flip-flops [5]. A simple cascaded chain of flip-flops only provides power-of-2 division but other division ratios may be obtained by use of additional logic. One might also use a digital counter clocked by the input frequency that generates an output pulse for each *n*th input pulse.

The relationship between the output frequency of a PLL and the divider ratios is given by

$$f_{\mathrm{PLL}} = \frac{N}{R} f_{\mathrm{ref}} \tag{5}$$

where f_{PLL} is the output frequency of the PLL, N is the VCO output divider ratio, R is the reference input divider ratio and f_{ref} is the reference frequency. These dividers are shown in the PLL schematic shown in Figure 9, as compared to Figure 2, which does not include dividers.

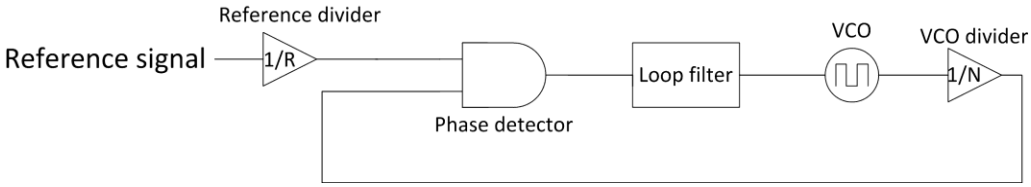


Figure 9 Schematic block diagram of a PLL, including dividers on the reference and VCO signals. Note that the dividers are frequency dividers. In the frequency or phase domain they would simple be modelled as gain blocks.

[5] The comparison frequency is the frequency which the VCO's output is divided down to and then compared to the reference at. It may be the same as the reference frequency or the reference frequency divided by an integer value.

[6] Another name for a frequency divider is "prescaler", although in the context of PLLs some people might separate prescalers from the frequency dividers and only use the word prescaler to refer to high-frequency dividers present before the "regular" frequency dividers.

Note that the dividers shown in Figure 9 are integer dividers, i.e. they divide the signal's frequency by an integer. A PLL utilising these types of dividers is known as an integer-N PLL. In addition to this, there are other types of PLLs as well, known as fractional-N PLLs. Fractional-N PLLs incorporate dividers that can divide by a fractional ratio. Fractional dividers are usually implemented as so called delta-sigma modulators, which is in essence an integer divider with a time-varying divisor. By appropriately varying the divisor in time, fractional division ratios may be obtained. For example, by varying the divisor as 100, 100, 100, 101, 100, 100, 100, 101, … in time, a division ratio of 100.25 would be obtained.

Switching the division ratio between 2 numbers as demonstrated in the example above would be called a first order delta-sigma modulator. A third order delta-sigma modulator would instead vary between $2^2 = 4$ divisor values to achieve the desired fractional ratio. Likewise, higher orders would vary between 2^n divisor values.

2.2 Transfer functions

A PLL can be analysed mathematically by the use of transfer functions, well known from control theory. A basic transfer function defines the relationship between the input and the output of a linear time-invariant system. Viewed in the frequency domain, the frequency response will usually vary depending on the frequency of the input signal. A number of transfer functions can be developed in order to describe different aspects of the PLL.

An important use of transfer functions is to investigate how noise injected at different places along the PLL loop will affect the PLL output. For example, noise present in the reference signal will be low-pass filtered (due to the loop filter), where as noise output from the VCO will be high-pass filtered. This can readily be shown by deriving the transfer functions. Another important use of transfer functions is to analyse the time domain (transient) response of PLLs when applying a frequency step on the reference input for example. This can be used for deriving lock-in times etc.

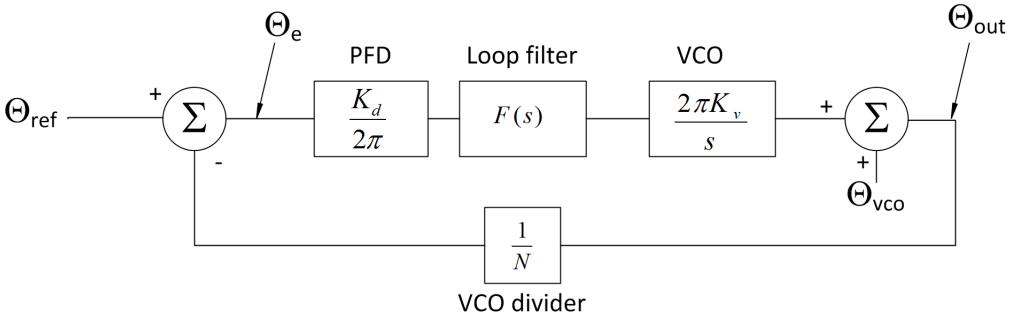


Figure 10 Schematic diagram of a PLL illustrating the transfer functions for the different parts in the phase domain.

Figure 10 shows a basic PLL schematic with its various components and their respective transfer functions. It is important to note that this is a phase domain model, i.e. the inputs to the PFD and the output of the VCO are modelled as phase signals. This makes sense since phase is the principal quantity of importance in a PLL. In addition to that, the transfer function for frequency signals would be equal to the one for phase signals, since frequency is the derivative of phase.

Using the linear PFD approximation, its output will be proportional to the phase error so it can be modelled as a simple gain step. This holds true at least for “small” phase errors. Many PFDs suffer from non-linearities for example in the way of having a so called “dead zone”, which will be covered in detail in the following chapter. A simple gain step model is however accurate enough for most purposes.

The VCO’s transfer function can be seen to be $\frac{K_{\mathrm{v}}}{s}$, i.e. an ideal integrator. This can be explained by the fact that the VCO’s output frequency depends directly on its input signal, in accordance with (2). Knowing that frequency is the derivative of phase, one may conclude that phase is the integral of frequency. Thus, the VCO’s output phase is the integral of its input signal [12].

The loop filter’s transfer function in the Laplace domain may be derived using KCL and KVL as mentioned and demonstrated in the chapter about the loop filter.

The dividers are simply modelled as being gain steps with a gain of $\frac{1}{N}$ [8].

Using the above definitions and schematic diagram, one may write an expression for the PLL’s open-loop gain as

$$H_{\mathrm{OL}}(s) = \frac{K_{\mathrm{d}}}{2\pi} F(s) \frac{2\pi K_{\mathrm{v}}}{s} \frac{1}{N} \tag{6}$$

which is the gain expression of interest when defining a PLL’s loop bandwidth and phase margin. The loop bandwidth (or loop filter cut-off frequency) is defined as the frequency at which the open-loop gain has a magnitude of 1 (or 0 dB). The phase margin is defined as the phase of the open-loop gain at that frequency minus -180 degrees.

In order to examine the transient response of a PLL after applying a frequency step at its input, for example, one may be interested in the relation between the reference input phase and the phase error. This may be derived as

$$\Theta_{\mathrm{e}}(s) = \frac{s}{s + K_{\mathrm{d}} K_{\mathrm{v}} F(s)/N} \Theta_{\mathrm{ref}}(s) \tag{7}$$

where $\Theta_{\text{ref}}(s)$ is the reference phase and $\Theta_{\text{e}}(s)$ is the phase error. Using this relation, it is possible to perform an inverse Laplace transform in order to obtain the time-domain response. It is important to keep in mind that a frequency step on the reference input would correspond to a phase ramp in the phase-domain.

Other transfer functions of interest are the ones showing how noise injected at different parts of the PLL affect the output phase. Intuitively, one may reason that noise coming from parts of the circuit before the loop filter will be low-pass filtered, since the loop filter is a low-pass filter. The exception to this would be noise coming from the VCO, which would be injected after the loop filter and thus be high-pass filtered instead. This is indeed the case and the PLL's final noise properties will be defined by reference and PFD noise inside the loop filter bandwidth and by VCO noise outside the loop filter bandwidth.

In order to show this qualitatively, one may derive transfer functions for both VCO and reference noise. The VCO noise transfers as

$$H_{\text{VCO}}(s) = \frac{\Theta_{\text{out}}(s)}{\Theta_{\text{VCO}}(s)} = \frac{sN}{sN + K_{\text{v}}K_{\text{d}}F(s)} \tag{8}$$

which will give a high-pass response. In order to be able to derive the transfer function for VCO noise to the output, one needs to keep in mind that the system is LTI. Since it is LTI, the superposition principle applies and the reference noise may be set to 0. The reference noise, on the other hand, transfers to the output as

$$H_{\text{ref}}(s) = \frac{\Theta_{\text{out}}(s)}{\Theta_{\text{ref}}(s)} = \frac{NK_{\text{v}}K_{\text{d}}F(s)}{sN + K_{\text{d}}K_{\text{v}}F(s)} \tag{9}$$

which will give a low-pass response, due to the lone s in the denominator. Using these transfer functions, it is possible to predict the impact noise coming from both the reference and the VCO will have on the output. Noise coming from other parts of the circuit such as the PFD, dividers, etc. will be transferred in a fashion similar to the reference noise, i.e. low-pass filtered, but with some scaling.

2.3 Noise

Noise is an inescapable property of all practical electrical circuits. Noise basically means that in addition to the desired voltage there will be a stochastically varying noise component, slightly offsetting the desired voltage level.

There are many different causes of noise. One type of noise that is present everywhere and in all electrical circuits is thermal noise. Thermal noise is caused by the random motion of electrons (or holes, if they are the dominant charge carrier) when they are heated above the absolute freezing point (0 K). Thermal noise voltage in a resistor can be described by the equation

$$v_{\mathrm{n}}^2 = 4kTR\Delta f \tag{10}$$

where v_{n} is the noise voltage, k is Boltzmann's constant, T is the temperature, R is the resistance and Δf is the bandwidth [2]. To calculate the noise power for a given bandwidth and temperature, utilising the fact that in a circuit matched for maximum power transfer half the voltage drop will be over the generator and half the voltage drop over the load, one may use the following relation

$$P = \frac{\left(\frac{V}{2}\right)^2}{R} = \frac{V^2}{4R} = kT\Delta f \tag{11}$$

which can be written in dBm units as

$$P_{\mathrm{dBm}} = 10\log_{10}(kT\Delta f \times 1000) \tag{12}$$

since dBm is defined as decibels in relation to 1 mW. This leads to the conclusion that for a resistor at room temperature ($T = 293$ K), the thermal noise in a 1 Hz bandwidth will give a noise power of

$$P_{\mathrm{dBm}} = 10\log_{10}(k \times 293 \times 1 \times 1000) = -174\,\mathrm{dBm} \tag{13}$$

The above might also be converted into a dBc[7] noise floor, by subtracting the carrier power in dBm from the noise power. Thus, for a VCO with an output power of 10 dBm, the thermal noise floor would be at -174 - 10 = -184 dBc. This is of course not achievable in practical circuits but fixed-frequency crystal oscillators can come pretty close (around 13 dB higher [6]).

[7] dBc stands for decibel in relation to the carrier, i.e. it is defined as the signal power in dBm (or any other decibel unit) minus the carrier power in dBm.

In addition to thermal noise, circuits like VCOs also suffer from shot noise and flicker (also called $1/f$) noise. Shot noise is caused by the fact that the charge carriers are so few in number (relatively speaking) that the electrical properties can vary significantly depending on when the circuit is observed. Flicker noise is a bit more esoteric and the mechanisms causing it are not fully explained theoretically. What is evident however is that this type of noise has a clear frequency dependence with noise power that depends on the inverse of the frequency [2]. In the case of an oscillator, the low frequency noise gets upconverted to the carrier frequency [7].

2.3.1 Phase noise

In theory, the output of an ideal oscillator can be described as a simple sinusoid. In reality, however, there are no ideal oscillators. All oscillators suffer from a certain amount of noise, which leads to deviations from the ideal behaviour. One such deviation is so called phase noise. The output from a practical oscillator can be described by a mathematical function such as

$$s(t) = A(t)\cos(\omega_0 t + \Phi(t)) \tag{14}$$

where $A(t)$ describes amplitude variations and $\Phi(t)$ describes phase variations (phase noise). Amplitude noise is generally not a big problem, since it tends to be much smaller than the phase noise and is relatively easily filtered by using limiters [7]. Phase noise, on the other hand, is very hard to get rid of.

The random phase variations result in a broadening of the fundamental frequency line in a power spectrum density plot, as seen in Figure 4. This means that the oscillator outputs energy not only at the oscillation frequency but also at frequencies close to the oscillation frequency (and in some cases quite far away). The phase noise close to the carrier is known as “close-in” phase noise whereas the phase noise further away is known as broadband phase noise.

When specifying phase noise for an oscillator, a power spectral density plot is usually given (which is usually referred to as $L(f)$). This plot usually shows the phase noise power in dBc/Hz. In the past there have been varying mathematical definitions of $L(f)$ but the current standard as adopted by IEEE specifies that it should be defined as

$$L(f) = \frac{1}{2} S_{\Phi}(f) \tag{15}$$

where $S_{\Phi}(f)$ is the single sideband power spectral density of $\Phi(t)$ [13]. The above definition may be hard to relate to, but put in words one may say that $L(f)$ is defined as the amount of power in a 1 Hz bandwidth at the specified frequency offset, compared to the carrier power.

Phase noise can be very troublesome in communication systems, for example in situations where one needs to up-mix or down-mix a signal (i.e. move a baseband signal to a higher frequency or extract a baseband signal from an RF signal by moving it to a lower frequency using a mixer). If the local oscillator (LO) signal is given by an oscillator with phase noise (which it always is in practical cases), it will “smear” the baseband signal and the resulting upmixed signal will be a frequency domain convolution of the LO and IF signals. For example, in the case where a baseband signal at 1 MHz is to be upconverted to a frequency of 100 MHz one needs an LO at 99 MHz. Ideally the resulting RF signal at 100 MHz would only contain energy at exactly 100 MHz. Due to phase noise it will however contain energy at frequencies near 100 MHz as well. This is illustrated in Figure 11.

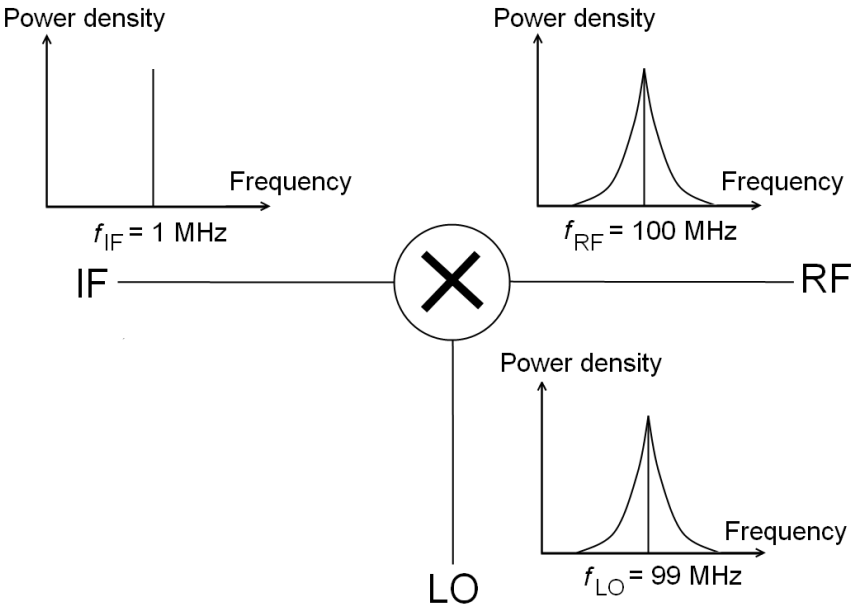


Figure 11 Upmixing situation where an IF at 1 MHz is being upmixed to 100 MHz by use of a noisy local oscillator (PLL).

Phase noise can also lead to a phenomenon which is known as reciprocal mixing. Reciprocal mixing occurs in receivers when an RF signal is to be downmixed to an intermediate frequency. If other signals are present on the RF input at nearby frequencies, they might “leak” into the frequency space where one would expect to find the desired signal on the IF side, due to smearing caused by phase noise. If the nearby disturbing signal is stronger than the desired signal, the disturbing signal may even be the dominant one at the IF frequency.

2.3.2 Noise in a VCO and PLL

As mentioned above, phase noise in a VCO is usually caused by many different mechanisms, some of them being thermal noise, flicker noise and shot noise. Since flicker noise has $1/f$ characteristics, it will be very dominant close to the carrier but will at some point be drowned in thermal noise. Close to the carrier frequency a VCO will generally display phase noise that varies with the inverse of the cube of the frequency, i.e. as $1/f^3$. This is the region where flicker noise is dominant. The reason for the inverse cubic f dependence (compared to simply inverse f dependence as expected for flicker noise) is that the VCO's thermal noise gets upconverted and transfers to the output with a $1/f^2$ dependence. Multiplied by flicker noise ($1/f$) it will result in a $1/f^3$ dependence. Further out, where the flicker noise has become small, a simple $1/f^2$ dependence due to thermal noise will be seen. At some point the noise level will hit the wideband thermal noise floor and flatten out. It is generally agreed upon that $1/f$ noise in the actual transistors used is responsible for the $1/f^3$ noise seen in a VCO, but studies have shown that the cut-off frequency for $1/f^3$ noise may not be the same as the transistor's flicker noise cut-off frequency [7]. A typical VCO noise spectrum is shown in Figure 4.

In addition to the phase noise shown in the figure, which is relatively uniform, the power spectral density plot of a VCO might display unwanted peaks at certain frequencies. These are known as spurious frequencies or "spurs" for short. Spurs can be caused by a number of different things. Some of them are fundamental, such as spurs at the harmonic frequencies of a VCO. These will always be present but are generally easy to filter since they will be far away from the carrier frequency. Other spurs can be more obscure in nature and can for example be caused by electromagnetic interference from a trace close to the VCO on the PCB.

When using a VCO connected in a phase-locked loop, the output power spectral density will most likely display a number of additional spurs, compared to the pure VCO output. For example, one will often see a spur at the PFD's comparison frequency and harmonics thereof. This spur is usually caused by the PFD's so called "dead zone" or transistor mismatch.

The PFD's dead zone is a region where the phase error is so small that the PFD is unable to generate an output signal to correct it. Since the transistors used in the PFD have a certain turn-on time, if the phase error is so small that it would lead to an output pulse shorter than the transistors' turn-on time, no output will be produced. This problem can in theory be avoided by using a PFD with current output and introducing something known as an anti-backlash pulse. The anti-backlash pulse basically ensures that the PFD will keep both of its current outputs (sink and source) active for a certain minimum time each cycle. This delay should then be chosen to be at least as long as the transistors' turn-on time, in order to ensure that even a very small phase error will result in a correction pulse which is longer than the turn-on time. It should however not be too long either, as that will make the reference spur problems worse [9].

In theory, the charge pump's anti-backlash pulses to source and sink current will be of equal magnitude and equal length. In reality, however, the transistors used are never perfectly calibrated, which leads to a small mismatch in the amount of current sourced/sunk. This mismatch leads to one of the pulses being longer than the other which in turn leads to a ripple on the tune line producing a reference spur. The reference spur may also be caused by the comparison frequency simply leaking through the PFD from the reference side to the output side due to poor isolation. In addition to spurs at the comparison frequency, in a fractional-N PLL one may see spurs at fractions of the comparison frequency due to its fractional division [3].

Noise in any form, whether it be thermal noise, spurious noise or flicker noise, will lead to the VCO/PLL not behaving as it would be expected to in an ideal noiseless world. The deviation from the ideal behaviour can be measured in a number of different ways. One simple way is to simply produce a power spectral density plot of the VCO's/PLL's output (i.e. a phase noise plot) to clearly show in which frequency regions the noise lies and how large it is compared to the carrier signal. Such a plot will however consist of very many data points and can become impractical in cases where one would like to express the performance with a single number. For such cases, a number of different measures of an oscillator's noise performance have been developed.

A commonly used measure of an oscillator's noise performance is its phase error. Noise will lead to fluctuations in the oscillator's phase. The phase error is then defined as the instantaneous deviation in phase from the ideal value. One may also calculate a root-mean-square (RMS) value for the phase error. In addition to the phase error, one may calculate another quantity known as the timing jitter, which is directly related to the phase error. Timing jitter is a time-domain phenomenon and describes how the zero crossings of the oscillator waveform vary in time. This is of great importance for clocking applications, i.e. when one uses a PLL as a clock source to drive for example an ADC/DAC.

Another measure of an oscillator's noise performance is its frequency error. Instantaneous frequency is defined as

$$f(t) = \frac{1}{2\pi}\frac{\mathrm{d}\Phi}{\mathrm{d}t} \tag{16}$$

where $\Phi(t)$ is the phase. The frequency error is then defined as the deviation in frequency from the ideal frequency. As thus, the frequency error of course bears some relation to the phase error but for a given RMS phase error it is not possible to calculate the corresponding frequency error since one needs several data points to perform the derivation.

For digital communication systems, especially those using some sort of IQ modulation, it is common to use an error measure known as the error vector magnitude (EVM). IQ modulation exploits the fact that one may use both the amplitude and the phase to carry information in a signal. A signal may be written in general form as shown in (14). Assuming that one has a phase reference by which to decode the phase, both $A(t)$ and $\omega(t)$ may be used to transfer information. In practise it is however much easier to modulate a signal's amplitude than its phase. To exploit this, one may use a so called IQ modulator.

An IQ modulator utilises the fact that (14) may be rewritten, using standard trigonometric identities, as

$$s(t) = I(t)\cos(\omega_0 t) - Q(t)\sin(\omega_0 t) \tag{17}$$

where $I(t) = A(t)\cos(\Phi(t))$ and $Q(t) = A(t)\sin(\Phi(t))$. Hence one can produce an output signal with both modulated phase and amplitude by simply supplying two signals with modulated amplitude [10]. Multiplication and addition is simple to implement in hardware using mixers and combiners.

When using IQ modulation, one may produce a so called I-Q plot. An I-Q plot uses the I (in-phase) signal as x coordinate and the Q (quadrature) signal as y coordinate in a 2-dimensional coordinate system. One may then assign different "codes" to different regions of the I-Q plane. This is how many digital (and some analogue) communication schemes work. One may then use EVM as a measurement of how accurate the modulation is. In a non-ideal world, each received (or transmitted) code will be represented by a vector in the I-Q plane which will differ slightly from the ideal reference vector, due to noise.

The difference vector between the ideal reference and the actual transmitted or received vector is called the error vector magnitude. It is defined as

$$\mathrm{EVM}(\%) = \sqrt{\frac{|\vec{v}_\mathrm{e}|^2}{|\vec{v}_\mathrm{r}|^2}} \times 100 \quad (18)$$

where $\vec{v}_\mathrm{e}$ is the error vector and $\vec{v}_\mathrm{r}$ is the reference vector. In words, it is the square root of the ratio of the power of the error vector to the power of the reference vector. Different cellular standards usually place some sort of restriction on the size of the EVM for a transmitter, in order to avoid coding errors. An example I-Q plot with EVM illustrated is shown in Figure 12.

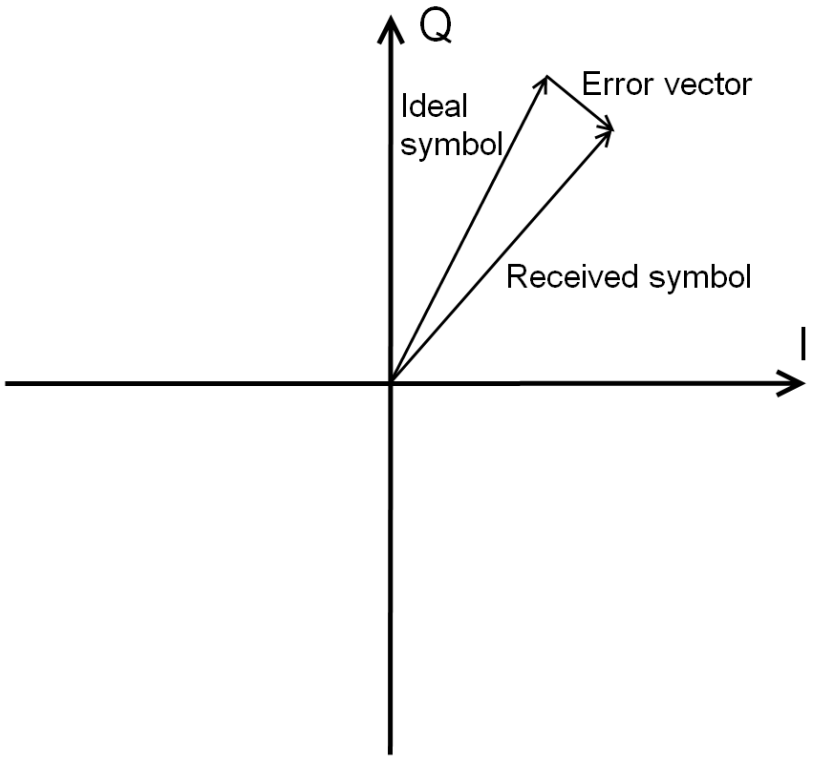


Figure 12 I-Q plot showing illustrating the concept of the error vector, the difference between an ideal symbol and an actual received (or transmitted) symbol.

2.3.3 Integrated phase noise quantities

Several of the performance indicators mentioned in the previous sub-chapter, such as phase error, frequency error and EVM, can be obtained by integrating the phase noise curve (i.e. the power spectral density plot). It is not possible, however, to reverse the operation and obtain the full phase noise curve from a single quantity such as the phase error. Thus, there may be many different phase noise curves that result in the same RMS phase error. Depending on the application, the oscillators with the different phase noise curves but equal RMS phase error may have entirely different performance. It is therefore important to know when to use a single number such as the RMS phase error and when to specify the full phase noise curve.

The RMS phase error in radians may be obtained from the phase noise curve by performing the following integration [3]

$$\sigma_{\Phi} = \sqrt{2\int_{f_1}^{f_2} L(f)\,\mathrm{d}f} \tag{19}$$

where $L(f)$ is the single sideband power spectral density of the signal's phase fluctuations (which may also be defined as the power contained in a 1 Hz bandwidth around the offset frequency divided by the carrier power). It is important to note that $L(f)$ has to be in linear units (as opposed to dBc/Hz, which would be a "logarithmic unit") when performing the integration. Arguments for the validity of integrating a frequency domain quantity to obtain the time domain jitter may be made using Parseval's theorem [11]. The EVM is approximately related to the phase error simply as

$$\mathrm{EVM} = 100 \times \sigma_{\Phi} \tag{20}$$

where the result is in the unit percent [%].

In the integration above, the integration has been performed between the frequencies f_1 and f_2. The integration limits depend upon the system in which the oscillator is used. Most systems are only affected by phase noise in a certain limited frequency range. For example, in a WCDMA system the lower limit is defined by the frequency under which phase noise does not affect the quality of the signal due to limited slot length and correction algorithms. This will be discussed in more detail in the following chapter. The upper limit is usually the bandwidth of the signal, which would be 3.84 MHz in a WCDMA system.

Similarly the RMS time jitter may be obtained by taking

$$\sigma_{\mathrm{t}} = \frac{1}{f}\frac{\sigma_{\Phi}}{2\pi} \tag{21}$$

where f is the frequency of oscillation. Time jitter is a quantity that is easier to relate to in the time domain, as compared to phase error. As described in the previous section, the time jitter is a very important quantity for digital applications, where the times of the zero crossings are the primary measure of an oscillator's stability. The RMS time jitter is a measure of how large the time variations of the zero crossings due to noise are.

The RMS frequency error, also called residual FM, of an oscillator with a specified phase noise curve may be obtained by performing the following integration

$$\sigma_{\mathrm{f}} = \sqrt{2\int_{f_1}^{f_2} L(f)\cdot f^{2}\,\mathrm{d}f} \tag{22}$$

and is a measure of how much the instantaneous frequency differs from the desired carrier frequency on average.

3 Practical simulation and measurements

In order to predict the performance of a PLL in different situations, one may of course use the theory provided in the previous chapter to perform mathematical calculations. This can be useful and worthwhile, but oftentimes a simulation tool is to be preferred.

There are several different simulation tools available for simulating PLL responses. Some are tailor-made for that specific purpose and have no other uses, like ADIsimPLL®[8], which is able to simulate several aspects of a PLL, such as its phase noise response (given the reference phase noise, VCO phase noise and PLL parameters). It also has the ability to simulate simple time domain responses, such as the time it takes to lock.

For more advanced simulations and situations where more flexibility is desired, however, something more generic is desired, such as general purpose electronic circuit simulation software. Some examples of such software would be Agilent ADS®, Cadence SpectreRF® or indeed any SPICE derivative.

When choosing a simulation tool, it is important to decide on whether you want to simulate your PLL using so called behavioural models or transistor-level models. Not all software includes behavioural models. A behavioural model simply defines the PLL components in terms of their behaviour, without regard to practical electrical circuits. For example, the VCO is defined to perform entirely as an ideal VCO would be expected to perform, simply taking the voltage on its input tuning line and multiplying it by K_v. A transistor-level model would be significantly much more complex, most likely involving several transistors and other components. It would also more accurately portray the non-idealities inherent in practical VCOs.

In theory, it would of course be more accurate to always use transistor-level models. It is however not practical for a number of reasons, some of them being: 1) It is often impossible to obtain transistor-level models of circuits bought from commercial circuit vendors 2) Transistor-level models require additional simulation time compared to behavioural models 3) Transistor-level models are very hard to work with and require a huge number of things to be changed in order to change a single top-level “PLL parameter”. Many aspects of a PLL are entirely possible to simulate to a high degree of accuracy using only behavioural models.

[8] Free PLL simulation software provided by Analog Devices. Available for download at http://forms.analog.com/form_pages/rfcomms/adisimpll.asp (2011-02-23)

This thesis's main focus has been on simulating PLLs using Agilent ADS®. ADS is a very popular electronic design automation software, produced by Agilent (formerly HP). It is focused mainly on high-frequency and microwave design. It incorporates several different simulation engines, both SPICE-style circuit simulation and digital DSP simulation. It also offers tools for layout, EM simulation, signal integrity etc. ADS includes many behavioural models and examples suited for simulating PLLs which have been used in this thesis.

The following sub-chapters will detail three different simulation aspects: steady state, transient and system-level. Steady state simulation focuses on simply obtaining the phase noise response of a PLL, given its reference phase noise, VCO phase noise, loop filter parameters etc. This kind of analysis does not require any time domain simulation and can be calculated simply through the use of the transfer functions derived in the theory chapter.

Transient simulation focuses on using a behavioural PLL model which can be simulated in the time domain. Using this kind of a model one can simulate things such as lock-in time, frequency stepping, pushing and pulling. It is also possible to see the effects of PFD leakage leading to spurious frequencies in the output spectrum.

System-level simulation focuses on using the above-mentioned DSP simulation engine that is built into ADS, called ADS Ptolemy. Taking a PLL's phase noise curve as input it is possible to simulate an entire WCDMA signalling chain to see what effect that PLL would have on the system performance if it was used as the LO in an upmixing transmitter for example.

3.1 Steady state simulation

The steady state simulation of a PLL aims to simulate things that do not vary in the time domain, such as the PLL's phase noise response due to steady state factors. Given the reference phase noise, VCO phase noise, PFD noise and loop filter parameters it is possible to obtain an estimate of what the resulting phase noise curve on the output would look like. One may also want to include phase noise resulting from the dividers but this is generally negligible in comparison to the other sources [15].

ADS provides a lot of the functionality needed for steady state phase noise simulations through examples. To simulate the basic phase noise response one can use simple components such as ideal voltage transformers to simulate a gain step. One does not need to use full-fledged behavioural PLL components. The simulation may then be carried out with a simple AC simulation controller to simulate the phase noise at different frequency offsets from the carrier.

To access the ADS examples for steady state phase noise response simulation, one may press **DesignGuide > PLL** in a schematic window and then choose "**Select PLL Configuration**" in the dialogue box. One may then choose to simulate the phase noise response.

A slightly modified example schematic is shown in Figure 13.

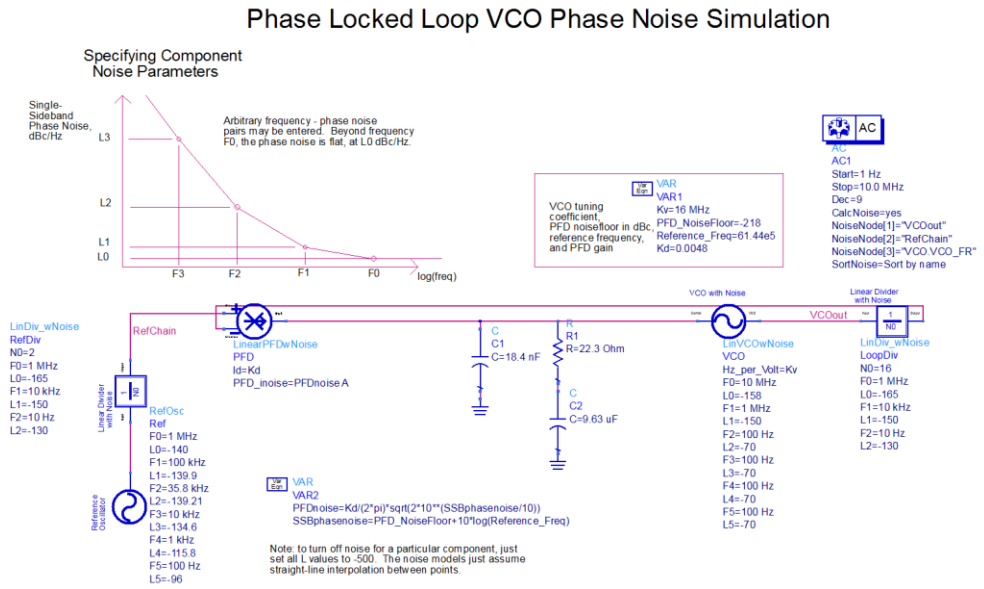


Figure 13 ADS schematic window showing an AC simulation for a PLL's steady state phase noise response.

Certain convenient modifications may be made to the example. For example, it is convenient to introduce a **VAR** item to calculate the PFD noise current from the PFD's noise figure of merit. PFD noise is usually specified in the datasheet as a noise figure of merit in dBc/Hz. This may then be scaled according to the PFD comparison frequency to obtain an estimate of the PFD's noise.

In order to convert the noise figure of merit to a noise current, the following equation may be used

$$i_{\mathrm{PFD}} = \frac{K_{\mathrm{d}}}{2\pi}\sqrt{2\times 10^{\frac{L_{\mathrm{SSB}}}{10}}} \tag{23}$$

where

$$L_{\mathrm{SSB}} = L_{\mathrm{PFD,fom}} + 10\log(f_{\mathrm{ref}}) \tag{24}$$

and $L_{\mathrm{PFD,fom}}$ is the PFD's noise figure of merit in dBc/Hz and f_{ref} is the comparison frequency. A noisy current source is a more natural way of modelling the PFD noise in ADS.

As can be seen in Figure 13, all PLL components are present. On the left side is the reference signal, with a specified phase noise mask. The reference signal is then passed through a divider, to divide it down to the comparison frequency. It is important to note that the phase noise scales with the division factor, due to the definition of phase noise. The noise itself is unchanged but due the the longer period time of a lower frequency signal the phase error will be smaller. In dBc, the phase noise will decrease by $20\log(N)$ dBc/Hz, where N is the division factor.

In ADS, the reference signal source is simply implemented as a voltage noise source, interpolating the phase noise between the offset frequencies where it is specified. There is no “carrier signal”. This is OK since the only thing being modelled is how the phase noise transfers to the PLL output.

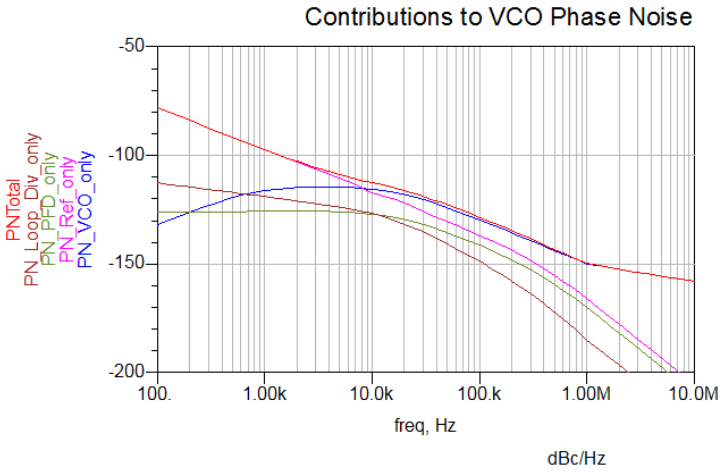
The divider is modelled as an ideal voltage transformer with a gain of $\frac{1}{N}$.
Note that there is no actual change of frequency taking place. The simply voltage division will however reflect what would happen in the frequency division case, since the phase noise will be scaled down by a division factor of N. It also has a certain phase noise mask associated with it but as mentioned previously the divider phase noise is usually very negligible in comparison to other noise sources.

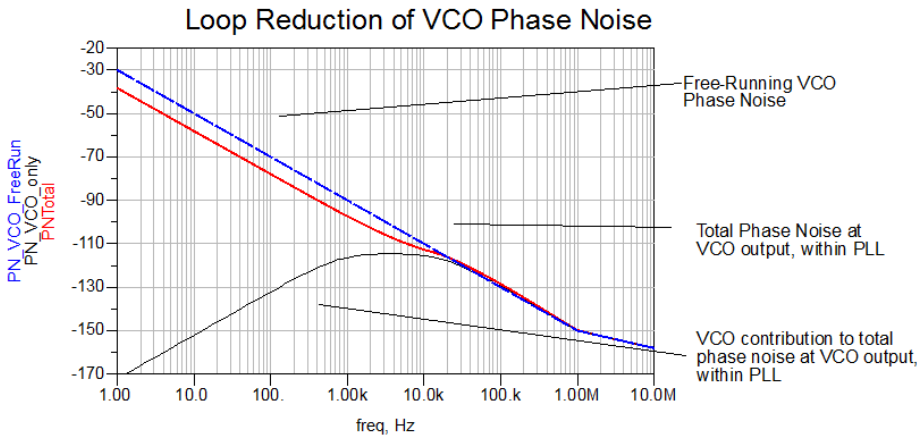
After the divider comes the PFD, which is modelled as an ideal voltage summation (in this case a subtraction since one of the voltages is inverted) of the two input signals controlling a current source with a gain corresponding to the PFD gain. The PFD’s noise is added with a noise current given by (23).

The PFD output is then low-pass filtered by the loop filter, which uses regular RLC components. After the loop filter comes the VCO, which is modelled as an ideal integrator by use of a voltage controlled current source charging a capacitor which is in turn connected to a voltage controlled voltage source acting as an output buffer. A noise voltage source is then connected in series with the VCO, adding its phase noise.

At the end comes the VCO divider and the feedback line, which completes the PLL loop.

The resulting output from this simulation is shown in Figure 14, in which one can clearly see the resulting phase noise response and what the major noise contributors are. It is evident that the reference noise is dominating up until the loop filter’s cut-off frequency, where the VCO noise starts to dominate instead. In addition to this, the loop reduction of the VCO phase noise is shown. One may also add integrated phase noise quantities according to the equations given in the theory chapter. By integrating the phase noise it is possible to calculate phase error, time jitter, EVM and frequency error. This is easily achieved by the use of measurement equations and ADS’s **integrate()** function.





Integrated phase noise quantities

PhaseError	FrequencyError
9.049 m	51.36
JitterError	**EVM**
25.57 f	15.79 m

100 kHz to 1.25 MHz

Figure 14 Data display output from the steady state phase noise simulation in ADS.

A steady state phase noise response simulation is a very useful tool for getting an idea of what the PLL's resulting phase noise will be. It is worth noting however that many effects are not taken into account, such as the spurious frequency signals mentioned in the PFD theory chapter. In general one will see a number of spurs (i.e. peaks) in the phase noise response of any real PLL. The spurs can, as mentioned previously, be caused by PFD leakage and mismatch. They may also be caused by other factors such as disturbing signal sources interacting with the VCO's tuning line due to electromagnetic interference (EMI). This could for example be caused by digital circuits on the same PCB as the PLL.

3.2 Transient simulation

In order to simulate signals that are varying in time, a different type of simulation is needed. Whereas the steady state simulation was based on transfer functions which could be evaluated with respect to time, transient simulations work by simulating the circuit's response to a time varying input signal. This allows for simulations of several time-dependent phenomena, such as lock-in time, the appearance of spurs, pushing, pulling, etc.

The first thing to decide when doing a transient simulation is which simulation engine to use. ADS offers a multitude of different simulation engines such as Alternating Current (AC), Harmonic Balance (HB), S-parameter, Transient and Circuit Envelope. Some of these, such as the AC and S-parameter simulation engines, are only intended to be used for linear frequency domain simulations, i.e. to see how a linear circuit's response varies over frequency. The AC simulation uses traditional concepts such as voltages and current while the S-parameter simulation uses S-parameters, which are often used at RF and microwave frequencies where voltages and currents are hard to measure. The harmonic balance simulation calculates the steady state response of a circuit at a certain frequency and can be used to simulate non-linear networks. While it can give some hint of the time domain response in the steady state, it can not be used to simulate things such as lock-in processes.

In order to perform true time domain simulations, one has to use either the transient or the circuit envelope simulation engine. The transient simulation engine is based on the SPICE circuit simulator developed at Berkeley. As input to the simulation, one must specify a time step and an end time. The simulator will then simulate the circuit's transient response at every time step up until the end time. This can become problematic for PLL circuits, since simulations can require a very high number of time steps in order to be meaningful.

When simulating circuits containing a VCO, a general rule of thumb is to use a time step that is $\approx \frac{1}{10 f_{\text{osc}}}$ where f_{osc} is the oscillation frequency of the undivided VCO [8]. In many practical PLL circuits the output VCO frequency is very high, sometimes several thousand times the comparison frequency. A lock-in process often involves many thousand cycles at the comparison frequency which then equals several million cycles at the VCO frequency. With a time step that's one tenth of a VCO cycle, this can translate to tens of millions simulation steps, which can take a very long time to simulate even on modern computers. In addition to the time it takes, it also requires a very large amount of memory which tends to be an even bigger problem.

In order to remedy this situation, Agilent developed something called the circuit envelope simulation engine. Circuit envelope is basically a combination of two other simulation engines, namely harmonic balance and transient. Its basic operating principle is to simulate the "RF" (i.e. the high frequency) part of the signal using harmonic balance, while simulating the baseband signals (the envelope) using the transient simulation engine. This is very useful for modulated signals where a baseband signal with a bandwidth of perhaps a couple of MHz is modulated onto an RF carrier in the GHz range. It can also be used for PLL circuits, by allowing the time step to be set to be just large enough to cover a bandwidth necessary to simulate the lock-in time for example. Such a simulation will run significantly faster using the circuit envelope simulation engine compared to the transient simulation engine. For phase noise simulations however, large bandwidths are often needed even with circuit envelope simulations.

To simplify the transient simulations in this thesis, high-level ADS models were created for a PLL (i.e. PFD plus dividers) and VCO, both models incorporating noise sources. Several models were created to fit different simulation scenarios. ADS incorporates two different classes of behavioural PFD components, namely baseband and tuned components. The baseband PFD, called **PhaseFreqDetCP** (where CP stands for charge pump) works in both transient and circuit envelope simulations and models reference signal feed-through, i.e. the PFD's output may contain significant spur energy at harmonic frequencies. The downside is that in order for the simulation to be accurate the time step must be set to about one tenth of the reference period, which means a relatively long simulation time.

The other PFD component, called **PhaseFreqDetTuned**, is a so called "tuned" PFD component and only works in circuit envelope simulations where the inputs are RF carriers. It does not model reference feed-through and only takes the part of the input signal at the nominal carrier frequency into account. The advantage of this model, however, is that it can be used with time steps in the same order of size as the PLL loop bandwidth.

The PLL top-level component utilising the baseband PFD is shown in Figure 15. First comes the reference and VCO dividers, then the PFD. At the PFD's output two types of noise are added. The first is a constant frequency current source intended to model the leakage currents through the PFD. These can be caused either by the transistors not turning off completely or by other leakage paths through the PFD [14]. The other noise source is due to the PFD's internal noise, which is modelled by (23).

The top-level component with the tuned PFD is almost identical except for the fact that the **PhaseFreqDetCP** component has been replaced by a **PhaseFreqDetTuned** component.

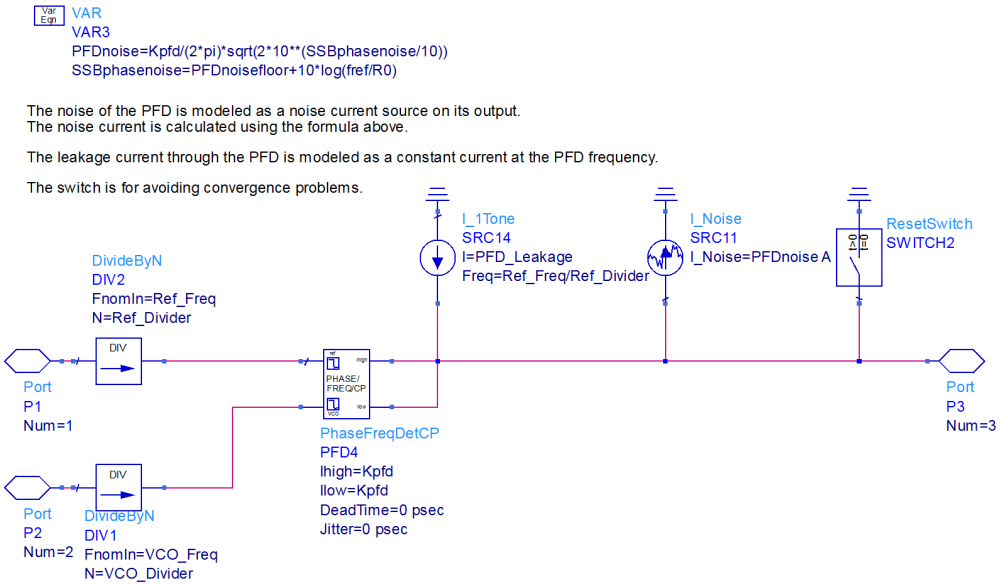


Figure 15 Top-level PLL component **PLL_BaseBand_ChargePump** for use in baseband simulations.

The VCO top-level component schematic is shown in Figure 16. It uses a behavioural VCO model with added phase noise. The phase noise is generated by an **OSCwPhNoise** component. By use of an FM demodulator (the VCO is in effect a frequency modulator) one may generate the appropriate voltage to add at the tuning side of the VCO in order to match the output phase noise. The noise voltage is simply added in series with the normal VCO tuning signal by use of an ideal voltage controlled voltage source with a gain of 1 (and zero output impedance). There are several other ways to add phase noise to an ideal VCO, for example by placing a phase modulator at the output.

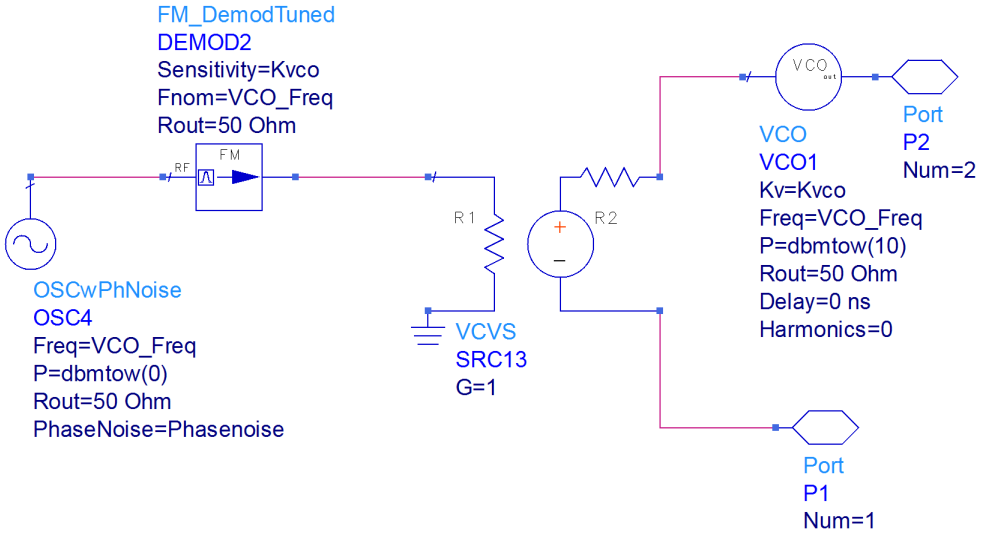


Figure 16 Top-level VCO component **VCO_with_PhNoise_BaseBand**.

As mentioned previously, it is possible to simulate several different time-dependent phenomena using behavioural models and circuit envelope simulations. In the following sub-chapters each of these will be examined in more detail.

3.2.1 Lock-in time

Lock-in time measures the time it takes for the PLL to lock onto a changed reference signal. In a PLL there are several different lock procedures, depending on how far apart the reference and VCO signals are from the start. If the signals are far apart in frequency a slow lock procedure will take place, referred to in [5] as a “pull-in” procedure. This is what happens when you apply a frequency step on the reference input or, more commonly, change the divide ratio for example. If you only apply a phase shift, on the other hand, the lock procedure will be much quicker. It is this lock-in procedure that is of interest for most frequency generation scenarios where the PLL is used as a clock source.

In order to simulate this type of quick lock-in procedure, the PLL is allowed to reach a phase-locked steady state where both reference and VCO signals are in phase. The phase of the reference signal is then changed by 180 degrees and the time taken to reach the phase-locked state again is measured. The ADS schematic used for the lock-in time simulation is shown in Figure 17.

Prepared (also subject responsible if other)		No.		
EMARFAX				
Approved	Checked	Date	Rev	Reference
		2011-01-17	PA1	

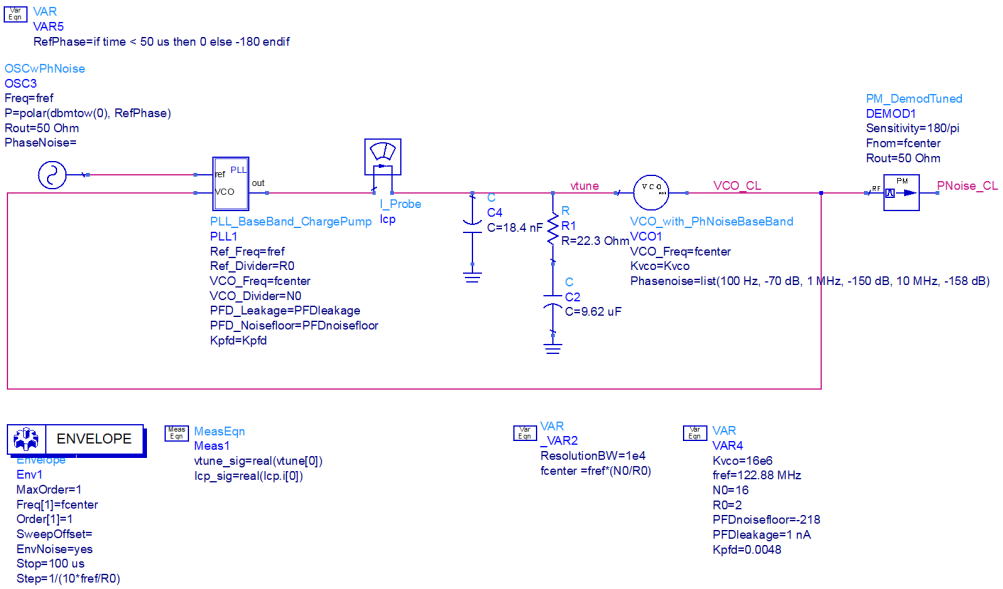


Figure 17 ADS schematic for the transient lock-in time simulation.

As can be seen it is a very standard PLL circuit, with the important part being the **VAR** item that defines **RefPhase**. After 50 µs the loop is assumed to have reached steady state. At that point, the reference phase is changed from 0 to -180. The lock-in time may then be defined as the time it takes until the tuning voltage has stabilised once again (i.e. reached steady state). A measurement like this may also easily be performed in the lab, by changing the reference signal's phase by 180 degrees.

The results of the simulation can be seen in Figure 18, the abrupt change in tuning voltage after 50 µs (when the reference phase is changed) can clearly be seen. It then takes about another 50 µs to stabilise, which would be the lock-in time.

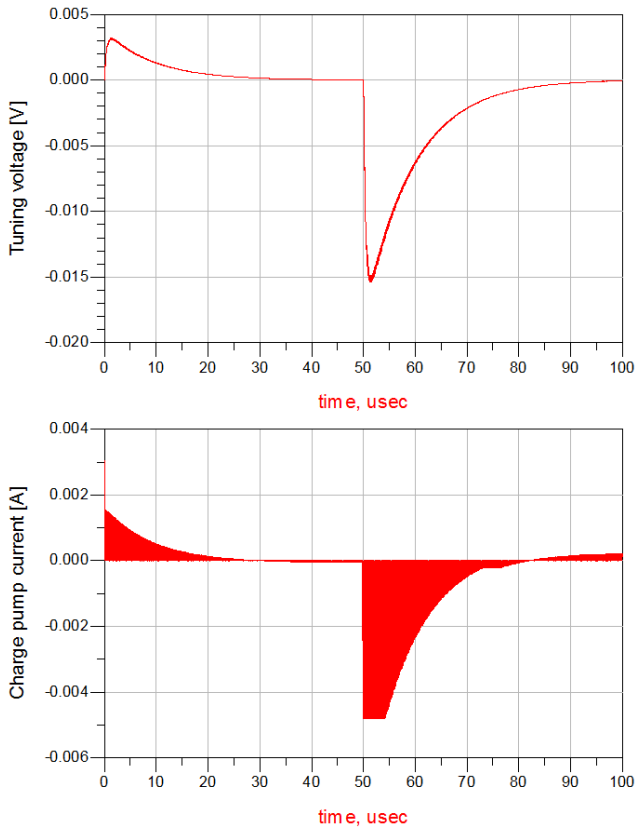


Figure 18 Data display window for lock-in time simulation.

3.2.2 Spurious frequencies

Another time domain phenomenon of simulation interest is the appearance of spurious frequencies (“spurs”) at the PLL output. A spurious frequency, as explained in earlier chapters, is an unexpected peak in the phase noise curve, meaning that at a certain offset frequency from the carrier there’s a lot more signal energy than one would expect given just $\frac{1}{f}$ and thermal noise. There are a number of different types of spurs, some of the more common ones being reference spurs and interference spurs.

Reference spurs appear at offsets at the reference frequency or harmonics thereof from the carrier. They are usually caused by PFD mismatch (sourcing/sinking transistors in the charge pump not perfectly calibrated) or PFD leakage (transistors not completely turned off, which leads to the reference signal leaking through the charge pump).

Interference spurs are caused by nearby traces on the PCB carrying signals that interfere with the PLL (crosstalk). Especially digital signals can be very problematic, since they often use relatively high clock frequencies and generate many harmonics due to the digital nature of the signal (sharp edges). Also, digital circuits are not as susceptible to problems caused by crosstalk and may as such not be well isolated.

Prepared (also subject responsible if other)		No.		
EMARFAX				
Approved	Checked	Date	Rev	Reference
		2011-01-17	PA1	

It is possible to simulate both leakage and mismatch reference spurs in ADS. Interference spurs are however hard to simulate in a simple model like the one employed here. In order to simulate these types of spurs one would need a full-featured 3D electromagnetic (EM) simulator and a complete model of the physical layout of the PCB. In addition to the two types of spurs mentioned here there are others that one tends to see in practical circuits. For example, fractional-N PLLs may have more spurs than comparable integer PLLs due to their fractional division. For more information about different types of spurs one may refer to a book such as [3].

In order to simulate reference spurs, an ADS test bench without the traditional sources of phase noise is used. Otherwise the spurs might be drained in other noise. The size of the reference spurs is largely dependent on the loop filter bandwidth in relation to the reference frequency. If the reference frequency is very far outside the loop bandwidth the spur will be heavily suppressed. If on the other hand the loop filter bandwidth is large compared to the reference frequency the spur will not be attenuated so much before reaching the output. Thus it might be a good idea to use a large reference frequency in the PLL in order to avoid the reference spur. The problem with this in a frequency generation situation is that the steps between output frequencies will be very large as well.

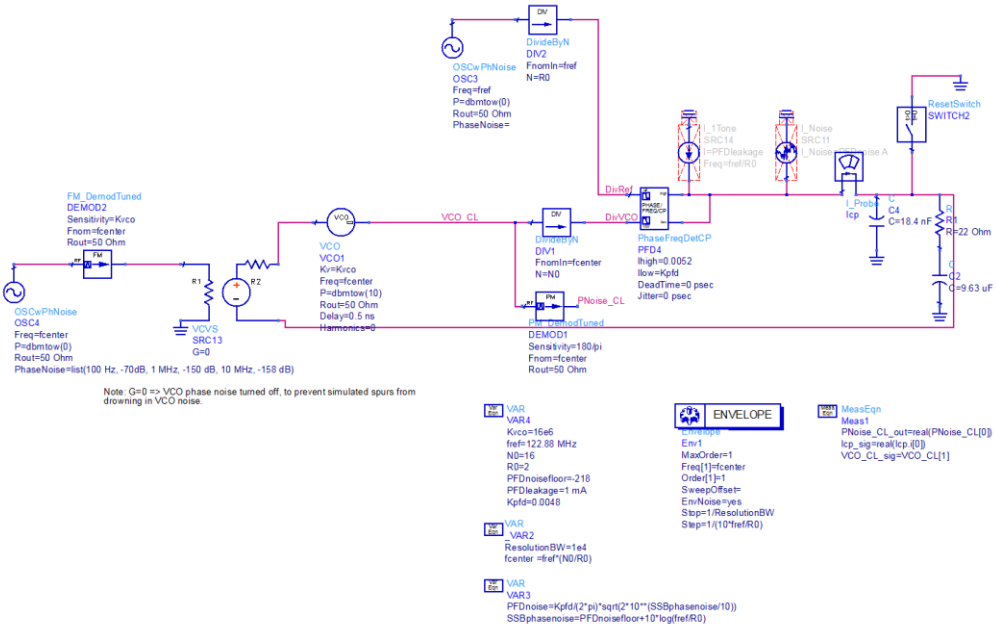


Figure 19 ADS schematic for the reference spur simulation.

The schematic used for the reference spur simulation can be seen in Figure 19. On the left side is a VCO which may have phase noise added to its output. This is however turned off by default, to avoid draining the reference spurs in other phase noise. After the VCO there is a baseband PFD, which is able to simulate the effect of reference feed-through. One may also set the PFD imbalance here, if the transistors are not perfectly matched.

On the PFD output side, before the loop filter, two current sources are used to add noise. One adds reference noise caused by PFD leakage, which is modelled as a constant current source at the reference frequency. The other adds randomly varying phase noise caused by the PFD noise. The PFD noise is turned off by default, in the same way as the VCO noise, in order to prevent the reference spurs from drowning in other phase noise.

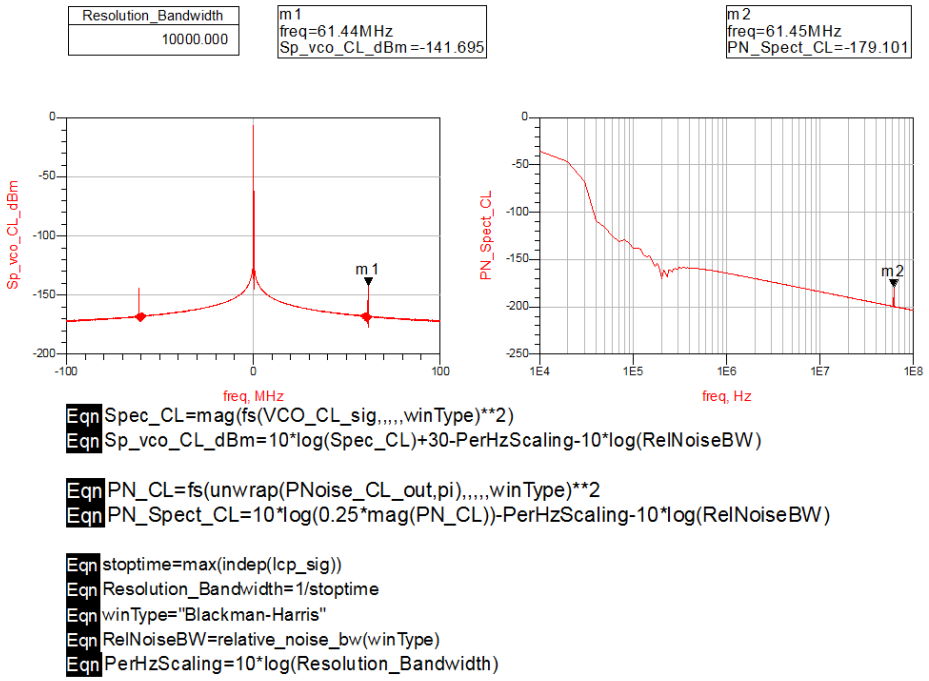


Figure 20 ADS data display window belonging to the reference spur simulation.

The output of the reference spur simulation is shown in Figure 20. It is important to note that the PLL is not phase-locked at the beginning of the simulation which leads to reference spurs appearing in the output spectrum. In the simulation shown here this is the major cause of reference spurs. Since the reference frequency is far outside the loop bandwidth the spurs get strongly attenuated.

The type of simulation performed above will only simulate the reference spurs, which is the only “naturally occurring” type of spur in an integer PLL. As mentioned above, in a fractional-N PLL one would see a number of spurs at fractions of the comparison frequency in addition to reference spurs. In reality, however, one usually sees a large number of spurs not related to the comparison frequency as well. These are usually caused by electromagnetic interference (crosstalk). One way to model such spurs is described in the next sub-chapter.

3.2.3 Pushing

Pushing is generally defined as a time domain electrical disturbance on the VCO's supply voltage, which leads to a change in its output frequency. This parameter is highly VCO dependent and is usually specified in the VCO's datasheet as a MHz/V value. This supply line disturbance may be generated in any number of ways, but a common source is the power supply itself. Especially switching power supplies can generate a lot of high frequency noise that gets coupled to the VCO's supply line.

When simulating pushing using behavioural model components and in a normal circuit simulator it is not possible to see the effects of parasitic coupling between the different traces on the PCB. It should however be possible to quantify the resulting disturbance in terms of a disturbing signal on the tuning line instead. Provided that that this is the case, one might model it in ADS to see what sort of impact it would have on the transient properties of the PLL and the phase noise.

For VCOs, the inherent supply pushing noise may be included in the phase noise profile given by the supplier. As stated previously, the effect of supply noise is very VCO specific but it is certainly not negligible in most cases. Some general studies have reported phase noise degradations of 8 dBc/Hz across the spectrum due to supply noise pushing [16].

In order to simulate the effect of pushing in ADS, a test bench as shown in Figure 21 is used. The test bench contains two circuits, one PLL without any pushing and one PLL with an added voltage controlled voltage source connected in series on the tuning line, in order to be able to add a periodic disturbance. In this case the disturbance is modelled by a pulse source with a certain frequency and a certain magnitude.

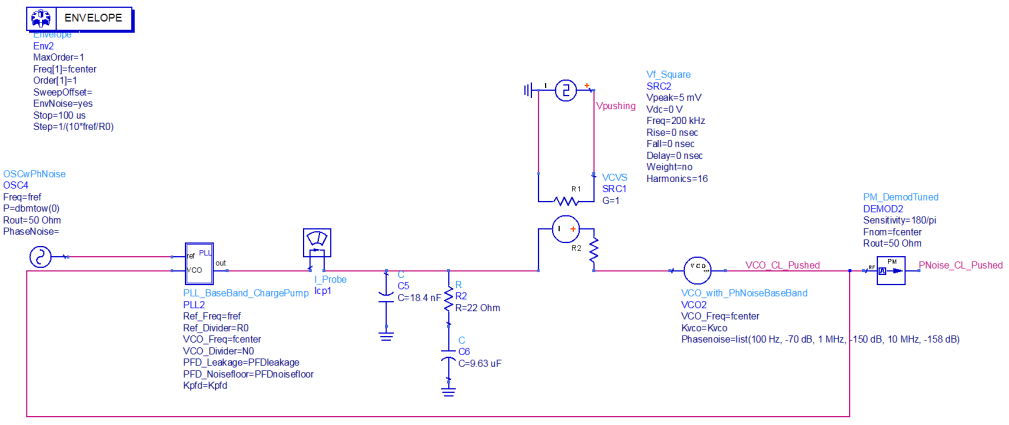


Figure 21 ADS schematic of the pushing simulation with the disturbance on the VCO tuning line. Only the circuit that is affected by pushing is shown, the reference circuit being equal but with no voltage controlled voltage source added.

Simulations like these can be problematic when the pushing signal has a frequency that is low (in the kHz range). In order to simulate over several of the disturbing signal's periods one needs a large stop time. In addition to this, in order to simulate the phase noise and VCO properly one needs a very small time step. Simulations can thus take very long time to complete and often result in ADS running out of memory.

The results of the pushing simulation are shown in Figure 22. A disturbing square wave signal of 5 mV at a frequency of 200 kHz (which might come from a switching power supply for example) has been injected in the VCO tuning node. As can be seen, this leads to large spurs in the output phase noise spectrum at frequencies of 200 kHz, 600 kHz, etc.

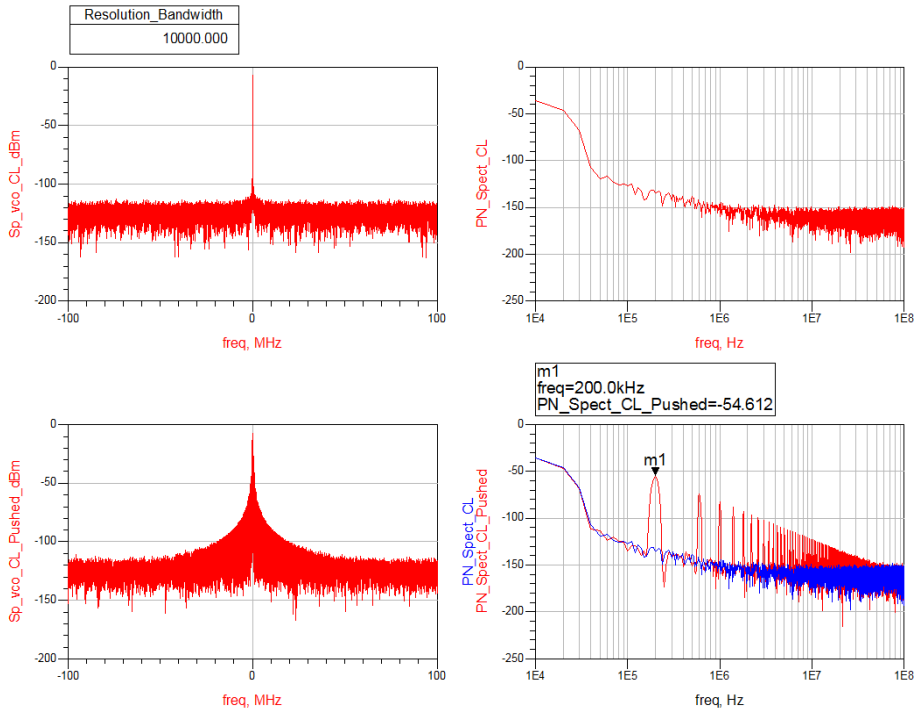


Figure 22 ADS data display window belonging to a pushing simulation where a disturbance of 5 mV at a frequency of 1000 Hz has been injected in the VCO tuning node.

In addition to modelling the disturbance as a noise signal that is coupled to the tuning line, one may also add disturbing signals at other places along the loop, using the same type of voltage controlled voltage source connected in series. An example of this where a 5 mV 200 kHz disturbing square wave signal has been introduced at the reference input is shown in Figure 23. One can clearly see the low pass effect for disturbances on the reference input when compared to disturbances on the VCO tuning line.

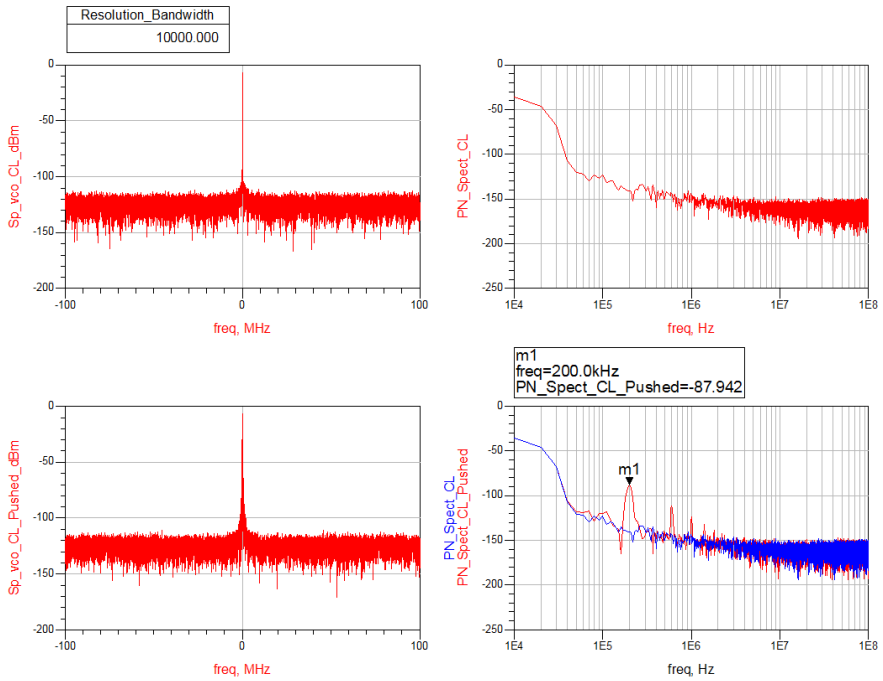


Figure 23 ADS data display window of the pushing simulation with a disturbance in the shape of a 5 mV 200 kHz square wave on the reference input.

3.2.4 Pulling

Pulling refers to the situation where a changing load seen by the VCO will affect its output frequency. The reason for this happening is manifold, and depends on the type of VCO used.

There are two basic types of VCO designs: harmonic oscillators and relaxation oscillators. Harmonic oscillators use some sort of LC tank circuit or a crystal that resonates at a certain frequency to define the oscillation frequency. Relaxation oscillators can have different topologies, one example being ring oscillators where the resonance frequency is instead decided by the delay around the ring.

In harmonic oscillators where an LC tank is used as the resonator, the load will usually be connected over the tank. Instead of simply seeing the LC tank, the VCO will see the LC tank in parallel with the load. Provided that the load has a non-zero reactance, this will lead to a shift in the resonance frequency, since the resonance frequency is determined by the tank reactance. This effect is commonly known as pulling.

In PLLs, another sort of pulling effect may occur as well, in addition to the reactance of the load changing the tank reactance. Since the VCO load might not be matched to the line, one may have reflections from the VCO travelling back to the PFD. This reflection will be phase-shifted (due to the length of the line) and will thus affect the VCO phase seen at the PFD.

Prepared (also subject responsible if other)		No.		
EMARFAX				
Approved	Checked	Date	Rev	Reference
		2011-01-17	PA1	

Since PLL simulations in ADS are performed using behavioural models, it is unfortunately impossible to correctly model the pulling effect due to a changed tank reactance. In order to be able to simulate this one needs to use a transistor-level VCO model, which adds a lot of complexity to the simulations.

What is possible to simulate in ADS, however, is the effect a mismatched load will have on the PLL in form of reflections changing the phase seen at the PFD. A schematic for simulating this can be seen in Figure 24.

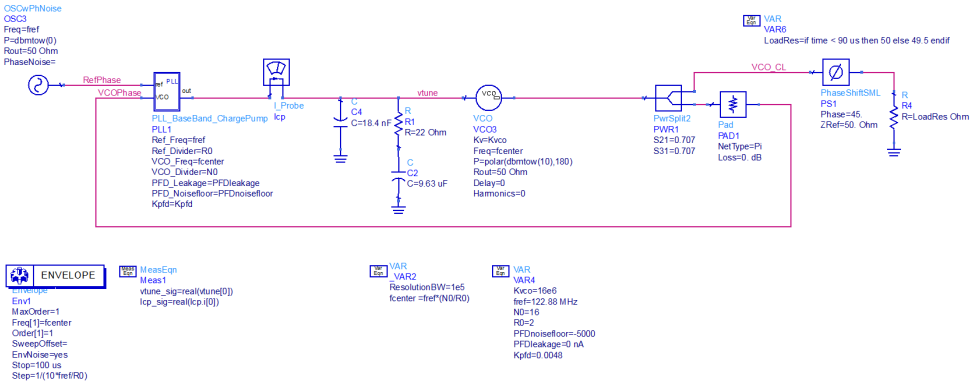


Figure 24 ADS schematic for pulling simulation.

As can be seen in Figure 24, the simulation is based on a standard PLL circuit but with a power splitter at the VCO output. The power splitter is used in order to ensure that all nodes are matched to 50 Ω. In one of the power splitter legs there is a load impedance and a phase shifter. The phase shifter is used to simulate line length. The value of the load impedance will be changed by a **VAR** item halfway through the simulation in order to see the effect it will have on the PLL's output frequency. The 0 dB pad is used in the power splitter's other leg in order to, once again, ensure that the impedance seen is 50 Ω.

The results of the pulling simulation are shown in Figure 25. The phase shift has in this case been set to 45 degrees, which will lead to a 90 degree phase shift for the reflected signal (since it gets phase shifted once in the forward direction and once in the backward direction), which will produce a maximum phase error when added to the VCO signal, since the two vectors will be orthogonal to each other. As can be seen, the changed load at 90 µs leads to a jump in the phase error at the PFD's output.

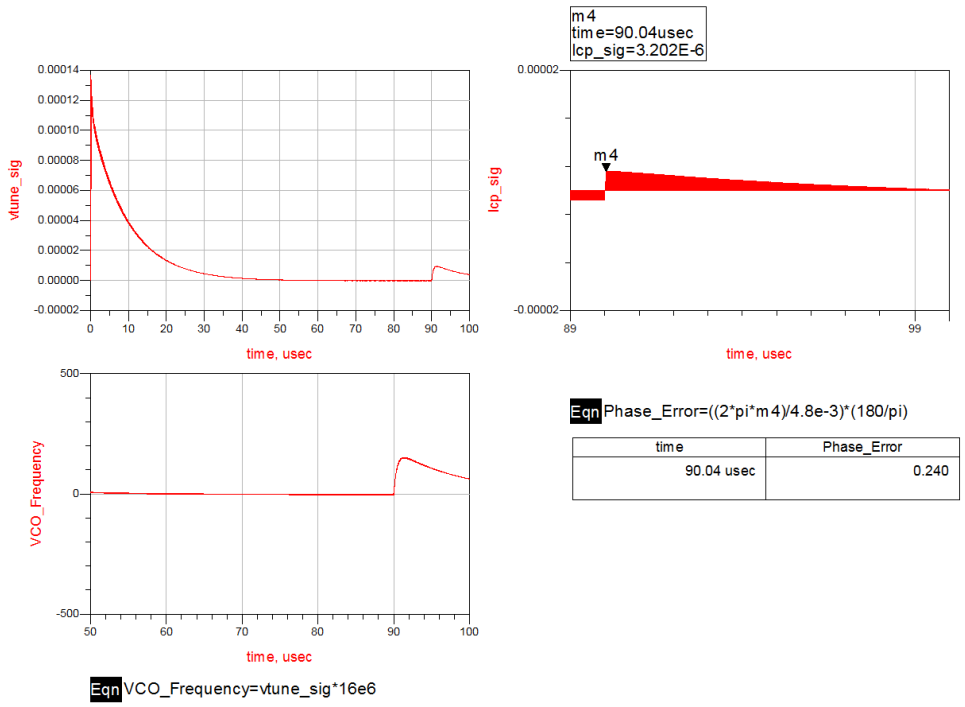


Figure 25 ADS data display window for the pulling simulation.

The phase error in the data display is calculated by use of the charge pump current at the 90 µs mark. A quick theoretical calculation can be made to support the accuracy of this result. The mismatched load in this case is 49.5 Ω as opposed to 50 Ω. This leads to a reflection coefficient of

$$T = \frac{Z_L - Z_S}{Z_L + Z_S} = \frac{49.5 - 50}{49.5 + 50} = -0.005 \quad (25)$$

which leads to a phase shift angle of

$$\Theta_{err} = \arctan(-0.005) = -0.287° \quad (26)$$

if one assumes that the reflected signal is phase shifted by 90 ° with respect to the VCO signal. This result is relatively close to the value reported by ADS (0.240 °).

3.3 System-level simulation

In contrast to the transient simulations performed in the previous chapter, this chapter will cover so called system-level simulations. They are also performed in the time domain but instead of simulating just a PLL circuit they will be focused on simulating an entire WCDMA receiver chain. A noisy PLL may then be used as the local oscillator for the upmixing in the WCDMA transmitter, in order to see what effect LO noise will have on the whole WCDMA chain's communication quality. This is illustrated in Figure 26.

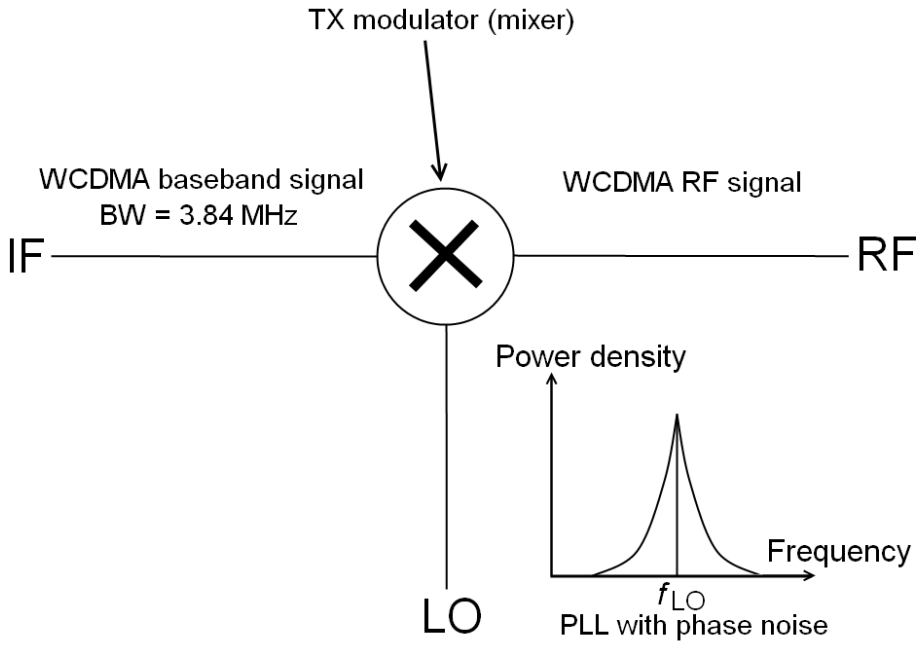


Figure 26 WCDMA transmitter with noisy PLL used for upmixing to the carrier (RF) frequency.

In order to perform this type of simulation, it is necessary to use ADS' Ptolemy simulator. ADS includes two types of basic simulation engines, one being a standard RF & Analogue simulator and the other being a digital signal processing (DSP) simulator called Ptolemy. While the RF & Analogue simulator is based upon concepts such as resistance, voltages and currents, the DSP simulator (Ptolemy) is based on the concept of sampling discrete quantities and performing some kind mathematical signal processing on them, the way that you would in a DSP. As such, one needs to somehow represent a PLL in the DSP world.

ADS includes a feature known as co-simulation, which makes it possible to simulate an RF/Analogue circuit as a small part of a larger design in the DSP world. In theory it would thus be possible to simulate a complete PLL circuit acting as the LO in a system-level simulation. In practise, however, it seems that the co-simulation feature does not work for networks which produce an output without any input (e.g. an oscillator). In addition to this, co-simulating an entire PLL circuit adds a substantial amount of overhead and increases the simulation time needed greatly.

In addition to the time and complexity needed to co-simulate a complete PLL circuit, it may simply not be worth it due to the small gains to be had. In a WCDMA system, one parameter of specific interest is for example the EVM, which was touched upon in the previous chapter. EVM is a measure of how much the transmitted (or received) symbols differ from the ideal ones. A large EVM value means that the transmitter (or receiver) is inducing a lot of noise. EVM may be caused by a number of different things, but the EVM produced by the LO is generally almost entirely attributable to its phase noise. This means that it is not necessary to simulate the entire PLL circuit, as long as one is able to simulate an oscillator with the same phase noise profile. Simulating an oscillator with a given phase noise profile is a simple task in Ptolemy.

ADS includes complex models of WCDMA signalling chains. In many of these it is possible to introduce some sort of device under test (DUT) in order to introduce some type of error. This chapter will mainly deal with the WCDMA EVM measurement, but using similar methods it is possible to simulate many other things such as adjacent channel leakage ratio (ACLR), code domain power, spurious emissions, etc.

In addition to simulating WCDMA signalling chains the Ptolemy simulator may also be used for many other types of simulations. One such example would be simulating an ADC/DAC clocked by a PLL with phase noise. ADS includes relatively complete examples for this as well that can be used with minor modifications. This type of simulation will be described briefly in this chapter.

3.3.1 System-level EVM simulations

In order to simulate a system-level EVM measurement, one may use the ADS examples provided in **examples/WCDMA3G/WCDMA3G_Bs_Tx_prj**. There is a specific example called BS_Tx_EVM that includes a complete WCDMA transmitter (signal generator) that outputs an RF WCDMA signal. At the other end there is a specialised EVM measurement device that downmixes to IF and calculates the EVM of the received symbols compared to the reference (ideal) symbols.

The (slightly) modified test bench is shown in Figure 27. The signal generator has been modified to allow for simulations with an external LO signal in the transmitter. The modified signal generator with an external LO input can be seen on the left and the EVM measurement instrument on the right. The underlying model for the WCDMA signal generator is very complex and includes a very large number of components but during the upmixing from IF to RF a component called **QAM_Mod** is used, which takes two input signals (I and Q signals, as described in the theory chapter) and produces an RF signal. This component may be replaced by another component called **QAM_ModExtOsc** which allows one to specify an external oscillator signal (the **QAM_Mod** component simply uses an ideal internal oscillator at the specified frequency).

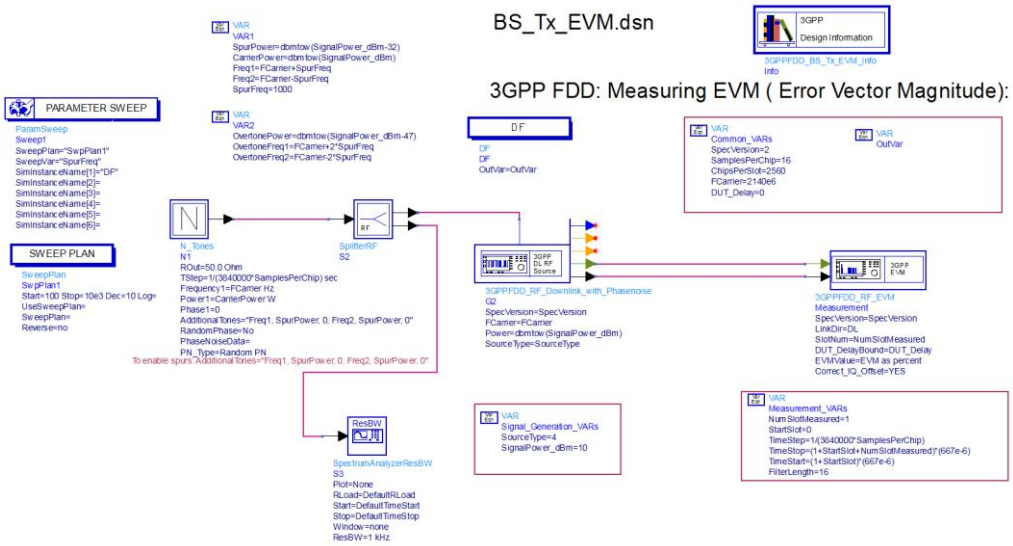


Figure 27 ADS schematic of EVM test bench.

Using the **QAM_ModExtOsc** component instead, it is possibly to use a component called **N_Tones** as the external oscillator source. The **N_Tones** component includes the ability to specify phase noise on its output, as a number of frequency offset/phase noise level pairs. It is also possible to specify spurs in the form of additional tones at a given power level.

Once these modifications are made it is very simply to simulate the EVM for a given phase noise profile. Of course, one needs to keep in mind that only the upmixing LO's EVM contribution is taken into account. In reality there are of course several other different contributors to a WCDMA system's EVM.

The resulting data display screen for a system-level EVM simulation shows the EVM result as a percentage. The WCDMA standard is developed by 3GPP, which is an international organisation that includes many national standard bodies as its members (or "partners"). According to the 3GPP release 10 specification, the maximum acceptable EVM percentage for a base station transmitter is 17.5% when transmitting a composite signal that includes QPSK and 12.5% when transmitting a composite signal that includes QAM-16[9].

An important question with respect to EVM in a WCDMA system is whether phase noise at all frequencies affect the EVM in the same way, or if phase noise at certain frequencies is more critical than phase noise at other frequencies. The reason that one might expect phase noise at certain offset frequencies to be less critical is that WCDMA uses "slots" that are of limited duration in time.

[9] The 3GPP release 10 standard for base station radio transmission and reception is available at http://www.3gpp.org/ftp/Specs/archive/25_series/25.104/25104-a00.zip (2011-03-04)

An in-depth explanation of WCDMA is outside the scope of this thesis but it is based on a spreading modulation scheme, where code symbols are spread out over a number (depending on the spreading factor) of chips, in order to increase the bandwidth. Chips are transmitted at a rate of 3.84 Mcps (mega chips per second) and the basic unit of transmission is the “frame”, which is 10 ms long. Each frame is then divided into 15 slots, leading to 2560 chips per slot and a slot length of 667 µs [17].

WCDMA also includes a number of error correcting features and has the ability to compensate for a static or linearly varying phase error. Based on this and the limited slot length, one would expect slowly varying phase errors (i.e. at small offset frequencies from the carrier) to be a lot less critical to the transmission quality than rapidly varying phase errors, since they will simply not have “time” to affect the signal during a single slot.

The error correction features found in WCDMA may be modelled as performing a linear regression on the phase error in order to remove static and linearly varying phase error. Assuming the received signal is

$$s(t) = \cos(\omega t + \Phi_s(t)) \tag{27}$$

where $\Phi_s(t)$ is the received signal’s phase and the reference signal is

$$r(t) = \cos(\omega t + \Phi_r(t)) \tag{28}$$

where $\Phi_r(t)$ is the reference signal’s phase, the instantaneous phase error may be defined as

$$e(t) = \Phi_r(t) - \Phi_s(t) \tag{29}$$

A linear regression (least squares) over time is then performed on $e(t)$ to fit a function

$$e_{est}(t) = a_1 + a_2 t \tag{30}$$

to $e(t)$. The resulting phase error after correction will then be

$$e_{res}(t) = e(t) - e_{est}(t) \tag{31}$$

In reality there is no way of knowing the actual reference signal, but for purposes of modelling the spur offset frequency’s effect on EVM, this model is accurate.

An easy way to measure the effect that phase noise at different offset frequencies has on the EVM is to generate an LO signal (as per Figure 26) with a spur at a specific offset frequency and then sweep this spur frequency. By keeping the spur power constant and recording the resulting EVM for each offset frequency, one may obtain a transfer function for spur offset frequency to EVM.

In order to visualise the transfer function from spur frequency offset to EVM percentage, one may perform a parameter sweep of the variable defining the spur frequency offset. The noisy LO is generated by an **N_Tones** source which includes two spur components (double-sided) at variable frequency. The result of this is shown in Figure 28. At low frequency offsets the EVM is not affected by the spur. At frequencies around 800 Hz it begins to affect the EVM and at a cut-off frequency somewhere around 1500 Hz it saturates to its maximal EVM impact.

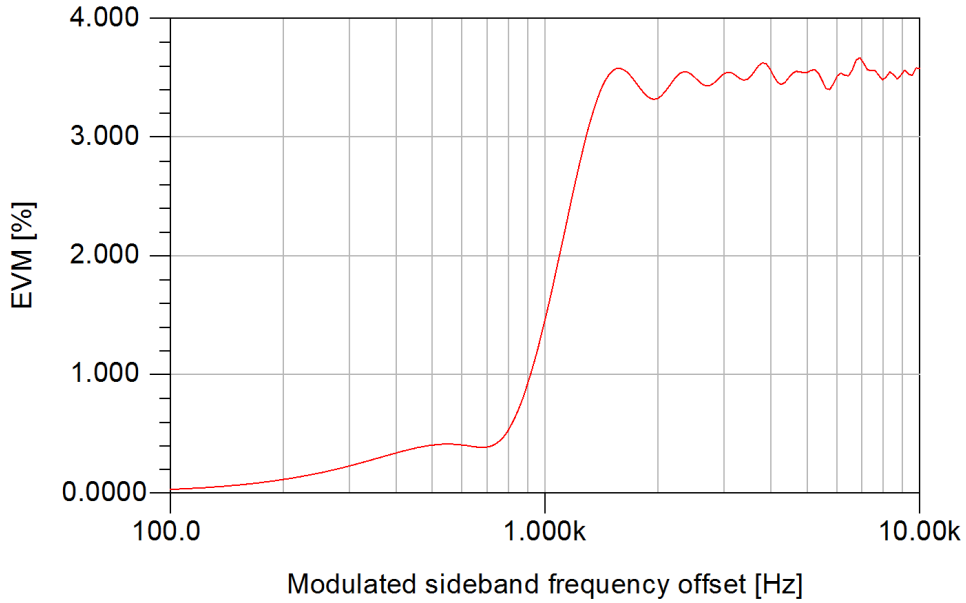


Figure 28 ADS results of a swept spur (modulated sideband) frequency offset simulation in the range 100 Hz to 10 kHz, for a spur that is -32 dBc.

For a WCDMA signal one also needs to take into account how EVM is affected by spurs that lie outside the bandwidth of the signal, i.e. outside 3.84 MHz. A simulation in the frequency range 100 kHz to 5 MHz is shown in Figure 29, showing that the effect on EVM starts to decrease around 1 MHz and is almost nonexistent at 4 MHz. This is due to the fact that the bandwidth of a WCDMA signal is 3.84 MHz. When integrating a phase noise curve in order to calculate the resulting EVM in a WCDMA system, an integration between 1000 Hz and 2 MHz provides an accurate estimate for a flat phase noise curve (the integration does not run all the way up to 3.84 MHz due to the phase error above 2 MHz having less and less of an effect on the EVM, as shown in Figure 29). In reality it might lead to a slight over-estimation of the EVM if the phase noise is not flat above 2 MHz.

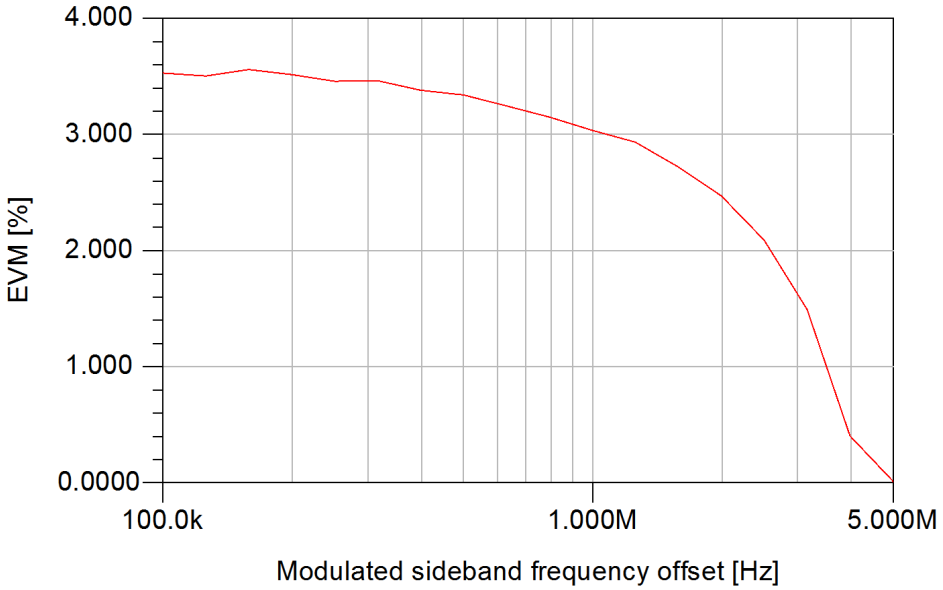


Figure 29 ADS results of a swept spur (modulated sideband) frequency offset simulation in the range 100 kHz to 5 MHz, for a spur that is -32 dBc.

The transfer function from spur frequency offset to EVM percentage can also be theoretically verified in MATLAB, by simulating a reference signal and a noisy signal with a spur at a specified frequency offset. The spur signal may be modelled as

$$s_{\text{spur}}(t) = \cos(\omega t + C\cos(\omega_{\text{spur}} t)) \tag{32}$$

where ω is the carrier frequency, ω_{spur} is the spur frequency offset (in rad/s) and C is a constant related to the size of the spur. The reference signal and the noisy signal may then be converted to analytical signals, by the use of the so called Hilbert transform. An analytical signal is complex valued and is convenient to work with when dealing with instantaneous phase.

The phase error after WCDMA's correction algorithms may then be found in accordance with (27)-(31). This may be done for varying spur frequency offsets, in order to see how a spur at a specified frequency translates to phase error, after WCDMA's correction algorithms and with the limited slot length taken into account. The results of such a MATLAB simulation are shown in Figure 30. In the MATLAB simulation the value for C in (32) was simply set to 0.1.

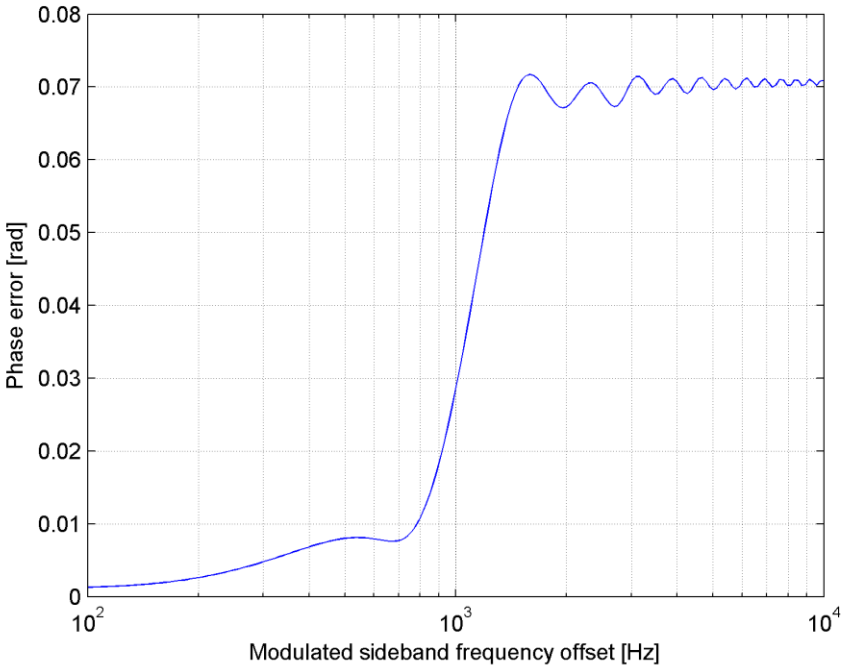


Figure 30 MATLAB simulation of how a swept modulated sideband (spur) affects the EVM (or phase error, in this case).

It is obvious that the correlation between the MATLAB simulation and the ADS simulation is very good, with regard to how spur energy at a certain frequency offset transfer to EVM (or phase error, which in this case is simply a scaling of EVM). The correlation is also good with theoretical phase error weight functions developed within Ericsson [20].

The effect of the spur's offset frequency on EVM is also possible to measure in the lab. This was done using a measurement setup with two signal generators and a spectrum analyser. One signal generator was set to generate a WCDMA signal (with a bandwidth of 3.84 MHz) at an IF frequency of 100 MHz. The other signal generator was set to generate an LO frequency at 2040 MHz. In addition to this, AM modulation was enabled. AM modulation with a constant modulation frequency will give rise to spurs on both sides of the carrier at the modulation frequency, i.e. similar to what you'd see when having a noisy carrier with spurious frequency components.

A mixer was then used to mix the WCDMA IF component at 100 MHz with the LO component at 2040 MHz, in order to have a WCDMA RF signal at 2140 MHz. This signal was then measured using a spectrum analyser with WCDMA decoding capabilities. The AM modulation frequency was then manually tuned to cover spur offset frequencies between 100 Hz and 10000 Hz in logarithmic steps and the measured EVM was recorded at each offset. The result is shown in Figure 31, which once again shows good correlation with the theoretical and simulated results. Note that the transfer function was measured for different AM modulation levels in order to ensure that the spur power and harmonics were not affecting the measurement (eg. due to mixer non-linearity).

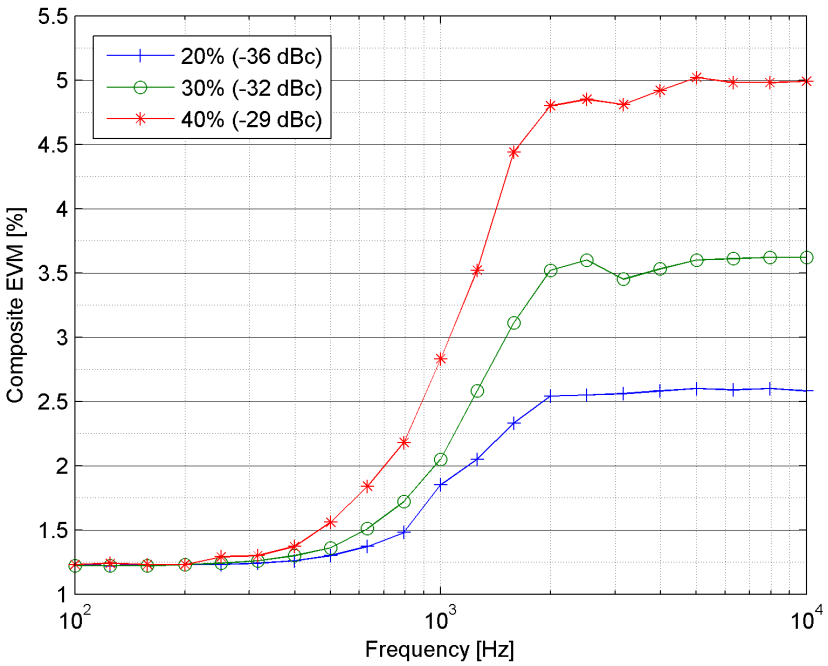


Figure 31 Measured transfer function from spur frequency offset to composite EVM for different modulation levels.

3.3.2 System-level ADC simulations

Another type of system simulation that might be of interest is to simulate an ADC that is clocked by a noisy oscillator (i.e. an oscillator containing phase noise). In the clocking world, one often speaks of oscillator jitter instead of oscillator phase noise. These two are different ways of referring to the same phenomenon.

When an ADC is clocked by an oscillator that contains phase noise, the clocking phase noise will get convoluted with the output spectrum [19]. As such, it will of course affect the signal-to-noise ratio (SNR). The effect on the SNR is derived in detail in [18].

Templates for simulating ADCs clocked with noisy oscillators are pre-existing in ADS and this sub-chapter will thus be very brief, not saying much more than stating the fact that they exist. For simulating an ADC clocked with a noisy oscillator one may use the example found under **DesignGuide > Analog/Digital Conversion > Analog to Digital Conversion > Test With Clock > SNR**. This test bench is complete except for the fact that it uses an ADC component which is ideal. One may want to change the ADC model to a Ptolemy component called **ADC_Timed** to simulate an ADC with a true RF clock and see the frequency domain convolution effects.

The ADC test bench (with the **ADC_Timed** component instead of the **ADC_with_clock_cosim** component) is shown in Figure 32. As in the previous EVM test bench, an **N_Tones** component is used to simulate a noisy clock with phase noise.

Prepared (also subject responsible if other)		No.		
EMARFAX				
Approved	Checked	Date	Rev	Reference
		2011-01-17	PA1	

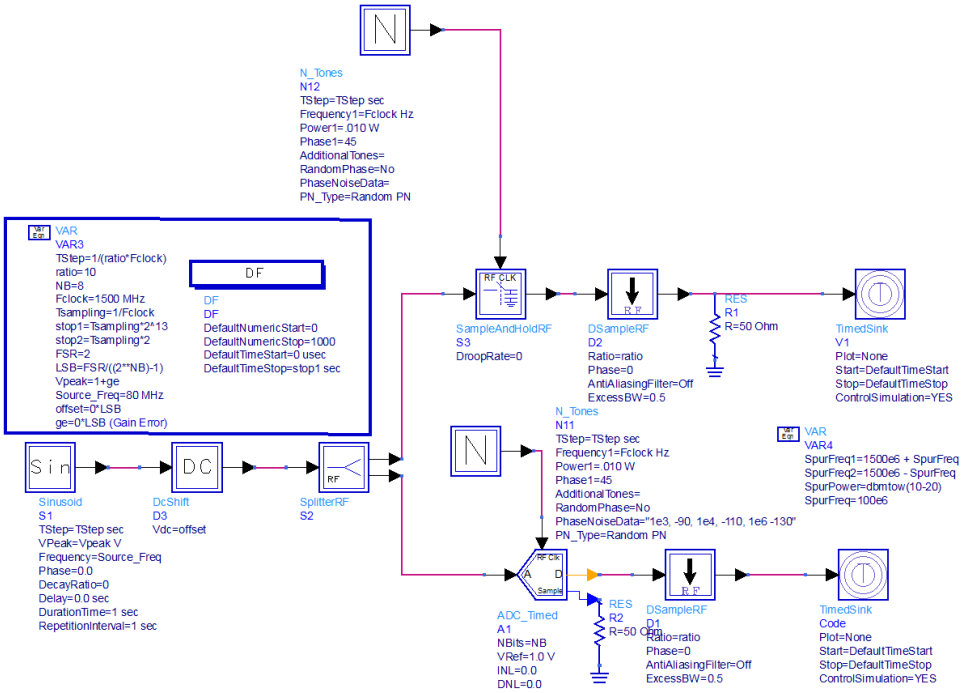


Figure 32 ADS schematic of ADC test bench.

The results of an ADC simulation are shown in Figure 33, where the frequency domain convolution effects are clearly visible. One can see the main tone at 80 MHz and several harmonics. Each harmonic has a distinct phase noise spectrum due to the convolution with the oscillator's phase noise. The SNR is shown as a dB value. In addition to SNR simulations, ADS' design guides also include examples to simulate INL, SINAD, SFDR, etc.

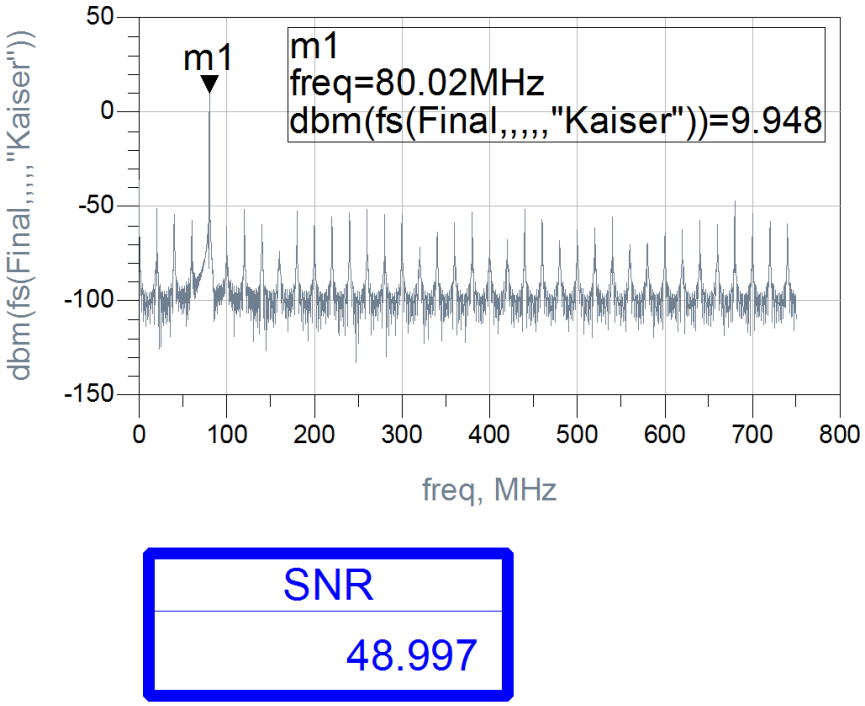


Figure 33 ADS data display belonging to the ADC simulation (only the output frequency spectrum and SNR display is shown), for an ADC clocked at 1500 MHz with an 80 MHz input signal.

It is important to note that the wideband phase noise of an oscillator has a very large effect on an ADC's SNR, as compared to the upmixing situation when an oscillator is used as an LO in a WCDMA chain, where wideband phase noise does not have a large effect on the EVM. For sampling clocks it is therefore very important to have the thermal noise floor as low as possible.

4 Results and conclusions

This chapter will try to summarise the results obtained. Since the scope of the thesis was quite general in its nature and did not include a goal of simulating or designing any specific PLL, the results will also be rather general and more resemble guidelines about what is possible to simulate in ADS and what is not. Specific results will however be given for how phase noise at different offset frequencies affects the output EVM in a WCDMA system.

The first part of the simulations focused on steady-state simulations. These types of simulations predict the resulting phase noise curve of a PLL, given its reference signal noise, VCO noise, loop filter parameters, etc. No transient simulation takes place. Simulations like these are possible to do in several ways, either by simply implementing the transfer functions in MATLAB or by using a dedicated PLL simulation program such as ADIsimCLK® or ADIsimPLL®. In addition to this, it is of course also possible to perform these simulations in ADS as was demonstrated in this thesis.

Steady state simulations are well established and quite reliable. Given correct input data the predicted output phase noise curve will generally correspond quite closely to what you would actually see if you measured it in the lab. However, one must keep in mind that several aspects of the PLL are not possible to simulate in a steady state simulation. For example, there is no way to simulate the locking process, since no transient simulation takes place. There is also no practical way of modelling disturbances such as pushing, pulling, etc. since these all take place in the time domain. The results of the ADS steady state simulations agree very well with simulations performed in for example ADIsimCLK. Using ADS for steady state simulations might be beneficial in cases where vendor tools are not satisfactory (i.e. when using PLLs from several different vendors). A downside to using ADS is that fractional spurs are not modelled in the steady state, if using fractional-N PLLs. One might also assume that vendor tools generally give more accurate results since they may include special algorithms to simulate a particular PLL's quirks.

In order to simulate locking processes, one needs to perform a time domain simulation. It is necessary to simulate over a large number of time steps and the simulations can be very time and memory consuming. Using time domain simulations it is also possible simulate the appearance of spurious frequencies that can be caused both by reference passthrough and other disturbances such as pushing. For pushing it is important to note that it is to some extent possibly to simulate such scenarios, but the models are severely limited since they are based on behavioural models and not the physical reality.

Most spurious frequencies that appear in the measured output spectrum of a real PLL are impossible to simulate using behavioural models in ADS, since they appear as a result of parasitic coupling etc. that is present in physical circuits but not in ADS simulations. It is to some extent possible to simulate the effect that a pushing signal would have on the PLL by injecting a voltage at different places along the loop.

Pulling is possible to simulate but is severely limited by the fact that ADS' behavioural model for a VCO does not include any type of tank circuit which reactance can be affected by a load (and thus changing the resonance frequency). The pulling results are thus not worth much in practical cases, since it is only possible to simulate one part of the phenomenon, namely the effect of reflections caused by mismatched loads.

Both the pushing and pulling simulations would benefit greatly from being simulated with a transistor level VCO instead of the behavioural model. Unfortunately, using a transistor level model adds a lot of complexity. Pushing and pulling might be easier to simulate on just a (transistor level) VCO and not an entire PLL.

ADS provides good facilities for simulating entire WCDMA signalling chains in its Ptolemy simulator. Using this type of simulations it is possible to simulate the effect of a spur on the system EVM, for example. It is also possible to investigate how phase noise at different frequency offsets affects EVM. Combining the results of simulations, theoretical calculations and actual measurements, one can draw the conclusion that phase noise below 1000 Hz or above 3.84 MHz has a very small effect on EVM.

In addition to being able to simulate EVM in a WCDMA system, ADS' Ptolemy simulator also has the capability to simulate ADCs clocked by noisy oscillators.

5 Discussion

PLLs continue to find use in almost all communication systems as well as in a multitude of other applications. PLLs are essential components for frequency synthesis, phase synchronisation, carrier recovery etc. In the future, some of these tasks may be better performed using other components. The current trend is to move as much as possible into the digital domain. The ideal digital system would have everything taking place inside a DSP only to be converted to an analogue signal by the use of a DAC at the very last stage. It is possible to build digital PLLs using so called “direct digital synthesis” (DDS). This replaces many of the traditional PLL components with digital counterparts. As of now the output frequencies are however limited [14].

PLLs are and will certainly continue to be very important electrical components in the foreseeable future. Therefore it is of course very relevant to be able to perform reliable simulations of PLLs. This thesis has shown that it is important to know what is practically possible to simulate and what is not. Certain aspects of PLLs, such as the steady state phase noise response, are certainly possible (and simple) to simulate. Other aspects, such as pulling, are almost impossible to simulate in practical cases.

Simulations can be a very helpful tool in the design process and allows for easy testing of new ideas and improvements. However it is important to stress that simulations cannot be trusted without verifying them in real life as well. Real designs tend to suffer from a lot of problems that the simulations do not take into account.

Simulating PLLs is a broad and quite complex topic due to the long simulation times needed in combination with the need for advanced models if one wants to accurately simulate all aspects of the PLL. In the future it would be of great interest to evaluate the possibilities of replacing the behavioural VCO model with a transistor level VCO model. This would presumably make both pushing and pulling simulations a lot more flexible and realistic. It would also be of interest to investigate how electromagnetic (EM) simulators can be used to simulate PLLs. This would allow for more accurate results with regard to for example spurious frequencies, provided that one was able to simulate an entire PCB with several disturbing signals.

6 References

[1] Harte, Lawrence. (2004) *Introduction to WCDMA: Physical Channels, Logical Channels, Network, and Operation*. [Electronic] ALTHOS: Books24x7.

[2] Razavi, Behzad. (2001) *Design of analog CMOS integrated circuits*. New York: McGraw-Hill Higher Education.

[3] Banerjee, Dean. (2006) *PLL Performance, Simulation and Design, Fourth Edition*. [Electronic] National Instruments.

[4] Tiebout, Marc. (2006) *Low Power VCO Design in CMOS*. Heidelberg: Springer Berlin.

[5] Best, Roland. (2007) *Phase-locked loops: design, simulation and applications, 6th ed*. New York: McGraw-Hill Professional.

[6] Kester, Walt. (2009) *MT-008: Converting Oscillator Phase Noise to Time Jitter*. Analog Devices application note.

[7] Hajimiri, Ali. (1998) A General Theory of Phase Noise in Electrical Oscillators. *IEEE Journal of Solid-State Circuits*, vol. 33, no. 2, pp. 179-194.

[8] Kundert, Ken. (2006) *Predicting the Phase Noise and Jitter of PLL-based frequency synthesizers*. Designer's Guide Consulting. Available online from http://www.designers-guide.org (downloaded 2011-03-03)

[9] Lee, Han-il et al. (2002) Scheme for no Dead Zone, Fast PFD Design. *Journal of the Korean Physical Society*, vol. 40, no. 4, pp. 543-545.

[10] Anderson, John B. (2005) *Digital Transmission Engineering*. New Jersey: IEEE Press.

[11] Silicon Laboratories. (2010) *AN256: Integrated phase noise*. Silicon Labs application note.

[12] Demir, Alper. (1997) Analysis and Simulation of Noise in Nonlinear Electronic Circuits and Systems. Berkeley: University of California. (Ph.D. thesis)

[13] Ferre-Pikal, E.S. et al. (1997) Draft revision of IEEE STD 1139-1988 standard definitions of physical quantities for fundamental, frequency and time metrology-random instabilities. *Proc. of the IEEE International Frequency Control Symposium*, pp. 338-357.

[14] Kern, Paul. (2007) Direct digital synthesis enables digital PLLs. *RFDesign*, July 2007 issue, pp. 26-30.

[15] Xanthopoulos, Thucydides. (2009) *Clocking in Modern VLSI Systems*. Marlboro: Springer US.

[16] Fetche, Rude et al. (2000) Analysis of the Effects of Supply Noise Coupling on Phase Noise in Integrated LC CMOS oscillators. *Radio and Wireless Conference*, pp. 199-202.

[17] Ojanperä, Tero and Prasad, Ramjee. (2001) *WCDMA: Towards IP Mobility and Mobile Internet*. Boston: Artech House.

[18] Kobayashi, Haruo et al. (1999) Aperture jitter effects in wideband ADC systems. *Proc. of the 6th IEEE International Conference on Electronics*, pp. 1705-1708.

[19] Brannon, Brad and Barlow, Allen. (2006) *AN-501: Aperture Uncertainty and ADC System Performance*. Analog Devices Application Note.

[20] Sahlman, Karl Gösta. (2009) *Mathematics for Phase and Frequency Error on Time Slot Basis*. Ericsson internal technical report no. 14/0363-10/FCP 103 9166 Uen.

Prepared (also subject responsible if other)		No.		
EMARFAX				
Approved	Checked	Date	Rev	Reference
		2011-01-17	PA1	

APPENDIX

A. MATLAB code

```
% Simulate "theoretical" EVM/phase error resulting from an LO with an added
% modulated sideband at a certain frequency offset.

clear all;
close all;

Fcarrier = 1e6; % Carrier frequency

T = 10e-3/15;   % Slot time ("sampling time")
N = 10000;      % Number of points

t = linspace(0, T, N);

% Generate our reference signal, both in time-domain and phasor-domain using
% the Hilbert transform.
signal = cos(2*pi*Fcarrier*t);
complex_signal = hilbert(signal);

f = [];
pe_rms = [];
pe_rms_no_corr = [];

% 201 evenly spaced steps from 10^2 to 10^4 in logarithmic scale
for j=1:201
    Fspur = 10^(2+(j-1)/100);
    spur_signal = cos(2*pi*Fcarrier*t + 0.1*cos(2*pi*Fspur*t));
    complex_spur_signal = hilbert(spur_signal);

    phase_error = phase(complex_signal)-phase(complex_spur_signal);

    % Perform a linear regression ("Best fit straight line")
    p = polyfit(t, phase_error, 1);

    % Calculate RMS phase error, as a deviation from the linearly regressed
    % line
    phase_error_rms = sqrt(sum((phase_error - polyval(p,t)).^2)/N)
    phase_error_no_correction = sqrt(sum(phase_error.^2)/N)

    % Save our values
    f(j) = Fspur;
    pe_rms(j) = phase_error_rms;
    pe_rms_nocorr(j) = phase_error_no_correction;
end

% Plot transfer function
figure(1)
semilogx(f, pe_rms);
xlabel('Modulated sideband frequency offset [Hz]');
ylabel('Phase error [rad]');

% Plot transfer function without error correction
figure(2)
semilogx(f, pe_rms_nocorr);
xlabel('Modulated sideband frequency offset [Hz]');
ylabel('Phase error [rad]');
```

B. Instruments used

In order to measure the transfer function from spur frequency offset to resulting phase noise, the following devices were used:

- Mini-Circuits ZEM-4300 mixer
- Rhode & Schwarz SMU200A signal generator with WCDMA add-on
- Rhode & Schwarz SMIQ06B signal generator for AM modulated spur
- Rhode & Schwarz FSG 9 kHz – 8 GHz spectrum analyser