Efficient Simulation Techniques for Modulated Delta-Sigma Fractional-N Synthesizers

A delta-sigma PLL is an attractive solution for agile frequency synthesis or even direct modulation. Theoretically, this design offers the advantage of eliminating spurious emissions. However, in practice, this is not always the case. Simulation techniques give the designer a tool to analyze the performance of the synthesizer, as well as a means to avoid surprises.

By Cyril Descleves

Efficient simulation methods exist to analyze spurs (spurious outputs) in a fractional-N delta-sigma PLL. A delta-sigma PLL is an attractive solution for agile frequency synthesis or even direct modulation. This architecture can meet requirements such as low power consumption and simple topology, and is suitable for high-level integration.

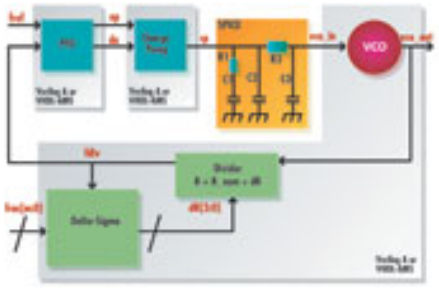
The following information investigates possible reasons for the presence of fractional spurs in the output spectrum of a delta-sigma PLL. It describes the basic operation of the delta-sigma PLL and its application in frequency synthesis and direct modulation, or GMSK.

From Fractional-N to Delta-sigma PLLs

Historically, fractional-N PLLs have been used to allow many more distinct output frequencies, compared to simpler integer-N PLLs using the same reference frequency.

Fractional PLLs tend to generate so-called *fractional spurs*, because of the deterministic alternating pattern used to switch the division ratio.[1, 2, 3]

Periodic phase shifts are inevitable and create large unwanted spurs in the output spectrum. It is often difficult to make sure that these spurs will be low enough to avoiding degrading the performance to unacceptable levels.

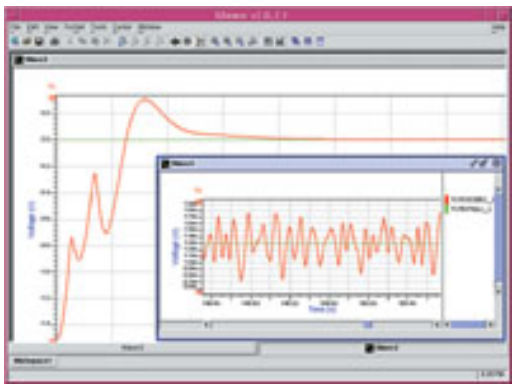


[1]

Using a delta-sigma converter to modulate the division ratio, instead of a periodic N/(N+1) modulation pattern, the benefits of an arbitrarily fine frequency resolution can be retained while eliminating these fractional spurs &#151 at least theoretically.[2, 3, 4, 5]. In this case, the division ratio is continuously modulated by the output code of the delta-sigma modulator. The average value of the division ratio provides the desired output frequency.

Figure 1 shows the basic architecture for the delta-sigma PLL. In a frequency synthesis application, the delta-sigma modulator is driven with a constant digital input, labeled &quotfrac&quot in Figure 1, coded on *m* bits.

This width, *m*, defines the delta-sigma reference and its periodicity. Depending on its order *n*, the modulator generates an output code ranging from &#150 2^{n}&#150 1 to 2^{n}&#150 1&#150 1. Third-order or fourth-order modulators are among the most common choices. The average value of the output code is equal to &quotfrac,&quot with *m*-bit accuracy. Thus, to increase the frequency resolution, the delta-sigma reference must be increased, which means that the width of the internal registers and accumulators of the delta-sigma must be enlarged.



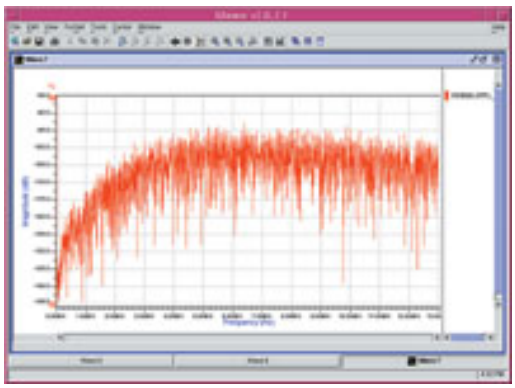
[2]

The delta-sigma is a 100% digital block. It can be clocked by either the reference clock signal or the divider output, although using the divider output signal is reported to yield better performance.

The Problem of Spurs in the Output Spectrum

Theoretically, the delta-sigma PLL should not generate any spurs in its output spectrum as the delta-sigma performs its well-known noise shaping function. The only obvious reason why spurs could be present is if the delta-sigma output itself contains fractional spurs. Then they would inevitably propagate to the output.

Delta-sigma modulators have an unfortunate natural tendency to generate fractional spurs in their output spectrum when driven with constant inputs. The frequencies where these spurs appear are easily predicted. The spurs are usually the largest when the input code is close to either 0, *ref*/2 or *ref*, where *ref* is the maximum input code to the delta-sigma. Various techniques exist to counter this effect and obtain a &quotclean&quot (i.e., fractional-spur-free) spectrum out of the delta-sigma itself. Adding a small random value to the input (dithering) is one such method. This increases the noise level, but it can be kept within acceptable limits.[4] It is also possible to slightly modify the architecture of the input stages so that the delta-sigma does not generate any spurs by itself.

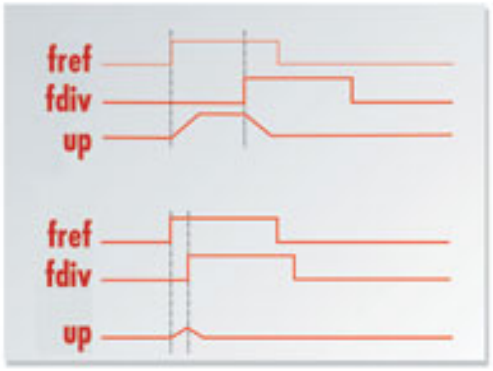


[3]

In order to simulate such PLLs, the first thing checked is that the delta-sigma itself does not generate any spur when driven with constant inputs.

Even with a clean delta-sigma output, it is, however, still quite common to observe spurs in a delta-sigma synthesizer. They are caused by the non-linearities in the PFD, CP or VCO. Following is a review of the possible sources of non-linearity.

Ideally the phase-frequency detector should have a linear phase transfer function, but the presence of a dead-zone, for example, creates a non-linearity in the phase transfer function. Similarly, the charge pump should have perfectly symmetrical *sink* and *source* currents, but this is never exactly so, which may also create a non-linearity. The VCO does not have a perfectly linear voltage-to-frequency transfer function &#151 there is always some level of distortion. Unless special care is taken to resynchronize the divider's output, small delay variations depending on the divide ratio can also create a non-linearity.



[4]

All of these effects potentially combine to create output spurs. The location of the

spurs is predictable; however, their exact amplitude quite often remains difficult to predict. The techniques described here can help to predict these effects though simulation.

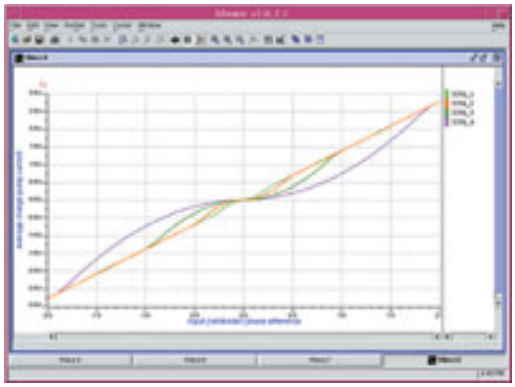
Frequency Synthesis

In the case of a frequency synthesis application, the delta-sigma PLL is used to generate precise frequencies with fine resolution. The delta-sigma is driven with a constant input code, and its output modulates the division ratio.

Modeling and Simulation Methodology

SPICE-level simulation of PLLs is, unfortunately, well-known as a tough problem. The major source of difficulty is the ratio of time constants involved in the system. In frequency synthesis for wireless communication applications, the typical VCO frequency is in the range of a few gigahertz, and the reference frequency is a few tens of megahertz.

If using SPICE-type algorithms, the simulation has to follow the fastest signal; in this case the VCO, during a total simulation time often in the range of milliseconds. Thus, a huge number of time points have to be computed, and the associated CPU time is prohibitive in most realistic cases. Even when a designer is ready to spend the necessary CPU time, the output files are often extremely large, making their post-processing analysis difficult, if not impossible.

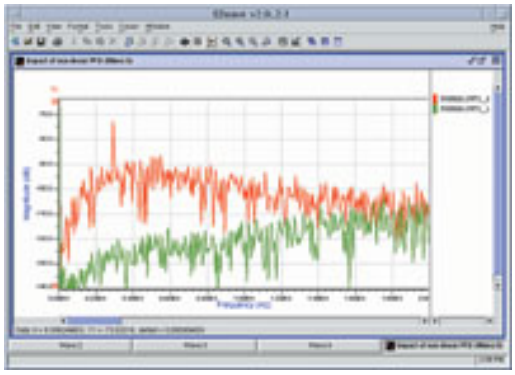


[5]

Two techniques are used to avoid this problem. First, behavioral models are used in a top-down approach; i.e., starting with ideal behavioral models and gradually

replacing certain blocks with non-ideal models or even their SPICE representation to obtain greater accuracy. Notice that these models are continuous time models; they are not sampled. They can be used in mixed-signal and RF simulators. In this first step transient simulation is used, but with fast models (see below: &quotFirst Step: Closed-loop Simulation&quot). The VCO input signal is obtained, from a closed-loop simulation, both in the time domain and in the frequency domain, through FFT.

Second, to obtain the true output modulation spectrum, a mixed time-frequency simulation algorithm, such as the *modulated steady-state* algorithm is used. This way the VCO is not simulated with a plain transient algorithm. Instead the VCO is driven with the signal obtained in the first step, and the true modulation spectrum is obtained at the VCO output, in a fraction of the CPU time that would be required if using transient simulation.



[6]

First Step: Closed-loop Simulation

Figure 1 also shows the modeling strategy for the first step. The charge pump and the frequency detector are modeled in Verilog-A (or VHDL-AMS). The loop filter is still modeled in SPICE (only R and C components). The VCO, divider and delta-sigma are lumped into a single model, also in Verilog-A or in VHDL-AMS. Merging the VCO and the divider into a single model prevents explicit generation of the VCO output signal at a few gigahertz. As already explained, this fast signal is the main reason for the slowness of regular transient simulations.

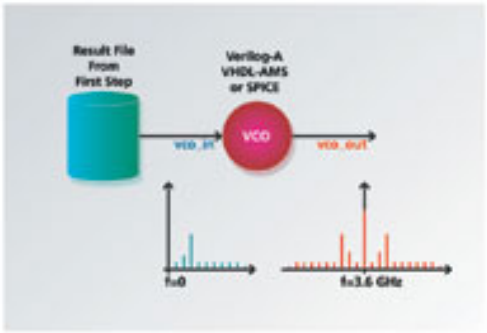
The charge pump and phase-frequency detector are simple models. The charge pump model includes a parameter to model the gain mismatch between the *source* and *sink* currents.

Additional outputs are added to the model for convenience and to monitor the delta-sigma output code and the normalized instantaneous frequency. A parameter is provided to force some non-linear distortion on the voltage-to-frequency transfer function of the VCO.

Closed-loop Simulation with Ideal Models

An ideal simulation is used to begin, where all models have no distortion or non-linearity. In this ideal case, the output is checked to be sure that it does not contain any fractional spurs.

These closed-loop simulations take three minutes of CPU time on a 2 GHz Linux laptop. Figure 2 shows the dynamics of the locking process in the closed-loop PLL.



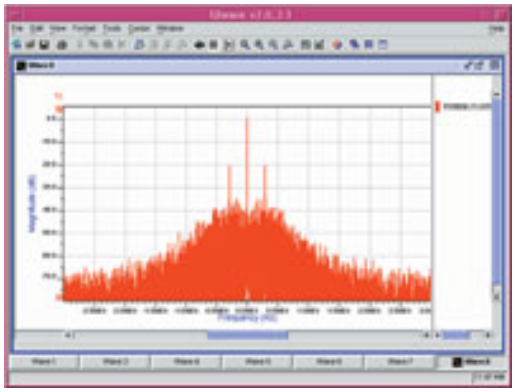
[7]

A delta-sigma PLL has no steady-state solution because the division ratio is changing continuously. Thus the control voltage, even when the PLL is &quotlocked,&quot is changing continuously, modulating the VCO output frequency. The shape of the control voltage when the PLL is &quotlocked&quot is shown in the smaller embedded window (Wave: 6) in Figure 2. The expected average value is also shown on the picture.

Figure 3 shows the FFT of this same VCO control voltage. We can check in this plot that no fractional spurs are present in the VCO input spectrum.

Closed-loop Simulation with Non-linear Models

To implement a non-linearity in the phase transfer function, a model is created of the PFD with finite rise and fall times for the outputs (the UP and DN signals that control the charge pump). It works as follows:
&#149 When the rise/fall times of the pulses are short, the amount of charge transferred to the loop filter is almost directly proportional to the phase shift between the PFD input signals.
&#149 When these rise/fall times become large, the almost-rectangular pulses that would be produced by an ideal PFD driven by almost-in-phase signals become triangular (see Figure 4) and the area below said triangle decreases when the phase difference becomes smaller.



[8]

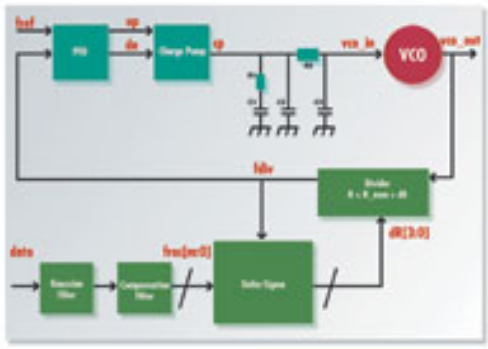
Thus, the net charge transferred to the loop filter is lower than it would be with an ideal model, creating the desired distortion. Notice that this technique is only a convenient way to create a distorted transfer function, but any other technique could be used instead.

To visualize the transfer function of the PFD/CP block, the PFD and CP are connected, and they drive the PFD with two signals that have slightly different phases (from &#150 2&#176 to +2&#176). The output current *I* of the CP is monitored and plotted versus this phase difference *delta_phi*.

Figure 5 shows these phase transfer functions for different values of the rise/fall times parameters of the PFD. When these parameters are equal to zero, an ideal transfer function is obtained (a straight line through the [*I* = 0, *delta_phi* = 0] point). When it is increased, the shape of the transfer function becomes more and more distorted around the central *delta_phi* = 0 area. The width of the distorted zone also increases. Thus, changing rise/fall time is indeed a way to force some non-linear behavior onto the behavior of the PFD. This is used to monitor the impact upon the fractional spurs.

Using a non-linear PFD model this time, the same closed-loop simulations are rerun as before.

These closed-loop simulations take three minutes of CPU time on a 2 GHz Linux laptop. Such a quick simulation time opens the door to easy &quotwhat-if&quot analysis, and this is certainly one of the main points learned from the experiments described in this article.



[9]

Figure 6 shows the spectrum of the VCO input (red), compared to the same spectrum from the ideal simulation (green). The PFD is highly distorted, using high values for the rise/fall time parameters. The noise floor at low frequencies rises significantly, but this time a fractional spur emerges by 15 dB to 20 dB above the noise floor. The spur is located exactly where &quotexpected,&quot given the value of the input fraction feeding the delta-sigma. In this case, the delta-sigma is driven with *frac* = 11, and *n_ref* = 10, *fref* = 26 Meg; the fractional spur is thus expected at $(2.frac+1)/(2^{11}).fref$, i.e. 291.992 kHz. In addition to the PFD, another possible source of non-linearity is a gain mismatch in the charge pump. When designing a charge pump, it is quite difficult to achieve identical *source* and *sink* currents, and some level of mismatch between these currents always remains. This results in a non-linearity with respect to the phase difference, so the assumption is that it would create fractional spurs as well. The charge pump model used in these simulations has a parameter modeling such a mismatch. However, none of the experiments with this mismatch parameter have led to the observation of fractional spurs. The noise floor increases with the amount of mismatch, but no particular spurs emerge from the noise floor. The gain mismatch parameter was varied from 0.01% up to an unrealistic 10%.

Actually, this type of non-linearity might not necessarily lead to the generation of spurs. Instead, it may simply force a static offset, and the PLL will still operate

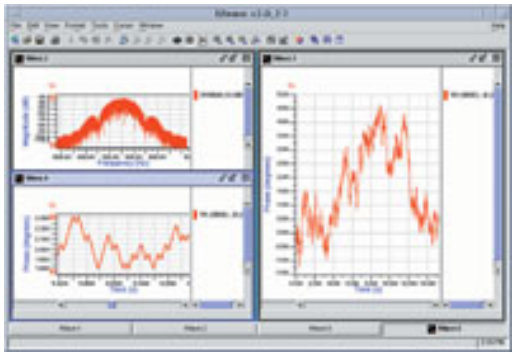
around a linear point. To study the effects of non-linearities in the CP itself, more detailed effects should be included, such as finite charge/discharge times of the commands to the UP and DN switches and possibly the asymmetry between these times.

Yet another source of non-linearity would be slightly different delays in the divider, depending on the divide ratio. Modifying the model of the VCO/divider to add such variable delays to the output would be quite easy.

Second Step: MODSST Simulation

Figure 7 shows the simulation strategy for the second step. In this simulation, a single model is used, the VCO. The goal is to obtain the true modulation spectrum at the output of the VCO, that is, a spectrum centered on 3.6 GHz. To do so, a transient simulation is not run, which would lead to unacceptable CPU time. Instead the MODSST algorithm from ADMS RF is used.

When &quotlocked,&quot the PLL generates an input control to the VCO that is *almost* constant. *Almost*, but *not quite*, of course. Because of the pseudo-random nature of the delta-sigma output code, the VCO input changes in random fashion, centered on a DC value that yields the average output frequency corresponding to the delta-sigma input code.



[10]

However, consider that the spectrum of the VCO output signal does not change abruptly over time. Instead, a frequency modulation operation, where the modulation input has a small amplitude, and a much smaller bandwidth, is seen. Thus the MODSST algorithm is expected to perform much better than a plain transient algorithm, in terms of CPU time.

The control input is driven with the signal obtained from the first step. By using a VCO model with the same characteristics as the one used in the closed-loop simulation of the first step, particularly its gain and transfer function, this procedure is exact.

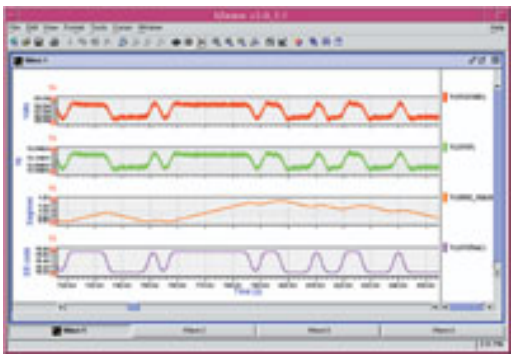
When in closed-loop, the VCO &quotsees&quot exactly the signal that is used to drive it, so the output spectrum provided by this open-loop simulation is the same spectrum as the one that exists when the PLL operates in closed-loop. Using a slightly different VCO model, or even a SPICE model, the results must be interpreted carefully. The qualitative behavior will be correct, but not the quantitative results of course.

Figure 8 shows the output spectrum from the VCO, when driven with the signal obtained in the first step. The spectrum is automatically centered around $f = 0$, to allow more convenient reading of the offsets to the carrier, directly. The fractional spurs present in the control input voltage (see simulations from the first step) are, of course, present in the output spectrum. They are located at 3.6 GHz &#177 291.992 kHz, as expected. This simulation takes about 30 minutes of CPU time on a 2 GHz Linux laptop. In comparison, a transient simulation would require several days of CPU time, and require sophisticated post-processing in order to obtain a usable FFT spectrum.

Direct Digital Modulation

It is also possible to use such delta-sigma frequency synthesizers for GMSK or GFSK modulation. For direct GMSK or GFSK modulation, the delta-sigma input is not held constant as in a frequency synthesis application. Instead it is changed according to the digital bit stream to be modulated, as shown in Figure 9.

The data stream is fed into a Gaussian filter, followed by a pre-compensation digital filter, before driving the delta-sigma.[7] The pre-compensation filter compensates the PLL transfer function so that higher modulation rates can be achieved. All the filtering and delta-sigma modulation is performed digitally, which allows high levels of integration.



[11]

The following information shows how to drive the PLL with a GMSK signal, not taking into account the pre-compensation. The digital pre-compensation filter would typically be represented as Verilog or VHDL code. This would make virtually no difference in terms of CPU time. First, a GMSK signal is generated, using built-in digitally modulated source models from the simulator. Second, this signal is used to drive a closed-loop simulation. Additionally, the VCO input voltage is saved, which this time will carry the modulation information. Finally, the saved VCO control voltage is used to run a mixed time-frequency simulation and compute the true output spectrum. Then the modulation spectrum is compared when ideal and non-ideal models of the PFD are used.

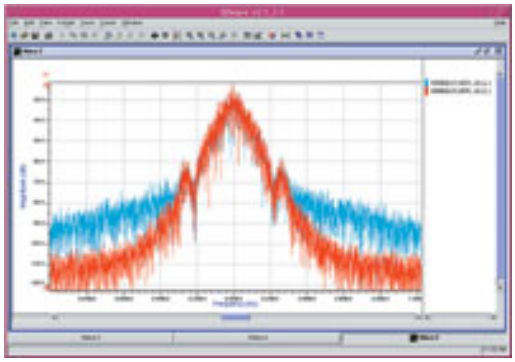
To compute a realistic modulation spectrum, we need a few hundred symbols. With a 3.7 &#181s symbol rate, the CPU time will remain quite acceptable.

Figure 10 (right and bottom-left windows) shows the GMSK signal (this is actually the phase of a GMSK signal). In a second step, the time-derivative of this signal is used to obtain the instantaneous frequency deviation signal needed to drive the delta-sigma with. A PRBS 19 input data sequence has been used, thus the &quotrandom&quot nature of the signal. The phase of a GMSK is constantly changing from one symbol to the next. The derivative of the phase signal with respect to time is constant during the symbols, and corresponds to the maximum frequency deviation (67.5 kHz in this case). The plot in the top-left window also shows the classical spectrum shape of this GMSK signal.

This GMSK signal is used to drive othe closed-loop delta-sigma PLL model, and thus obtain the VCO control voltage. This is similar to what has been done when using the model for frequency synthesis application, except that now the VCO/divider model used has an additional input for the fraction code. This fraction code is computed from the GMSK signal.

Then the instantaneous frequency deviation is derived, and the delta-sigma is driven with this signal.

Starting from the top are the VCO control voltage, the instantaneous frequency deviation, the phase in degrees and the delta-sigma code. The frequency deviation signal is a Gaussian-filtered (beta = 0.35) version of the input binary data stream. The complete simulation is run over 5 ms, which represents more than 1300 symbols. This simulation takes 45 minutes. Finally, a MODSST simulation of the VCO is run when driven by the modulated control voltage, to obtain the spectrum of the transmitted signal.



[12]

This time the VCO model is a SPICE transistor-level representation.

Figure 12 shows the spectra corresponding to an ideal PFD (red) and to a non-ideal one (blue). The non-ideal spectrum does not exhibit any *localized* fractional spur, as was the case with frequency synthesis (see Figure 6). Instead, the noise level outside the PLL bandwidth (about 300 kHz) increases, which is the expected effect.

Conclusion

The combination of modeling techniques using standard analog behavioral languages (Verilog-A and VHDL-AMS) together with efficient mixed time-frequency algorithms, makes it easy to analyze the impact of non-linearities upon the existence of spurs or noise in the output spectrum of delta-sigma PLLs. CPU times of these simulations are quite reasonable and a single simulation environment is used.

References

1. de Muer, Brian and Michael Steyaert. &quotCMOS Fractional-N Synthesizers,&quot KAP, 2003.
2. Hegazi, E. and A.A. Abidi. &quotA 17 mW Transmitter and Frequency Synthesizer for 900-MHz GSM Fully Integrated in 0.35-&#181m CMOS,&quot *IEEE Journal of Solid State Circuits*, Vol 38, n5, May 2003.
4. McMahill, D.R. and C.G. Sodini. &quotA 2.5-Mb/s GFSK 5.0-Mb/s 4-FSK Automatically Calibrated Sigma-delta Frequency Synthesizer,&quot *IEEE Journal of Solid State Circuits*, Vol 37, n1 January 2002.
3. Leenaerts, Domine, Johan van der Tang and Cicero Vaucher. &quotCircuit Design for RF Transceivers,&quot Kluwer AP, 2001.
5. Rategh, Hamid R. and Thomas H. Lee. &quotMulti-GHz Frequency Synthesis & Division &#151 Frequency Synthesizer Design for 5 GHz Wireless LAN Systems,&quot Kluwer AP, 2001.
6. Rhee, W., B.S. Song and A. Ali. &quotA 1.1 GHz CMOS Fractional-N Frequency Synthesizer with a 3-b Third-order Delta-sigma Modulator,&quot *IEEE Journal of Solid State Circuits*, Vol 35, n10, Oct 2000.
7. Riley, Thomas A.D., et al. &quotTechniques for In-band Phase Noise Reduction in Delta-sigma Synthesizers,&quot *IEEE Trans. on Circuits and Systems*, Vol 50, n11, Nov 2003.

Acknowledgments
Philippe Level and Serge Ramet from the Cellular Terminal Division at ST Microelectronics in Grenoble, France, shared their technical expertise about delta-sigma PLLs. This work would not have been possible without their contribution and constructive feedback.

About the Author
Cyril Descleves graduated from the Ecole Supérieure d'Electricité, France, in 1985. He is currently with Mentor Graphics, where he is Product Marketing Manager in the Deep SubMicron division, in charge of the analog and RF simulation products. Prior to joining Mentor, he was with Dolphin Integration as the Engineering Manager. Prior to this, he managed several projects in digital and mixed-signal IC design, device characterization and CAD methodology.

Links:
[1] http://www.wirelessdesignmag.com/sites/wirelessdesignmag.com/files/legacyimages/0512/wd512coverstory_1_lrg.jpg
[2] http://www.wirelessdesignmag.com/sites/wirelessdesignmag.com/files/legacyimages/0512/wd512coverstory_2_lrg.jpg
[3] http://www.wirelessdesignmag.com/sites/wirelessdesignmag.com/files/legacyimages/0512/wd512coverstory_3_lrg.jpg

[4] http://www.wirelessdesignmag.com/sites/wirelessdesignmag.com/files/legacyimages/0512/wd512coverstory_4_lrg.jpg
[5] http://www.wirelessdesignmag.com/sites/wirelessdesignmag.com/files/legacyimages/0512/wd512coverstory_5_lrg.jpg
[6] http://www.wirelessdesignmag.com/sites/wirelessdesignmag.com/files/legacyimages/0512/wd512coverstory_6_lrg.jpg
[7] http://www.wirelessdesignmag.com/sites/wirelessdesignmag.com/files/legacyimages/0512/wd512coverstory_7_lrg.jpg
[8] http://www.wirelessdesignmag.com/sites/wirelessdesignmag.com/files/legacyimages/0512/wd512coverstory_8_lrg.jpg
[9] http://www.wirelessdesignmag.com/sites/wirelessdesignmag.com/files/legacyimages/0512/wd512coverstory_9_lrg.jpg
[10] http://www.wirelessdesignmag.com/sites/wirelessdesignmag.com/files/legacyimages/0512/wd512coverstory_10_lrg.jpg
[11] http://www.wirelessdesignmag.com/sites/wirelessdesignmag.com/files/legacyimages/0512/wd512coverstory_11_lrg.jpg
[12] http://www.wirelessdesignmag.com/sites/wirelessdesignmag.com/files/legacyimages/0512/wd512coverstory_12_lrg.jpg