DESIGN AND SIMULATION OF FRACTIONAL-N PLL FREQUENCY SYNTHESIZERS

Mücahit Kozak and Eby G. Friedman

Department of Electrical and Computer Engineering
University of Rochester
Rochester, New York 14627-0231

ABSTRACT

A fast simulation environment has been developed using MATLAB™ and CMEX™ for behavioral level simulation of Delta-Sigma (ΔΣ) based Fractional-N PLL frequency synthesizers. The simulator uses a difference equation approach with a uniform time step. To support a uniform time step in the simulation, the continuous-time *average current-to-voltage* loop filter transfer function is modeled as a discrete-time *charge difference-to-voltage* transfer function. Simulation results are presented on a type-II fourth-order PLL frequency synthesizer employing a third-order MASH ΔΣ modulator.

1. INTRODUCTION

Delta-Sigma (ΔΣ) based Fractional-N PLL synthesizers are extensively used in wireless communication applications as a local oscillator to generate accurately defined frequencies. The technique offers high switching speed, low phase noise, and narrow channel spacing [1]. The design of Fractional-N PLL synthesizers, however, requires an iterative design process due to the large set of system parameters that must be optimized to achieve the desired phase noise, settling time, and fractional spur rejection. In addition, a ΔΣ modulator used to instantaneously alter the feedback division modulus introduces excessive phase noise and fractional spurs. A behavioral level simulator is required to reduce the design turnaround time as well as assess the phase noise contribution and fractional spur rejection of the ΔΣ modulator before the physical design phase. The need for a behavioral level simulator is strengthened by the characteristic that both the PLL and the ΔΣ modulator are nonlinear systems. Mathematical approximations based on a small signal analysis and a white noise assumption do not accurately characterize the system behavior. Transistor level simulators such as SPICE or SPECTRE are not suitable for fast simulation of such complex systems.

Simulating a ΔΣ based Fractional-N PLL frequency synthesizer is a non-trivial task due to the mixed-signal nature of the system, and the time varying nature of the division modulus in the feedback loop. Moreover, since the output frequency is two to three orders of magnitude higher than the loop filter time constants, an exorbitant amount of samples are required for an accurate simulation. In recent years, various behavioral level simulators for Fractional-N PLL synthesizers have been reported to address these challenges. In [2], Perrott developed a custom C++ simulator for the behavioral simulation of Fractional-N PLL systems with uniform time steps based on an area conservation principle to minimize the adverse effects of signal quantization. In [3], Brigati *et al.* developed a simulation environment using MATLAB™ and SIMULINK™. A time-domain simulator has also been reported by Fan in [4]. Cassina *et al.* developed an event-driven simulator with non-uniform time steps using Verilog [5].

In this paper, a new simulation environment is developed for Fractional-N PLL frequency synthesizers based on a mixed MATLAB™ and CMEX™ platform. The continuous-time *average current-to-voltage* transfer function of the charge pump loop filter is modeled as a discrete-time *charge difference-to-voltage* transfer function, enabling the use of a uniform time step during simulation. Due to the simple integration with MATLAB™ and faster execution speed, CMEX™ is preferred to stand-alone C code [6]. Compared to previously reported simulators [2]-[5], the proposed simulator is the fastest known simulator achieved to date.

This paper is organized as follows. In Section 2, the simulation environment is described together with the system level design of a PLL. Simulation results for the power spectral density (PSD), phase noise, and settling time are presented in Section 3. Additionally, a method for eliminating fractional spurs is demonstrated. Finally, some conclusions are offered in Section 4.

2. BEHAVIORAL LEVEL SIMULATION

This section is composed of two subsections. The first subsection outlines the design of a type-II fourth-order PLL. The simulation model of the PLL is described in the second subsection.

2.1 Design of the Loop Filter

A block diagram of a Fractional-N PLL frequency synthesizer is shown in Figure 1. The circuit includes a phase-frequency detector (PFD), a charge pump loop filter, a Voltage Controlled Oscillator (VCO), a programmable multi-modulus divider, and an all-digital ΔΣ modulator. The static input word K is processed by a ΔΣ modulator to produce an encoded oversampled sequence. This sequence is used to alter the division modulus of

* This research was supported in part by the Semiconductor Research Corporation under Contract No. 2003-TJ-1068, the DARPA/ITO under AFRL Contract F29601-00-K-0182, the National Science Foundation under contract No. CCR-0304574, the Fulbright Program under Grant No. 87481764, grants from the New York State Office of Science, Technology & Academic Research to the Center for Advanced Technology – Electronic Imaging Systems and to the Microelectronics Design Center, and by grants from Xerox Corporation, IBM Corporation, Intel Corporation, Lucent Technologies Corporation, and Eastman Kodak Company.

a multi-modulus divider in the feedback loop. Essentially, the average value of the encoded ΔΣ output is equal to the DC input word *K*, resulting in an output frequency at a fractional multiple of the reference frequency.

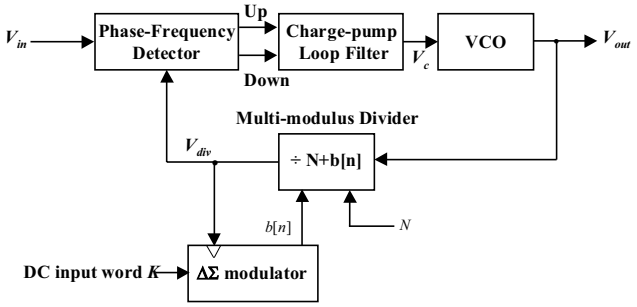


Figure 1. Fractional-N PLL frequency synthesizer

As a demonstrative example, a Fractional-N PLL frequency synthesizer for GSM900 receiver applications with a frequency range of 880 MHz to 915 MHz, a channel spacing of 200 kHz, and a settling time of 10 μs is targeted, where the reference frequency is 20 MHz.

Typically, a higher-order loop filter is used in Fractional-N PLL frequency synthesis applications to provide adequate suppression of the reference spurs as well as the high frequency phase noise from the ΔΣ modulator. The charge pump loop filter topology is a third-order passive network as shown in Figure 2. The use of a higher-order loop filter, however, requires careful design consideration, as the PLL is prone to instability. The *average current-to-voltage* transfer function of the loop filter is

$$F(s)=\frac{V_c(s)}{I_{avg}(s)}=\frac{D\left(s+1/\tau_1\right)}{\tau_2\tau_3 s^3+\left[\tau_2+\left(\frac{D}{R_2}+1\right)\tau_3\right]s^2+\left(\frac{D\tau_3}{\tau_1 R_2}+1\right)s}, \quad (1)$$

where $D=R_1C_1/\left(C_1+C_2\right)$, $\tau_1=R_1C_1$, $\tau_2=R_1C_1C_2/\left(C_1+C_2\right)$, and $\tau_3=R_2C_3$. The open loop transfer function of the PLL can be determined from the following expression,

$$G(s)=\frac{K_dK_vF(s)}{sN_{mean}}, \quad (2)$$

where K_d and K_v is the PFD constant and the VCO constant measured in A/rad and rad/(sec*volt), respectively. N_{mean} is the geometric mean of the maximum and minimum division ratio required to span the desired frequency band (in this case, $N_{mean}=44.86$). The open loop transfer function of the PLL has a zero located at $w_z=-1/\tau_1$, two poles at the origin, and two additional high frequency poles, denoted as w_{p1} and w_{p2}. Note that as long as $w_{p2}>>w_{p1}$, the non-zero poles can be approximated by $w_{p1}\approx-1/\tau_2$ and $w_{p2}\approx-1/\tau_3$.

To achieve a 10 μs settling time, the unity gain frequency of the open loop transfer function is located at $w_u=2\pi\cdot200$ krad/sec. 60° of phase margin is chosen to provide good settling behavior, dictating that $1/\tau_1=2\pi\cdot50$ krad/sec and $1/\tau_2=2\pi\cdot800$ krad/sec. The high frequency pole is located at $1/\tau_3=2\pi\cdot6.6$ Mrad/sec to provide an additional 20 dB attenuation of the reference spurs. With these passive component values, the open loop transfer function of the PLL is displayed in Figure 3, where the unity gain frequency is 199.18 kHz and the phase margin is 59.8°.

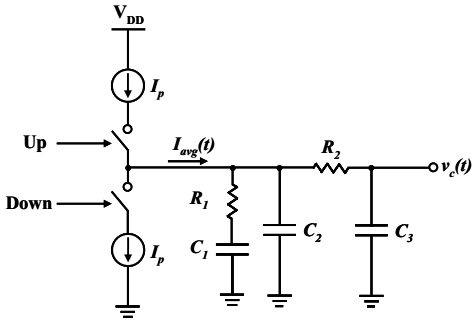


Figure 2. A passive third-order charge pump loop filter

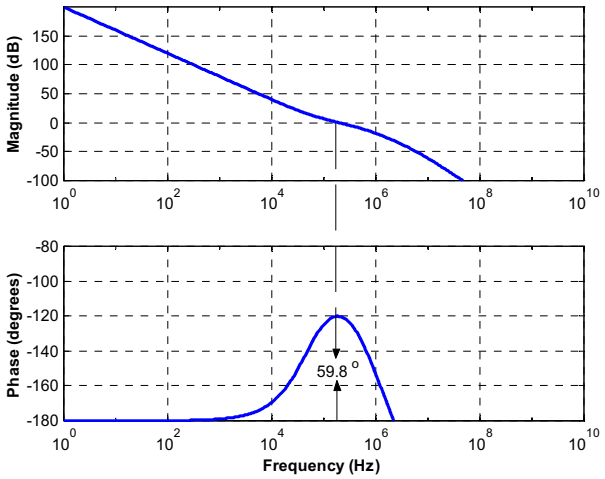


Figure 3. Open loop transfer function of the PLL

2.2 Simulation Model

In this section, a difference equation model is described which uses uniform time steps to simulate Fractional-N PLL frequency synthesizers. A mixed MATLAB™ and CMEX™ platform is used as illustrated in Figure 4, where the main MATLAB **.m** file calls a custom subroutine written in CMEX (which stands for C for MATLAB executable). This configuration results in a high degree of versatility in a simulation environment, as CMEX files easily integrate with MATLAB.

The custom CMEX™ subroutine is compiled into a **.dll** (dynamically linked library) file, and performs most of the computational complexity involved in simulating a Fractional-N PLL synthesizer. The main MATLAB .m file is used to determine the specifications, calculate the passive component values of the loop filter, and perform PSD estimation and visualization.

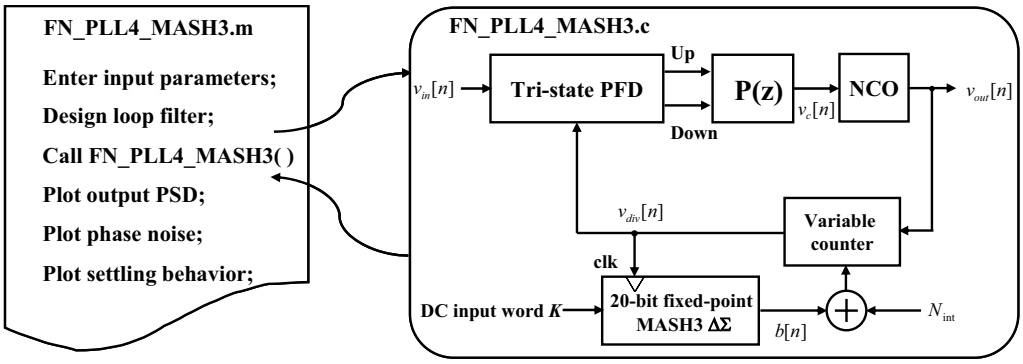


Figure 4. Simulation model of the overall Fractional-N PLL frequency synthesizer. A custom CMEX subroutine simulating a difference equation model of the PLL synthesizer is called from the main MATLAB file to achieve fast simulation speeds.

Among the PLL building blocks, the VCO and the multi-modulus divider are the easiest to model in a software environment. As shown in Figure 4, the VCO and multi-modulus divider are modeled as a Numerically Controlled Oscillator (NCO) and a variable counter, respectively.

A tri-state model of the PFD used in the simulation is shown in Figure 5. A 20-bit fixed-point model is also incorporated in the simulator for a third-order 1-1-1 MASH ΔΣ modulator (hereinafter referred to as MASH3). The difference equation model for the loop filter, however, requires a special technique due to the continuous-time nature of the filter.

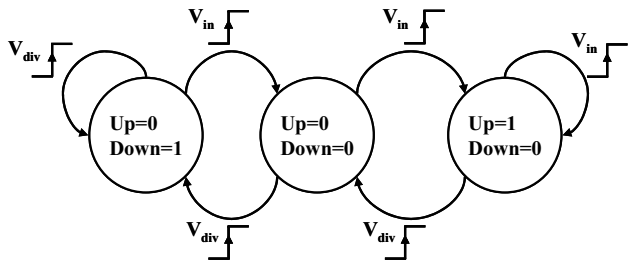


Figure 5. Tri-state model for the PFD

In difference equation simulations of PLL systems, a fixed time step is used. The time step is chosen between one-tenth and one-hundredth of the period of the output frequency to provide reasonably accurate results. Otherwise, the effect of the sampling operation would have detrimental consequences on the accuracy of the simulator. By choosing such a small simulation time step, any change in voltages and currents during a simulation time step is insignificant. Hence, the average current flowing in a branch can be accurately represented by the change in charge divided by the uniform time step [7]. The *average current-to-voltage* transfer function in (1) can thereby be converted into an equivalent discrete-time *charge difference-to-voltage* transfer function using

$$P(z) = \frac{V_c(z)}{\Delta Q(z)} = \frac{2}{T_s}\frac{1}{1-z^{-1}} F\left(\frac{2}{T_s}\frac{1-z^{-1}}{1+z^{-1}}\right), \qquad (3)$$

where T_s is the simulation time step. Substituting (1) into (3), and after algebraic manipulations,

$$P(z) = \frac{a_3 z^{-3} + a_2 z^{-2} + a_1 z^{-1} + a_0}{b_4 z^{-4} + b_3 z^{-3} + b_2 z^{-2} + b_1 z^{-1} + b_0}. \qquad (4)$$

In equation (4), the coefficients of the numerator are given as follows; $a_3 = -2D/T_s + D/\tau_1$, $a_2 = -2D/T_s + 3D/\tau_1$, $a_1 = 2D/T_s + 3D/\tau_1$, and $a_0 = 2D/T_s + D/\tau_1$. Likewise, the coefficients of the denominator are $b_4 = -A + B - C$, $b_3 = 2A - 2C$, $b_2 = -2B$, $b_1 = -2A + 2C$, and $b_0 = A + B + C$, where

$$A = \frac{\tau_2 \tau_3 4}{T_s^2},\ B = \left[\tau_2 + \left(\frac{D}{R_2} + 1\right)\tau_3\right]\frac{2}{T_s}, \text{ and } C = \frac{D\tau_3}{\tau_1 R_2} + 1. \qquad (5)$$

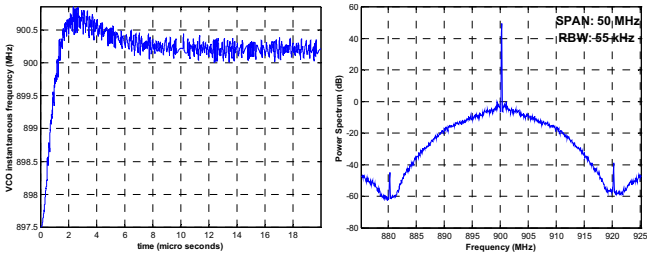
As a result, the continuous-time charge pump loop filter is transformed into an equivalent discrete-time filter. The reason for this transformation is that it is more convenient to calculate the charge difference during two consecutive time steps rather than calculating the average current [7]. At each time step, the charge difference is $\Delta q[n] = I_p T_s$ if Up=1 and Down=0, and $\Delta q[n] = -I_p T_s$ if Up=0 and Down=1, otherwise $\Delta q[n] = 0$.

3. SIMULATION RESULTS

The aforementioned simulation model is used to determine the settling behavior, the output PSD, and the phase noise of a Fractional-N synthesizer with the specifications provided in Section 2.1. The settling behavior and the output PSD of the synthesizer are shown in Figures 6(a) and (b), respectively, for an output frequency of 900.200 MHz. The simulation time step is set to 1/32 of the period of the output frequency. The simulated phase noise of the synthesizer is depicted in Figure 7.

As clearly shown in Figures 6 and 7, no fractional spurs are generated when the output frequency is set to 900.200 MHz (*i.e.*, K=(00000010100011110101)$_2$). The generation of the fractional spurs, however, is dependent on the value of the DC input word. For input words with sufficient activity at or near the Least Significant Bit (LSB) position, no fractional spurs are generated. For simple rational DC inputs, however, a significant amount of fractional spurs is generated. This phenomenon is illustrated in Figure 8(a), where the output frequency is set to 905 MHz

(K=(00100000000000000000)$_2$). In Figure 8(a), a significant amount of fractional spurs is due to the poor randomization of the quantizer error sequence in the ΔΣ modulator.



(a) **(b)**

Figure 6. Simulation of the Fractional-N PLL frequency synthesizer, (a) settling behavior, and (b) PSD

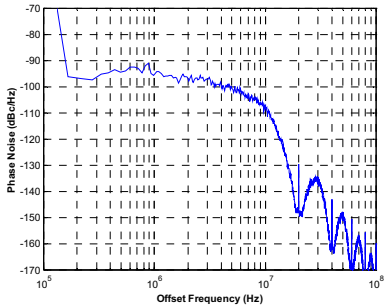
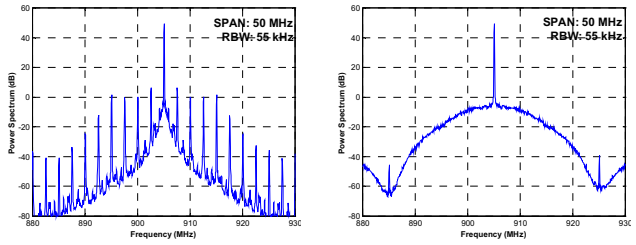


Figure 7. Phase noise of the Fractional-N PLL frequency synthesizer. Note that since all other building blocks are ideal, the only contributor to the phase noise is the ΔΣ modulator

There are two different approaches to eliminating these undesirable fractional spurs. The first method involves one LSB dithering the DC input word K. This method is effective in eliminating the fractional spurs. However, the resolution of the synthesizer is compromised because a change in the DC input directly shifts the output frequency. A one LSB dither in the DC input word shifts the output frequency by 19 Hz for a 20-bit implementation, thereby limiting the ultimate resolution of the synthesizer (usually, a resolution of 1 to 2 Hz is required).



(a) **(b)**

Figure 8. Fractional spur rejection using initial condition, (a) zero initial condition, and (b) "1" LSB initial condition

The second method involves imposing a small initial condition on the first accumulator of the ΔΣ modulator [8]. Because the long term average of the ΔΣ output is not dependent on the value of the initial condition, with this method the output frequency can be synthesized with greater accuracy. In Figure 8(b), a "1" LSB initial condition is imposed on the first accumulator, completely eliminating fractional spurs resulting from the ΔΣ modulator [8].

The execution speed of this simulator on a Pentium™ II 400 MHz laptop (with 384 Mbytes of RAM) is 14 and 8 seconds for 10 and 5.5 millions of samples, respectively, at the output of the synthesizer. The Fractional-N PLL simulator reported by Perrott in [2] obtains 5 millions samples in 80 seconds, while the simulator reported in [4] takes about a day to complete a simulation run.

4. CONCLUSIONS

A behavioral level simulation environment has been developed for Fractional-N PLL frequency synthesizers on a mixed MATLAB™ and CMEX™ platform [6]. A uniform simulation time step is allowed by appropriately modeling the continuous-time *average current-to-voltage* loop filter transfer function as a discrete-time *charge difference-to-voltage* transfer function. The simulator enables the exhaustive behavioral level simulation of Fractional-N PLL frequency synthesizers in a fast and accurate manner. The simulation results demonstrate the effectiveness of the "1" LSB initial condition imposed on the first integrator of the ΔΣ modulator in rejecting fractional spurs.

5. REFERENCES

[1] T. A. Riley, M. A. Copeland, and T. A. Kwasniewski, "Delta-Sigma Modulation in Fractional-N Frequency Synthesis," *IEEE Journal of Solid-State Circuits*, Vol. 28, No. 5, pp. 553-559, May 1993.

[2] M. H. Perrott, "Fast and Accurate Behavioral Simulation of Fractional-N Frequency Synthesizers and other PLL/DLL Circuits," *Proceedings of the IEEE/ACM Design Automation Conference*, pp. 498-503, June 2002.

[3] S. Brigati, F. Francesconi, A. Malvasi, and A. Pesucci, and M. Poletti, "Modeling of Fractional-N Division Frequency Synthesizers with Simulink and Matlab," *Proceedings of the IEEE International Conference on Electronics, Circuits, and Systems*, Vol. 2, pp. 1081-1084, September 2001.

[4] Y. Fan, "Modeling and Simulation of ΣΔ Frequency Synthesizers," *Proceedings of the IEEE Symposium on Industrial Electronics*, Vol. 1, pp. 684-689, June 2001.

[5] M. Cassia, P. Shah, and E. Bruun, "A Spur-free Fractional-N ΣΔ PLL for GSM Applications: Linear Model and Simulations," *Proceedings of the IEEE International Symposium on Circuits and Systems*, Vol. 1, pp. 1065-1068, May 2003.

[6] *MEX-Files Guide*, The Mathworks Inc., Natwick, Massachusetts, 2003.

[7] D. Johns and K. Martin, *Analog Integrated Circuit Design*. New York: John Wiley & Sons, 1997.

[8] M. Kozak and I. Kale, *Oversampled Delta-Sigma Modulators: Analysis, Applications, and Novel Topologies*. Boston: Kluwer Academic Publishers, 2003.