RFIT2007-IEEE International Workshop on Radio-Frequency Integration Technology, Dec. 9-11, 2007, Singapore

A Simulink Model for All-Digital-Phase-Locked-Loop

Xiaoyan Wang, Yeung Bun Choi, Mingkyu Je and Wooi Gan Yeoh
Integrated Circuits and Systems Laboratory
Institute of Microelectronics, Singapore
wangxy@ime.a-star.edu.sg

Abstract—**A simulink model for All-Digital-Phase-Locked-Look (ADPLL) is proposed in this paper. The study is based on ADPLL implemented in an all-digital RF transceiver. Simulation results in simulink give the performance overview of the ADPLL.**

I. INTRODUCTION

Using digital techniques to implement RF circuits has become a new trend in recent years. An all-digital transceiver has been proposed recently [1]. Although it is intended to implement all the blocks as digital circuit, certain analog circuits are not replaceable, such as oscillator. Therefore, the modeling and simulation of such kind of system becomes an important issue. A simulation platform based on behavior model and digital circuit is needed in this case.

In this paper, we focus on one of the key block in a transceiver, All-Digital-Phase-Locked-Loop (ADPLL). A VHDL model has been proposed in [1], taking advantage of event-driven process. Another ADPLL model implemented in Matlab is proposed in [2], taking advantage of the flexibility of modeling and mathematical manipulation that characterizes Matlab [2]. In this paper, we set our simulation platform in simulink, taking fully use of the build-in blocks provided by the software. Simulations are done in both frequency and time domain to validate the model proposed in this paper.

There has been such model for the traditional analog PLL [3]. However, in such case, since most of the blocks are analog circuits, the model of the PLL remains at behavior level, and gives more indication about system performance, rather than guidance for circuit design. While for ADPLL, as most of the blocks are digital circuits, simulink model of such blocks gives a direct guidance to the later real circuit design.

II. ADPLL DESCRIPTION

The ADPLL structure used to validate the simulink model is proposed in [1], and the block diagram is shown in Fig. 1. Similar to analog PLL, an ADPLL comprises three sub blocks as well, *i.e.*, phase detector, loop filter and digital-controlled-oscillator (DCO). The frequency tuning in DCO is realized by switching on and off the capacitance through digital control bits in stead of voltage in VCO. In this case, the tuning step is

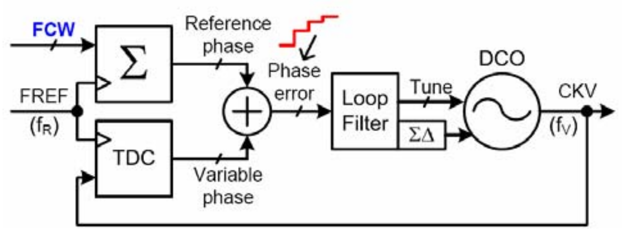


Figure 1. Block diagram of ADPLL [1].

limited by the finest capacitance value controlled by the LSB digital bit. A solution to achieve even finer tuning step is to dither the LSB digital bit, thus controlling its time-averaged value with a finer resolution [1]. The time-dithering is realized by using an Σ-Δ converter. The structure proposed in [1] works in phase domain instead of frequency domain, therefore, the phase detector is realized as a simple subtractor. The loop filter can be realized as a gain factor since there is no high frequency element in the output of the phase detector.

Referring back to Fig. 1, Frequency Control Word (FCW) is to select the channel for PLL. Time-to-Digital Converter (TDC) is used to convert the phase error into digital signal with fine resolution. Assuming FCW is an arbitrary number Y, and the reference frequency is F_{ref}. In the lock status, DCO oscillates at the frequency $Y\times F_{ref}$.

The working principle of ADPLL is described as below. In each period of the reference signal, the number of the DCO output periods is counted, and then compared with FCW. The difference therefore represents the phase error between DCO output and reference signal. This phase error is then separated into integer and fractional part. The integer part is to control switching capacitance in DCO, therefore to do the coarse tuning. The fractional part is fed to an Σ-Δ converter, and then to control the DCO for fine tuning.

III. SIMULINK MODEL

Based on the above discussion, the whole ADPLL can be simplified into two loops working concurrently, *i.e.*, the integer part of FCW, and the fractional part of FCW. The block diagram is as shown in Fig. 2, where loop A is the integer loop, and loop B is the fractional loop.

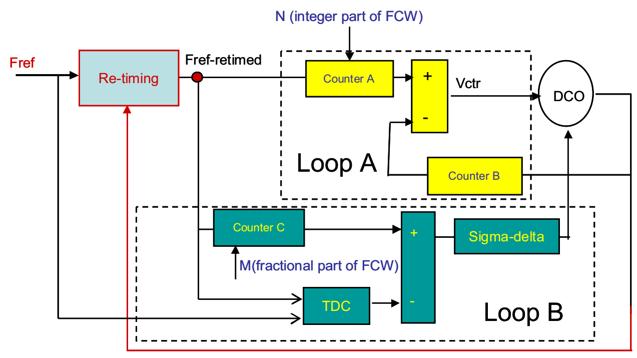


Figure 2. Block diagram of ADPLL in simulink.

Assuming first FCW is an integer number *N*, only loop A is needed. In loop A, there are two counters A and B. Counter A accumulates by *N* at each rising edge of the reference signal, and counter B accumulates by *1* at each rising edge of the DCO output signal. The output from counter A and B are compared at the rising edge of reference signal and the result is the digital control bits of DCO. A timing diagram is shown in Fig. 3. Assuming FCW is *8*, the center frequency of DCO is $6\times F_{ref}$, Kvco is *Fref/bit*, and DCO oscillates at the center frequency by default. In the first cycle, counter A gives a *8*, and counter B gives *6*. Therefore the phase error is *2*, *i.e.*, the control bit is *2*. Thus, the output frequency of DCO changes from $6\times F_{ref}$ to $8\times F_{ref}$. Since the phase detector works continuously, the output remains 2 in the following cycles, so does the output frequency remain at $8\times F_{ref}$.

Now considering the case that when FCW is a non-integer number. Referring back to Fig. 2, there is a re-timing block after the reference signal. This block is to synchronize the reference signal with the DCO output. The waveform of the reference signal, re-timed reference signal and DCO output are shown in Fig. 4. It has been derived in [1] that ε is the fractional part of the phase error between reference signal and DCO output. The ε is measured by TDC, and then compared with the fractional part of FCW. The result is a fractional number, representing the fractional part of the phase error. This number is thus converted into a train of ‘1’ and ‘0’ by an Σ-Δ converter. Loop A and B work concurrently when FCW is non-integer, to tune the DCO output at the right frequency.

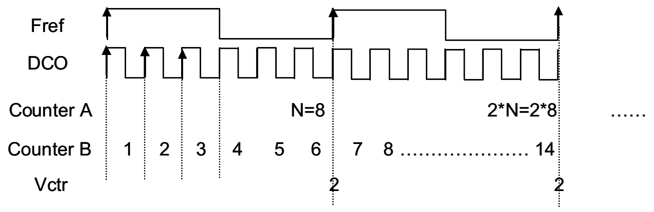


Figure 3. Waveform of counter A and B in integer phase comparator.

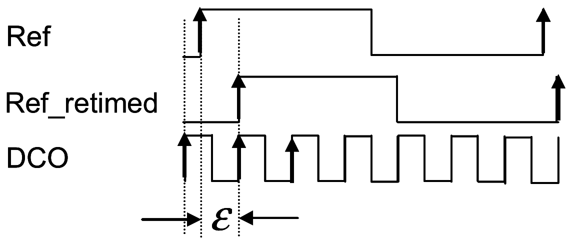


Figure 4. Waveform to demonstrate the fractional phase difference.

A. System Model

A system model is set up first, for close-loop frequency domain analysis. The model is shown in Fig. 5. The non-linear components, such as sample/hold device and Σ-Δ converter are not included in this case. The input/output of the model are phases of the reference signal and DCO output. Integrator is needed to convert from frequency to phase. The input phase is multiplied with FCW, and then compared with DCO output phase. The phase difference, going through a gain stage, controls the DCO frequency. The simulation result is shown in Fig. 6(a), which indicates the 3-dB cut-off frequency of the closed-loop ADPLL. The loop bandwidth can be adjusted by varying the gain factor, however, it compromises with phase noise performance.

The second step is to include the non-linear components as shown in Fig. 7. The blocks added are DCO control word converter and an Σ-Δ converter. In the first block, the phase error is sampled and held at a sampling time of *1/Fref*, and then separated into integer and fractional part. The integer part controls the DCO frequency directly, while the fractional part is converted first by an Σ-Δ converter, and then to control the DCO frequency. A differentiator is added to convert phase information into frequency information.

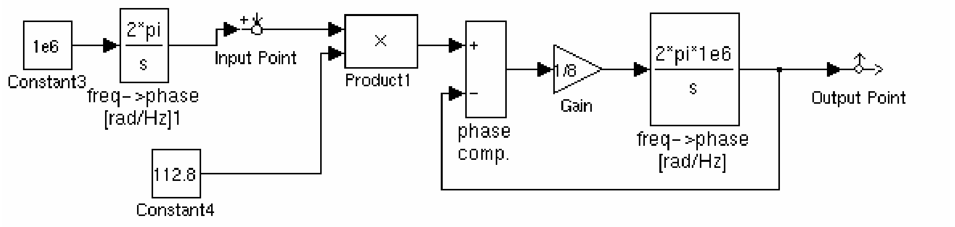
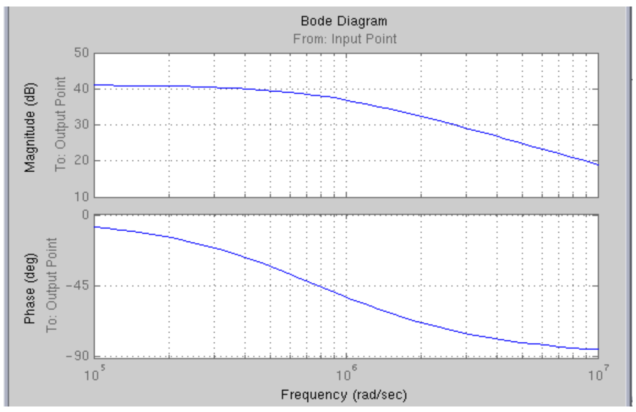
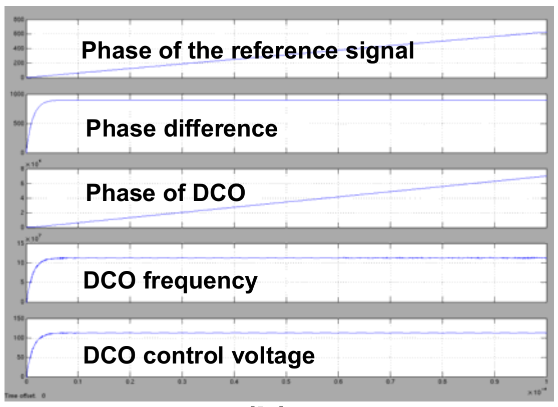


Figure 5. System model of ADPLL for LTI analysis.



(a)



(b)

Figure 6. (a) Frequency response of the model shown in Fig. 5; (b) Output waveform of the model shown in Fig. 7.

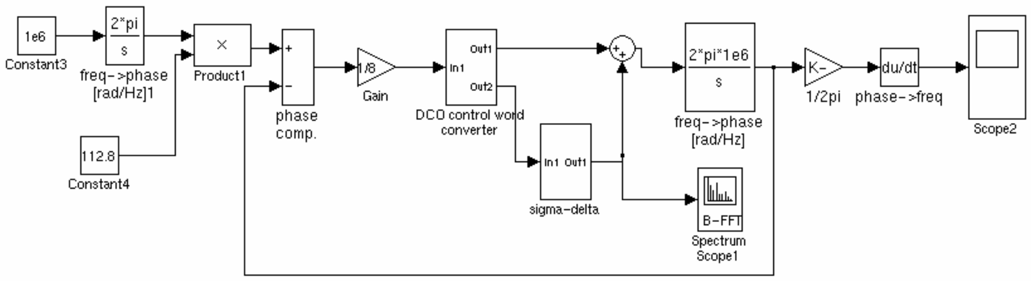


Figure 7. System model of ADPLL for time-domain analysis.

The simulation result is shown in Fig. 6(b). As can be seen, the phase of reference signal and DCO keep increasing, while the phase error settles down after *5μs*, so does the DCO frequency.

B. Time-Domain Model

In the last step, a time-domain ADPLL model is set up, as shown in Fig. 8. There are five major blocks in the model: integer phase comparator, TDC as fractional phase comparator, gain stage, Σ-Δ converter and DCO. The integer phase comparator is implemented as two counters, as introduced in section II. The TDC follows the exact circuit in [1]. The metastability faced in the real circuit is not discussed here for system analysis. The summation of the fractional and integer part of phase error passes through the gain stage, and in this case it is divided by $\alpha=8$. The output from the gain stage is separated into integer and fractional part again. The integer part goes directly to control the output frequency of DCO. The fractional part is first limited to resolution of *0.1*, and then goes to the Σ-Δ converter. The resolution limiter is necessary to keep the whole system in lock status, and is related to the TDC resolution, and thus the resolution of tuning frequency. The Σ-Δ converter implemented is of 3rd-mesh structure. The clock frequency is set to be ¼ of the DCO frequency. The DCO generates a sinusoidal signal of a certain frequency according to the control bits. The reference frequency is set at *1MHz*, FCW is set as *112.8*, TDC resolution is *100KHz*, gain factor is *1/8*, the center frequency of VCO is *100MHz* and Kvco/bit is *1MHz*. Therefore, in the lock state, the DCO should be oscillating at *112.8MHz*. The output is downconverted to baseband and the Fourier transform is done to show the spectrum.

Fig. 9 and 10 show the spectrum of the DCO output, Σ-Δ output, and the waveform of the integer control bit and fractional control bit of DCO. The settling time as shown is about *50μs*. As can be seen in Fig. 9, the output spectrum of DCO shows a similar profile as that from the Σ-Δ converter, meaning the noise is shaped by the converter. However, there are some harmonics shown in both spectrums, which can also be improved by adopting or improving the converter structure. Fig. 10 shows the output waveform of the integer control bit, fractional control bit before and after the *1/10* resolution limiter. As can be seen, the integer part settles at *12* in the end. Afterwards, the fractional part settles at *0.8*.

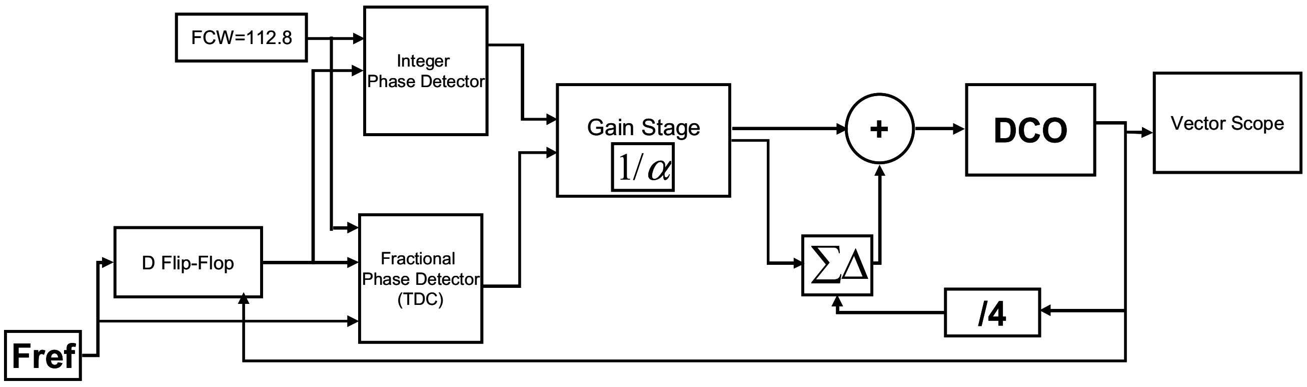


Figure 8. Time-domain model of ADPLL.

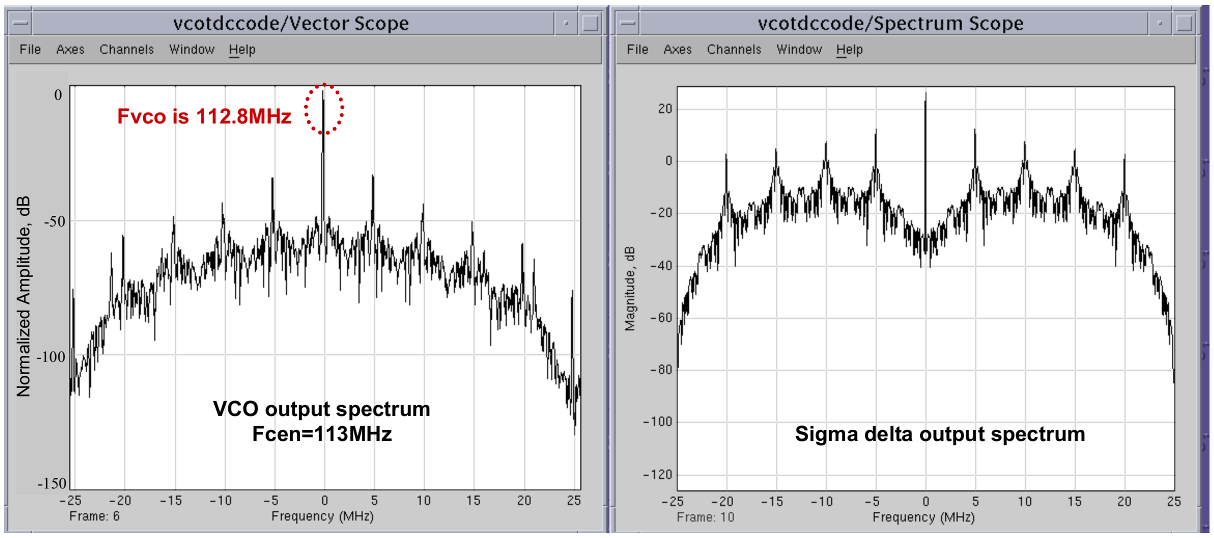


Figure 9. Output spectrum of DCO and Σ-Δ converter in ADPLL model in Fig. 8.

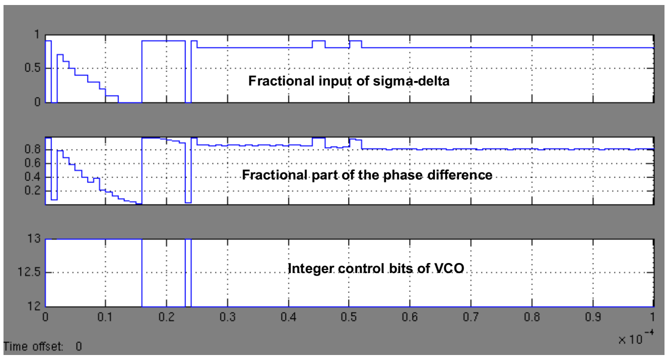


Figure 10. Output waveform of ADPLL model in Fig. 8.

IV. CONCLUSION

This paper introduced the modeling of ADPLL in simulink, including one model at system level for LTI analysis, and one at block level for time-domain analysis. Simulations are done in both cases, to observe the setting time and phase-lock process.

REFERENCES

[1] R. B. Staszewski and P. T. Balsara, “All-Digital Frequency Synthesizer in Deep-Submicron CMOS” Wiley-Interscience

[2] J. Zhuang, Q. Du and T. Kwasniewski., “Event-Driven Modeling and Simulation of an Digital PLL,” Proceeding of 2006 IEEE International Behavioral Modelling and Simulation Workship, Sep 2006, pp 67-72.

[3] M. H. Perrott, M. D. Trott and C. G. Sodini., “A Modeling Approach for Σ-Δ Fractional-N Frequency Synthesizers Allowing Straightfoward Noise Analysis” IEEE JSSC, vol 37, Aug. 2002, pp. 1028–1038.