NTNU – Trondheim
Norwegian University of
Science and Technology

Python Framework for Design and Simulation of Integer-N ADPLLs

Cole Nielsen

Electronic Systems Design, Specialization Project

Submission date:	December 2019
Supervisor:	Trond Ytterdal, IET
Co-supervisor:	Carsten Wulff, IET

Norwegian University of Science and Technology
Department of Electronic Systems

Abstract.

A pure-Python simulation and design automation framework for the design of integer-N all digital phase locked loop (ADPLL) frequency synthesizers is presented in this paper. The framework enables (a) automatic design of optimized discrete-time and quantized digital PLL loop filters, and (b) discrete-time simulation to verify filter and PLL designs for phase noise performance, lock-time and stability subject to variation using Monte-Carlo sampling. Simulation is performed with a discrete-event time domain simulator utilizing behavioral PLL component models, which permit accurate modeling effects of time-discretization and quantization in a ADPLL. Filter design automation is achieved via optimization of a prototype loop filter in a PLL for minimum total phase noise power and lock time, where lock time is maximally constrained. The optimizer utilizes continuous phase transfer function approximations of discrete PLL loop, allowing computationally fast estimates of phase noise and lock time. Thus continuous-to-discrete time conversion and coefficient quantization of the optimal filter is applied to yield a end loop-filter design. Second order optimization is utilized to minimize loop filter design error due to quantization effects.

Problem description.

To develop a standalone PLL simulation framework to aid and facilitate a later master's thesis project regarding the design of an all-digital, ultra-low power PLL (ULPPLL) frequency synthesizer. This framework is intended to address and ease all-digital PLL design and simulation challenges at a high level. Specifically it should allow for speedy PLL simulation defined with system and component- level specifications (e.g. desired frequency, gain of digitally contolled oscillator, phase detector resolution, divider modulus). This is to allow for development of component level specifications and validation of ideal PLL performance (phase noise, lock time, stability) before transistor level implementation. Furthermore, the framework should automate design of any pure-digital PLL components, namely loop-filter design, to accelerate overall time of PLL implementation.

Contents

1 Introduction 10

2 Theory 11
- 2.1 Continuous PLL Model . . . 11
  - 2.1.1 PLL Synthesizer architecture . . . 12
  - 2.1.2 Divider . . . 12
  - 2.1.3 Phase detector . . . 12
  - 2.1.4 Loop Filter . . . 12
  - 2.1.5 VCO . . . 13
  - 2.1.6 Continuous PLL Transfer function . . . 13
  - 2.1.7 PI-loop filter design . . . 14
  - 2.1.8 PI-controller peaking compensation . . . 15
  - 2.1.9 Alternative PID controller permutations . . . 16
- 2.2 ADPLL - digital, discretized PLL Model . . . 16
  - 2.2.1 Divider . . . 17
  - 2.2.2 TDC . . . 17
  - 2.2.3 Loop Filter . . . 18
  - 2.2.4 DCO . . . 19
  - 2.2.5 Discrete-time PLL transfer function . . . 20
- 2.3 ADPLL Noise Model . . . 21
  - 2.3.1 TDC noise . . . 21
  - 2.3.2 DCO noise . . . 22
  - 2.3.3 Divider noise . . . 23
  - 2.3.4 Loop filter noise - direct type I . . . 24
  - 2.3.5 PLL noise sensitivity transfer functions . . . 25
  - 2.3.6 PLL phase noise and output PSD relationship . . . 26
  - 2.3.7 PLL output-referred noise PSD . . . 26

3 Methods 27
- 3.1 Behavioral, discrete event PLL simulation . . . 27
  - 3.1.1 Phase noise modeling . . . 27
  - 3.1.2 Measurement of phase noise, spurs . . . 27
  - 3.1.3 Monte-carlo sampling . . . 27
- 3.2 Loop filter optimization . . . 27
  - 3.2.1 Estimation of settling time . . . 28
  - 3.2.2 Estimation of PLL phase noise . . . 28
  - 3.2.3 Optimization algorithm . . . 29

4 Discussion 29
- 4.1 Divider noise constraint . . . 30
- 4.2 Example exercise . . . 30

5 Conclusion 31

6 Baseband phase noise in radio with PLL 31
- 6.1 Modulated Signal . . . 31
- 6.2 Local oscillator - noisy PLL . . . 31

6.3 Baseband phase noise 31

A Appendix - Schedule **34**

B Appendix - Code **34**

C DCO tuning **35**

C.1 Backgate tuning 35

C.2 Ring oscillator frequency derivation 35

C.2.1 Finding $\langle g_{ch} \rangle$ and C 36

C.2.2 Handling unequal NMOS/PMOS 37

C.2.3 Solving for oscillator frequency and power 37

C.3 Ring oscillator backgate tuning derivation 38

C.4 DCO Gain Uncertainty 39

List of Figures

1 Basic phase feedback network. 11
2 Basic PLL. 12
3 PI-controller PLL pole-zero locations. 14
4 Example PI-PLL responses with varied ζ. 15
5 Basic ADPLL. 16
6 Digital divider signals. 17
7 TDC model. 18
8 Implementation of filter. 19
9 Discrete time PLL model. 20
10 TDC quantization noise models. 21
11 Quantization as via additive error signal. 21
12 DCO additive noise model. 22
13 Voltage to phase noise conversion. 22
14 Divider phase noise. 23
15 Direct type I filter implementation with quantization. 24
16 Full PLL additive noise model. 25
17 Model for ring oscillator. 35
18 Approximate model for ring oscillator inverter delay cell. 35
19 Backgate-tuned ring oscillator with coarse tuning capacitor bank. 38

List of Tables

1 System-level specifications 30
2 Component-level specifications. 30

Abbreviations.

BBPD	Bang-bang phase detector
BW	Bandwidth
CMOS	Complementaty metal oxide semiconductor
DAC	Digital to analog converter
DCO	Digitally controlled oscillator
DFT	Discrete Fourier transform
DSP	Digital signal processing
FDSOI	Fully depleted silicon on insulator
FET	Field effect transistor
FFT	Fast Fourier transform
FM	Frequency modulation
FRAC	Fractional
FSK	Frequency shift keying
IF	Intermediate frequency
INT	Integer
LF	Loop filter
LO	Local oscillator
LSB	Least significant bit
MOSFET	Metal oxide semiconductor field effect transistor
NMOS	N-channel MOSFET
OTW	Oscillator tuning word
PD	Phase detector
PLL	Phase locked loop
PMOS	P-channel MOSFET
PN	Phase noise
PSD	Power spectral density
PSK	Phase shift keying
RC	Resistor-capacitor
RF	Radio frequency
RO	Ring oscillator
RX	Receiver
SNR	Signal to noise ratio
SSB	Single side band
TDC	Time to digital converter
TSPC	True single phase circuit
ULPPLL	Ultra-low power PLL
VCO	Voltage controlled oscillator
WURX	Wake up receiver

1 Introduction

Phase locked loops are extraordinarily useful frequency synthesizers that are vital to the operation of virtually all wired and wireless communication systems of today. The trend towards increasingly lower power wireless devices poses an acute need to reduce PLL power consumption. This is a challenge as PLLs typically rank among the highest power consuming components of a radio, and are necessarily so to limit phase noise. Reducing analog PLL power consumption can be a prohibitive challenge as the performance of analog loop filters degrade as a result of unavoidably lower charge pump current. However, recent CMOS process nodes with minimum gate lengths as small as 7nm allow for all-digital PLLs as an attractive alternative to analog designs due to low power consumption associated with the implementation of a digital loop filter. Digital loop-filters have a unique advantage where they can be scaled indefinitely as process nodes advance, while suffering no loss in performance.

All-digital PLLs introduce new challenges in the process of design in the ultra-low power domain. Low power design is complemented by low complexity design, which in a PLL transcribes to low resolution of phase detectors, digitally tunable oscillators, and loop filter digital data paths. In other words, effects of quantization are strong. Whereas analog designs and high power/resolution digital designs can be effectively analyzed (even by hand) with transfer functions in the Z-domain, time domain simulation is necessary to capture quantization effects in low power, low resolution ADPLLs. This, of course, presents a challenge in manual design of PLLs if simulation is an integral part to the most basic modeling, and thus motivates the creation of an automated design solution. Currently, no PLL design framework is known that automates PLL design for the needs of ultra low power PLLs as characterized here. Of the most prominent pre-existing frameworks, CppSim [add reference...] features a utility for design of continuous loop filters, however for this it does not provide any level of performance optimization, nor does it support discrete-time and quantized digital designs.

Thus in this paper, a new framework written in pure-Python is introduced, which uniquely addresses issues of ultra-low power ADPLL design. Specifically, design of integer-N type PLLs is focused on, as the impetus of this work is an integer-N PLL design project. Topics presented are (a) automatic design and optimization of ADPLL loop filters given target system and component level specs for the PLL, and (b) behavioral time domain PLL simulation for accurate analysis and verification of PLL performance, with an integrated Monte-Carlo sampling variation analysis engine.

A brief outline of the paper is as follows. An introduction to PLL theory is in section 2. Simulation and optimization methods are discussed in section 3. An example design exercise, a comparison to existing solutions, and general discussion considerations for using the framework are in section 4. Finally, section 5 concludes.

include main contributions

2 Theory

In its most basic form, a phase locked loop is a phase-based feedback system whose output tracks or maintains a fixed phase relationship to an input signal. Such a system is uniquely suited to the task of frequency synthesis, which is the process of generating derivative frequencies from some reference frequency. Given reference and output phase signals Φ_{ref} and Φ_{out}, a PLL can be modeled as in figure 1, with feedforward and feedback networks A(s) and B(s).

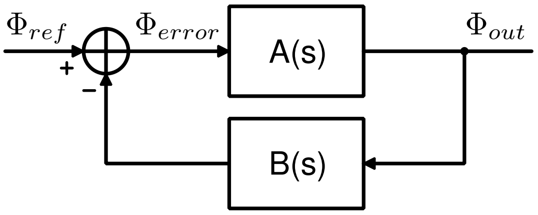


Figure 1: Basic phase feedback network.

The closed loop phase response T(s) for Φ_{ref} to Φ_{out} is therefore:

$$\mathrm{T}(s) = \frac{\Phi_{out}(s)}{\Phi_{ref}(s)} = \frac{A(s)}{1 + A(s)B(s)} \tag{1}$$

A particular case of interest is when B(s) = 1/N, where N is a constant, and the loop gain A(s)B(s) $>>$ 1, the closed loop response is:

$$\frac{\Phi_{out}(s)}{\Phi_{ref}(s)} \approx \frac{A(s)}{A(s)B(s)} = \frac{1}{B(s)} = N \tag{2}$$

We see that the phase through the PLL is multiplied by a factor of N. If the input phase signal is a sinusoid with frequency ω_{ref}, and likewise the output with ω_{out}, then $\phi_{ref}(t) = \omega_{ref} t$ and $\phi_{out}(t) = \omega_{out} t$. Thus:

$$\frac{\Phi_{out}}{\Phi_{ref}} = \frac{\omega_{out} t}{\omega_{ref} t} \approx N \quad \rightarrow \quad \omega_{out} \approx N\omega_{ref} \tag{3}$$

Therefore, it is observed that a PLL allows for the generation, i.e. synthesis, of a new frequency from a reference frequency signal. Given a feedback divider ration of 1/N, the PLL multiplies the reference frequency by a factor of N. In the following sections, more advanced models for PLL will be developed, extending the concept introduced here. Specifically, the theory of digital, discrete-time PLLs will be developed and extended from a continuous phase model of a basic PLL.

2.1 Continuous PLL Model

Although PLLs are practically limited to using discrete-time sampling in real-word hardware, continuous models can still be applied in their analysis and design. Thus a continuous PLL model is developed in this section to aid in the later discussed discretized PLL modeling.

2.1.1 PLL Synthesizer architecture

The traditional architecture for implementing a PLL frequency synthesizer [3] is shown in figure 2. This basic PLL is comprised of four components: (1) a phase detector, denoted by PD, (2) a loop filter, denoted by H_{LF}(s), (3) a voltage controlled oscillator, denoted by VCO, and (4) and phase divider, denoted by "÷ N" in the figure. These components are explained in the following sections.

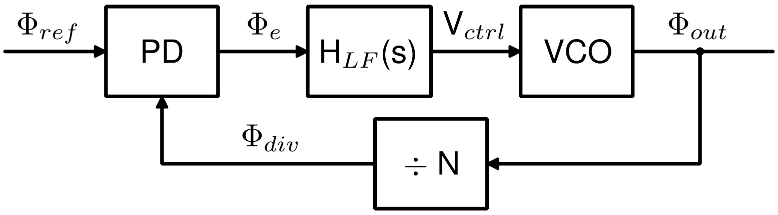


Figure 2: Basic PLL.

2.1.2 Divider

The phase divider is used as the feedback path in the PLL, where the division modulus N controls the frequency multiplication of the PLL. The tranfer function of the divider is:

$$\mathrm{H}_{div}(s) = \frac{\Phi_{div}(s)}{\Phi_{out}(s)} = \frac{1}{\mathrm{N}} \tag{4}$$

2.1.3 Phase detector

The phase detector is used to measure the phase of the feedback signal (i.e. divider output) in relation to the reference phase, in order to establish a phase error signal Φ_e used to control the tuning of the PLL.

$$\Phi_e(s) = \Phi_{ref}(s) - \Phi_{div}(s) \tag{5}$$

2.1.4 Loop Filter

The PLL loop filter is used to control the phase-frequency response of PLL, which affects transient PLL behavior, as well as phase noise performance (see section 2.3). As will later be seen, low pass response is desired in a PLL, so the loop filter must be designed accordingly. Generally, this can be designed arbitrarily to have P poles and Z zeros, as such:

$$\mathrm{H}_{LF}(s) = \frac{\Sigma_0^Z b_j s^j}{\Sigma_0^P a_k s^k} \tag{6}$$

Rational choice of the loop filter will ensure that the PLL will be stable and minimize steady state phase error. In order to achieve zero steady state error, the loop filter must contain a pole at zero, in other words an integrator. A PLL containing such a pole is classified as a Type II

PLL, and a PLL omitting the pole is considered Type I. A logical approach to loop filter design for zero-phase error is to treat it as a PID controller, where:

$$\mathrm{H}_{LF}(s) = sK_d + K_p + \frac{K_i}{s} = \frac{K_d}{s}\left(s^2 + s\frac{K_p}{K_d} + \frac{K_i}{K_d}\right) \tag{7}$$

Such a loop filter contains two zeros and one integrator pole at zero. The gain parameters K_p, K_i, K_d can be tuned to achieve the desired bandwidth and stability for the PLL. The impacts of loop filter design will be further considered in sections 2.1.6-2.1.9.

2.1.5 VCO

The voltage controlled oscillator is an oscillator with frequency controlled by an input signal V_{ctrl}. The VCO is characterized by its gain $K_{DCO} = \partial f/\partial \mathrm{V}_{ctrl}$, and the nominal oscillation frequency f_0. Analyzed in terms of phase, an oscillator can be seen as a time-phase integrator:

$$\Phi_{VCO}(t) = \Phi_{out}(t) = \int 2\pi(K_{DCO}\mathrm{V}_{ctrl}(t) + f_0)\mathrm{dt} \tag{8}$$

In s-domain, where frequency offsets will be represented via initial conditions for modeling purposes, the VCO transfer function is therefore:

$$\mathrm{H}_{VCO}(s) = \frac{\Phi_{VCO}(s)}{\mathrm{V}_{ctrl}(s)} = \frac{\Phi_{out}(s)}{\mathrm{V}_{ctrl}(s)} = \frac{2\pi K_{DCO}}{s} \tag{9}$$

2.1.6 Continuous PLL Transfer function

Now that the continuous PLL synthesizer is understood at a component level, the closed loop dynamics of the PLL can be analyzed. First the PLL loop gain is determined:

$$\mathrm{L}(s) = \mathrm{H}_{LF}(s)\mathrm{H}_{VCO}(s)\mathrm{H}_{div}(s) = \frac{2\pi K_{VCO}K_d}{\mathrm{N}}\frac{1}{s^2}\left(s^2 + s\frac{K_p}{K_d} + \frac{K_i}{K_d}\right) \tag{10}$$

With the phase detector as the feedback summation point, the closed loop response of the PLL from reference to output is:

$$\mathrm{T}(s) = \frac{\Phi_{out}(s)}{\Phi_{ref}(s)} = \frac{2\pi K_{VCO}\left(s^2K_d + sK_p + K_i\right)}{s^2\left(1 + \frac{2\pi K_{VCO}K_d}{\mathrm{N}}\right) + \frac{2\pi K_{VCO}}{\mathrm{N}}\left(sK_p + K_i\right)} = \mathrm{N}\frac{\mathrm{L}(s)}{1 + \mathrm{L}(s)} \tag{11}$$

It should be noted that in the closed loop configuration, this PLL phase transfer function contains two poles and two zeros. This is not a low pass response as desired for a satisfactory PLL phase noise power spectrum, as will later be discussed. In order achieve low pass operation, the derivative term K_d must be set to zero, yielding a PI controller for the loop filter (with one zero and two poles):

$$\mathrm{T}(s) = \frac{\Phi_{out}(s)}{\Phi_{ref}(s)} = \frac{2\pi K_{VCO}\left(sK_p + K_i\right)}{s^2 + \frac{2\pi K_{VCO}}{\mathrm{N}}\left(sK_p + K_i\right)} = \mathrm{N}\frac{\mathrm{L}(s)}{1 + \mathrm{L}(s)} \tag{12}$$

Steady state zero phase error can be verified by solving the closed loop $\Phi_e(s)$ for s=0:

$$\Phi_e(s)|_{s=0} = \left(\Phi_{ref}(0) - \frac{\Phi_{out}(0)}{\mathrm{N}}\right) = \Phi_{ref}(0)\left(1 - \frac{\Phi_{out}(0)}{\mathrm{N}\Phi_{ref}(0)}\right) = \Phi_{ref}(0)\left(1 - \frac{\mathrm{N}}{\mathrm{N}}\right) = 0 \tag{13}$$

2.1.7 PI-loop filter design

Given a PI-controller loop filter, which can be optionally represented using a pole at zero and a zero with $\omega_z = K_i/K_p$:

$$\mathrm{H}_{LF}(s) = K_p + \frac{K_i}{s} = \frac{K_i}{s}\left(\frac{s}{\omega_z} + 1\right) \tag{14}$$

Selection of (not-necessarily optimal) PI controller gains can be easily derived from overall PLL settling time requirements. Suppose that settling time t_s is defined such that the PLL settles within $\pm\delta$ of the final value for a step input. If the initial and final PLL output frequencies are f_i and Nf_{ref}, and settling with $\pm f_{tol}$ is desired, $\delta = f_{tol}/|f_i - Nf_{ref}|$. Setting time is therefore:

$$t_s = -\tau \ln(\delta) \tag{15}$$

Thus, to find settling time, a value for the PLL time constant τ must be derived. Rewriting equation 12 with substitutions $\omega_z = K_i/K_p$ and $\mathrm{K} = 2\pi K_{VCO}K_i/\mathrm{N}$:

$$\frac{\Phi_{out}(s)}{\Phi_{ref}(s)} = \mathrm{N} \cdot \frac{s\frac{K}{\omega_z} + K}{s^2 + s\frac{K}{\omega_z} + K} \tag{16}$$

If the second order denominator can be redefined in terms of a natural frequency ω_n and damping ζ, such that:

$$s^2 + s\frac{K}{\omega_z} + K = s^2 + s2\zeta\omega_n + \omega_n^2 \tag{17}$$

It is then found that $\omega_n = \sqrt{K}$, and $\omega_z = \sqrt{K}/2\zeta$. The poles of equation 16 are then located at s = $\zeta\sqrt{K} \pm \sqrt{K}\sqrt{1-\zeta^2}$. The settling time of the PLL will be determined by the real portion of dominant pole of equation 16, specifically $\tau = 1/|\min(\Re(\{s_{p1}, s_{p2}\}))|$. Based on the pole-zero plot of figure 3, it can be observed that the dominant pole location is maximized with $\zeta = 1$. The pole-zero loci orientations are based on increasing ζ values. According to Razavi [2], ζ is usually "chosen to be $> \sqrt{2}/2$ or even 1 to avoid excessive ringing."

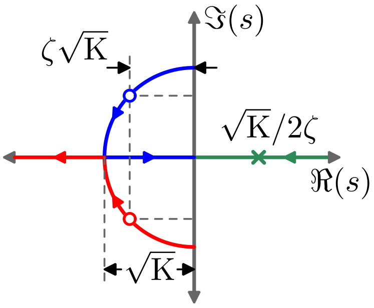


Figure 3: PI-controller PLL pole-zero locations.

To illustrate the effect of the damping coefficient ζ, figure 4 illustrates the example frequency and step responses of a PI-controlled PLL with N=1. Notice excessive peaking and ringing for $\zeta < \sqrt{2}/2$. The peaking observed in the frequency response is unavoidable with the PI-PLL due to the inherent zero in the transfer function. Its effect can be reduced with large ζ, however this will increase PLL settling time.

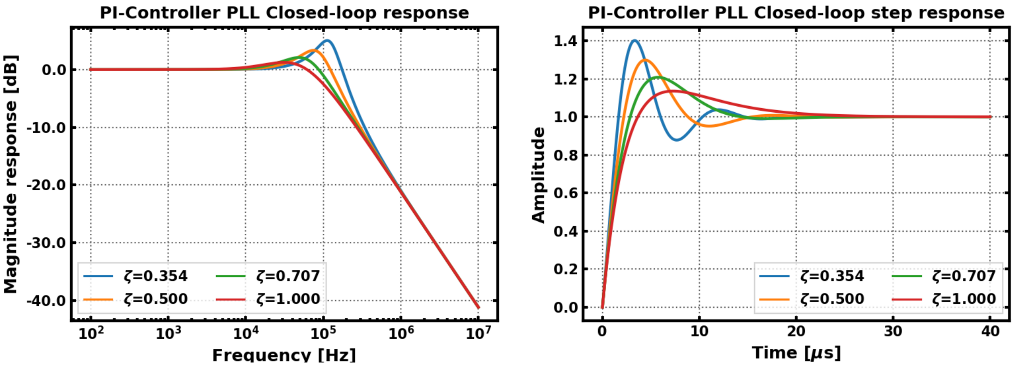


Figure 4: Example PI-PLL responses with varied ζ.

If ζ is constrained to ≤ 1:

$$\tau = \frac{1}{|\min(\Re(\{s_{p1}, s_{p2}\}))|} = \frac{1}{\zeta\sqrt{K}} \tag{18}$$

Thus:

$$t_s = \frac{-\ln(\delta)}{\zeta\sqrt{K}} = \frac{-\ln\left(\frac{f_{tol}}{|f_i - Nf_{ref}|}\right)}{\zeta\sqrt{K}} \tag{19}$$

Based on specification for settling time and damping ζ, the values for K and ω_z can be determined. If K_{VCO} and N are also specified, the PI gain coefficients can be solved additionally.

$$\omega_z = \frac{-\ln(\delta)}{2t_s} = \frac{-\ln\left(\frac{f_{tol}}{|f_i - Nf_{ref}|}\right)}{2t_s} \tag{20}$$

$$K = \frac{\ln^2(\delta)}{\zeta^2 t_s^2} = \frac{\ln^2\left(\frac{f_{tol}}{|f_i - Nf_{ref}|}\right)}{\zeta^2 t_s^2} \tag{21}$$

$$K_i = \frac{\text{NK}}{2\pi K_{VCO}} \tag{22}$$

$$K_p = \frac{K_i}{\omega_z} \tag{23}$$

2.1.8 PI-controller peaking compensation

To compensate for closed loop peaking, the original PI-controller loop filter of equation 14 can be modified with the addition of a single tunable pole at ω_p. The closed loop response becomes third order, which complicates direct analysis and design of the loop filter. However, utilizing the automated filter optimization approach described later in this paper resolved issues regarding filter design in this case.

$$\text{H}_{LF}(s) = \frac{K_i}{s}\frac{\left(\frac{s}{\omega_z}+1\right)}{\left(\frac{s}{\omega_p}+1\right)} \tag{24}$$

2.1.9 Alternative PID controller permutations

If individual terms within the PID-controller are dropped, different controller permutations (PD, ID, PI, P, I, D) can be achieved. As mentioned before, inclusion of an integral term is needed to ensure the desired zero steady state error for a PLL. This leaves ID and I-controllers as possible alternative solutions to PI for the loop filter. However, it can be easily found that neither of these controllers result in a stable PLL, which leaves a PI-controller as the only viable PID implementation.

I-only controller

Setting the K_p and K_d terms of equation 11 to zero yields:

$$\frac{\Phi_{out}(s)}{\Phi_{ref}(s)} = \frac{2\pi K_{VCO} K_i}{s^2 + \frac{2\pi K_{VCO}}{\mathrm{N}} K_i} \tag{25}$$

This closed loop transfer function results in a pair of poles at $\pm\sqrt{2\pi K_{VCO} K_i / \mathrm{N}}$. This will never be stable, as it can only be manifested as (1) a pair of poles on the imaginary axis, which is an oscillator, or (2) a real pole in the right-half plane and a real pole in the left-half plane, the former of which is not causally stable.

ID-controller

Setting the K_p term of equation 11 to zero yields:

$$\frac{\Phi_{out}(s)}{\Phi_{ref}(s)} = \frac{2\pi K_{VCO}\left(s^2 K_d + K_i\right)}{s^2\left(1 + \frac{2\pi K_{VCO} K_d}{\mathrm{N}}\right) + \frac{2\pi K_{VCO}}{\mathrm{N}} K_i} \tag{26}$$

The poles of this transfer have the same form as the I-only controller, and this PLL-controller configuration is unstable for the same reasons as the I-only controller PLL.

2.2 ADPLL - digital, discretized PLL Model

Based on the continuous PLL theory, a model for digital, discretized PLLs (i.e. ADPLLs) can be adapted. The general approach here is to utilize the bilinear transformation between continuous s-domain models to the discrete z-domain models. As commonly cited in PLL literature from a seminal paper by Gardner [1], if the sampling frequency $f_s > 10 \cdot \mathrm{BW}_{loop}$, where BW_{loop} is the PLL loop bandwidth, the effects of time sampling are easily ignored for purposes of analysis. Thus the design methods established in this paper are predicated on $f_s > 10 \cdot \mathrm{BW}_{loop}$.

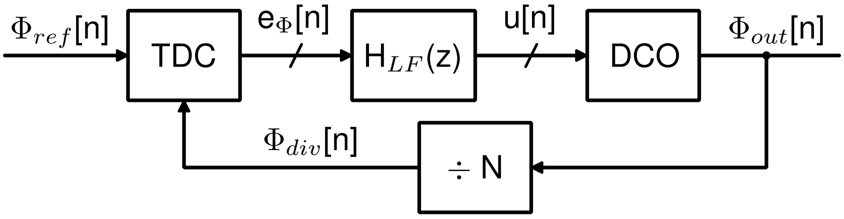


Figure 5: Basic ADPLL.

The basic architecture of an ADPLL is shown in figure 5. Here, compared to the continuous PLL, the phase detector has been replaced with a time to digital converter (TDC), the loop filter $\mathrm{H}_{LF}(s)$ with a discrete loop filter $\mathrm{H}_{LF}(z)$, and the VCO with a digitally controlled oscillator (DCO). In this architecture, the TDC, loop filter, and DCO are digital.

2.2.1 Divider

A digital divider functions by counting input cycles. With a divider modulus N, the output of the divider will have an active edge transition (considered to be rising edge as shown here) every N-cycles. Phase information is inferred from active edge timing, which occurs with time interval N/f_{osc}, and is equal to the point at which output phase equals a multiple of 2π. Thus a digital divider does not provider continuous phase information, but rather a sampled phase signal with rate f_{osc}/N.

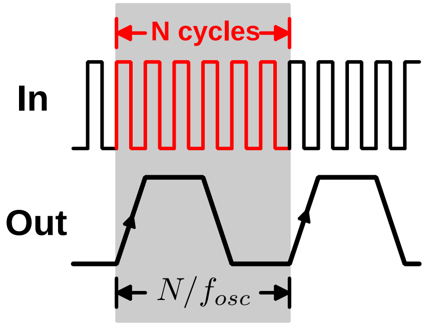


Figure 6: Digital divider signals.

For PLL transfer function modeling, a digital divider behaves identically to the continuous case:

$$\Phi_{div}[n] = \frac{\Phi_{out}[n]}{\mathrm{N}} \tag{27}$$

Application of the z- and s-domain transformations:

$$\mathrm{H}_{div}(z) = \mathrm{H}_{div}(s) = \frac{\Phi_{div}(z)}{\Phi_{out}(z)} = \frac{1}{\mathrm{N}} \tag{28}$$

2.2.2 TDC

The TDC is a digital, quantized representation of the the phase detector. It takes input phase signals Φ_{div}[n] and Φ_{ref}[n], and outputs a digital phase error word $e_{\Phi}[n]$. Figure 7 shows the basic TDC model architecture. Being digitized, a TDC will have limited resolution in phase, equivalent to M steps per reference cycle. This is a minimum step size in time of Δt_{step} = $1/Mf_{ref}$. Since the output of the TDC is digital, the model applies a scale factor $\mathrm{M}/2\pi$ and floor rounding, so 1 least significant bit (LSB) of $e_{\Phi}[n]$ equates to Δt_{step} timing error between Φ_{div}[n] and Φ_{ref}[n].

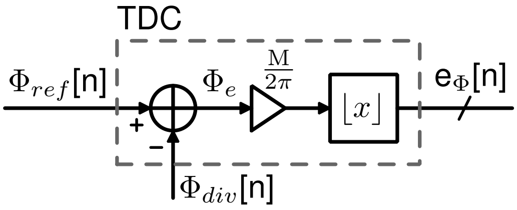


Figure 7: TDC model.

In sampled-time equation form:

$$e_\Phi[n] = \left\lfloor \frac{\mathrm{M}}{2\pi}(\Phi_{ref}[n] - \Phi_{div}[n]) \right\rfloor \tag{29}$$

For purposes of PLL transfer function calculation, the TDC z- and s-domain representation is equation 31, which accounts for phase-to-digital domain conversion gain. Effects of quantization will be handled in section 2.3.

$$e_\Phi(z) = \frac{\mathrm{M}}{2\pi}(\Phi_{ref}(z) - \Phi_{div}(z)) \tag{30}$$

$$\mathrm{H}_{TDC}(z) = \mathrm{H}_{TDC}(s) = \frac{\mathrm{M}}{2\pi} \tag{31}$$

2.2.3 Loop Filter

The loop filter design will be derived from the continuous PI-controller loop filter with optional peaking compensation (equation 24) via application of the bilinear transform. The bilinear transform specifically allows for the conversion of a continuous transfer function to discrete representation, and vice versa. This, however is conditioned on satisfaction of Nyquist sampling criteria, and in the case of PLLs it is recommended that $f_s > 10 \cdot \mathrm{BW}_{loop}$ to ensure transformation accuracy [1]. A high level of oversampling allows for the following definition of the bilinear transform, where $1/\Delta T_s$=f_{ref}:

$$\begin{aligned} z^{-1} &= e^{-s\Delta T_s} && \text{(definition of z-space)} \\ &= \sum_{k=0}^{\infty} \frac{(-s\Delta T_s)^k}{k!} && \text{(exponential Taylor series)} \\ &\approx 1 - s\Delta T_s && (\text{if } |s\Delta T_s| = 2\pi \mathrm{BW}_{loop} \cdot \Delta T_s << 1) \end{aligned}$$

Thus the bilinear transform identities are:

$$z^{-1} = 1 - s\Delta T_s \tag{32}$$

$$s = \frac{1}{\Delta T_s}(1 - z^{-1}) \tag{33}$$

Applying 33 to equation 24 yields the z-domain loop filter:

$$\mathrm{H}_{LF}(z) = \mathrm{H}_{LF}(s)\big|_{s=\frac{1}{\Delta T_s}(1-z^{-1})} = \left.\frac{K_i}{s}\frac{\left(\frac{s}{\omega_z}+1\right)}{\left(\frac{s}{\omega_p}+1\right)}\right|_{s=\frac{1}{\Delta T_s}(1-z^{-1})} \tag{34}$$

$$= k_i \Delta T_s \frac{\omega_p}{\omega_z}\frac{(1+\omega_z \Delta T_s)-z^{-1}}{(1+\omega_p \Delta T_s)-z^{-1}(2+\omega_p \Delta T_s)+z^{-2}} \tag{35}$$

Equation 35 is transformed to a digitally implementable representation by converting into the canonical representation of 36, which then determines the tap coefficients for the sampled-time difference equation 37.

$$\mathrm{H}_{LF}(z) = \frac{\sum_{j=0}^{M} b_j z^{-j}}{1+\sum_{k=1}^{N} a_k z^{-k}} \tag{36}$$

$$y[n] = -\sum_{k=1}^{N} a_k y[n-k] + \sum_{j=0}^{M} b_j x[n-j] \tag{37}$$

The transformed 35 is directly implementable in digital hardware with a direct type 1 IIR filter shown in figure 8, with the filter coefficients given by equations 38-41. The filter coefficients must be quantized into finite resolution fixed point words for a complete digital implementation. Effects of quantization will be discussed in section 2.3.

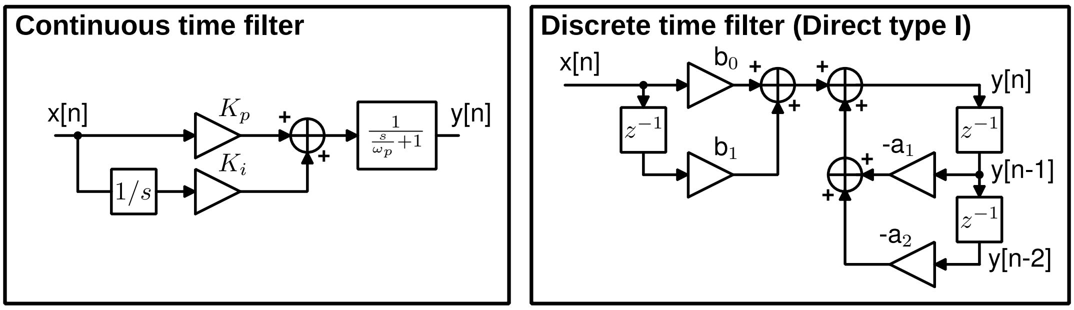


Figure 8: Implementation of filter.

$$a_1 = -\frac{2+\omega_p \Delta T_s}{1+\omega_p \Delta T_s} \tag{38}$$

$$a_2 = \frac{1}{1+\omega_p \Delta T_s} \tag{39}$$

$$b_0 = \frac{K_i \omega_p \Delta T_s}{\omega_z}\frac{1+\omega_z \Delta T_s}{1+\omega_p \Delta T_s} \tag{40}$$

$$b_1 = \frac{K_i \omega_p \Delta T_s}{\omega_z}\frac{1}{1+\omega_p \Delta T_s} \tag{41}$$

2.2.4 DCO

The digitally controlled oscillator varies from a VCO by only accepting a digital (quantized) frequency tuning signal, called the oscillator tuning word (OTW). A DCO modeled in discrete time as a recursive phase integrator, dependent on (a) the DCO gain K_{DCO}, equal to the frequency tuning of the oscillator per OTW LSB , (b) the digital OTW u[n], and (c) the sampling period T=f_{ref}^{-1}.

$$\Phi_{out}[n] = \Phi_{out}[n-1] + 2\pi K_{DCO} u[n] \Delta T_s \tag{42}$$

Application of the z-transform yields:

$$\mathrm{H}_{DCO}(z) = \frac{\Phi_{out}(z)}{u(z)} = \frac{2\pi K_{DCO} \Delta T_s}{1 - z^{-1}} \tag{43}$$

Application of the bilinear transform to the DCO transfer function yields:

$$\mathrm{H}_{DCO}(s) = \frac{\Phi_{out}(s)}{u(s)} = \frac{2\pi K_{DCO} \Delta T_s}{1 - (1 - s\Delta T_s)} = \frac{2\pi K_{DCO}}{s} \tag{44}$$

2.2.5 Discrete-time PLL transfer function

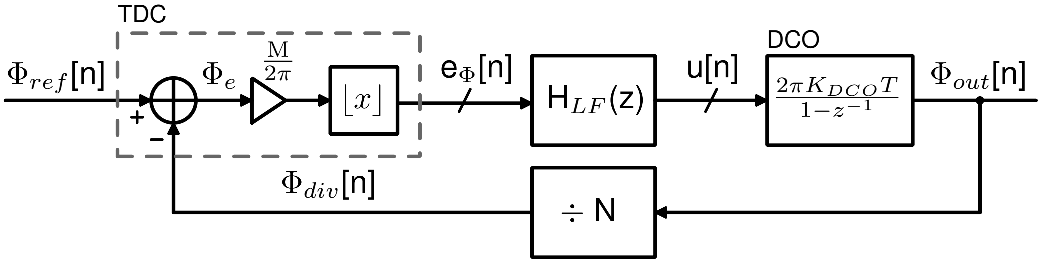


Figure 9: Discrete time PLL model.

The transfer function for the discrete-time PLL can be computed in the z-domain, and also approximated continuously. The open loop z-domain transfer function is:

$$\begin{aligned} \mathrm{L}(z) &= \mathrm{H}_{TDC}(z)\mathrm{H}_{LF}(z)\mathrm{H}_{DCO}(z)\mathrm{H}_{DIV}(z) \qquad (45) \\ &= 2\pi K_{DCO} K_i \Delta T_s^2 \frac{\mathrm{M}}{\mathrm{N}} \frac{\omega_p}{\omega_z} \frac{(1 + \omega_z \Delta T_s) - z^{-1}}{(1 + \omega_p \Delta T_s) - z^{-1}(3 + 2\omega_p \Delta T_s) + z^{-2}(3 + \omega_p \Delta T_s) - z^{-3}} \qquad (46) \end{aligned}$$

The closed loop z-domain PLL phase transfer function is:

$$\mathrm{T}(z) = \frac{\Phi_{out}(z)}{\Phi_{ref}(z)} = \frac{2\pi K_{DCO} K_i \Delta T_s^2 \mathrm{M} \frac{\omega_p}{\omega_z}(1 + \omega_z \Delta T_s) - z^{-1}}{\begin{pmatrix} (1 + \omega_p \Delta T_s + 2\pi K_{DCO} K_i \Delta T_s^2 \frac{\mathrm{M}}{\mathrm{N}} \frac{\omega_p}{\omega_z}(1 + \omega_z \Delta T_s)) - z^{-1}(3 + \\ 2\omega_p \Delta T_s + 2\pi K_{DCO} K_i \Delta T_s^2 \frac{\mathrm{M}}{\mathrm{N}} \frac{\omega_p}{\omega_z}) + z^{-2}(3 + \omega_p \Delta T_s) - z^{-3} \end{pmatrix}} \tag{47}$$

The s-domain approximation of the transfer function is:

$$\mathrm{L}(s) = \mathrm{H}_{TDC}(s)\mathrm{H}_{LF}(s)\mathrm{H}_{DCO}(s)\mathrm{H}_{DIV}(s) = \frac{\mathrm{M}}{\mathrm{N}} \frac{K_{DCO} K_i}{s^2} \frac{\left(\frac{s}{\omega_z} + 1\right)}{\left(\frac{s}{\omega_p} + 1\right)} \tag{48}$$

And in closed loop configuration the s-domain PLL phase transfer function is:

$$\mathrm{T}(s) = \frac{\Phi_{out}(s)}{\Phi_{ref}(s)} = \frac{\mathrm{M} K_{DCO} K_i \left(\frac{s}{\omega_z} + 1\right)}{s^3 \frac{1}{\omega_z} + s^2 + \frac{\mathrm{M}}{\mathrm{N}} K_{DCO} K_i \left(\frac{s}{\omega_z} + 1\right)} = \mathrm{N} \frac{\mathrm{L}(s)}{1 + \mathrm{L}(s)} \tag{49}$$

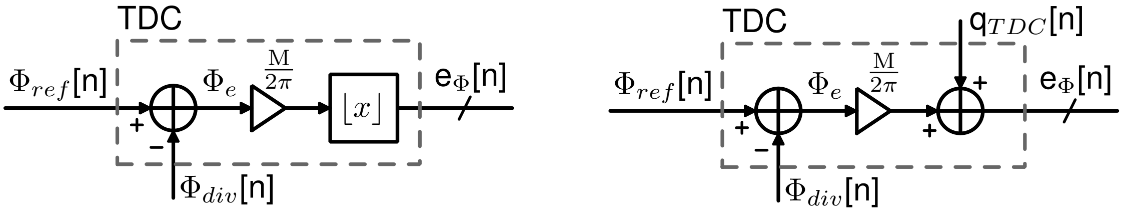
Incidentally, the s-domain approximation is significantly simpler and will is preferred in this paper for purposes of analysis.

2.3 ADPLL Noise Model

The predominant sources of noise in the discrete-time ADPLL are chiefly quantization (in the TDC, loop filter, and DCO), along with thermal noise (in the DCO, divider and TDC). The noise generated by these quantization sources will be discussed in the following sections.

2.3.1 TDC noise

The predominant phase noise source in the TDC is due to quantization. A straightforward approach to model quantization noise is to utilize the model of figure 10b to represent quantization.



(a) TDC Model.

(b) TDC additive noise model.

Figure 10: TDC quantization noise models.

Using this model, the quantized signal e_Φ[n] is the sum of the its unquantized representation $\Phi_e \frac{M}{2\pi}$ with a quantization error signal $\mathrm{q}_{TDC}[n]$. Figure 11 illustrates this process.

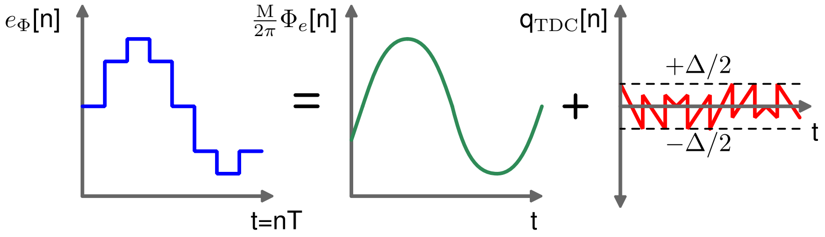


Figure 11: Quantization as via additive error signal.

The quantization noise signal has the statistical property that it is uniformly distributed in the range $[-\Delta/2, \Delta/2]$, i.e. $P_q(Q = q) = \mathrm{U}(-\Delta/2, \Delta/2)$ if Δ is the quantization step size. The power of the TDC quantization noise signal is:

$$\sigma^2_{q_{TDC}} = \int_{-\infty}^{\infty} q^2 P_q(Q = q)dq = \int_{-\Delta/2}^{\Delta/2} \frac{q^2}{\Delta}dq = \frac{\Delta^2}{12} \tag{50}$$

Since e_Φ[n] is digital signal, the minimum step size is Δ=1 LSB. The TDC quantization noise power is therefore $\sigma^2_{q_{TDC}} = 1/12$ LSB^2. The po If the quantization noise is assumed to be white, and the TDC is sampled at f_{ref}, the quantization PSD is:

$$S_{qn_{TDC}}(f) = \frac{P_{q_{TDC}}}{\Delta f} = \frac{\sigma^2_{q_{TDC}}}{f_{ref}} = \frac{\Delta^2}{12 f_{ref}} = \frac{1}{12 f_{ref}} \quad \frac{[\text{LSB}]^2}{[\text{Hz}]} \tag{51}$$

2.3.2 DCO noise

Noise in a DCO is resulting from (a) quantization of the oscillator tuning word u[n], and (b) from thermal and stochastic sources. In the digital PLL, the OTW quantization occurs in the loop filter, so this will be analyzed in the later loop filter section (2.3.4). Thus oscillator thermal/stochastic noise will be considered, to develop a phase noise model where oscillator phase noise $\Phi_{n_{DCO}}$ is an additive process to the oscillator phase Φ_{osc}, thus Φ_{out} = $\Phi_{osc} + \Phi_{n_{DCO}}$.

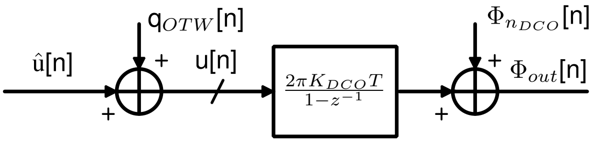


Figure 12: DCO additive noise model.

Oscillator phase noise due to stochastic and uncorrelated circuit and supply noise can be analyzed as additive voltage disturbance δv_n with variance $\sigma^2_{v_n}$ to the oscillator waveform V_{osc} at any given time. In a stable, noiseless oscillator, amplitude is inherently tied to signal phase, i.e. $V_{osc}(\Phi = \omega t)$. With additive noise, given $\frac{dV_{osc}(\Phi)}{d\Phi}$ is finite ∀t, a small voltage disturbance from noise δv_n will be coupled as a disturbance $\delta\Phi_n$ in the oscillator phase, shown in figure 13. The phase evolution of the noisy oscillator for an infinitesimal time increment δt with such a disturbance is:

$$\Phi_{out}(t + \delta t) = \Phi_{out}(t) + \delta\Phi_n + \omega_{osc}\delta t \tag{52}$$

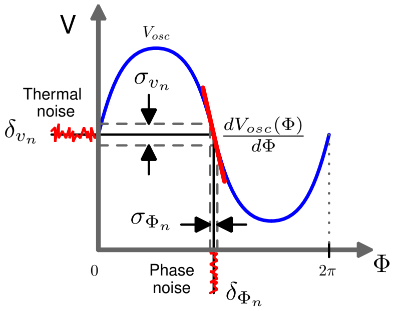


Figure 13: Voltage to phase noise conversion.

Assuming δv_n is Gaussian white noise, $\delta\Phi_n$ is sampled stochastically at any instant based on the probability distribution 53, dependent on the current oscillator phase Φ_{out} and the noiseless voltage-phase relation $V_{osc}(\Phi)$. It will be assumed that like the source noise δv_n, $\delta\Phi_n$ is white spectrum.

$$P(\delta\Phi_n|\Phi_{out}) = \text{Norm}\left(\mu = 0, \sigma = \sigma_{v_n}\left(\left.\frac{dV_{osc}(\Phi)}{d\Phi}\right|_{\Phi=\Phi_{out}}\right)^{-1}\right) \tag{53}$$

Spectral analysis of the noisy oscillator phase can be made utilizing discrete time-modeling. Converting 52 into a sampled signal with time step δt

$$\Phi_{out}[n+1] = \Phi_{out}[n] + \omega_{osc}\delta t + \delta\Phi_n[n|\Phi_{out}[n]] \tag{54}$$

Computing the z-transform, and splitting the result into the oscillation Φ_{osc} and phase noise Φ_n components:

$$\Phi_{out}(z) = \frac{\omega_{osc}\delta t}{z-1} + \frac{\delta\Phi_n(z)}{z-1} = \Phi_{osc}(z) + \Phi_n(z) \tag{55}$$

$$\Rightarrow \Phi_n(z) = \frac{\delta\Phi_n(z)}{z-1} \tag{56}$$

Application of the bilinear transform to 56 can be used to approximate the continuous phase noise spectrum, if $s = j\omega$

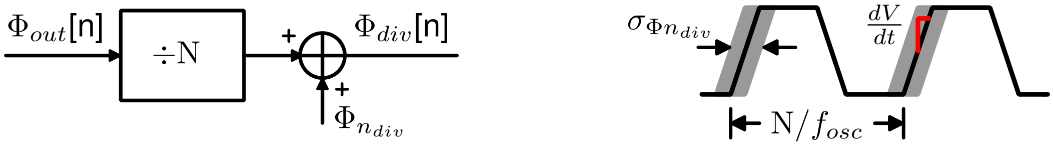
$$\Phi_n(s) = \Phi_n(z)|_{z=1-s\delta t} = \left.\frac{\delta\Phi_n(z)}{z-1}\right|_{z=1+s\delta t} = \frac{\delta\Phi_n(s)}{s\delta t} \tag{57}$$

The phase noise PSD is therefore:

$$S_{\Phi n_{DCO}}(f) = \lim_{\Delta T\to\infty}\frac{1}{\Delta T}\left|\frac{\delta\Phi_n(j2\pi f)}{j2\pi f\delta t} * \mathcal{F}\left\{\text{rect}\left(\frac{t}{\Delta t}\right)\right\}\right|^2 = \frac{S_{0\Phi n_{DCO}}}{f^2} \tag{58}$$

Following that the phase disturbance signal $\delta\Phi_n(t)$ is white spectrum, a constant value for its PSD $S_{0\Phi n_{DCO}}$ can be defined. The value for $S_{0\Phi n_{DCO}}$ is highly dependent on implementation and is best extracted by means of curve fitting simulation or physical measurement.

2.3.3 Divider noise



(a) Divider noise model.

(b) Digital divider output jitter.

Figure 14: Divider phase noise.

Divider noise is manifested as jitter (with RMS distribution in time of $\sigma_{tn_{div}}$) on the the divider output. This is due to effects of stochastic and uncorrelated voltage noise coupling into the signal phase, much like in the case of oscillator phase noise. If the divider is a digital circuit, with edge rate dV/dt, and subject to thermal noise in the form of a voltage v_n, with noise power of $\sigma_{v_n}^2$, the divider phase noise power added to the divider output is:

$$\sigma^2_{\Phi n_{div}} = \omega^2_{ref}\sigma^2_{tn_{div}} = \omega^2_{ref}\left(\frac{dV}{dt}\right)^{-2}\sigma^2_{v_n} \tag{59}$$

At lock, the output of a digital divider will have an update rate $f_{osc}/\mathrm{N} \approx f_{ref}$, which can be treated as the sampling rate of the output phase signal $\Phi_{div}[n]$. Thus if the divider phase noise power is confined into a bandwidth equal to f_{ref}, the spectral density of divider noise is:

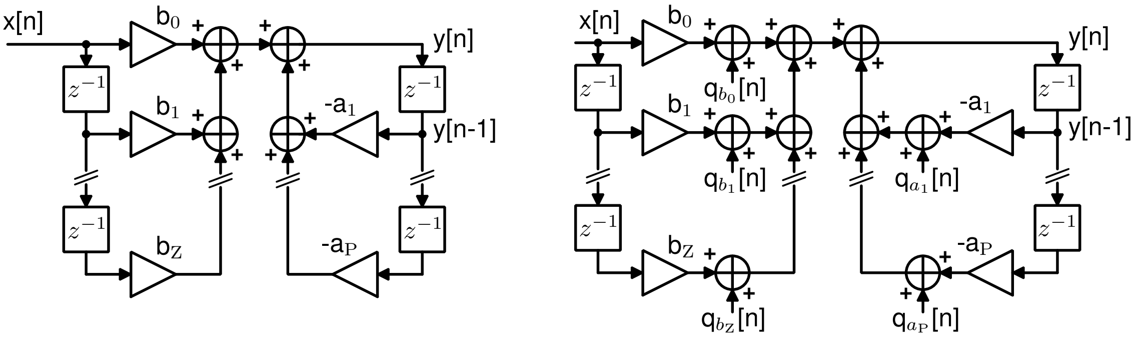
$$S_{\Phi n_{div}}(f) = \frac{\sigma^2_{\Phi n_{div}}}{f_{ref}} = 2\pi\omega_{ref}\sigma^2_{tn_{div}} = 2\pi\omega_{ref}\left(\frac{dV}{dt}\right)^{-2}\sigma^2_{v_n} \quad \frac{[\text{rad}]^2}{[\text{Hz}]} \tag{60}$$

2.3.4 Loop filter noise - direct type I

In a digital loop filter, quantization noise arises from rounding errors due to finite precision in the arithmetic circuits that implement the filter. Quantization noise power here will be derived under the assumption of a direct type-I filter implementation, with B bits in each fixed point word throughout the loop filter. In a digital implementation of the canonical z-domain transfer function 61 as the direct type-I structure of figure 15a, delays are constructed using registers, adders with digital adders, and the filter coefficient gain terms $\{a_1, ...a_N; b_0, ..., b_M\}$ with digital multipliers. The registers and adders do not introduce extra round-off error beyond that already existing, however the multipliers will if the products of B bit words, yielding 2B bits, are mapped onto B bit words.

$$\mathrm{H}_{LF}(z) = \frac{\sum_{j=0}^{Z} b_j z^{-j}}{1+\sum_{k=1}^{P} a_k z^{-k}} \tag{61}$$

Quantization in this case can be represented by adding a quantization error signal q_x[n] to the result of each ideal multiplication, as shown in figure 15b. This is the same approach for TDC quantization noise in section 50. The noise power associated with each q$_x$[n] is identical, with $\sigma_{qx}^2 = 1/12$ LSB^2.



(a) Ideal structure. **(b)** With added quantization error signals.

Figure 15: Direct type I filter implementation with quantization.

Assuming the quantization error signals of each multiplier are uncorrelated with all other multipliers, the output-referred noise power of the filter can be computed as the sum of the output-referred individual contributions. These contributions can be determined via solving for the transfer function from each source q$_x$[n] of each quantization noise to the output y[n]. In the case of the direct type I filter structure, all quantization sources q$_x$[n] have the same transfer characteristic to the output y[n]:

$$\frac{Y(z)}{Q_x(z)} = \frac{1}{1+\sum_{k=1}^{P} a_k z^{-k}} \tag{62}$$

Applying the bilinear transform to 62, with high oversampling where N$\cdot$BW$_{loop}10 < f_{ref}$, and N is the number of poles in the system.

$$\left.\frac{Y(z)}{Q_x(z)}\right|_{z^{-1}=1-sT} \approx \frac{1}{1+\sum_{k=1}^{P} a_k - s\sum_{k=1}^{P} k a_k} \tag{63}$$

The output power spectral density is then for one error source is, confined to a bandwidth defined by the (sampling) reference frequency f_{ref}:

$$S_{qx}(f) = \frac{\sigma_{qx}^2}{f_{ref}} \left| \frac{Y(s)}{Q_x(s)} \right|^2_{s=j2\pi f} \approx \frac{1}{12 f_{ref}} \left| \frac{1}{1 + \sum_{k=1}^{P} a_k - j2\pi f \sum_{k=1}^{P} k a_k} \right|^2 \quad \frac{[\text{LSB}]^2}{[\text{Hz}]} \tag{64}$$

Given P poles and Z zeros in the filter, the total output quantization PSD of the filter is 65. The total loop filter noise PSD linearly scales with the number of multipliers in the direct type-I filter implementation.

$$S_{qn_{LF}}(f) = (P + Z + 1) S_{qx}(f) \quad \frac{[\text{LSB}]^2}{[\text{Hz}]} \tag{65}$$

2.3.5 PLL noise sensitivity transfer functions

Having developed models for noise of generated by each PLL component, noise sensitivity transfer functions must be computer to refer each noise source to the PLL output in terms of phase. In the developed noise theory, thus far all noise sources have been modeled as additive phase components. The full system model illustrating this is:

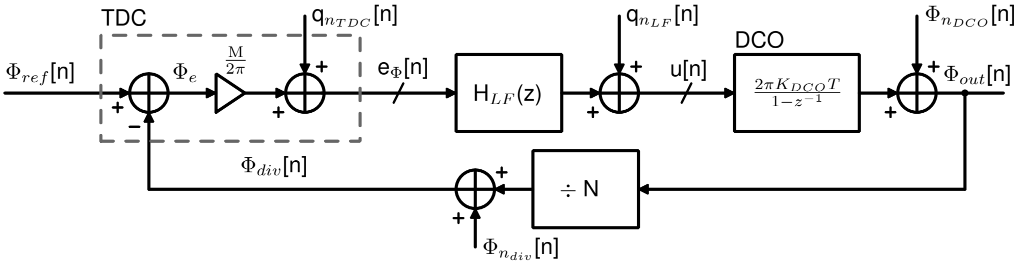


Figure 16: Full PLL additive noise model.

It is useful to define a transfer function $\hat{\text{T}}(s)$ which characterizes the normalized closed loop response from reference to output of the PLL. $\hat{\text{T}}(s)$ is defined in terms of the loop gain L(s).

$$\hat{\text{T}}(s) = \frac{\text{L}(s)}{1 + \text{L}(s)} \quad \text{s.t.} \quad \text{T}(s) = \frac{\Phi_{out}}{\Phi_{ref}} = \text{N}\hat{\text{T}}(s) \tag{66}$$

Solving for the closed transfer functions between each $q_{n_{TDC}}$, $q_{n_{LF}}$, $\Phi_{n_{DCO}}$ and $\Phi_{n_{div}}$ to the output Φ_{out} utilizing s-domain approximations yield:

$$\frac{\Phi_{out}(s)}{q_{n_{TDC}}(s)} = \frac{2\pi \frac{K_{DCO}}{s} \text{H}_{LF}(s)}{1 + \text{L}(s)} = 2\pi \frac{\text{N}}{\text{M}} \frac{\text{L}(s)}{1 + \text{L}(s)} = 2\pi \frac{\text{N}}{\text{M}} \hat{\text{T}}(s) \tag{67}$$

$$\frac{\Phi_{out}(s)}{\Phi_{n_{DCO}}(s)} = \frac{1}{1 + \text{L}(s)} = 1 - \hat{\text{T}}(s) \tag{68}$$

$$\frac{\Phi_{out}(s)}{q_{n_{LF}}(s)} = \frac{2\pi \frac{K_{DCO}}{s}}{1 + \text{L}(s)} = 2\pi \frac{K_{DCO}}{s} (1 - \hat{\text{T}}(s)) \tag{69}$$

$$\frac{\Phi_{out}(s)}{\Phi_{n_{div}}(s)} = \frac{\text{M} \frac{K_{DCO}}{s} \text{H}_{LF}(s)}{1 + \text{L}(s)} = \text{N} \frac{\text{L}(s)}{1 + \text{L}(s)} = \text{N}\hat{\text{T}}(s) \tag{70}$$

2.3.6 PLL phase noise and output PSD relationship

When analyzing PLL noise, the noise power spectral density of the PLL output is of most interest. Up to this point, noise has been defined in terms of phase noise Φ_n, or an unwanted added component to the oscillator phase signal $\Phi_{osc} = \omega_{osc}t$. The PLL output phase signal Φ_{out} is thus:

$$\Phi_{out}(t) = \Phi_{osc}(t) + \Phi_n(t) = \omega_{osc}t + \Phi_n(t) \tag{71}$$

Computation of PSD requires the PLL output voltage waveforms. These here will be defined in terms of complex exponentials. Given an oscillation amplitude A_0:

$$V_{out} = \Re\left(A_0 e^{j\Phi_{out}(t)}\right) = \Re\left(A_0 e^{j\omega_{osc}t} e^{j\Phi_n(t)}\right) \tag{72}$$

Assuming the phase noise signal is zero mean, $\mathbb{E}[\Phi_n(t)] = 0$, and the power of phase noise signal is small, $\mathrm{Var}[\Phi_n(t)] << 1$, then the approximation $e^{j\Phi_n(t)} = 1 + j\Phi_n(t)$ can be applied by truncating the exponential Taylor series expansion.

$$\begin{aligned} V_{out} &= \Re\left(A_0 e^{j\omega_{osc}t} e^{j\Phi_n(t)}\right) = \Re\left(A_0 e^{j\omega_{osc}t} + j\Phi_n(t) A_0 e^{j\omega_{osc}t}\right) && (73)\\ &= A_0\cos(\omega_{osc}t) - \Phi_n(t) A_0 \sin(\omega_{osc}t) && (74)\end{aligned}$$

The result is a carrier cosine signal, and an orthogonal sine signal modulated by the phase noise Φ_n. From this, the spectral density of the phase noise relative to the carrier can be estimated. The power spectral density $S_{V_{out}}$ is computed in 75-77. Due to orthogonality of the sine/cosine components of 74, the cross terms that appear in the PSD are zero.

$$\begin{aligned} S_{V_{out}} &= \lim_{\Delta T\to\infty} \frac{1}{\Delta T} |\mathcal{F}\{V_{out}(t)\cdot \mathrm{rect}(t/\Delta T)\}|^2 && (75)\\ &= \lim_{\Delta T\to\infty} \frac{A_0^2}{\Delta T} |\mathcal{F}\{\cos(\omega_{osc}t)\cdot \mathrm{rect}(t/\Delta T)\}|^2 && (76)\\ &+ \lim_{\Delta T\to\infty} \frac{A_0^2}{\Delta T} |\mathcal{F}\{\Phi_n(t)\cdot \mathrm{rect}(t/\Delta T)\} * \mathcal{F}\{\sin(\omega_{osc}t)\cdot \mathrm{rect}(t/\Delta T)\}|^2 && (77)\end{aligned}$$

Single side band (SSB) noise power spectral density $\mathcal{L}(\Delta f)$ is defined as the phase noise PSD at offset Δf from the carrier frequency f_{osc}, normalized to the carrier power. Here the PSD carrier component is given by 76, and the noise component by 77.

$$\mathcal{L}(\Delta f) = \lim_{\Delta T\to\infty} \frac{1}{\Delta T} |\mathcal{F}\{\Phi_n(t)\cdot \mathrm{rect}(t/\Delta T)\}|^2 \Big|_{f=\Delta f} = S_{\Phi_n}(\Delta f) \tag{78}$$

Thus, the PLL output noise PSD relative to the carrier is equal to the PSD $S_{\Phi_n}(\Delta f)$ of the phase noise signal $\Phi_n(t)$.

2.3.7 PLL output-referred noise PSD

In terms of analysis, PLL noise PSD referred to the PLL output is of most interest. Thus far the following have been established: (a) noise spectrum generated by each individual PLL component, (b) the PLL phase noise sensitivity functions, and (c) the relationship between PLL output PSD and output phase noise. Now these can be combined to provide a final result for full PLL output-referred noise PSD. An important assumption here is all noise sources are uncorrelated, so their independent noise power contributions may be summed to find the total noise PSD. The PLL output phase noise PSD for each noise source is simply found by multiplying magnitude squared of the respective noise sensitivity function with the noise source PSD. Thus:

$$S_{\Phi n_{TDC,out}}(f) = S_{qn_{TDC}}(f) \left| \frac{\Phi_{out}(f)}{q_{n_{TDC}}(f)} \right|^2 = \frac{1}{12 f_{ref}} \left| 2\pi \frac{\mathrm{N}}{\mathrm{M}} \hat{\mathrm{T}}(f) \right|^2 \tag{79}$$

$$S_{\Phi n_{DCO,out}}(f) = S_{\Phi n_{DCO}}(f) \left| \frac{\Phi_{out}(f)}{\Phi_{n_{DCO}}(f)} \right|^2 = \frac{S_{0\Phi n_{DCO}}}{f^2} \left| 1 - \hat{\mathrm{T}}(f) \right|^2 \tag{80}$$

$$S_{\Phi n_{LF,out}}(f) = S_{qn_{LF}}(f) \left| \frac{\Phi_{out}(f)}{q_{n_{LF}}(f)} \right|^2 \approx \frac{K_{DCO}^2}{12 f_{ref} f^2} \left| \frac{1 - \hat{\mathrm{T}}(f)}{1 + \sum_{k=1}^{P} a_k - j2\pi f \sum_{k=1}^{P} k a_k} \right|^2 \tag{81}$$

$$S_{\Phi n_{div,out}}(f) = S_{\Phi n_{div}}(f) \left| \frac{\Phi_{out}(f)}{\Phi_{n_{div}}(f)} \right|^2 = f_{ref} \left| 2\pi \sigma_{tn_{div}} \mathrm{N} \hat{\mathrm{T}}(f) \right|^2 \tag{82}$$

The output SSB noise PSD at offset Δf relative to the carrier normalized to carrier power of PLL will be:

$$\mathcal{L}(\Delta f) = S_{\Phi n_{TDC,out}}(\Delta f) + S_{\Phi n_{DCO,out}}(\Delta f) + S_{\Phi n_{LF,out}}(\Delta f) + S_{\Phi n_{div,out}}(\Delta f) \tag{83}$$

Hypothetical ex. plot of all components? (in disco?)

3 Methods

Talk about how simulator is implemented: Discrete simulation models of phase noise, dco etc Filter optimization -phase noise and lock time estimate in frequency domain

Design recommendations: High sampling rate Minimum choice of TDC tesolution, constraints for divider jitter,

3.1 Behavioral, discrete event PLL simulation

3.1.1 Phase noise modeling

3.1.2 Measurement of phase noise, spurs

3.1.3 Monte-carlo sampling

Used to verify stability

3.2 Loop filter optimization

Reference phase noise does not matter, is always fixed. DCO and TDC phase noise should be highest. Will simulate with other noise sources, but framework will optimize loop filter design and provide recommendations for other parameters (max divider jitter) so DCO and TDC phase noise are dominant.

3.2.1 Estimation of settling time

Based on a continuous model of the PLL dynamics, the PLL closed loop phase transfer function $\mathrm{T}(s)$ is defined in the following form, where number of poles P > number of zeros Z. The transfer function is defined as a rational function of two polynomial functions of s.

$$\mathrm{T}(s) = \frac{\sum_{j=0}^{Z} b_j s^j}{\sum_{k=0}^{P} a_n s^n} \tag{84}$$

An estimate of the step response settling time of $\mathrm{T}(s)$ can by utilizing its representation in state space. This is given in 85, with input vector $\mathrm{U}(s)$, state vector $\mathbf{X}(s)$, and output $\mathbf{Y}(s)$. The state-space representation from a s-domain transfer function can be quickly solved computationally with available signal processing packages such as `scipy.signal`.

$$s\mathbf{X}(s) = \mathbf{A}\mathbf{X}(s) + \mathbf{B}\mathrm{U}(s) \tag{85}$$

$$Y(s) = \mathbf{C}\mathbf{X}(s) + \mathbf{D}\mathrm{U}(s) \tag{86}$$

The set of k eigenvalues $\{\lambda_1, ..., \lambda_N\}$ corresponding to poles for the system are found as the roots of 87.

$$|\mathbf{A} - \lambda\mathbf{I}| = 0 \tag{87}$$

With the constraint of number of poles > number of zeros, the system $\mathrm{T}(s)$ may be represented via partial fraction decomposition using the poles from the eigenvalues of state matrix $\mathbf{A}$ $\{\lambda_1, ..., \lambda_N\}$:

$$T(s) = \sum_{k=1}^{P} \frac{c_k}{s - \lambda_k} \tag{88}$$

The step response of this system will take the form as a sum of complex exponentials:

$$y(t) = c_1 e^{\lambda_1 t} + ... + c_k e^{\lambda_k t} \tag{89}$$

The dynamics of the step response are governed by the exponential components of y(t). If $\{\lambda_1, ..., \lambda_N\} \in \mathbb{C}$ where $\lambda_k = \sigma_k + j\omega_K$, the real portion of each λ_k will describe the transient behavior. The long term settling of y(t) will be dominated by the λ_k with the smallest valued real component, that is the dominant pole. This value is approximately the reciprocal time constant for the system. Settling time t_s can be considered as the interval required for the signal to drop within a tolerance band $\pm\delta_{tol}\mathrm{y}(\infty)$ about the final value $\mathrm{y}(\infty)$.

$$t_s = \tau \ln(\delta_{tol}) = \frac{\ln(\delta_{tol})}{\min(|\Re(\{\lambda_1, ..., \lambda_k\})|)} \tag{90}$$

This settling time estimate is computationally fast, as it requires only (a) computation of state matrix $\mathbf{A}$, (b) computation of the eigenvalues of $\mathbf{A}$, and (c) computation of settling time from the eigenvalue with minimum real component.

3.2.2 Estimation of PLL phase noise

It is assumed that the dominant output-referred phase noise contributions are due to the DCO thermal noise and the TDC quantization. If such is the case, total output integrated noise power is at a minimimum when the TDC and DCO contributions are approximately equal.

Thus S_{TDC} and S_{DCO} are the PLL output-referred noise PSD respectively for the TDC and DCO noise sources. The total PLL output noise PSD $S_{\Sigma}(f)$ is (N is the PLL divider modulus):

$$S_{\Sigma}(f) = f_{clk} \cdot |2\pi N \cdot G(f)|^2 S_{TDC} + |1 - G(f)|^2 S_{DCO} \tag{91}$$

Given a bandwidth of interest Δf (i.e. baseband bandwidth for radio applications), the total integrated phase noise power is:

$$P_{\phi noise} = 2 \int_0^{\Delta f} S_{\Sigma}(f) df \tag{92}$$

This can be computation solved for a grid of K values in the interval Δf, where each point represents a frequency bin f_{bin} = Δf/K. Therefore this estimate is implemented as such:

$$\hat{P}_{\phi noise} = 2 \sum_{k=0}^{K-1} S_{\Sigma}(k f_{bin}) f_{bin} \tag{93}$$

3.2.3 Optimization algorithm

Utilize BFGS optimization with constraints on settling time

4 Discussion

Discuss choices for loop filter optimizer (loop filter type) how was loop filter transfer function prototype arrived at? ie what is the best loop filter design

why direct type 1 structure is selected. lf noise via simulation with input noise optimization of data word precision loop filter error due to coefficient round-off

Why DCO+TDC is primary focus of phase noise optimization (other noise sources are easier to reduce below the limits of these two) Plot of BW vs tdc/dco noise, motivating optimum

Talk about why DCO flicker noise doesn't matter (i.e. Reference flicker is much greater) Also, we don't optimize for reference flicker, it will be constant no matter what, reference is separate from PLL and PLL reponse is flat N multiplier from ref to output. Discuss why only random-phase walk component of oscillator noise considered.

Compare to state of art (perrot) Perrot's pre-existing work: general purpuse simulation architecture, doesn't directly handle optimization for integer-N, especially in heavily quantized case

Models are not accurate for frequencies near or greater than reference frequency???

Recommendations for maximum divider jitter, loop filter resolution

4.1 Divider noise constraint

Output refered phase noise PSD of TDC:

$$S_{\Phi n_{TDC,out}} = \frac{1}{12 f_{ref}} \left| 2\pi \frac{N}{M} G(f) \right|^2 \tag{94}$$

Output refered phase noise PSD of divider:

$$S_{\Phi n_{div,out}} = f_{ref} \left| 2\pi N \sigma_{tn_{div}} G(f) \right|^2 \tag{95}$$

The output-referred phase noise for the TDC and divider have the same frequency dependence. So by setting $S_{\Phi n_{div,out}} < S_{\Phi n_{TDC,out}}$, a constraint to force PLL output divider less than TDC noise can be found:

$$\sigma_{tn_{div}} < \frac{1}{\mathrm{M} f_{ref}} = \Delta t_{step_{TDC}} \tag{96}$$

Must simply ensure that jitter of divider is much less than TDC resolution, which is a reasonable demand. Thus, it is reasonable to ignore divider noise in the phase noise optimization if divider noise can reasonably be made insignificant in the overall output phase noise.

4.2 Example exercise

Use WuRx design specs to motivate design example... put updated/better definition of specs relative to design example

Parameter	Value	Unit	Notes
Frequency	2.4-2.4835	GHz	2.4G ISM Band
Ref. frequency	16	MHz	Yields 6 channels
Power	≤ 100	μW	
Residual FM	≤ 107	kHz$_{RMS}$	BER $\leq$ 1e-2, f_{dev}=±250 KHz
Initial Lock Time	≤ 50	μs	Upon cold start
Re-lock Time	≤ 5	μs	Coming out of standby
Bandwidth	100	kHz	(nominally), tunable

Table 1: System-level specifications

Parameter	Value	Unit	Notes
DCO LSB Resolution	≤ 50	kHz	Determined from quantization noise.
DCO DNL	< 1	LSB	Ensures monotonicity
TDC Resolution	≤ 3.8	ns	
TDC Resolution (bits)	≥ 4.03	bits	

Table 2: Component-level specifications.

5 Conclusion

Extend to fractional-N. Loop filter design will be the same, difference is divider model.

6 Baseband phase noise in radio with PLL

This doesn't have a home yet, perhaps appendices... will be relevant for BER estimation for radio system (will be referenced in discussion)

6.1 Modulated Signal

A generalized FSK/PSK modulated signal $x_s(t)$ can be written in the following with phase trajectory $\phi_s(t)$. $\phi_s(t)$ is composed of a modulation component $\phi_m(t)$ and carrier component $\omega_c t$.

$$x_s(t) = A_s \cos(\phi_s(t)) = A_s \cos(\phi_m(t) + \omega_c t) \tag{97}$$

6.2 Local oscillator - noisy PLL

A noisy PLL can be realized as $x_{lo}(t)$ with phase trajectory $\phi_{lo}(t)$. $\phi_{lo}(t)$ is comprised of a phase noise component $\phi_n(t)$ and an oscillation $\omega_{lo}t$.

$$x_{lo}(t) = A_{lo} \cos(\phi_{lo}(t)) = A_{lo} \cos(\phi_n(t) + \omega_{lo} t) \tag{98}$$

6.3 Baseband phase noise

The signal in the baseband, $x_{bb}(t)$, is given by mixing $x_s(t)$ with the LO $x_{lo}(t)$. For analysis of the baseband phase noise, zero-IF ($\omega_c = \omega_{lo}$) is considered. Thus:

$$\begin{aligned} x_{bb}(t) &= A_s \cos(\phi_m(t) + \omega_c t) \cdot A_{lo} \cos(\phi_n(t) + \omega_{lo} t) && (99) \\ &= \frac{1}{2} A_s A_{lo} \left[\cos(2\omega_{lo} t + \phi_m(t) + \phi_n(t)) + \cos(\phi_m(t) - \phi_n(t))\right] && (100) \end{aligned}$$

The $2\omega_{lo}t$ component is assumed to be rejected due to limited mixer bandwidth. Thus the baseband signal is now:

$$\begin{aligned} x_{bb}(t) &= A_s \cos(\phi_m(t) + \omega_c t) \cdot A_{lo} \cos(\phi_n(t) + \omega_{lo} t) && (101) \\ =&\frac{1}{2} A_s A_{lo} \left[\cos(\phi_m(t) - \phi_n(t))\right] = \frac{1}{4} A_s A_{lo} \left[e^{j\phi_m(t)} e^{-j\phi_n(t)} + e^{-j\phi_m(t)} e^{j\phi_n(t)}\right] && (102) \end{aligned}$$

The phase noise is presumed to be comprised of random phase and frequency walk, such that $\langle \phi_n(t) \rangle = 0$. Also, the phase noise is presumed to be small in amplitude, such that

$\phi_n(t) << 1$, so a Taylor polynomial-based approximation of $e^{-j\phi_n(t)} = 1 - j\phi_n(t)$ can be made. Accordingly :

$$x_{bb}(t) \approx \frac{1}{4} A_s A_{lo} \left[e^{j\phi_m(t)}(1 - j\phi_n(t)) + e^{-j\phi_m(t)}(1 + j\phi_n(t)) \right] \tag{103}$$

$$= \frac{1}{4} A_s A_{lo} \left[\cos(\phi_m(t)) + \phi_n(t) \sin(\phi_m(t)) \right] \tag{104}$$

The above can be treated as a signal component $m(t) = \cos(\phi_m(t))$, and a noise component $n(t) = \phi_n(t)\sin(\phi_m(t))$. The Fourier transform of the noise component can be considered:

$$N(f) = \mathcal{F}\{\phi_n(t)\sin(\phi_m(t))\} = \Phi_n(f) * \mathcal{F}\{-\cos(\phi_m(t) + \pi/2)\} \tag{105}$$

$$= -j\Phi_n(f) * \mathcal{F}\{\cos(\phi_m(t))\} \tag{106}$$

The signal and noise power spectral densities are therefore approximately:

$$S_{MM} = |M(f)|^2 = |\mathcal{F}\{\cos(\phi_m(t))\}|^2 \tag{107}$$

$$S_{NN} = |N(f)|^2 = |\Phi_n(f) * \mathcal{F}\{\cos(\phi_m(t))\}|^2 \tag{108}$$

The inband noise and signal power of these components can be easily computed via integration:

$$P_x = \int_{-\Delta f/2}^{+\Delta f/2} S_{XX} df \tag{109}$$

SNR is therefore the ratio of P_M/P_N within a bandwidth of interest Δf, which is straightforward to determine computationally for arbitrary PLL phase noise and modulation spectral densities.

References

[1] F. Gardner. “Charge-Pump Phase-Lock Loops”. In: *IEEE Transactions on Communications* 28.11 (Nov. 1980), pp. 1849–1858. DOI: `10.1109/tcom.1980.1094619`.

[2] Behzad Razavi. *Design of analog CMOS integrated circuits*. McGraw-Hill Education, 2017.

[3] Behzad Razavi. “Design of monolithic phase-locked loops and clock recovery circuitsDa tutorial”. In: 1996.

A Appendix - Schedule

remove

Week Number	**Dates**	**Tasks**	**Outcomes**
36	2.9 - 8.9	Review PLL Design	Refreshed Knowledge
37	9.9 - 15.9	Modeling/simulation (set up)	–
38	16.9 - 22.9	Modeling/simulation	TDC/DCO Requirements
39	23.9 - 29.9	Modeling/simulation	Loop Filter/Digital Algorithms
40	30.9 - 6.10	Modeling/simulation	**Loop filter,** Ideal implementation in Cadence
41	7.10 - 13.10	Circuit Research	DCO/Divider topologies
42	14.10 - 20.10	Circuit Research	TDC/other topologies
43	21.10 - 27.10	Circuit Implementation	Digital logic (schematic)
44	28.10 - 3.11	Circuit Implementation	DCO (schematic)
45	4.11 - 10.11	Circuit Implementation	Divider/other (schematic)
46	11.11 - 17.11	Circuit Implementation (TDC)	
47	18.11 - 24.11	Circuit Implementation (TDC)	TDC (schematic)
48	25.11 - 1.12	Full Circuit testing	Testbenches, find bugs, design fixes
49	2.12 - 8.12	Full Circuit testing	Design Fixes/iteration
50	9.12 - 15.12	–	–

Legend: **Done** **Current** **Revised**

B Appendix - Code

Placeholder example...

```python
##################################################################
# Simulation loop
##################################################################

t0 = time.clock()
for n in range(SAMPLES)[1:]:
    clk_out[n] = clk.update()
    tdc_out[n] = TDC_SCALE*((TDC_OFFSET+tdc.update(clk=clk_out[n-1], xin=div_out[n-1]))%TDC_STEPS)
    lf_out[n]  = lf.update(xin=tdc_out[n-1], clk=clk_out[n-1])
    osc_out[n] = dco.update(lf_out[n-1])
    div_out[n] = div.update(osc_out[n-1], DIV_N)
tdelta = time.clock() - t0
print("\nSimulation completed in%f s"%tdelta)
```

Listing 1: excode

C DCO tuning

C.1 Backgate tuning

Tuning of a ring oscillator DCO through backgate terminal control will be considered. A general analysis of ring oscillator frequency will be made first to begin.

C.2 Ring oscillator frequency derivation

To analyze the oscillation frequency of a CMOS ring oscillator, an approximate model for a CMOS inverter will first be considered. A common model for delay in digital circuits [elmore delay model] is an RC circuit, where the MOSFET channels are approximated with an average conductance value $\langle g_{ch} \rangle$, and the output node is approximated to have a capacitance of C. With such a model, a ring oscillator would be assumed to have waveforms as decaying exponential, with time constant $\tau = \langle g_{ch} \rangle^{-1} C$, such as in Figure 17.

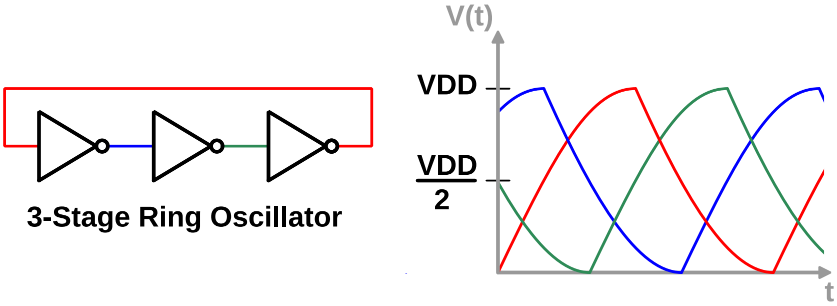
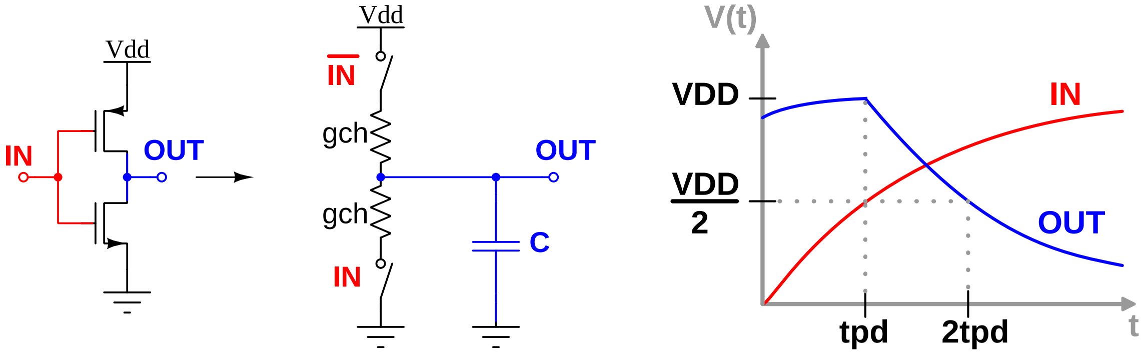


Figure 17: Model for ring oscillator.



(a) Inverter approximate model. **(b)** Inverter waveforms in ring oscillator.

Figure 18: Approximate model for ring oscillator inverter delay cell.

To calculate oscillation frequency ring oscillator from the RC model, several inferences are made:

- The switching point V_M of the inverters is $V_{DD}/2$, based on the assumption that the NMOS and PMOS are of equal strength.
- The output of an inverter will have a decaying exponential which starts coincident with

the passing of V_M at the input.

- The propagation delay t_{pd} for an inverter will be the time differential between the V_M crossing points on the input and output.
- The oscillator frequency will be f_{osc} = $1/2Nt_{pd}$, where N is the number of stages (i.e. defined by 2N propagation delays).

Following the definition of V_M, it is trivial to find that t_{pd} = $\tau \ln 2$. It is therefore known that:

$$f_{osc}^{-1} = 2Nt_{pd} = \frac{2\ln(2)NC}{\langle g_{ch}\rangle} \tag{110}$$

C.2.1 Finding $\langle g_{ch}\rangle$ and C

The node capacitance C is trivial to find based on the inverter gate capacitance and a lumped load capacitance term C_L:

$$C = C_{ox}\left(W_N L_N + W_P L_N\right) + C_L \tag{111}$$

The average channel conductance $\langle g_{ch}\rangle$ is more involved to find. To do so, several assumptions are made:

- L $>>$ L$_{min}$, so no velocity saturation, and therefore square law is applicable.
- NMOS and PMOS have equal V_t and transconducance.
- Output transition occur with the active FET in saturation during t_{pd}. This requires:
  - $V_{DD}/4 < V_t < V_{DD}/2$

Following those assumptions, $\langle g_{ch}\rangle$ can be computed via integral within the period t_{pd}:

$$\langle g_{ch}\rangle = \frac{1}{t_{pd}}\int_0^{t_{pd}} \frac{I_{out}(t)}{V_{out}(t)}dt \tag{112}$$

I_{out} is computed using the saturated MOSFET square law model an exponential waveforms assumptions. An I_{short} term is included to account for output current reduction from short-circuit conduction.

$$I_{out}(t) = \frac{k_n}{2}\left(\frac{W}{L}\right)_n \left[(V_{in}(t) - V_t)^2\right] - I_{short} = \frac{k_n}{2}\left(\frac{W}{L}\right)_n \left[\left(V_{DD}\left(1 - e^{-t/\tau}\right) - V_t\right)^2 - \left(\frac{V_{DD}}{2} - V_t\right)^2\right] \tag{113}$$

$k_n = \mu_n C_{ox}$, with the equal PMOS/NMOS assumption, $k_n\left(\frac{W}{L}\right)_n = k_p\left(\frac{W}{L}\right)_p$. V_{out} is simply a decaying exponential with a delay $t_p d$ versus the input:

$$V_{out} = V_{DD}e^{-(t-t_{pd})/\tau} \tag{114}$$

Now, computing the integral for $\langle g_{ch}\rangle$ yields:

$$\langle g_{ch}\rangle = \frac{1}{2}\mu_n C_{ox}\left(\frac{W}{L}\right)_n \left[V_{DD}\left(\frac{7}{8\ln 2} - 1\right) - V_t\left(\frac{1}{\ln 2} - 1\right)\right] \tag{115}$$

As a simplification, α is defined as:

$$\alpha = \left[V_{DD}\left(\frac{7}{8\ln 2} - 1\right) - V_t\left(\frac{1}{\ln 2} - 1\right)\right] \tag{116}$$

C.2.2 Handling unequal NMOS/PMOS

In the case of different threshold voltages for NMOS and PMOS:

$$f_{osc}^{-1} = N(t_{pdn} + t_{pdp}) = \ln(2)NC\left(\frac{1}{\langle g_{ch}\rangle_n} + \frac{1}{\langle g_{ch}\rangle_p}\right) = \frac{2\ln(2)NC}{\langle g_{ch}\rangle'} \tag{117}$$

A modified $\langle g_{ch}\rangle'$ is defined:

$$\langle g_{ch}\rangle' = 2\left(\frac{1}{\langle g_{ch}\rangle_n} + \frac{1}{\langle g_{ch}\rangle_p}\right)^{-1} = 2\frac{\langle g_{ch}\rangle_n\langle g_{ch}\rangle_p}{\langle g_{ch}\rangle_n + \langle g_{ch}\rangle_p} = 2\frac{\frac{1}{2}\mu_n C_{ox}\left(\frac{W}{L}\right)_n\alpha_n\frac{1}{2}\mu_p C_{ox}\left(\frac{W}{L}\right)_p\alpha_p}{\frac{1}{2}\mu_n C_{ox}\left(\frac{W}{L}\right)_n\alpha_n + \frac{1}{2}\mu_p C_{ox}\left(\frac{W}{L}\right)_p\alpha_p} \tag{118}$$

This is somewhat unmanagable, hovever enforcing $\mu_n C_{ox}\left(\frac{W}{L}\right)_n = \mu_p C_{ox}\left(\frac{W}{L}\right)_p$ for V_M to equal $V_{DD}/2$ gives:

$$\langle g_{ch}\rangle' = \frac{1}{2}\mu_n C_{ox}\left(\frac{W}{L}\right)_n\frac{2\alpha_n\alpha_p}{\alpha_n + \alpha_p} = \frac{1}{2}\mu_n C_{ox}\left(\frac{W}{L}\right)_n\alpha' \tag{119}$$

Thus α_n and α_p are found for the according threshold voltages and then $\langle g_{ch}\rangle$ can be found.

$$\alpha' = \frac{2\alpha_n\alpha_p}{\alpha_n + \alpha_p} \tag{120}$$

C.2.3 Solving for oscillator frequency and power

Solving for oscillator frequency:

$$f_{osc} = \frac{\mu_n C_{ox}}{4\ln 2NC}\left(\frac{W}{L}\right)_n\left[V_{DD}\left(\frac{7}{8\ln 2} - 1\right) - V_t\left(\frac{1}{\ln 2} - 1\right)\right] \tag{121}$$

If gate capacitance is the dominant load component, and PMOS/NMOS are equal sized such that $C = 2WLC_{ox}$:

$$f_{osc} = \frac{\mu_n}{8\ln 2N}\cdot\frac{1}{L^2}\left[V_{DD}\left(\frac{7}{8\ln 2} - 1\right) - V_t\left(\frac{1}{\ln 2} - 1\right)\right] \tag{122}$$

Power can also be calculated, knowing in digital circuits $P = fC_\Sigma V_{DD}^2$, where C_Σ is the total active capacitance. Thus:

$$P_{osc} = Nf_{osc}CV_{DD}^2 = \frac{\mu_n C_{ox}}{4\ln 2}\left(\frac{W}{L}\right)_n\left[V_{DD}\left(\frac{7}{8\ln 2} - 1\right) - V_t\left(\frac{1}{\ln 2} - 1\right)\right] \tag{123}$$

It should be noted that the power consumption is proportional to FET aspect ratio (W/L).

C.3 Ring oscillator backgate tuning derivation

Using the basic expressions for ring oscillator frequency, the operature under backgate biasing can be found. In UTBB-FDSOI processes, the threshold voltage of a FET varies with linear dependence on the applied back gate bias V_{BG} (relative to source). Given the body effect coefficient of a process, γ, V_t is:

$$V_t = V_{t0} - \gamma V_{BG} \tag{124}$$

Using this in the ring oscillator frequency equation:

$$f_{osc} = \frac{\mu_n C_{ox}}{4\ln 2 NC}\left(\frac{W}{L}\right)_n \left[V_{DD}\left(\frac{7}{8\ln 2} - 1\right) - V_{t0}\left(\frac{1}{\ln 2} - 1\right) + \gamma V_{BG}\left(\frac{1}{\ln 2} - 1\right)\right] \tag{125}$$

Equivalently, $f_{osc} = f_{0,osc} + \Delta f_{osc}(V_{BG})$, where:

$$\Delta f_{osc}(V_{BG}) = \gamma V_{BG} \frac{\mu_n C_{ox}}{4\ln 2 NC}\left(\frac{W}{L}\right)_n \left[\frac{1}{\ln 2} - 1\right] \tag{126}$$

And $f_{0,osc}$ is the frequency with no backgate bias. If the backgate is swept from 0 to V_{DD}, and the node capacitance is increasingly varied (C0 to C3), Figure 19 is observed. Note that the change in frequency is linear with to backgate bias.

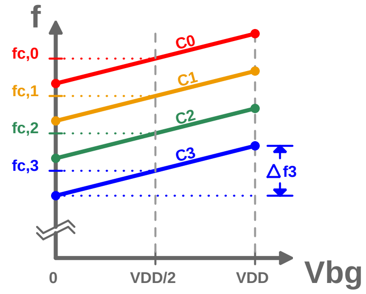


Figure 19: Backgate-tuned ring oscillator with coarse tuning capacitor bank.

If the backgate voltage is constrained in the range [0, V_{DD}], the center frequency f_c in the tuning range of the oscillator is then:

$$f_c = \frac{\mu_n C_{ox}}{4\ln 2 NC}\left(\frac{W}{L}\right)_n \left[V_{DD}\left(\frac{7}{8\ln 2} - 1 + \frac{\gamma}{2\ln 2} - \frac{\gamma}{2}\right) - V_{t0}\left(\frac{1}{\ln 2} - 1\right)\right] \tag{127}$$

The tuning range is also therefore:

$$\Delta f = \frac{\gamma V_{DD}}{2}\frac{\mu_n C_{ox}}{4\ln 2 NC}\left(\frac{W}{L}\right)_n \left[\frac{1}{\ln 2} - 1\right] \tag{128}$$

The fractional tuning range of the oscillator is:

$$\frac{\Delta f}{f_c} = \frac{1}{2} \cdot \frac{\gamma V_{DD}\left(1 - \ln 2\right)}{V_{DD}\left(\frac{7}{8} - \ln 2 + \frac{\gamma}{2} - \frac{\gamma}{2}\ln 2\right) - V_{t0}\left(1 - \ln 2\right)} \tag{129}$$

If a N-bit DAC is used to control the oscillator, the resulting DCO gain is therefore:

$$K_{DCO} = \frac{\Delta f}{2^{N_{DAC}}} = \frac{f_c}{2^{N_{DAC}+1}} \cdot \frac{\gamma V_{DD}\left(1 - \ln 2\right)}{V_{DD}\left(\frac{7}{8} - \ln 2 + \frac{\gamma}{2} - \frac{\gamma}{2}\ln 2\right) - V_{t0}\left(1 - \ln 2\right)} \tag{130}$$

C.4 DCO Gain Uncertainty

The DCO gain K_{DCO} is used in setting the loop filter coefficients, so the uncertainty of the DCO gain is of interest to allow for statistical analysis of the PLL across process variation. The uncertainty of K_{DCO} (normalized with nominal K_{DCO} value) as a function of V_{DD}, V_{t0} and γ is:

$$\sigma_{KDCO} = \sqrt{\left(\frac{\partial K_{DCO}}{\partial V_{DD}} \cdot \frac{\sigma_{VDD}}{K_{DCO}}\right)^2 + \left(\frac{\partial K_{DCO}}{\partial V_{t0}} \cdot \frac{\sigma_{Vt0}}{K_{DCO}}\right)^2 + \left(\frac{\partial K_{DCO}}{\partial \gamma} \cdot \frac{\sigma_{\gamma}}{K_{DCO}}\right)^2} \tag{131}$$

$$\frac{\partial K_{DCO}}{\partial V_{DD}} = \frac{f_c}{2^{N_{DAC}+1}} \cdot \frac{-\gamma V_{t0}(1-\ln 2)^2}{\left[V_{DD}\left(\frac{7}{8} - \ln 2 + \frac{\gamma}{2} - \frac{\gamma}{2}\ln 2\right) - V_{t0}\left(1 - \ln 2\right)\right]^2} \tag{132}$$

$$\frac{\partial K_{DCO}}{\partial V_{t0}} = \frac{f_c}{2^{N_{DAC}+1}} \cdot \frac{\gamma V_{DD}(1-\ln 2)^2}{\left[V_{DD}\left(\frac{7}{8} - \ln 2 + \frac{\gamma}{2} - \frac{\gamma}{2}\ln 2\right) - V_{t0}\left(1 - \ln 2\right)\right]^2} \tag{133}$$

$$\frac{\partial K_{DCO}}{\partial \gamma} = \frac{f_c}{2^{N_{DAC}+1}} \cdot \frac{V_{DD} \cdot (1-\ln 2)\left[V_{DD}\left(\frac{7}{8} - \ln 2\right) - V_{t0}\left(1 - \ln 2\right)\right]}{\left[V_{DD}\left(\frac{7}{8} - \ln 2 + \frac{\gamma}{2} - \frac{\gamma}{2}\ln 2\right) - V_{t0}\left(1 - \ln 2\right)\right]^2} \tag{134}$$

Simplified:

$$\sigma_{KDCO} = \frac{1}{\gamma V_{DD}\left[V_{DD}\left(\frac{7}{8} - \ln 2 + \frac{\gamma}{2} - \frac{\gamma}{2}\ln 2\right) - V_{t0}\left(1 - \ln 2\right)\right]} \cdot \sqrt{\left(\gamma V_{t0}(1-\ln 2)\sigma_{VDD}\right)^2 + \left(\gamma V_{DD}(1-\ln 2)\sigma_{Vt0}\right)^2 + \left(V_{DD}\left[V_{DD}\left(\frac{7}{8} - \ln 2\right) - V_{t0}\left(1-\ln 2\right)\right]\sigma_{\gamma}\right)^2} \tag{135}$$

TODO - extract γ, V_{t0} variance for FETs in process kit.