Digital PLL's, Part 3 – Phase Lock an NCO to an External Clock

Sometimes you may need to phase-lock a numerically controlled oscillator (NCO) to an external clock that is not related to the system clocks of your ASIC or FPGA. This situation is shown in Figure 1. Assuming your system has an analog-to-digital converter (ADC) available, you can sync to the external clock using the scheme shown in Figure 2. This time-domain PLL model is similar to the one presented in Part 1 of this series on digital PLL's [1]. In that PLL, we dealt with phases only, which resulted in a very simple model. Here, we have a sinusoidal reference signal instead of a phase reference signal, so we need to make some modifications.

Our new model adds a Hilbert transformer and a complex digital phase detector (PD). The Hilbert transformer converts the real input to a complex signal. The complex PD compares the phase of this signal to that of the complex sinusoidal output of the NCO. The model includes ADC quantization noise and the Gaussian noise of the ADC and external clock, allowing us to examine how they affect the NCO output phase noise. However, the model does not include the phase noise contributions of the PLL sample clock or the external clock.

In this post, I'll describe the components of the PLL and then perform a couple of simulations. A word of caution: I have not implemented this design in hardware, so there may be gotchas with the approach that I have not anticipated.

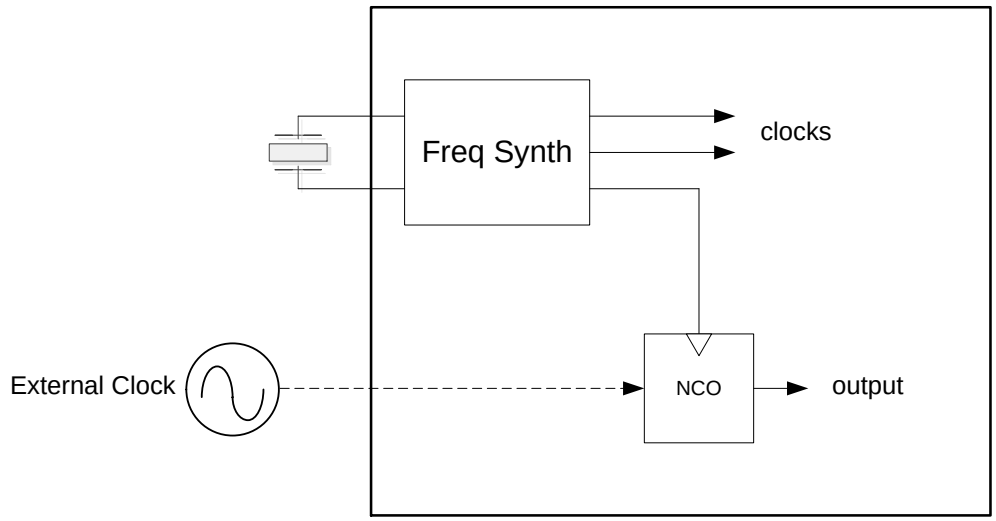

Figure 1. How do you phase-lock the NCO to an external clock that is unrelated to the system clocks?

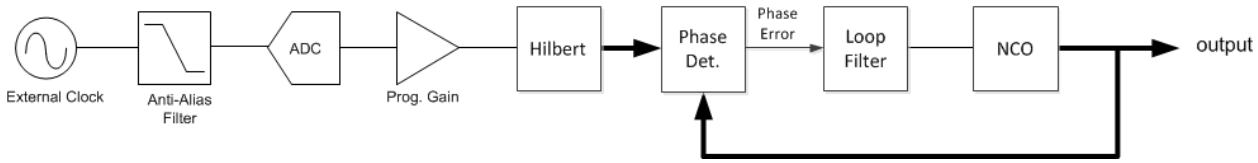

Figure 2. PLL to phase-lock an NCO to an external clock (wide lines are complex signals).

Hilbert Transformer

Here we'll use the same 31-tap Hilbert transformer presented in an earlier post [2]. Its Matlab code is part of the m-file listed in the Appendix. For sinusoidal input, the Hilbert Transformer generates I and Q sinusoids, with Q lagging I by π/2. The pure-imaginary frequency response of Q/I is plotted in Figure 3. For sample frequency of 40 MHz, this design allows an input frequency range of roughly 4 MHz to 16 MHz (flat region of the response).

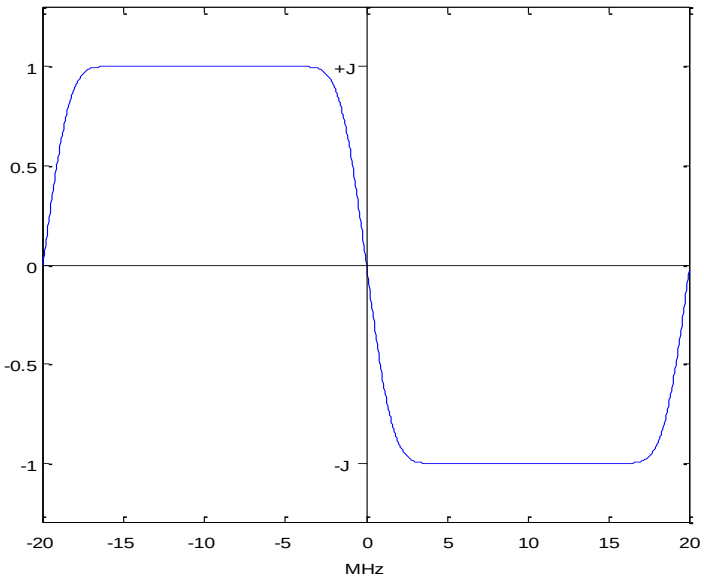

Figure 3. Frequency Response of Hilbert Transformer, f_s = 40 MHz.

Complex Digital Phase Detector

In the following, we refer to the external clock as the reference clock, and we make the simplifying assumption that it is sinusoidal. The phase detector computes the phase difference of two complex sinusoidal inputs [3]. A conceptual block diagram of this remarkably simple device is shown in Figure 4. Note that the NCO input has a negative exponent (inverted phase). The output is simply the imaginary part of the complex product of the two inputs:

$$y = Im\{A\,exp\left(j\left(\omega t + \phi_{ref}\right)\right) * exp\left(-j(\omega t + \phi_{nco})\right)\} = Im\{A \exp\left(j\left(\phi_{ref} - \phi_{nco}\right)\right\}$$

$$y = A\,sin\left(\phi_{ref} - \phi_{nco}\right)$$

$$y = -A\,sin\left(\phi_{nco} - \phi_{ref}\right) \qquad (1)$$

The terms involving ω have canceled, leaving a slowly rotating or stationary phase difference. Equation 1 is plotted in Figure 5 for A = 1. The useful phase error range is from -1/4 to +1/4 cycles or –π/4 to +π/4 radians. The phase detector gain is the slope of the curve near zero, which (since sin(x) ~= x for small x) is -A rad^{-1}. The gain in cycles^{-1} is thus -2πA.

Given phase detector inputs I_{ref}/Q_{ref} and I_{nco}/-Q_{nco} (where the negative sign inverts the NCO phase), the output is:

$$y = Im\{\left(I_{ref} + jQ_{ref}\right)(I_{nco} - jQ_{nco})\}$$

$$y = -I_{ref}Q_{nco} + Q_{ref}I_{nco} \qquad (2)$$

Equation 2 is implemented by the block diagram in Figure 6. As you can see, just two multipliers and an adder are used.

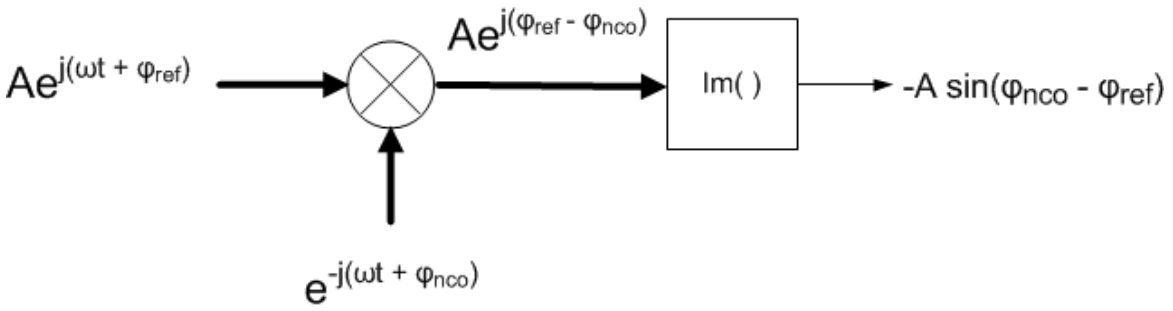

Figure 4. Complex Digital Phase Detector Conceptual Diagram (wide lines are complex).

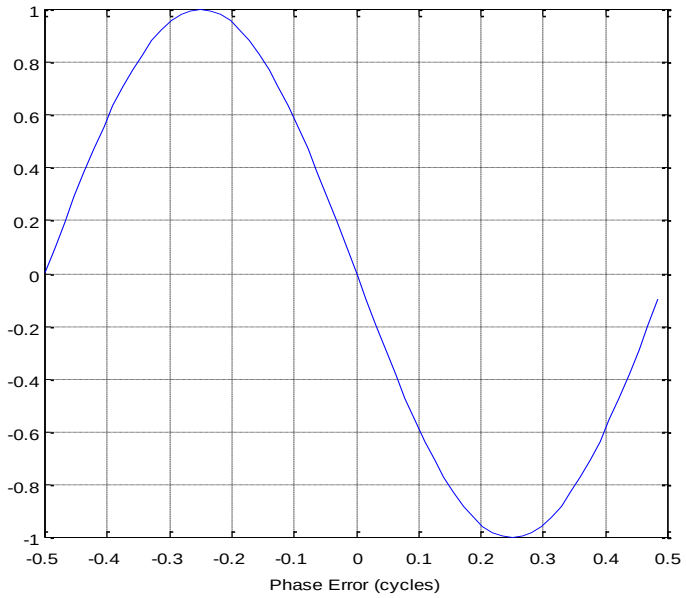

Figure 5. Phase Detector Characteristic, sinusoidal input amplitude = 1.

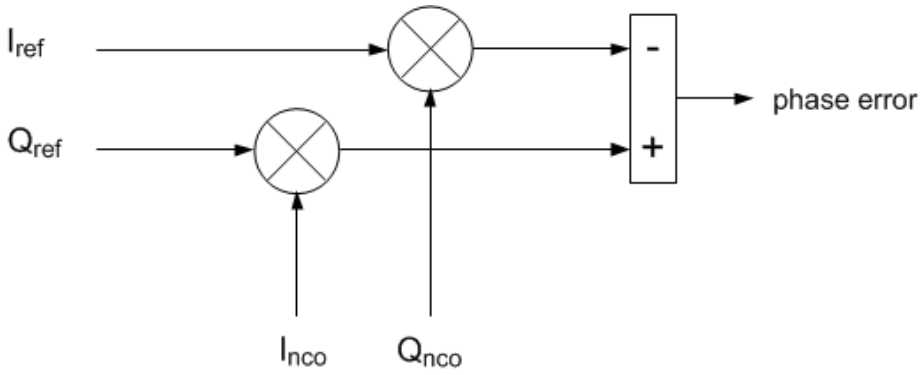

Figure 6. Implementation of Complex Digital Phase Detector.

Loop Filter

I discussed calculation of the loop filter coefficients for a 2nd order PLL in Part 2 [4]. The formulas for the coefficients are:

$$K_L = \frac{2\zeta\omega_n}{K_p} * \frac{T_s}{K_{nco}} \qquad (3)$$

$$K_I = \frac{\omega_n^2}{K_p} * \frac{T_s^2}{K_{nco}} \qquad (4)$$

Where K_L and K_I are the proportional and integrator gains of the loop filter in Figure 7, and the other parameters are defined as follows:

ζ	desired loop damping factor (dimensionless)
ω_n	desired loop natural frequency, rad/s
K_p	phase detector gain, cycle^{-1}
K_{nco}	nco gain (dimensionless)
T_s	sample time, s

In the model, I have assumed the reference clock is a sine with amplitude A = 1. This gives phase detector gain $K_p = 2\pi$ cycles^{-1}. I used damping factor $\zeta = 1$.

For larger values of loop natural frequency, K_L may be greater than one, causing the loop filter output level to exceed +/- 1 during acquisition. The loop filter output should clip (not roll-over). Clipping can hurt acquisition, so typically the clip level is set higher than the expected excursion of the loop filter output. It may be appropriate to set clip level to a value greater than +/-1.

NCO

We developed the difference equations for the NCO (shown in Figure 7) in Part 1 [1]. The NCO used here has a couple of modifications from the Part 1 example:

1. We quantize the phase to 20 bits and the I/Q outputs to 12 bits.
2. In Part 1, we used the NCO's phase output as the input to the phase detector; here we use the NCO's sine and cosine outputs.

The model uses the Matlab sin and cos functions and thus does not provide a method of computing those outputs. For a description of a look-up table (LUT) approach for computing sine and cosine, see [5]. For a description of a Cordic approach, see [6].

Anti-alias Filter, ADC, and Programmable Gain

An anti-alias filter is needed to filter clock harmonics and noise at frequencies above $f_s/2$. The ADC input does not have to be a pure sinusoid; for example, if the third harmonic of the clock is below $f_s/2$, the anti-alias filter does not need to attenuate it.

The model assumes an 8-bit ADC using the same sample rate as the NCO, and includes Gaussian noise on the input clock. This noise represents noise of the clock source and the input noise of the ADC. The SNR due to quantization noise of an ideal 8-bit ADC is 6.02*8 + 1.76 dB = 49.9 dB [7]. I somewhat arbitrarily chose a Gaussian noise amplitude σ = 0.0015 (See Appendix for Matlab code: `.0015*randn(1,N)`). For input clock amplitude = 1 V peak = .707 V rms, this gives an SNR of 20 $\log_{10}$(.707/.0015) = 53.5 dB. Thus the Gaussian noise is 3.6 dB below the quantization noise.

The gain of the phase detector is proportional to the reference clock level. The model does not include it, but a programmable gain block (or AGC), as shown in Figure 7, would be useful for setting the reference clock level. The gain might have programmable values of 1, 9/8, 10/8, … etc. For example, given an ADC full-scale input of 2 V_{pp}, if the clock level at the ADC input were 1.5 V_{pp}, the programmable gain could be set to 10/8, giving a clock level at the Hilbert transformer input of 1.5/2 * 10/8 = 0.94 with respect to full-scale.

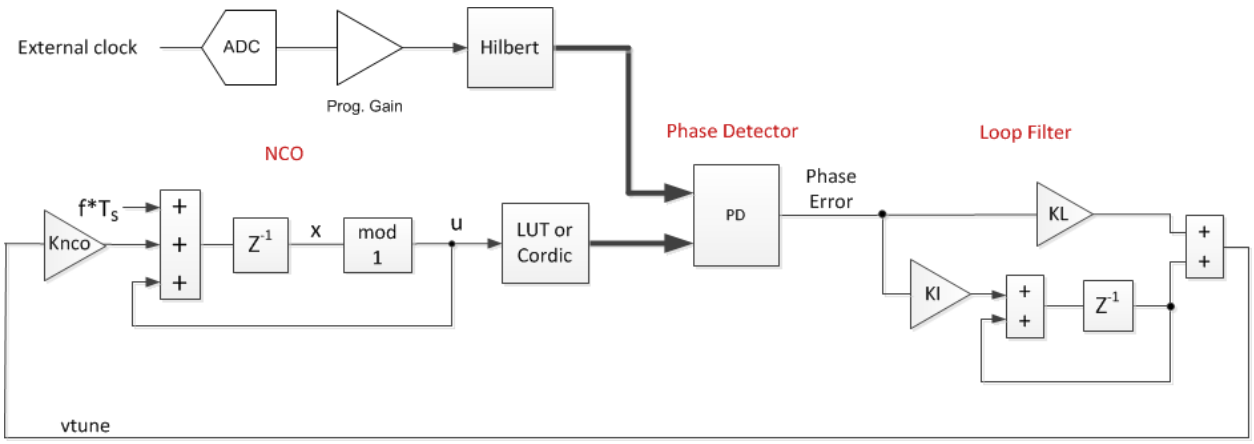

Figure 7. PLL Block Diagram showing NCO and Loop Filter parameters.

Simulation Example 1

This time-domain simulation solves the difference equations of the 2nd-order PLL, as was done in Part 1. The PLL model in Part 1 did not include quantization. Here, I have quantized a few key signals: the ADC output; the Hilbert transformer output; the NCO phase and NCO output. The m-file for this simulation is provided in the Appendix. Table 1 lists the parameter values used.

Table 1. Simulation Parameters

Symbol	Value	Units	Description
fs	40	MHz	sample frequency
fref	6.3001	MHz	external clock frequency
fnco	-100 ppm w.r.t. fref		NCO initial frequency
A	1		amplitude of external clock
--	53.5	dB	ref clock SNR = $20\log_{10}(.707/.0015)$
nbits	8	bits	resolution of ADC
Knco	1/4096		NCO gain constant
KI	6.4E-5		Loop filter integrator term
KL	0.41		Loop filter proportional term
--	20	bits	NCO phase quantization
--	12	bits	NCO output quantization
N	40,000		total number of samples
--	1	ms	time duration = N/fs

Note that the values of KI and KL used result in loop natural frequency of 2 KHz, with damping of 1.0. For a real-world design, we'd probably want to use a lower natural frequency (to minimize spurs and noise from the ADC), but I chose this value to keep simulation time manageable. Note that the loop bandwidth of the PLL is slightly greater than the natural frequency. The external clock frequency is 6.3001 MHz, and the NCO initial frequency is 100 ppm (630.01 Hz) below that.

The simulation results are shown in Figures 8 through 12. Figure 8 shows the spectrum of the reference clock at the output of the 8-bit ADC. The highest spur is roughly 75 dB below the carrier level. Figure 9 shows the output of the phase detector as the loop acquires lock. The fuzz on this signal is caused by the Gaussian noise and ADC quantization noise. Figure 10 shows the loop filter output (vtune). Figure 11 shows the spectrum of the NCO cosine output signal. The close-in spectrum of the NCO output is shown in Figure 12. Note that the model does not include the phase noise of the PLL sample clock.

In Figure 11, the highest spur is more than 100 dB below the carrier level – an improvement of more than 25 dB compared to the spurious at the ADC output. This occurs because the spurious level outside the 2 kHz bandwidth of the PLL is attenuated by the PLL's closed-loop response [4]. But given that the noise is amplitude noise, how does it affect the PLL at all? In the phase domain, you can view the noise

as a vector modulating the tip of the clock signal [8] (See Figure 14 in the next example). The vector has an amplitude and phase component. The amplitude component does not affect the PLL. The phase component, if inside the loop bandwidth, appears as noise in the NCO output. If outside the loop bandwidth, it is attenuated by the closed-loop response of the PLL.

Keep in mind that in a particular hardware implementation, spurs may fall within the loop bandwidth and thus will not be attenuated by the PLL. The next example looks more closely at the attenuation of noise outside the PLL loop bandwidth.

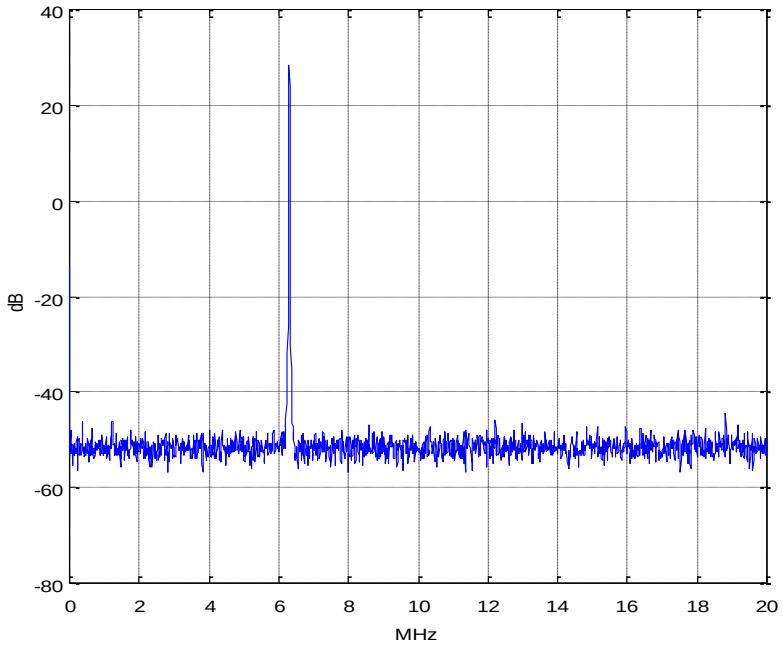

Figure 8. Spectrum at ADC output.

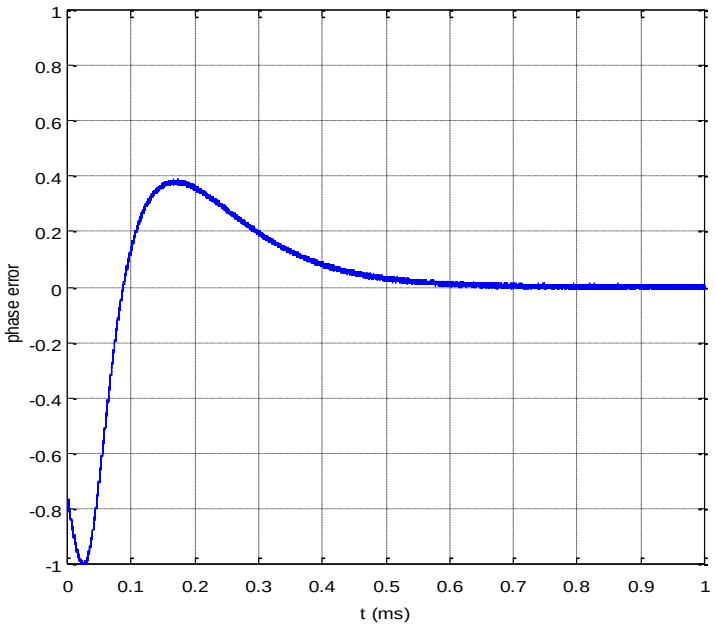

Figure 9. Phase Detector Output.

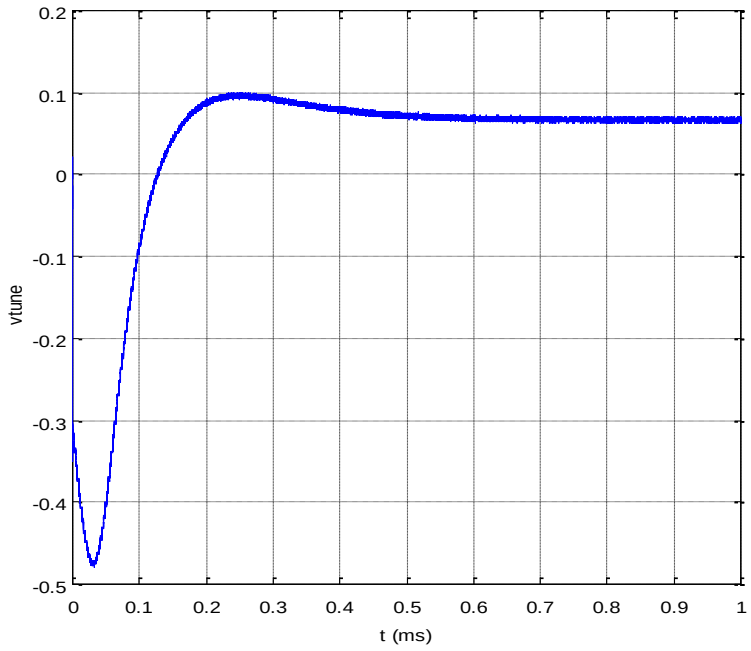

Figure 10. Loop Filter Output (vtune).

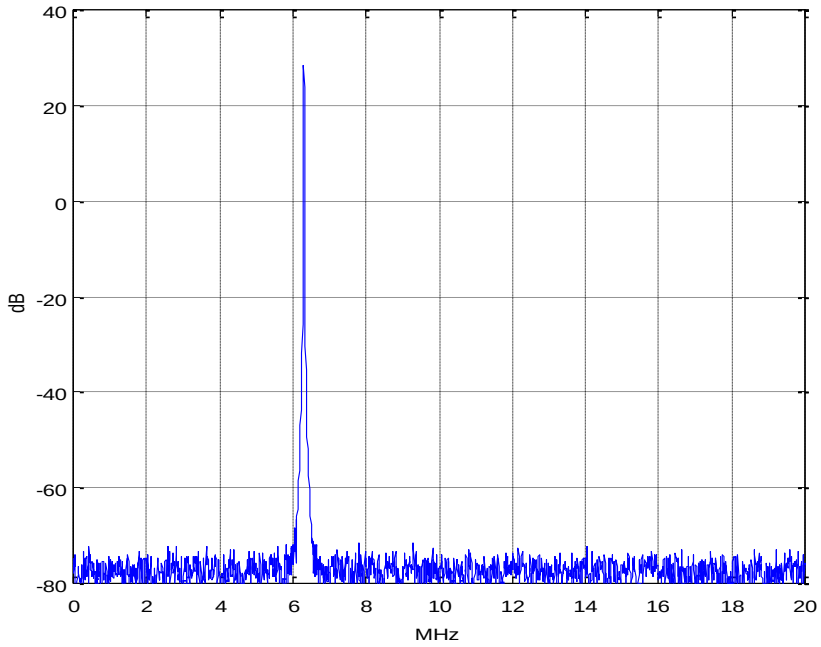

Figure 11. Spectrum of NCO cosine output.

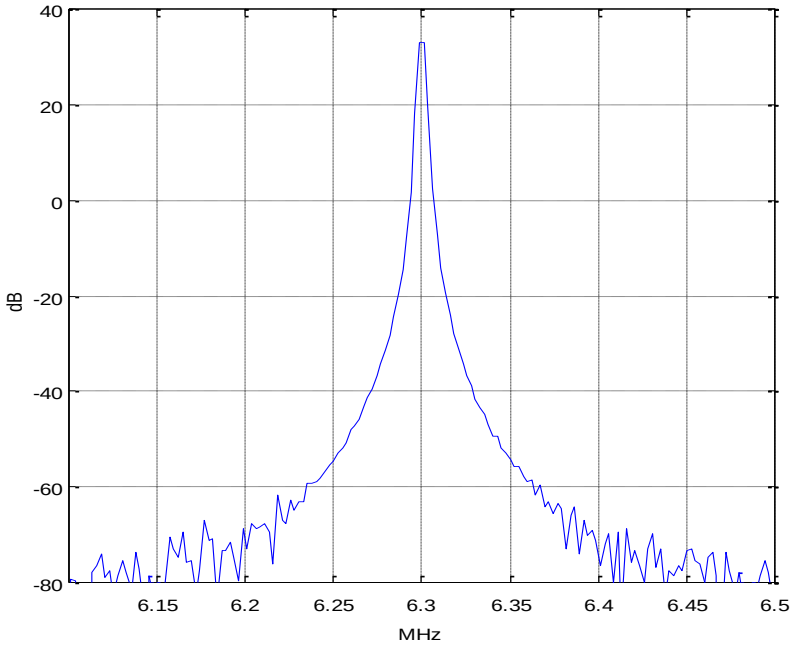

Figure 12. Close-in spectrum of NCO cosine output. Span = 400 kHz.

Simulation Example 2

In this example, we'll look at how the PLL's closed-loop response shapes the noise of the reference signal. We'll use most of the same parameters as in Example 1, but we'll increase the Gaussian noise level. We'll also change the loop natural frequency to 20 kHz and reduce the number of samples. These new parameters are coded as follows:

```
N= 20000                                    % number of samples
Vr= A*cos(phi_r) + .05*randn(1,N);          %  ref clock with Gaussian noise
KI= .0064;                          % loop filter integrator gain, fn = 20 kHz
KL= 4.1;                            % loop filter proportional gain, fn= 20 kHz
```

The reference clock SNR is $20\log_{10}(.707/.05)$ = 23 dB. It has the spectrum shown in the top of Figure 13. The spectrum of the NCO cosine output is shown in the bottom of Figure 13. You can see the shape of the PLL closed-loop response in the noise floor.

Figure 14 is a polar plot of the I/Q output of the Hilbert transformer, derotated to baseband. As discussed in Example 1, you can see that the phase vector is amplitude and phase modulated by the Gaussian noise (the noise causes the amplitude to exceed 1.0, which is a little flakey for a real application, but it works to demonstrate the concept). The phase detector computes the error due to the phase modulation, and the PLL attenuates components of the phase modulation outside the loop bandwidth. Figure 15 shows the I/Q output of the NCO, derotated to baseband: you can see that the phase noise is much less than that of the reference clock. Again, the model does not include the phase noise of the PLL sample clock. Figure 16 shows the increased NCO phase noise that results when f_n is widened to 200 kHz.

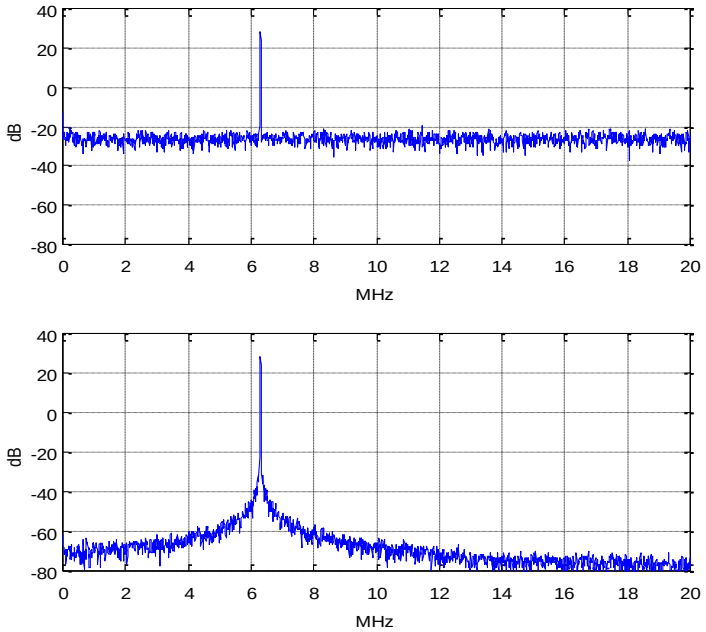

Figure 13. Top: Spectrum of reference clock with added Gaussian noise. `psd(Vr,2^12,fs/1e6)`
Bottom: Spectrum of NCO output for loop f_n = 20 kHz. `psd(y(5000:end),2^12,fs/1e6)`

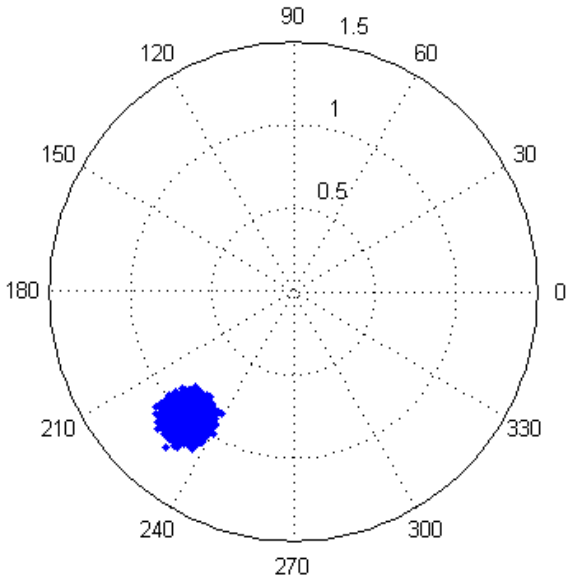

Figure 14. De-rotated phase of Hilbert Transformer output, showing Gaussian noise of reference clock.

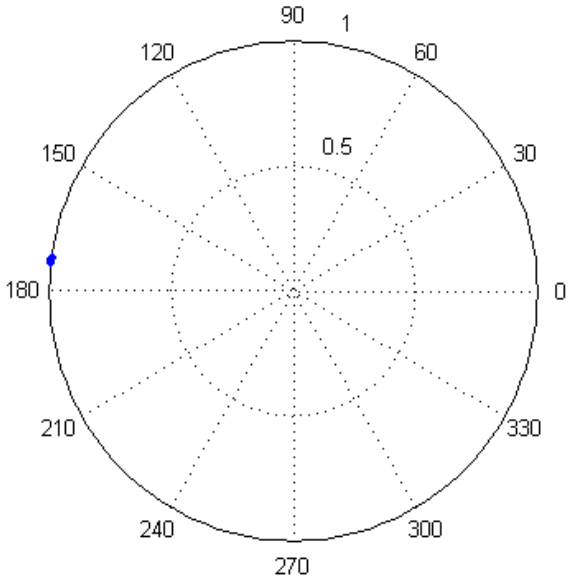

Figure 15. De-rotated phase of NCO output, f_n = 20 kHz.

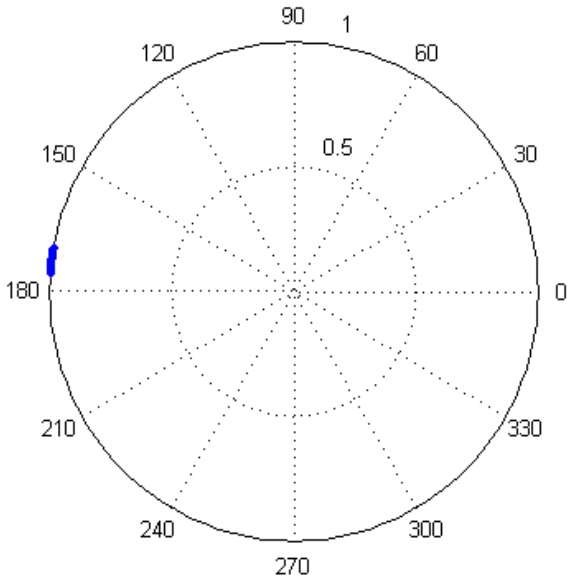

Figure 16. De-rotated phase of NCO output, f_n = 200 kHz.

References

1. Robertson, Neil, "Digital PLL's – Part 1", June, 2016
https://www.dsprelated.com/showarticle/967.php

2. Robertson, Neil, "Phase or Frequency Shifter Using a Hilbert Transformer", March, 2018,
https://www.dsprelated.com/showarticle/1147.php

3. Gardner, Floyd M., Phaselock Techniques, 3rd Ed., Wiley-Interscience, 2005, section 13.1.3.

4. Robertson, Neil, "Digital PLL's – Part 2", June, 2016
https://www.dsprelated.com/showarticle/973.php

5. Rice, Michael, Digital Communications, a Discrete-Time Approach, Pearson Prentice Hall, 2009, section 9.2.2.

6. Rice, section 9.4.

7. Kester, Walt, Ed., The Data Conversion Handbook, Newnes, 2005, Ch 2.
http://www.analog.com/en/education/education-library/data-conversion-handbook.html

8. Brannon, Brad, "Sampled Systems and the Effects of Clock Phase Noise and Jitter", Analog Devices Application Note AN-756, 2004
http://www.analog.com/media/en/technical-documentation/application-notes/AN-756.pdf

Neil Robertson May, 2018

Appendix Matlab time-domain model of PLL with external reference clock

```
% pll_ext_clk1.m       5/23/18 Neil Robertson

% Digital PLL model in time using difference equations.
% External reference clock and complex digital phase detector
% fn = 2 kHz   NCO initital freq error = -100ppm

N= 40000;                       % number of samples
fref= 6.3001e6;                 % Hz  external clock frequency
fs= 40e6;                       % Hz sample rate
Ts = 1/fs;                      % s  sample time

n= 0:N-1;                       % time index
t= n*Ts*1000;                   % ms

% 31-tap Hilbert xfmr
b= 2/pi * [-1/15 0 -1/13 0 -1/11 0 -1/9 0 -1/7 0 -1/5 0 -1/3 0 -1 0 1 0 ...
1/3 0 1/5 0 1/7 0 1/9 0 1/11 0 1/13 0 1/15];

b1= b.*blackman(31)';                       % windowed coefficients
b1= round(b1*2^12)/2^12;

b2= [0 0 0 0  0 0 0 0  0 0 0 0  0 0 0 1];    % delay of 15 samples (CT of
filter)

% Create reference signal

A = 1.0;                                        % reference signal amplitude
Vr= A*cos(2*pi*fref*n*Ts) + .0015*randn(1,N);   % ref signal with Gaussian
noise

nbits= 8;
Vr= floor(2^(nbits-1)*Vr)/2^(nbits-1);    % ADC quantization of ref signal

% Apply reference signal to Hilbert transformer
Ir= filter(b2,1,Vr);              % I channel HT output
Qr= filter(b1,1,Vr);              % Q channel HT output

% PLL

Knco= 1/4096;                     % NCO gain constant
KI= 6.4e-5;                       % loop filter integrator gain, fn= 2 kHz
KL= 0.41;                         % loop filter linear gain, fn= 2 kHz


fnco = fref*(1-100e-6);         % Hz NCO initial frequency
u(1) = 0;
int(1)= 0;
phase_error(1)= 0;
vtune(1) = 0;
```

```
% Compute difference equations
for n= 2:N;
    % NCO
    x = fnco*Ts + u(n-1) + vtune(n-1)*Knco;      % cycles  NCO phase
    u(n) = mod(x,1);                             % cycles  NCO phase mod 1
    u(n)= fix(2^20*u(n))/2^20;                   % quantized phase 20 bits
         Inco = cos(2*pi*u(n-1));
         Qnco = sin(2*pi*u(n-1));
         y(n)= round(2^11*Inco)/2^11;          % quantized NCO output 12 bits
    % Phase Detector
    pe= -Ir(n-1)*Qnco + Qr(n-1)*Inco;
    phase_error(n) = pe;

    % Loop Filter
    int(n) = KI*pe + int(n-1);                   % integrator
    vtune(n) = int(n) + KL*pe;                   % loop filter output
end


psd(Vr,2^12,fs/1e6)                            % plot spectrum at ADC output
xlabel('MHz'),ylabel('dB')
axis([0 20 -80 40]),figure

plot(t,phase_error),grid                       % plot phase detector output
axis([0 1 -1 1])
xlabel('t (ms)'),ylabel('phase error'),figure

plot(t,vtune),grid                             % plot loop filter output vtune
xlabel('t (ms)'),ylabel('vtune'),figure

psd(y(20000:end),2^12,fs/1e6)                % plot NCO cosine output spectrum
xlabel('MHz'),ylabel('dB')
axis([0 20 -80 40]),figure

psd(y(20000:end),2^14,fs/1e6)                  % plot close-in NCO spectrum
axis([fref/1e6-.2 fref/1e6+.2 -80 40])
xlabel('MHz'),ylabel('dB')
```