AN INTEGER-N CHARGE-PUMP PHASE-LOCKED LOOP WITH CONTROLLABLE PHASE OFFSET

BY

CHRIS YIM

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2019

Urbana, Illinois

Adviser:

Professor Pavan Kumar Hanumolu

ABSTRACT

In a phase-locked loop (PLL), the phase-offset is a result from non-idealities that usually need to be minimized to reduce the reference spurs at the output. However, in certain applications such as multi-buses, we would like to control this offset so we can align the clock with the data. This thesis will explore the idea of controlling the phase-offset at the output of a charge-pump PLL (CPLL) by setting the current mismatch in the charge-pump module.

To my parents, for their love and support.

ACKNOWLEDGMENTS

I would like to thank Prof. Hanumolu for introducing me to the field of analog integrated circuits through his courses and giving me the opportunity to join his research group. The last two years have been a great learning experience overall.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Motivation

Phase-locked loops (PLLs) are widely used in almost every application needing a clock. These PLLs are most commonly implemented as a charge-pump PLL (CPLL) because of its many advantages. Most of the focus on CPLLs has been to minimize the phase offset since that minimizes the reference spur at the output of the PLL. However, there may be applications where one needs to control the output phase offset. Instead of having a delay-locked loop (DLL) to add this delay, it is definitely possible to change the phase offset of the CPLL directly.

The focus of this thesis is to introduce a CPLL architecture that can be used for adding in phase offset while having small reference spurs at the output.

1.2 Outline

The chapters of this thesis are organized as follows:

- Chapter 2 reviews the standard design procedure for an integer-N C-PLL.
- Chapter 3 covers the causes of phase offset.
- Chapter 4 briefly explains the sample and hold loop filter.
- Chapter 5 shows the CPLL architecture and gives the results.
- Chapter 6 explains the improvements needed for this CPLL.

CHAPTER 2

DESIGN PROCEDURE OF AN INTEGER-N CPLL

2.1 Review of CPLL Basics

2.1.1 Introduction

The design of an integer-N CPLL has been studied and done many times in the past. For this reason, this chapter is as brief as possible. For more details, you may refer to [1] and [2]. Furthermore, the focus of this chapter will be on the design of a type II third-order PLL since it is most commonly used.

2.1.2 PLL Functionality

A phase-locked loop (PLL) is, simply put, a loop used to lock the phase. In other words, a PLL ensures a fixed relationship between the phase of the input and the output.

By locking the phase of the output signal, it is effectively also locking the frequency of the output. Most of the time, PLLs are used to generate a higher-frequency output clock based off a reference signal.

The reader may wonder why we use PLLs rather than frequency-locked loops (FLLs) if we only care about the frequency at the output. PLLs have many advantages over FLLs. A main advantage is that the phase is tracked perfectly. In other words, the duty cycle is also fixed.

2.1.3 Integer-N CPLL Small-Signal Model

The small-signal model of an integer-N CPLL is shown in the Fig. 2.1. Understanding this model is the key for determining the parameters for the

CPLL.

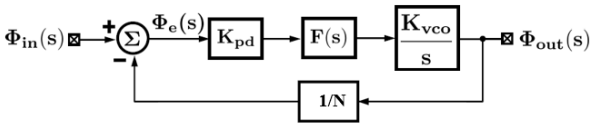

Figure 2.1: PLL Small-Signal Model

The variable used for this loop is phase, for the reasons given previously. The phase-frequency detector (PFD) determines the phase error between $\Phi_{in}(s)$ and $\Phi_{out}(s)$. This phase error is converted into current through the charge-pump (CP) block. The current is passed through the loop filter which integrates the error while also converting the signal into voltage. This voltage is fed to the voltage-controlled oscillator (VCO) which outputs the corresponding frequency and phase at the output which would minimize the error in the loop. This phase will be N-times greater than the phase at the input, due to the divider in the loop.

Next, the loop gain for this PLL can be written to analyze its small-signal behavior.

$$LG(s) = \frac{K_{vco} I_{cp}}{N 2\pi} F(s) \tag{2.1}$$

where for a Type II third-order PLL:

$$F(s) = \frac{s + \omega_z}{C_2 s(1 + \omega_p)}$$

and

$$\omega_z = \frac{1}{RC_1}$$

$$\omega_p = \frac{1}{R(C_1 || C2)}$$

Figure 2.2 shows the loop gain response.

2.1.4 PLL Bandwidth

One of the most important parameters for the PLL is the bandwidth (BW). This bandwidth corresponds to the modulation frequency at which the PLL

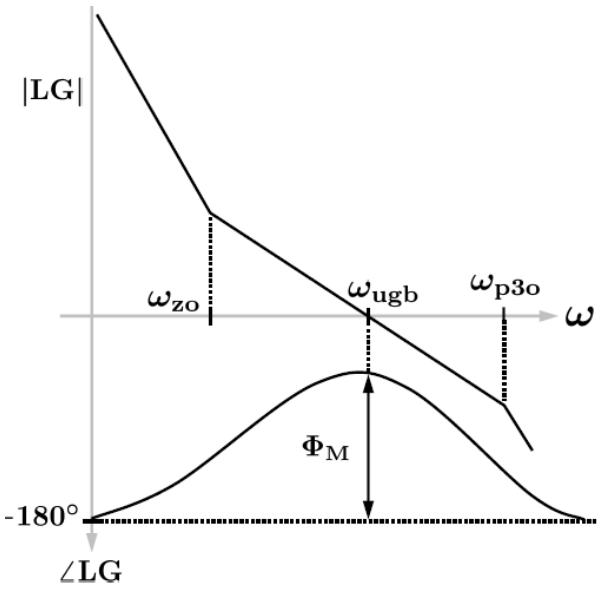

Figure 2.2: LG Frequency Response

begins to lose lock with the changing reference. It also determines the speed at which the PLL can lock.

Furthermore, a larger PLL BW minimizes more VCO noise, whereas a smaller PLL BW minimizes the reference noise (PFD/CP). A smaller PLL BW also minimizes the reference spurs at the output as well.

2.1.5 CPLL Design Procedure

The PLL design must be stable, or in other words have a good phase margin. To maximize the phase margin, one can write out equations using knowledge of the zero and the poles in the system.

The design procedure for an integer-N CPLL is given as follows:

1. Start with an approximation of PLL BW ω_{ugb} and phase margin Φ_M
2. $$K_c = \frac{C1}{C2} = 2(tan^2\Phi_M + tan\Phi_M\sqrt{tan^2\Phi_M + 1})$$
3. $$\omega_z = \frac{\omega_{ugb}}{\sqrt{\frac{C1}{C2} + 1}}$$

4. Choose R for low-noise

$$C_1 = \frac{1}{\omega_z R}$$

$$C_2 = \frac{C_1}{K_c}$$

5. Use an estimate of K_{vco}

$$I_{cp} = \frac{2\pi C_2 N}{K_{vco}} \omega_{ugb}^2 \sqrt{\frac{\omega_{p3}^2 + \omega_{ugb}^2}{\omega_z^2 + \omega_{ugb}^2}}$$

In summary, in this design procedure, entering in the PLL bandwidth, phase margin, K_{vco} and resistor (in the loop filter) gives the necessary C_1, C_2, and I_{cp}.

CHAPTER 3

PHASE OFFSET

3.1 Analysis

3.1.1 Equations for Phase Offset

The phase offset at the output of the PLL mainly comes from the mismatch in the values of the current pulses in the charge pump block.

For example, take the case where $I_{DN} > I_{UP}$, as shown in waveform in Fig. 3.1.

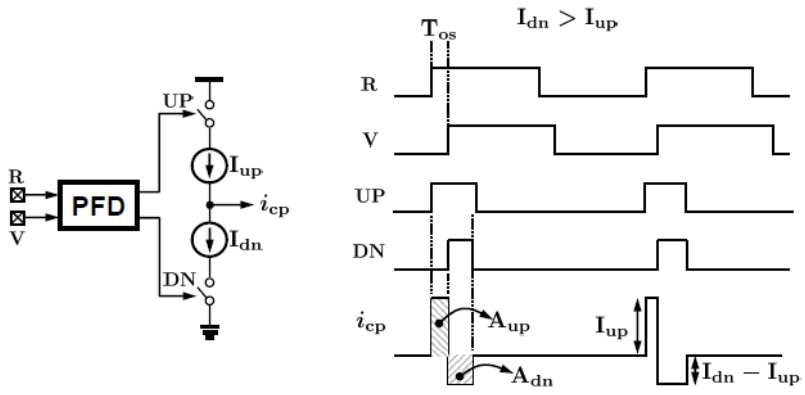

Figure 3.1: Waveform of Charge-Pump Mismatch

When the PLL is locked, there will be no net charge deposited into the capacitor in the loop filter. In other words, the areas of the current must sum to be zero. This can simply be written as the relationship:

$$I_{UP}T_{os} = (I_{DN} - I_{UP})T_{rst} \tag{3.1}$$

where T_{rst} is the reset delay in the PFD block.

The time offset T_{os} can be converted into phase by multiplying by $2\pi/T_R$ where T_R is the reference period. This gives the equation:

$$\Phi_{os} = 2\pi \frac{T_{rst}}{T_R} \frac{I_{DN} - I_{UP}}{I_{UP}} \tag{3.2}$$

Similarly, for $I_{UP} > I_{DN}$:

$$\Phi_{os} = 2\pi \frac{T_{rst}}{T_R} \frac{I_{UP} - I_{DN}}{I_{DN}} \tag{3.3}$$

3.1.2 Phase Offset Dependence on PFD Reset Delay

As shown in Equations 3.2 and 3.3, the phase offset can mainly be controlled by the magnitude of the I_{UP} and I_{DN} currents. However, the magnitude of the phase offset also depends on the T_{rst} in the PFD block, which is going to be fixed in the standard design.

With a larger T_{rst}, the output phase offset can reach its bounds with a smaller difference in I_{UP} and I_{DN}. On the other hand, it would mean that there will be less precision in the phase offset (assuming that the currents are controlled discretely). Furthermore, a larger T_{rst} would also mean more current noise from the CP block.

Therefore, when selecting a T_{rst}, a small T_{rst} of 200 ps was chosen.

CHAPTER 4

SAMPLE AND HOLD FILTER

4.1 Description

The sample and hold filter (S/H-filter) can be used to reduce the reference spurs at the output. This can be done with the schematic shown in Fig. 4.1.

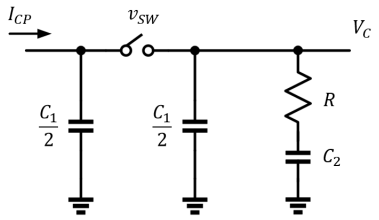

Figure 4.1: S/H Loop Filter Schematic

The switch is left open (hold) while the current is non-zero. Once the voltage before the switch is stable, the switch is closed (sample). By doing this, the voltage into the VCO ideally should not see any fluctuations, and all the reference spurs will be removed.

In the locked-state for the PLL, the net charge to the capacitor will still be zero.

4.2 Simulation

To verify the functionality, we created a simulink model for the integer-N CPLL with a S/H loop filter, as shown in Fig. 4.2. Specific details on the implementation is given in Appendix B.

A comparison of the VCO control voltage Vc is given in Fig. 4.3. The sample and hold loop filter removes the large ripple as expected.

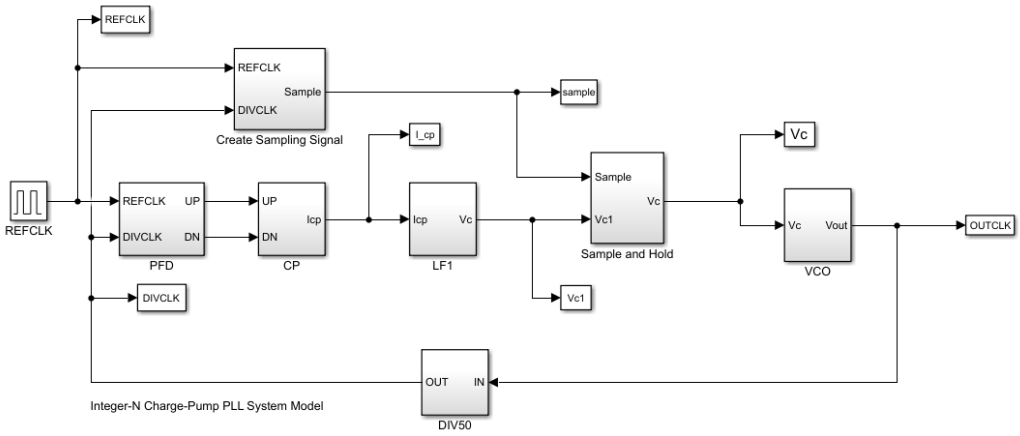

Figure 4.2: Integer-N CPLL with S/H Loop Filter

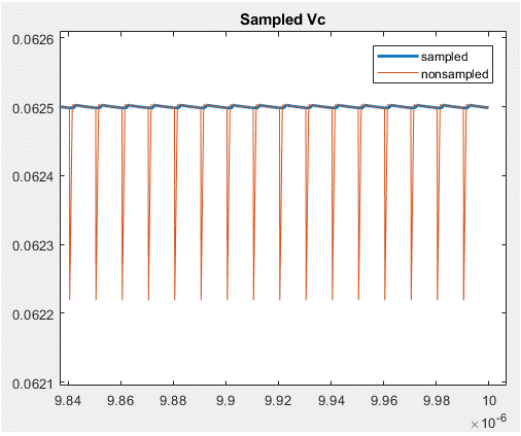

Figure 4.3: Effect of S/H on Vc

CHAPTER 5

RESULTS

5.1 Specific Modules

5.1.1 PFD

The phase-frequency detector is used to convert the phase error between the REFCLK and DIVCLK into voltage.

The schematic of the PFD is shown in Fig. 5.1. As described in Chapter 3, the reset delay was set at around 217 ps to allow for precision and less current noise. The two inverters shown in the transistor level are used to set the delay.

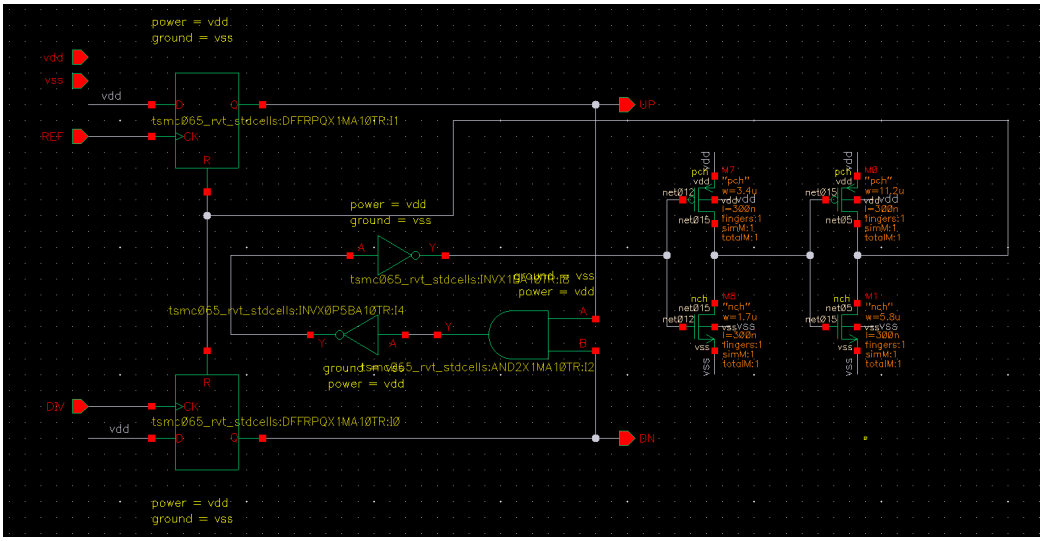

Figure 5.1: PFD Schematic

A sample waveform of the PFD is shown in Fig. 5.2. As expected, UP pulse (green) is triggered by the rising edge of REFCLK (purple), and DN pulse (orange) is triggered by the rising edge of DIVCLK (red). The DN

pulse shows that the reset delay is around 217 ps.

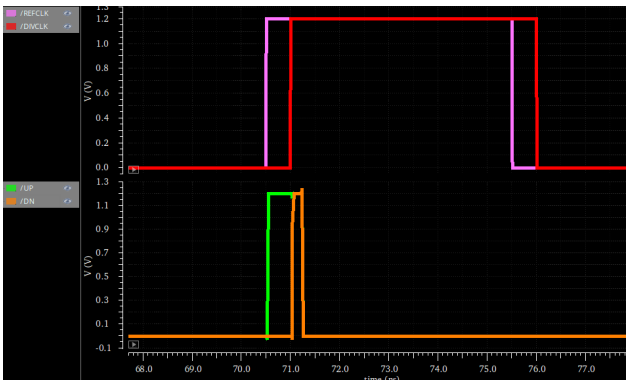

Figure 5.2: PFD Input and Output Voltage Waveforms

5.1.2 PFDBUF

The PFD buffer (PFDBUF) is used to generate differential signals in UPB and DN. Without this block, there would be a mismatch the delay, leading to unwanted phase offset.

The schematic of the module is shown in Fig. 5.3. First, there are four inverter stages because this made matching the delays easier. This is because total delay of this four inverter stage is $2\tau_{p,LH} + 2\tau_{p,HL}$. The transmission gate was simply matched to the delay of one of the inverter stages.

An example output waveform of the PFDBUF is shown in Fig. 5.4. The intersection of the waveform was designed to be around 0.6 V.

5.1.3 CP

The CP block is used to convert the voltage signals from the PFD into currents. This module is needed to create different magnitudes for the UP and DN signals. The architecture for this design is based on the one in [3]. The schematic for the CP block is shown in Fig. 5.5.

This module was designed by putting an ideal voltage source at 946 mV, which was the expected voltage for the VCO to generate 5 GHz. The sizes were also minimized to reduce the parasitic capacitance so that the current waveforms would have fewer spikes from switching.

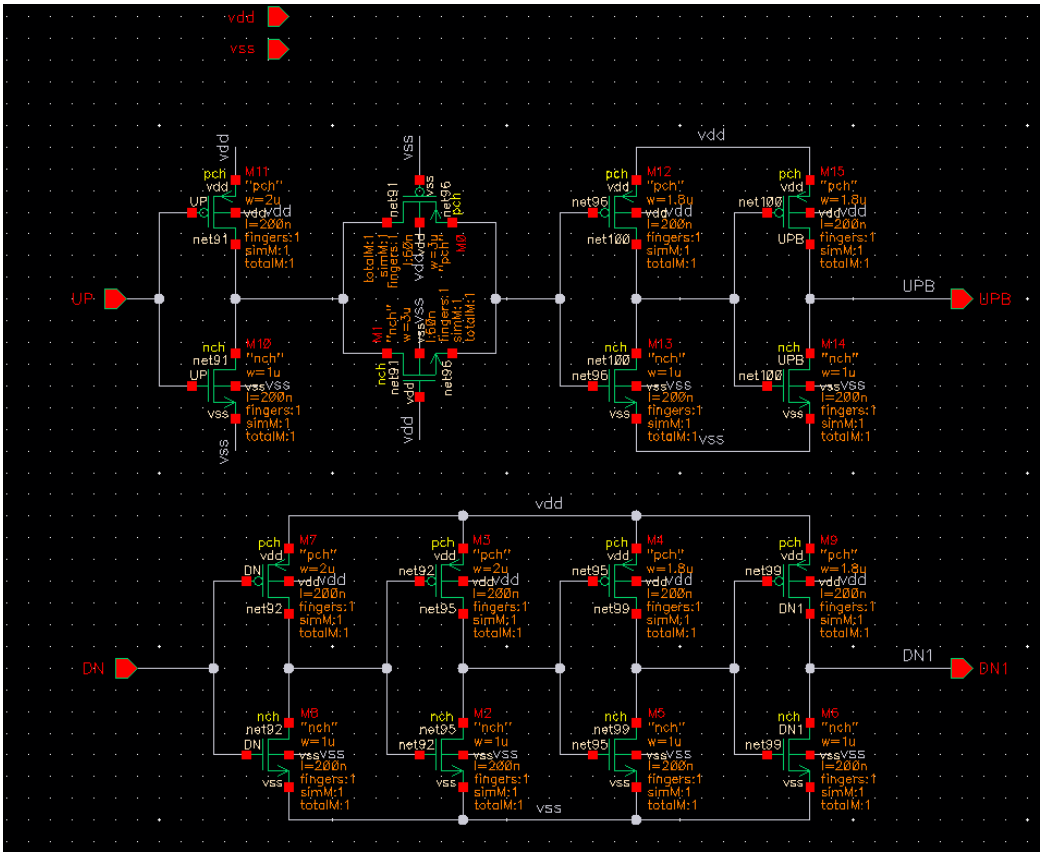

Figure 5.3: PFDBUF Schematic

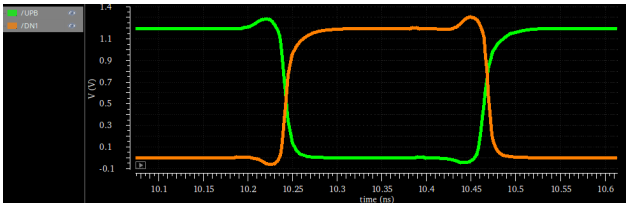

Figure 5.4: PFDBUF Output Voltage Waveform

An example output waveform is shown in Fig. 5.6. The currents are ideally the same, but due to differences in the PMOS and NMOS transistors there is some mismatch.

The currents are also tunable with approximately 1 uA precision. The outputs of the tuned UP currents are shown in Fig. 5.7, and the DN currents are shown in Fig. 5.8.

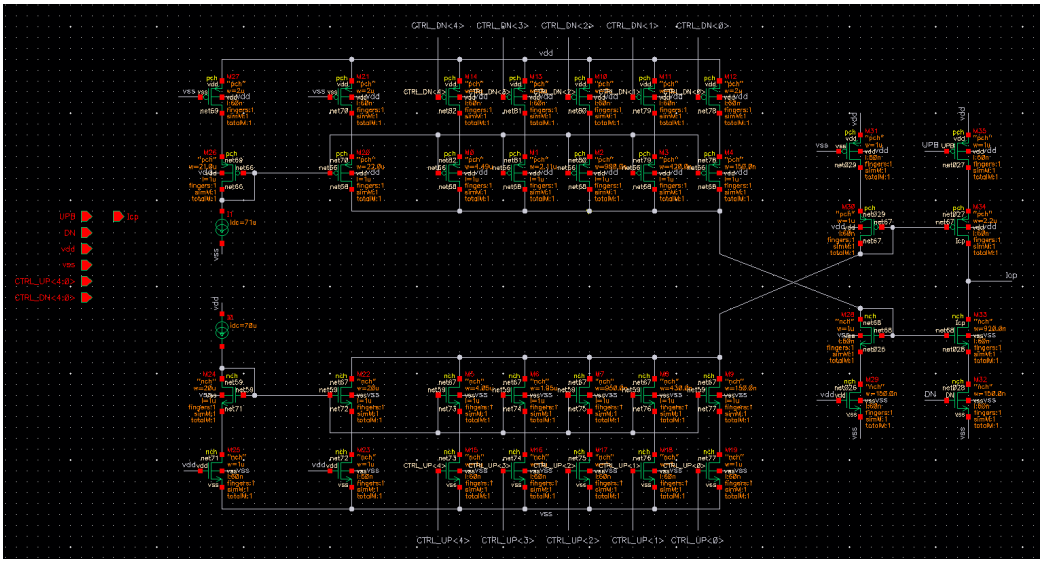

Figure 5.5: CP Schematic

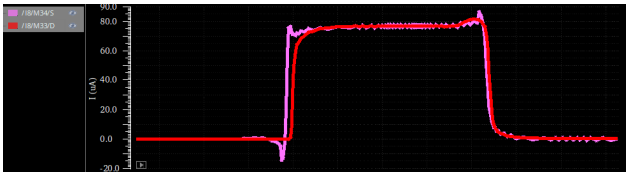

Figure 5.6: CP Output Current Waveform

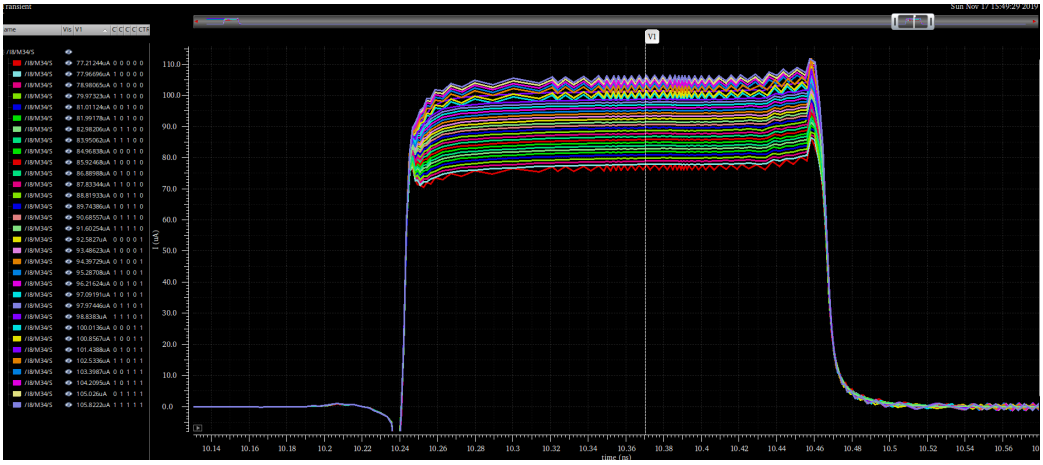

Figure 5.7: CP UP Current Waveforms

5.1.4 VCO

The top-level schematic of the VCO is shown in Fig. 5.9.

A five-stage single-ended ring oscillator was chosen as the VCO. This architecture was chosen for its simplicity, as the focus of this research was on

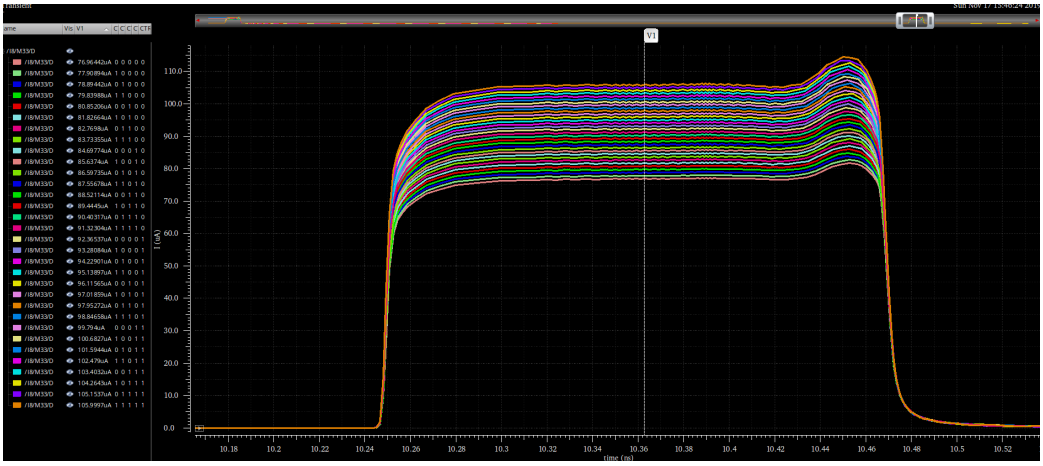

Figure 5.8: CP DN Current Waveforms

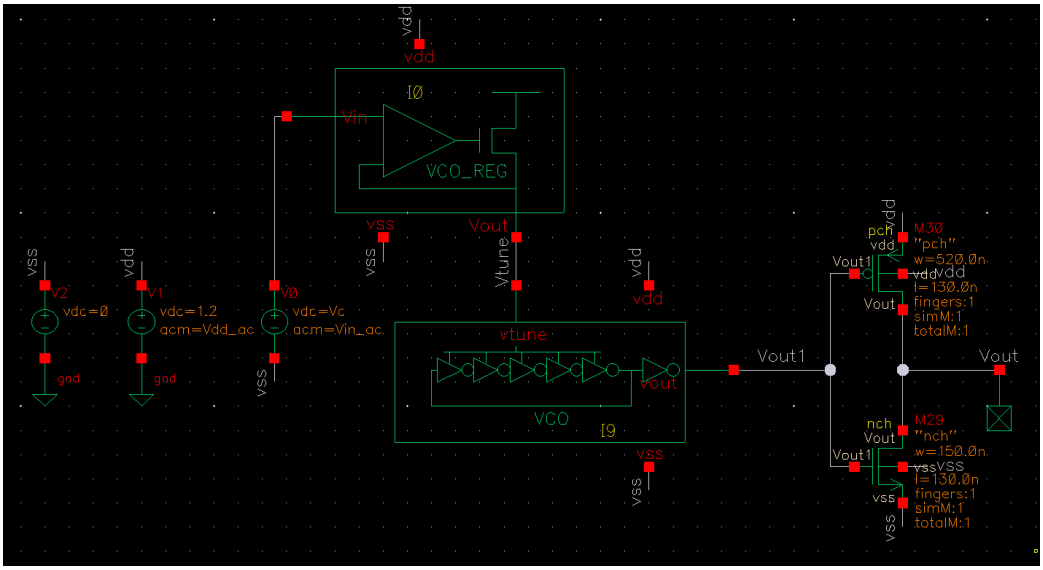

Figure 5.9: VCO Top-Level Schematic

the charge-pump module not the VCO.

In general, compared to LC-oscillators, ring oscillators have a wider tuning range and less area but have worse phase noise. Ring oscillators also can easily generate multi-phase outputs, which is necessary in many clock-data recovery (CDR) circuits.

The schematic for the ring oscillator is shown in Fig. 5.10. Note that the level shifter was simply comprised of inverters. This is not the optimal design, but it is functional.

Figure 5.11 gives the plot of voltage tuning range versus frequency. The VCO tuning voltage ranges from 0.6 V to 1.2 V, and the output frequency

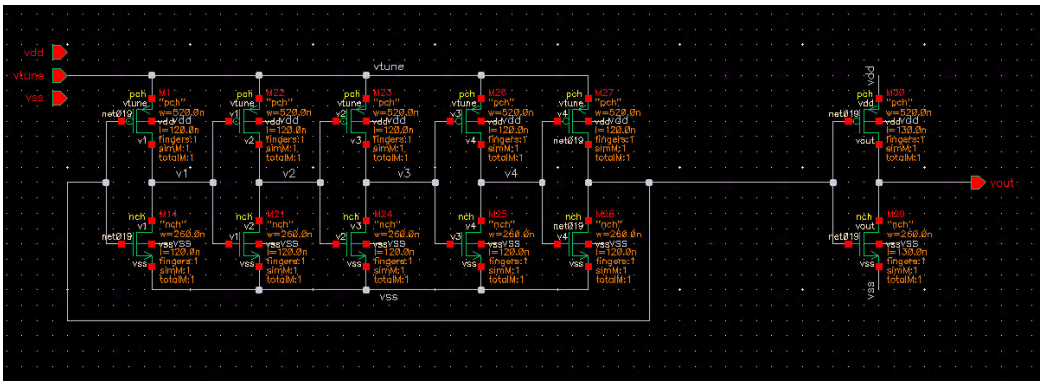

Figure 5.10: Ring Oscillator Schematic

ranges from 1.7 GHz to 6.2 GHz. The K_{VCO} near the output frequency of 5 GHz is 8.177 GHz/V.

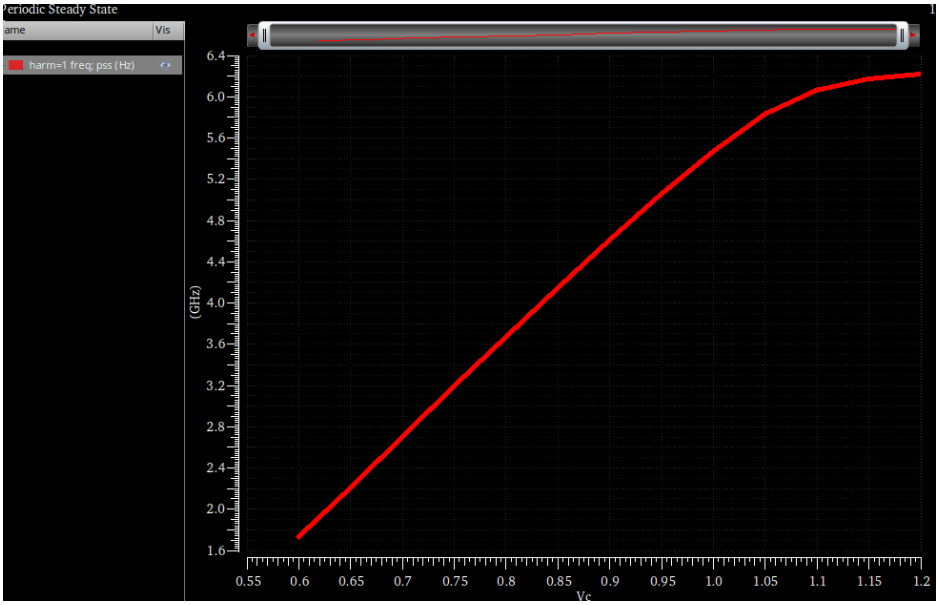

Figure 5.11: VCO Frequency vs. Voltage

The regulator for the VCO was a simple unity gain buffer configuration with a five-transistor operational transconductance amplifier (OTA) cascaded with a PMOS pass transistor. We chose a PMOS pass transistor since the expected voltage was going to be high, therefore we had to use an NMOS input OTA to reduce the power supply rejection. The schematic for this module is shown in Fig. 5.12.

The constraints on this module were that the bandwidth had to be much larger than the PLL bandwidth, and that the PMOS pass transistor must be

able to supply enough current to the ring oscillator.

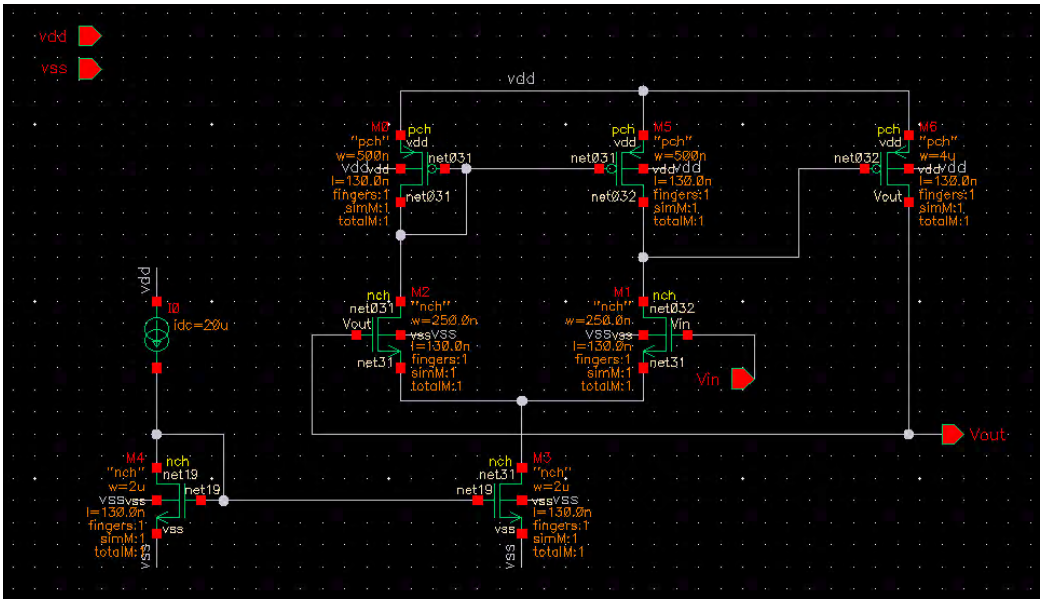

Figure 5.12: VCO Regulator Schematic

5.1.5 DIV

The divide-by-50 block (DIV) was simple to design because it is completely digital. It is created by cascading three divider blocks, a divide by two, divide by five, and divide by five, as shown in Fig. 5.13. The divide by two block comes first, since it is a simpler circuit and would operate more robustly at higher frequencies.

The divide by five circuit architecture is from USA Patent # US4703495A [4]. The schematic is shown in Fig. 5.14.

Figure 5.15 shows the transient waveform for the output of the DIV block with an ideal input clock of 5 GHz. The output has frequency of 100 MHz as expected.

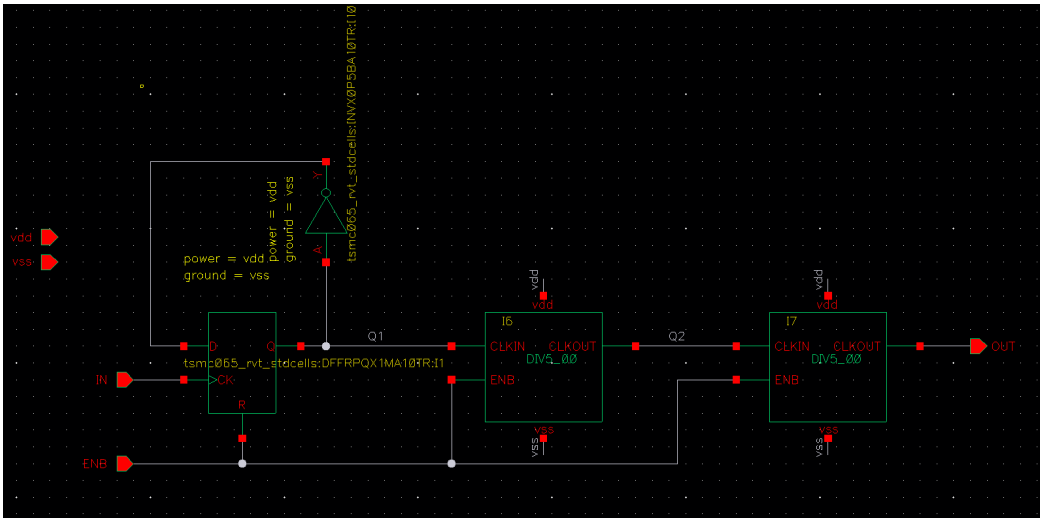

Figure 5.13: DIV Schematic

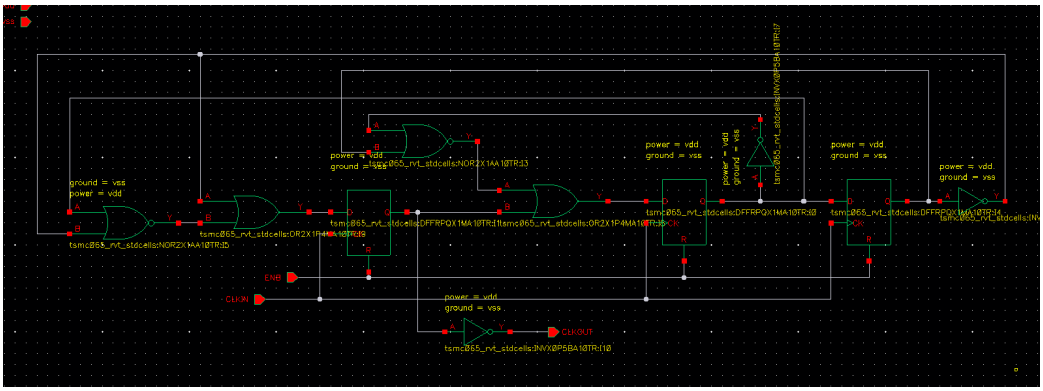

Figure 5.14: Divide by 5 Circuit Schematic

5.2 PLL

5.2.1 Schematics

The schematic of the entire PLL is shown in Fig. 5.16. The input and output nodes of the VCO regulator are initialized high for faster locking. The sizes of these switches are minimized to avoid affecting the loop filter capacitance.

5.2.2 Transient Simulation

A transient simulation of the entire PLL was first done to ensure that it would lock. As shown in Fig. 5.17, the PLL locks in about 1.2 μs. The input

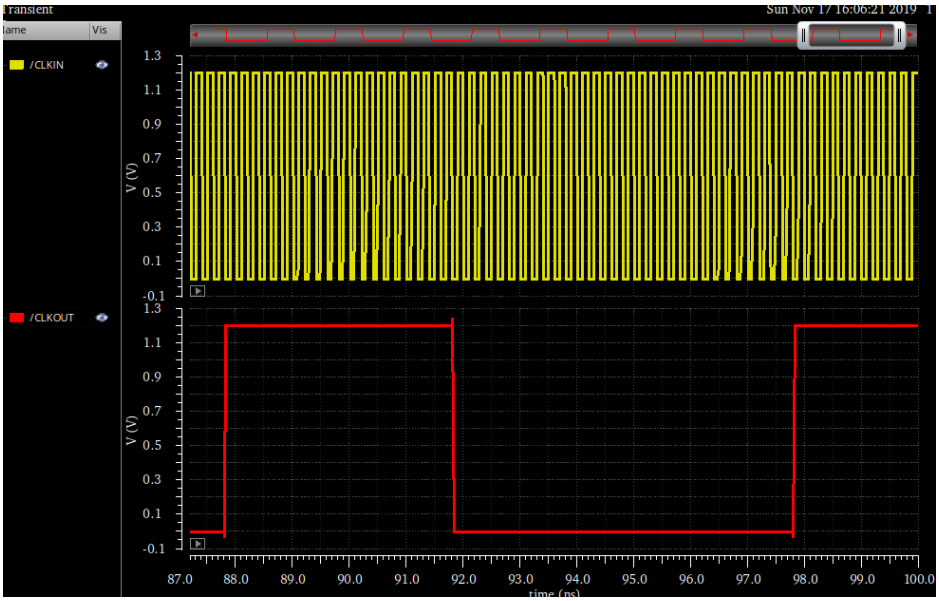

Figure 5.15: DIV Input vs. Output Waveform

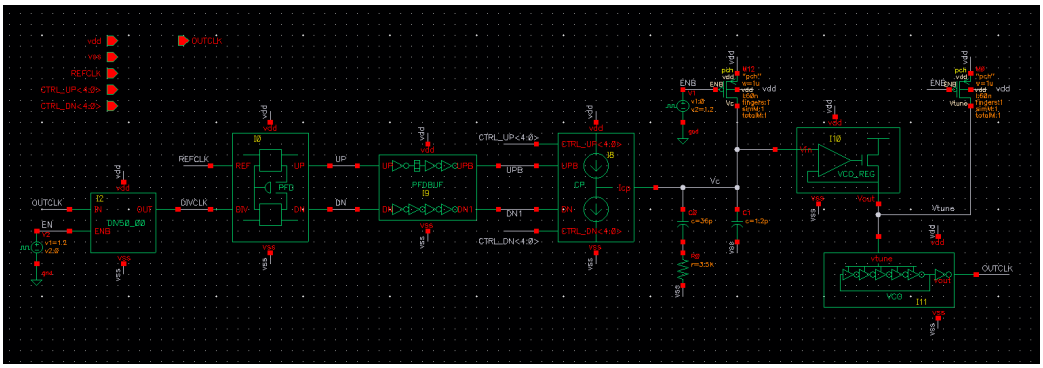

Figure 5.16: PLL Schematic

to the VCO regulator (red) has voltage ripples from the phase offset when locked. The output of the VCO regulator (green) is actually oscillating due to the current drawn from the ring oscillator.

The locked condition was verified by calculating the area of the currents to ensure that it was zero.

The output transient waveforms of the OUTCLK (red) and REFCLK (yellow) are shown along with UP current (green) and DN current (blue). Figure 5.18 shows that the UP and DN current pulses are closely matched, meaning that the rising edge of the OUTCLK is very close to REFCLK. To be exact, it is measured to be 0.4 ps apart at 0.6 V.

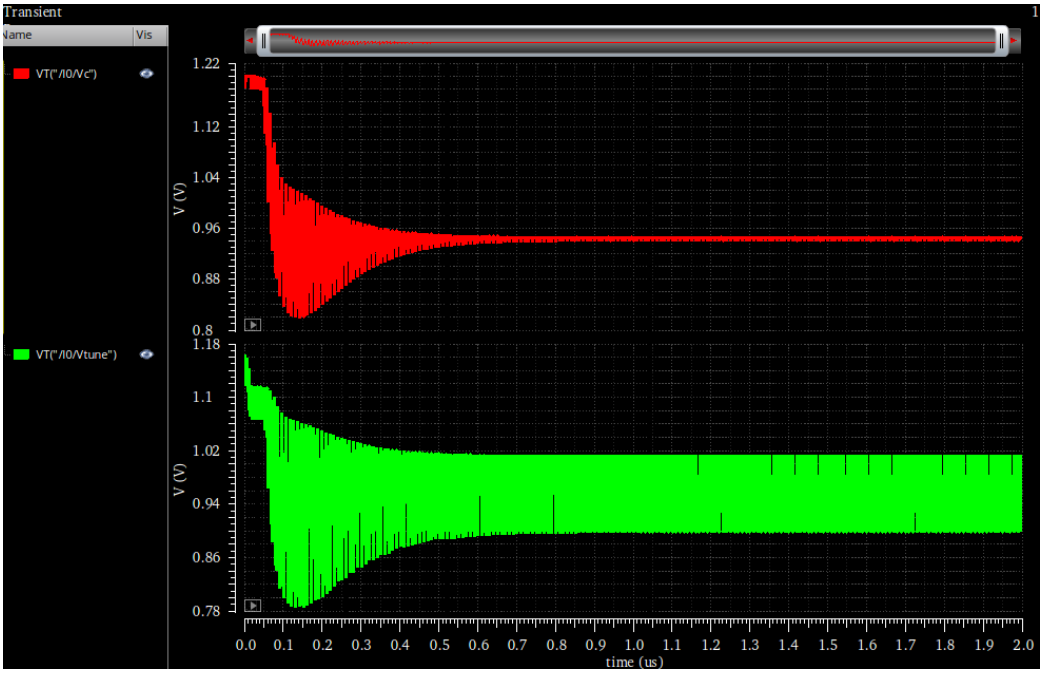

Figure 5.17: PLL Vtune Transient Waveform

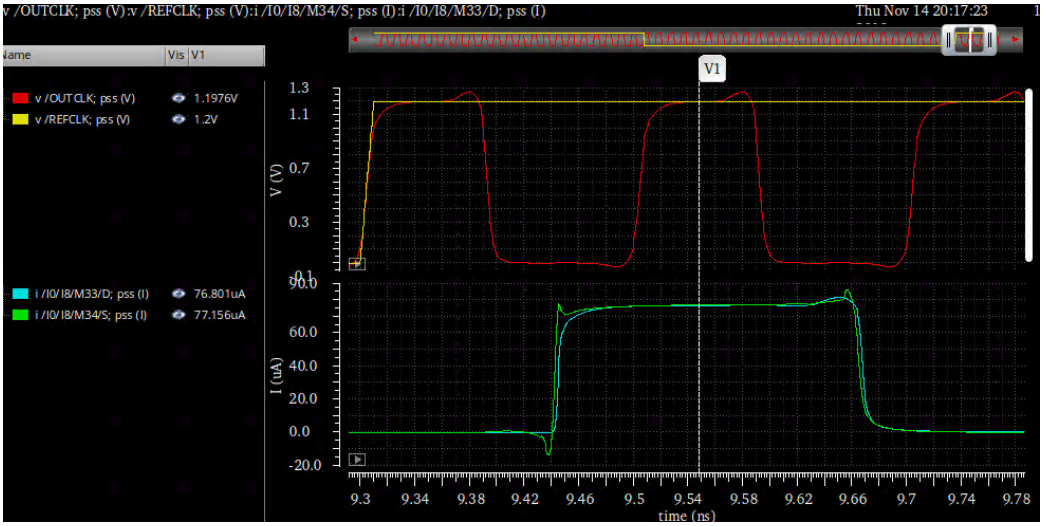

Figure 5.18: PLL Transient Waveform

5.2.3 Verification of Phase Offset

To verify that the phase offset would change as expected, a PSS analysis was run rather than a transient analysis since it is much faster.

The PLL output waveform is shown for different codes in Fig. 5.19. As expected, the phase offset changed accordingly. For example, for $UP = 00000$ (77.2 μ A) and $DN = 01000$ (84.7 μ A), the calculated time offset is 19.25

ps. The measured time offset is approximately 20 ps as well.

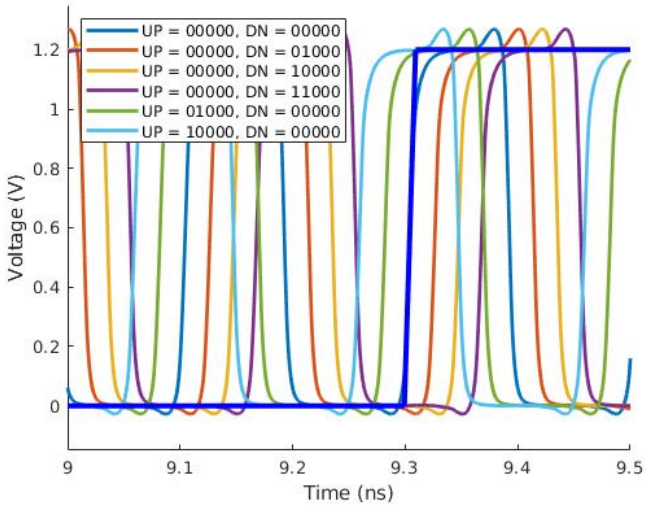

Figure 5.19: PLL Transient Waveform for Different Digital Codes

The bounds of the phase offset were tested by using the maximum DN current (blue) and maximum UP current (red) as shown in Fig. 5.20. The blue curve is delayed by 80 ps, and the red curve is ahead by 80 ps, which was also close to the expected results.

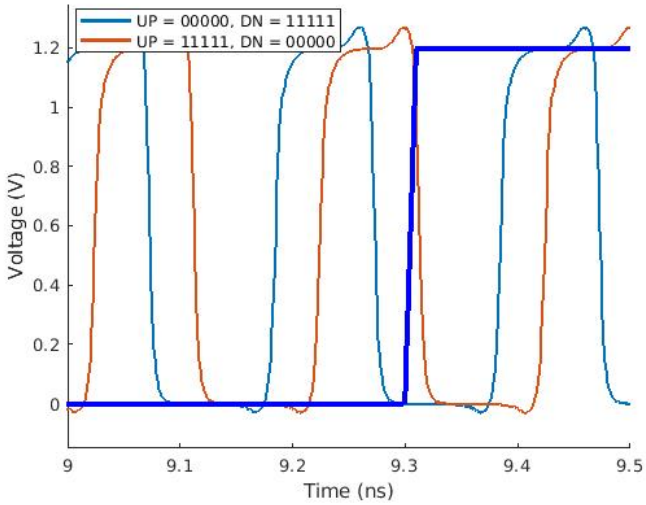

Figure 5.20: PLL Transient Waveform for Maximum Phase Offset

5.2.4 Output Phase Noise

The output phase noise shape was as expected (shown in Fig. 5.21). At lower frequencies, it is dominated by the noise contribution from the CP, and at higher frequencies it is dominated by the VCO noise.

The output phase noise was matched up to the expected in MATLAB as well. There are some differences due to differences in values.

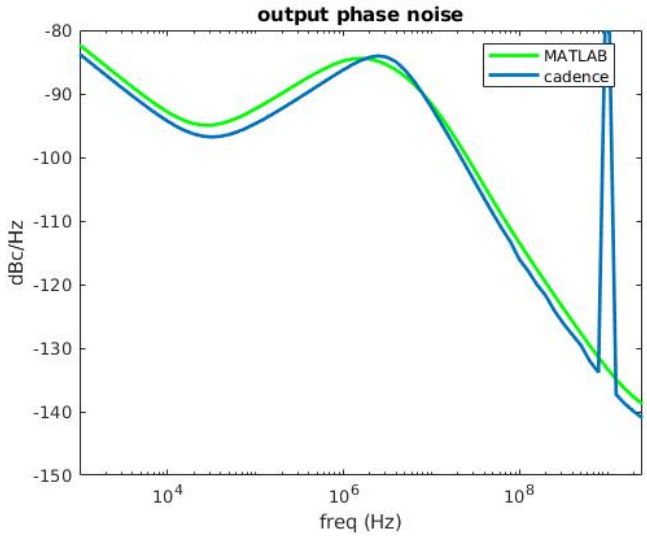

Figure 5.21: PLL Output Phase Noise

The root mean square (RMS) jitter from integration bounds of 1 kHz to 2.5 GHz is 5.13 ps. A bound of 2.5 GHz is used because it is Nyquist frequency, and a lower bound of 1 kHz is used as a conservative measure (it should most likely be around 10 kHz or more).

CHAPTER 6

FUTURE WORK

6.1 Improvements

Although the surface level functionality of changing the phase offset of a PLL by controlling the charge-pump current has been explored, there are many improvements that are needed.

One of the main advancements is implementing a S/H-filter, as shown in Chapter 4. Without it, as the phase offset increases, the reference spurs at the output would increase as well.

Another development would be reducing the VCO noise contribution by increasing the bandwidth. For the ideal PLL output phase noise plot, there should not be any peaking.

Other improvements would be changing the architecture of miscellaneous blocks. The ring oscillator, for example, is single-ended and differential-ended would have better performance. Furthermore, the architecture for the level-shifter at the output of the ring oscillator should be changed.

REFERENCES

[1] P. K. Hanumolu, "Lecture notes: Ece 599 phase locked loops," Oregon State University, Fall 2011.

[2] "First time, every time practical tips for phase locked loop design," 2009. [Online]. Available: http://www.ctan.org/tex-archive/macros/latex/contrib/supported/IEEEtran/

[3] Y. Liu, W. Rhee, D. Friedman, and D. Ham, "All-digital dynamic self-detection and self-compensation of static phase offsets in charge-pump plls," in *IEEE International Solid-State Circuits Conference (ISSCC'07)*, San Francisco, CA, Feb. 2007.

[4] B. J. Bereznak, "High speed frequency divide-by-5 circuit," U.S. Patent 4 703 495, Oct 27, 1987.

APPENDIX A

NOISE

A.1 MATLAB Code

```
%% Find loop parameters for CPLL
% Inputs: Bandwdith wc, Phase-margin pm, Divider ratio N,
%         Resistor R, VCO gain Kvco
% Outputs: Chargepump current Icp, C1, C2

clear all; clc;
% Loop parameters
wc = 2 * pi * 7e6; % PLL bandwidth
pm = pi * (70/180);
N = 50;
Kvco = 8.785226e9;
Kvco_rad = 2 * pi * Kvco;
R = 3.5e3; % From Noise Limitation

% Determine Icp, C1, C2
Kc = (2*(tan(pm)^2)) + (2*tan(pm)*sqrt((tan(pm))^2+1));
wz = wc/sqrt(Kc+1);
C1 = 1/(R*wz);
C2 = C1/Kc;
wz = 1/(R*C1); % LF zero
wp = (C1+C2)/(R*C1*C2); % LF pole due to C2
Icp = (2*pi*C2*N/Kvco_rad) * wc^2 * (sqrt(wc^2 + wp^2)/sqrt(wc^2 + wz^2))
```

```
% Noise Transfer Functions -------
Kpd_cp = Icp/(2*pi);
LF_TF = tf([1 wz],[C2 C2*wp 0]);
VCO_TF = tf([1],[1 0]); % VCO Transfer Function
LG = Kvco_rad * Kpd_cp * LF_TF * VCO_TF/N;

Hin = LG * N/(1 + LG); % Input
Hicp = Hin/Kpd_cp; % Chargepump Current
Hr = Kvco_rad * VCO_TF / (1 + LG); % Loop filter resistor
Hvco = 1/(1 + LG); % VCO

%% Import Data
Icpn_pss = csvread('PFDCP_02.csv',1,0);
Icpn_rms = Icpn_pss(:,2); % A/sqrt(Hz)
F = Icpn_pss(:,1); % Frequency vector 1kHz to 2.5GHz
Sicp = Icpn_rms .* Icpn_rms;

kT = 1.3806503e-23 * 300;
Sr = 4*kT*R;

% PN = csvread('VCO_only_00.csv',1,0);
PN = csvread('VCO_reg_02.csv',1,0);
SvcodB = PN(:,2); % dBc/Hz
Svco = 10 .^ (SvcodB/10) * 2; %% "2" to make it 2-sided PSD

div_noise = csvread('DIV_00.csv',1,0);
div_rms = div_noise(:,2); % V/sqrt(Hz)
Sdiv = div_rms .* div_rms;
```

```
[mag, ph] = bode(Hin,2*pi*F);
Hin2(1:length(F)) = mag(1,1,1:length(F)) .* mag(1,1,1:length(F));
[mag, ph] = bode(Hicp,2*pi*F);
Hicp2(1:length(F)) = mag(1,1,1:length(F)) .* mag(1,1,1:length(F));
[mag, ph] = bode(Hr,2*pi*F);
Hr2(1:length(F)) = mag(1,1,1:length(F)) .* mag(1,1,1:length(F));
[mag, ph] = bode(Hvco,2*pi*F);
Hvco2(1:length(F)) = mag(1,1,1:length(F)) .* mag(1,1,1:length(F));

Stot_icp = Sicp .* Hicp2';
Stot_r = Sr .* Hr2';
Stot_vco = Svco .* Hvco2';
Stot_div = Sdiv .* Hin2';
Stot = Stot_icp + Stot_r + Stot_vco + Stot_div;  % Integrated noise

%% Calculate jitter
Fref = 100e6;
Freq_jitter = F(1:end-1); %% Remove artifacts of PNOISE simulation
Stot_jitter = Stot(1:end-1);
Stot_integ = trapz(Freq_jitter,Stot_jitter);
RMS_jitter_rad = sqrt(Stot_integ); %% in Radians
Tvco = 1/(N*Fref);
RMS_jitter_sec = RMS_jitter_rad/(2*pi) * Tvco %% in ps
RMS_jitter_deg = RMS_jitter_rad/(2*pi) * 360; %% in deg
```

A.2 Results

The noise contributions from the VCO is shown in Fig. A.1. The noise is high-pass filtered to the output of the PLL, which is why the CP module contributes the most noise at lower frequencies.

The output noise contribution from each module is shown in Fig. A.2.

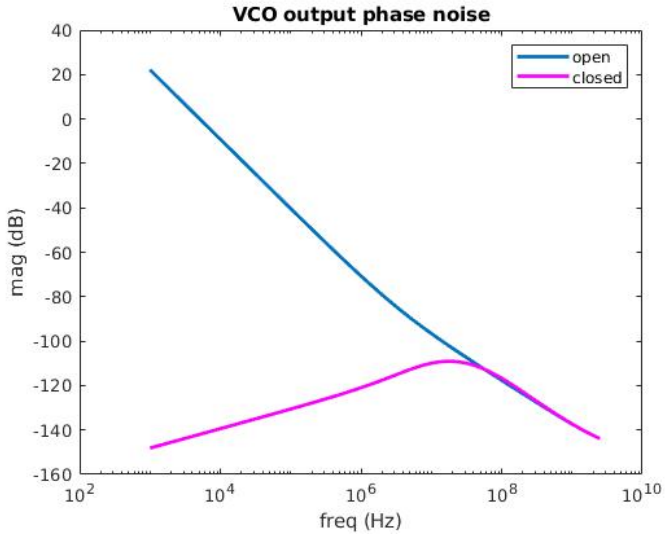

Figure A.1: VCO Open Loop vs. Closed Loop Noise

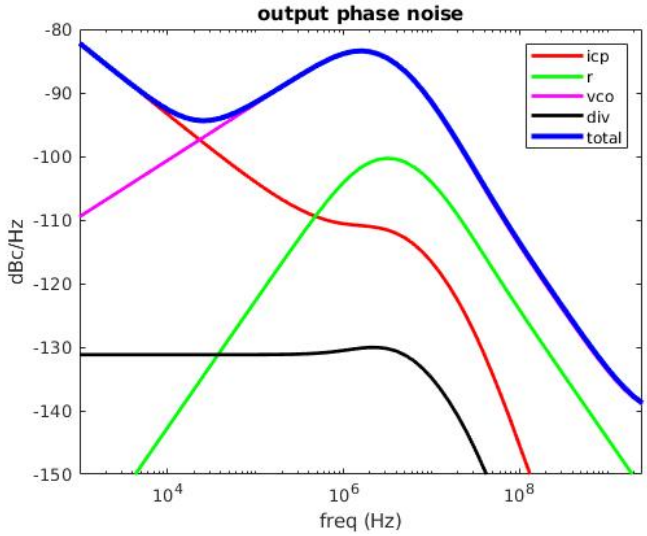

Figure A.2: Output Noise Contributions

APPENDIX B

SIMULINK PLL MODEL DETAILS

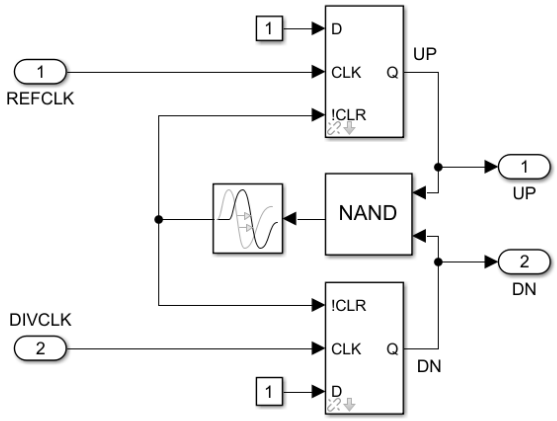

Figure B.1: PFD Simulink Model

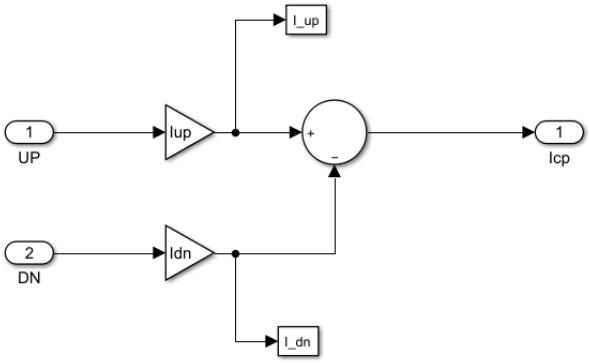

Figure B.2: CP Simulink Model

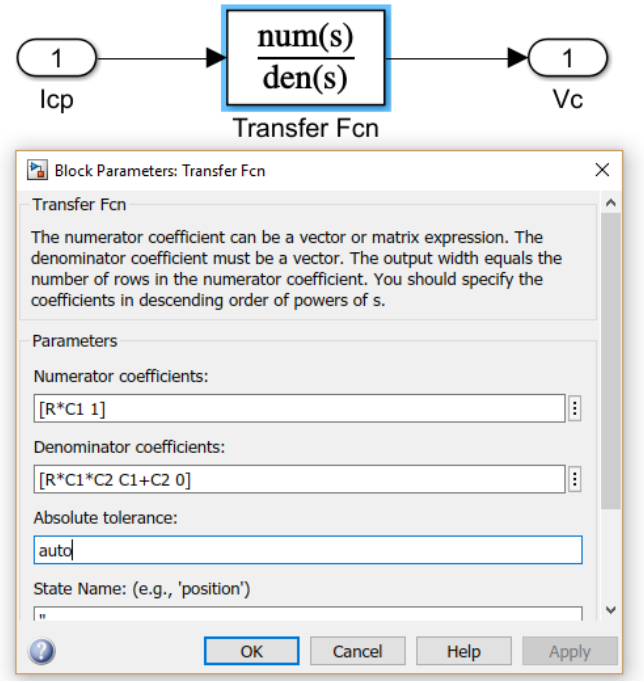

Figure B.3: LF Simulink Model

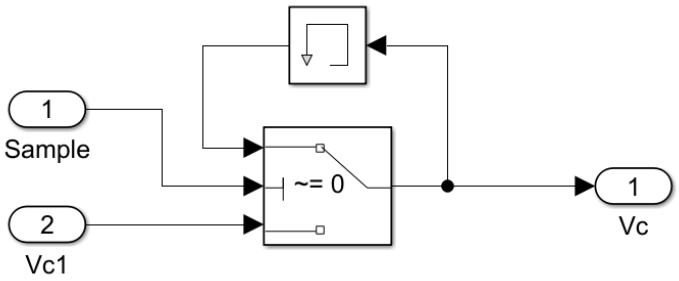

Figure B.4: Sample and Hold Simulink Model

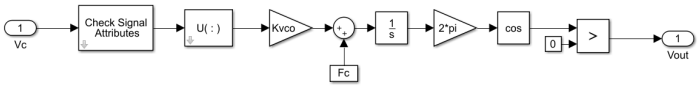

Figure B.5: VCO Simulink Model

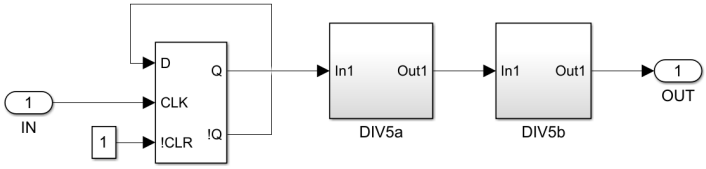

Figure B.6: DIV Simulink Model

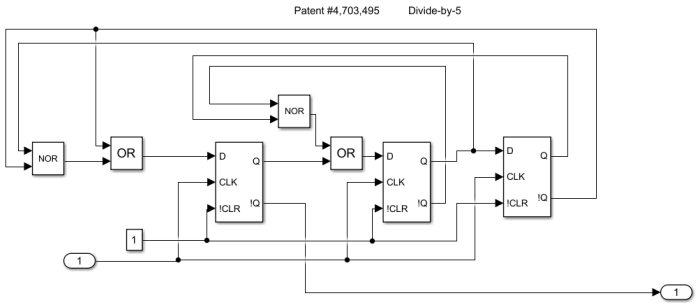

Figure B.7: Divide by 5 Simulink Model

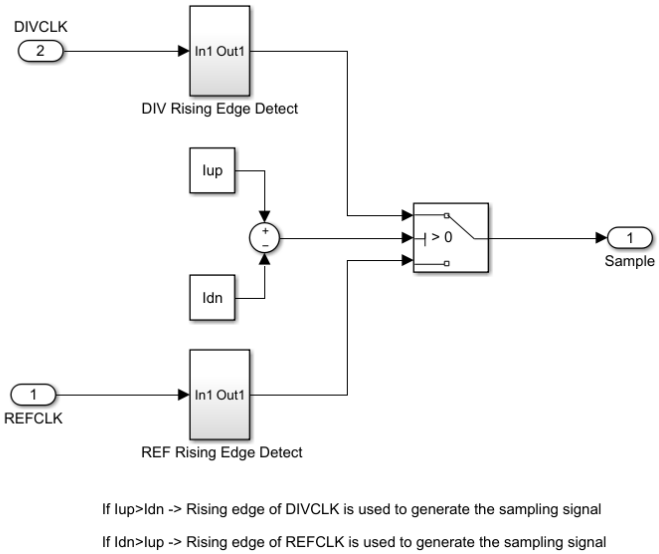

Figure B.8: Generate Sampling Signal Simulink Model

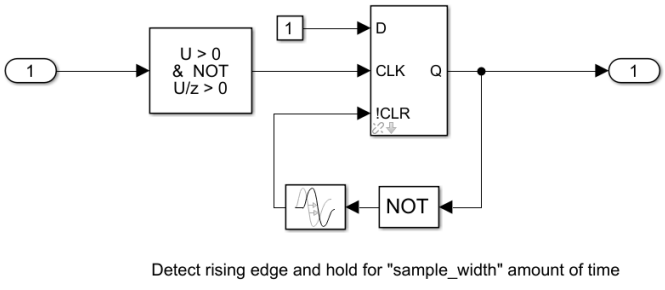

Figure B.9: Edge Detection Simulink Model

APPENDIX C

CADENCE SIMULATION SETTINGS

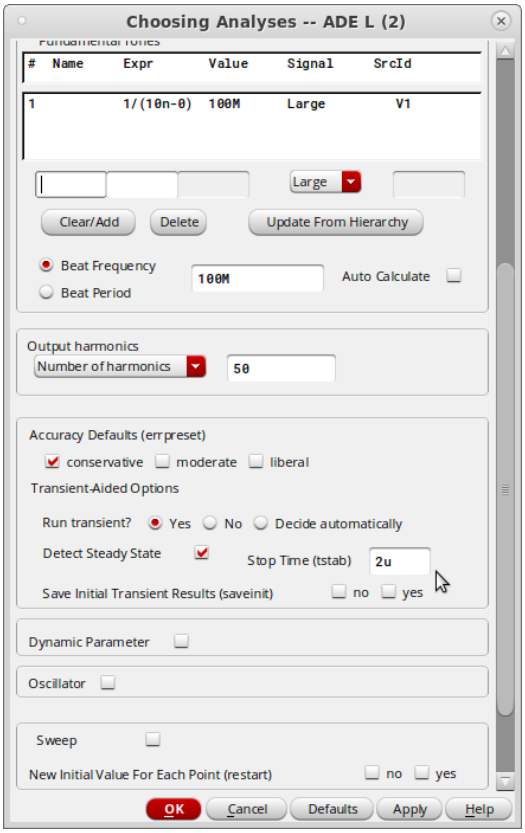

Figure C.1: PSS Settings

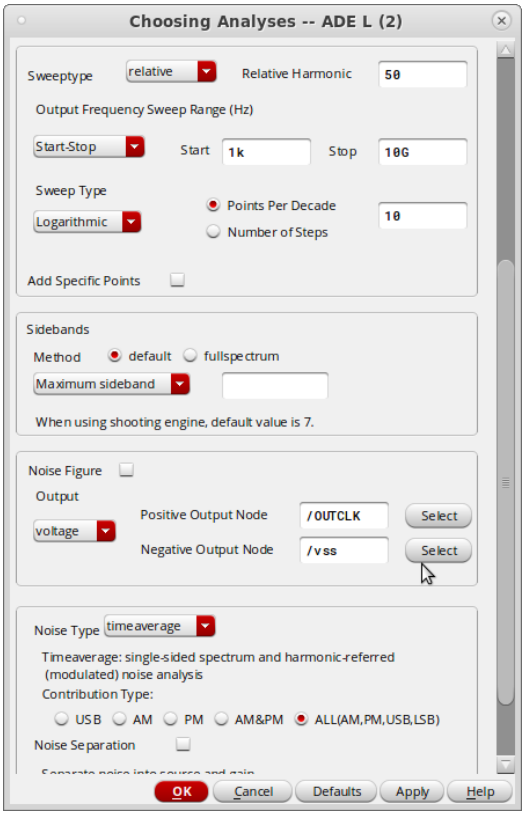

Figure C.2: PNOISE Settings