DESIGN OF A LOW POWER FRACTIONAL-N PLL FREQUENCY SYNTHESIZER IN 65NM CMOS

by

JACK RYAN CHAILLE

Submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Electrical, Computer, and Systems Engineering

CASE WESTERN RESERVE UNIVERSITY

May, 2022

CASE WESTERN RESERVE UNIVERSITY

SCHOOL OF GRADUATE STUDIES

We hereby approve the thesis/dissertation of

Jack Chaille

candidate for the degree of Master of Science *

Committee Chair

Hossein Lavasani

Committee Member

Pedram Mohseni

Committee Member

Christian Zorman

Date of Defense

4 April 2022

*We also certify that written approval has been obtained

for any proprietary material contained therein

Table of Contents

List of Tables

List of Figures

Acknowledgements

I would first like to thank Dr. Hossein Lavasani for his guidance as an academic advisor and professor. This project would not be where it is and I would not be where I am today without him.

In addition, I would also like to thank Dr. Pedram Mohseni and Dr. Christian Zorman for being a part of my thesis defense committee.

I would also like to thank several professors in particular who have played an instrumental role in my growth as an electrical engineer, including Dr. Evren Gurkan-Cavusoglu, Dr. Soumyajit Mandal, Dr. Kenneth Loparo, Dr. Wyatt Newman, and Larry Sears.

I would like to thank the many friends I made and classmates I met that made my college experience unforgettable.

Lastly, I would like to thank my parents for raising me to be the person I am today and supporting me through everything.

Design of a Low Power Fractional-N PLL Frequency Synthesizer in 65nm CMOS

Abstract

by

JACK RYAN CHAILLE

Frequency synthesizers play a crucial role in modern wireless communications as the local oscillator in a transceiver's upconverter and downconverter. One type of frequency synthesizer utilizes a phase-locked loop (PLL) to generate a frequency that is a multiple of a fixed reference. Integer-N PLLs are typically insufficient in modern wireless standards due to tight channel spacing putting a restrictive limit on reference frequency and bandwidth. Fractional-N PLLs, however, can precisely generate an output frequency that is a fractional multiple of the reference by toggling between division ratios. This thesis covers the design and simulation of a 1.2GHz low-power fractional-N PLL-based frequency synthesizer in 65nm CMOS. System-level and circuit-level design choices and simulations are shown, including a true single phase clock phase frequency detector, a charge pump, a 3rd order loop filter, a voltage-controlled oscillator, a toggleable frequency divider, and a 16-bit 2nd order delta sigma modulator.

1 Introduction

Generation of a reliable, stable, and low-noise local oscillator is critical for the operation of essentially any system that operates in the RF spectrum and interfaces with other devices wirelessly. As shown in Table 1.1, many wireless standards utilize channels that are tightly spaced relative to the frequency band. Some standards utilize frequency hopping, where the carrier frequency/ channel is continually changed during operation to minimize interference between adjacent channels or standards that operate in the same frequency band [1]. In the case of Bluetooth, a technique known as adaptive frequency hopping examines all channels and classifies how well each channel is working; this map is continuously updated and a device will only hop between satisfactory channels via a channel selection algorithm [2]. To precisely generate a desired frequency while also being able to quickly switch to another, frequency synthesizers are utilized in many modern electronic systems as local oscillators. Frequency synthesizers utilize at least one stable reference, such as a crystal oscillator, to generate many frequencies in a desired band of interest [1], [3].

Wireless Standard	Frequency Band	Channel Width/Spacing(s)	Number of Channels
Bluetooth	2.4GHz	1MHz	79
Bluetooth Low Energy (BLE)	2.4GHz	2MHz	40
IEEE 802.11ax (Wi-Fi 6, Wi-Fi 6e)	2.4GHz, 5GHz, 6GHz	20MHz, 40MHz, 80MHz, 160MHz	Depends on Band/Channel width: 2-59
GSM-850	850MHz	200KHz	124

Table 1. 1 Examples of Wireless Standards

In wireless communications, one of the most important applications of a local oscillator—and thus a frequency synthesizer—is in an RF transceiver, as shown in Figure 1.1. A local oscillator is utilized in both transmitting and receiving a desired signal centered around a carrier frequency. For transmission (TX), a local oscillator signal and intermediate frequency (IF) signal are combined in an upconverter stage and transmitted as an RF signal. Similarly, the receiving (RX) end converts a high-frequency incoming RF signal into a usable lower one through a downconverter stage [3], [4].

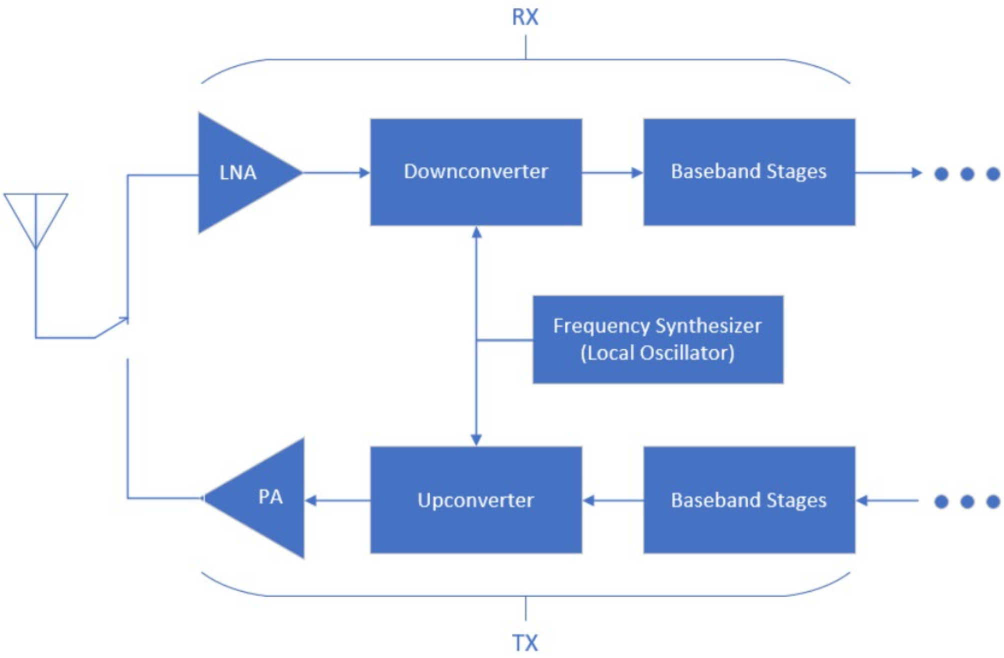


Figure 1. 1 Frequency synthesizer in an RF transceiver. Adapted from [3].

There are multiple frequency synthesizer architectures for different applications and specifications, but most frequency synthesizers fit within one of 3 categories: direct analog synthesis, direct digital synthesis, or indirect synthesis using a phase-locked loop (PLL). Direct analog synthesizers, such as the architecture shown in Figure 1.2, mix base frequencies and pass them through switched bandpass filters. Direct analog synthesizers are capable of impressive switching speeds and low phase noise; however, the number of frequencies that can be synthesized and the step size between them is limited by the number of base frequencies and mixer stages [5], [6].

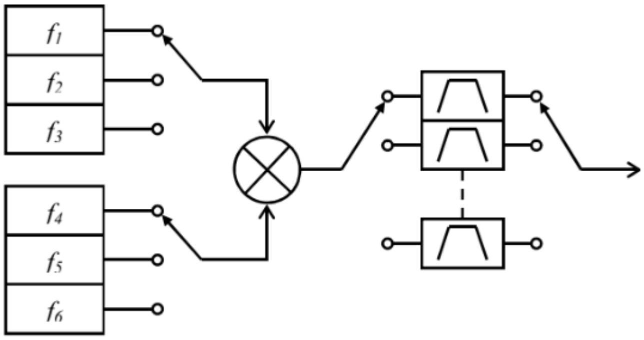


Figure 1. 2 A direct analog frequency synthesizer [5]

Direct digital synthesizers (DDS), such as Figure 1.3, utilize digital signal processing to construct a desired output frequency. A reference signal is fed into a phase accumulator, controlled by some binary word. The phase accumulator contains a frequency register that increments by the input digital word. This phase accumulator output is then fed into a phase-to-amplitude converter that generates the digital representation of the desired output signal. This phase-to-amplitude converter is either implemented via a ROM lookup table or with a CORDIC (COordinate Rotation DIgital Computer) algorithm, which calculates the desired output via vector rotation [7]. This is then finally processed through a digital-to-analog converter (DAC) to generate a sine wave. Direct digital synthesizers can achieve sub-Hz frequency resolution as well as switching times comparable to direct analog synthesizers—the primary limitation on speed comes from the digital control interface. However, quantization and DAC errors limit the bandwidth and can cause noticeable spurs in the output spectrum [5], [6].

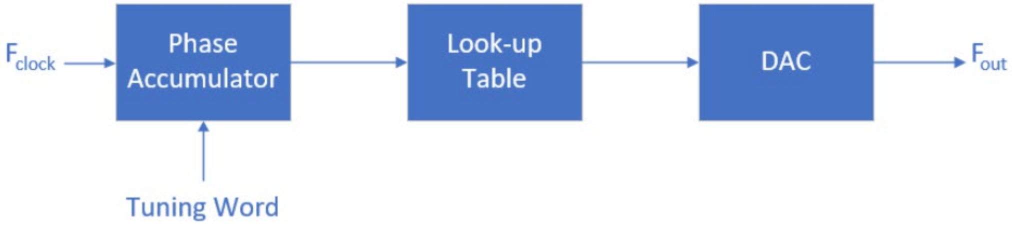


Figure 1. 3 A direct digital synthesizer. Adapted from [6]

Indirect frequency synthesizers, such as this work, are the most common and utilize a circuit known as a phase-locked loop (PLL) to synthesize a desired output frequency. The output signal of the PLL is generated by a voltage-controlled oscillator (VCO), which has its control voltage kept near-constant with the use of negative feedback via a phase detector and a low pass filter. By adding frequency dividers between the feedback signal and the output, the output frequency will be a multiple of the reference signal. When this multiple is an integer, the circuit is referred to as an integer-N PLL or integer-N frequency synthesizer [5], [8].

However, an integer-N PLL is typically not sufficient for wireless standards that have tight channel spacing relative to the frequency band. For example, Bluetooth operates between 2.4GHz and 2.48GHz with 1MHz channel spacing—if an integer-N PLL is used to generate these channels, the reference frequency cannot be above 1MHz and the total divide ratio must be between 2400 and 2480. Not only would such a divider consume a great amount of chip space and power, but the performance of the synthesizer would be severely degraded; maintaining loop stability with such a large divide ratio would require a small bandwidth, drastically increasing lock time and reducing suppression of oscillator phase noise [3].

In this case, a fractional-N PLL, as shown in Figure 1.4, is used to generate an output frequency that is a fractional multiple of the reference frequency. To achieve a fractional output division, a frequency divider with multiple divide ratios is employed and additional circuitry is employed to toggle between the ratios such that the average ratio is a non-integer.

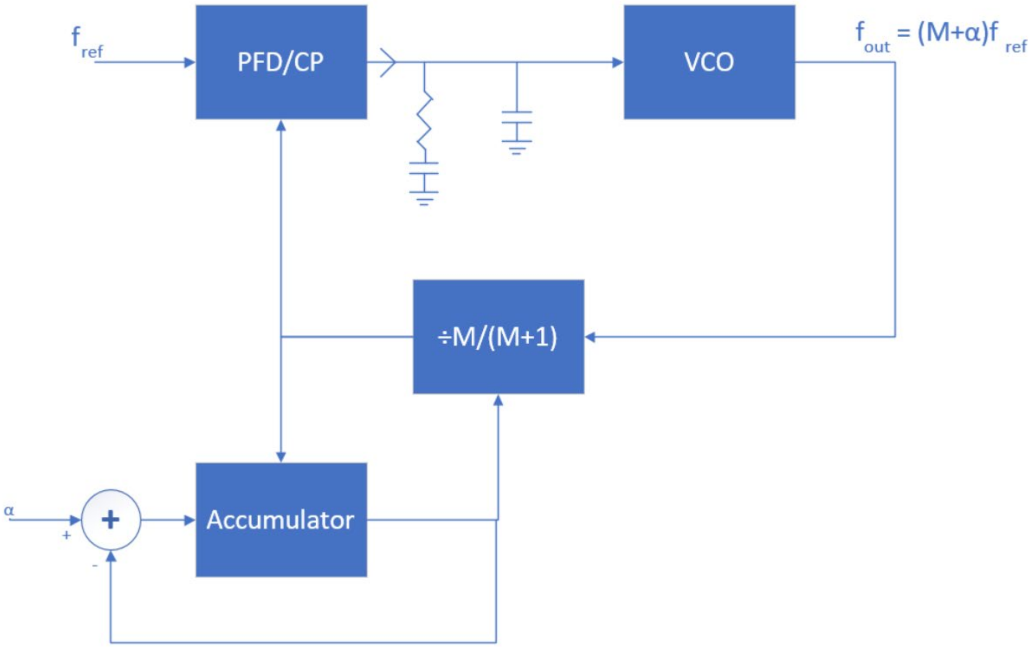


Figure 1. 4 A fractional-N PLL-based frequency synthesizer. Adapted from [3]

There is increased demand for low-power frequency synthesizers, as many devices that require them may be battery operated. This thesis will cover the system and circuit level design and simulation of a low-power 1.2GHz PLL-based fractional-N frequency synthesizer in 65nm CMOS. The main blocks of a basic PLL (phase detector, loop filter, VCO, and frequency divider) are outlined and expanded upon to show the need for fractional-N frequency synthesizers. The PLL is then examined as a linear system in the phase domain and the concept of phase noise is examined. Following this, the transistor-level design of each block of this fractional-N frequency synthesizer is outlined—notably, the use of a true single phase clock (TSPC) phase-frequency detector (PFD), charge pump (CP), toggleable frequency divider, and the use of a 2nd order delta-sigma modulator (DSM) to toggle the ratio of the divider. Finally, results are presented showing the output spectrum, transient response, and the power consumption.

2 Fractional-N PLL Fundamentals

Before describing the fractional-N PLL of Figure 1.1c, it is useful to examine the simplest form of PLL and examine why it is insufficient for frequency synthesis. A basic block diagram of a PLL is shown in Figure 2.1a. It consists of three blocks: a phase detector (PD), a low pass filter (LPF) and a voltage-controlled oscillator (VCO). V_{REF} is the is the reference signal inputted into the PLL, typically from a crystal oscillator. Its frequency is f_{REF} (in Hz) or ω_{REF} (in rads/s) and its phase is ϕ_{REF} (in rads) [3], [9].

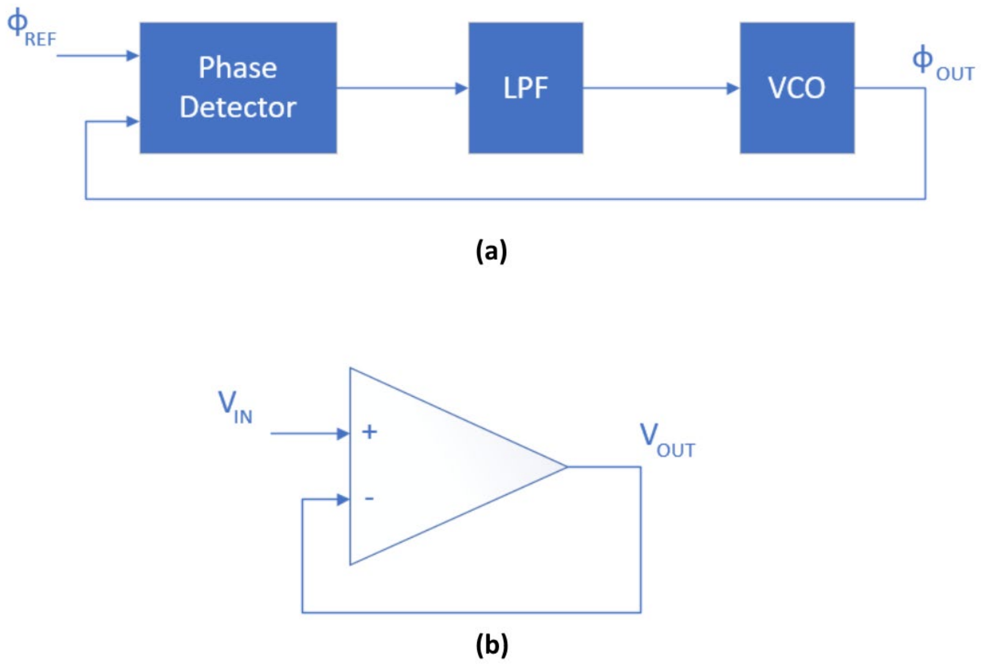


Figure 2. 1 (a) Basic PLL block diagram. (b) A voltage buffer. Adapted from [3]

Similar to how an op-amp can utilize voltage-domain negative feedback to equalize the voltage at its two input ports in Figure 2.1b, a PLL also utilizes negative feedback. Rather than equalize two voltages, however, a PLL equalizes the phase of two signals. Two signals are compared at a phase detector, which has an output proportional to their phase difference. This output is smoothed by a low pass filter to set the control voltage for a voltage-controlled oscillator (VCO), which has an output frequency proportional to input voltage [3], [4].

In the simplest phase-locked loops, these first blocks (a phase detector, low pass filter, and VCO) are all that are needed for the operation of the system—For the purposes of frequency synthesis, this is insufficient. To generate an output frequency larger than the reference frequency, a frequency divider is needed in the feedback path. Frequency dividers almost exclusively divide by integer ratios, so this will allow for integer-N frequency synthesis. For an integer-N PLL with input frequency f_{in}, output frequency f_{out}, and a divide-by-N frequency divider,

$$f_{OUT} = Nf_{IN} \tag{2.1}$$

The analysis of a PLL is primarily conducted in the phase domain and is described in further detail in Chapter 3. This chapter examines the individual blocks and subcircuits that make up a fractional-N PLL frequency synthesizer—the basic phase detector is first examined, followed by the more robust phase frequency detector and charge pump. The simplest form of loop filter is introduced (and expanded upon in Chapter 3), followed by the VCO and frequency dividers. Following this, the need for fractional-N frequency synthesis is elaborated on and the idea of toggling division ratios in introduced. Delta-sigma modulators are introduced as one circuit capable of toggling the frequency divider with minimal spurs.

2.1 Phase Frequency Detector

Before analyzing the phase frequency detector (PFD), it is helpful to analyze the simple phase detector (PD) and note its shortcomings that the PFD aims to improve upon. A phase detector takes two signals as an input (in the case of Figure 1, ϕ_{REF} and ϕ_{out}) and outputs a voltage related to their phase difference. This is almost exclusively a pulse whose width is proportional to the phase difference. The simplest implementation of a phase detector is an XOR gate, shown in Figure 2.2. Assuming the two input signals V_1 and V_2 are of the same frequency, an XOR gate is sufficient for detecting phase difference;

however, this is not always the case and more sophisticated architectures are often required [3].

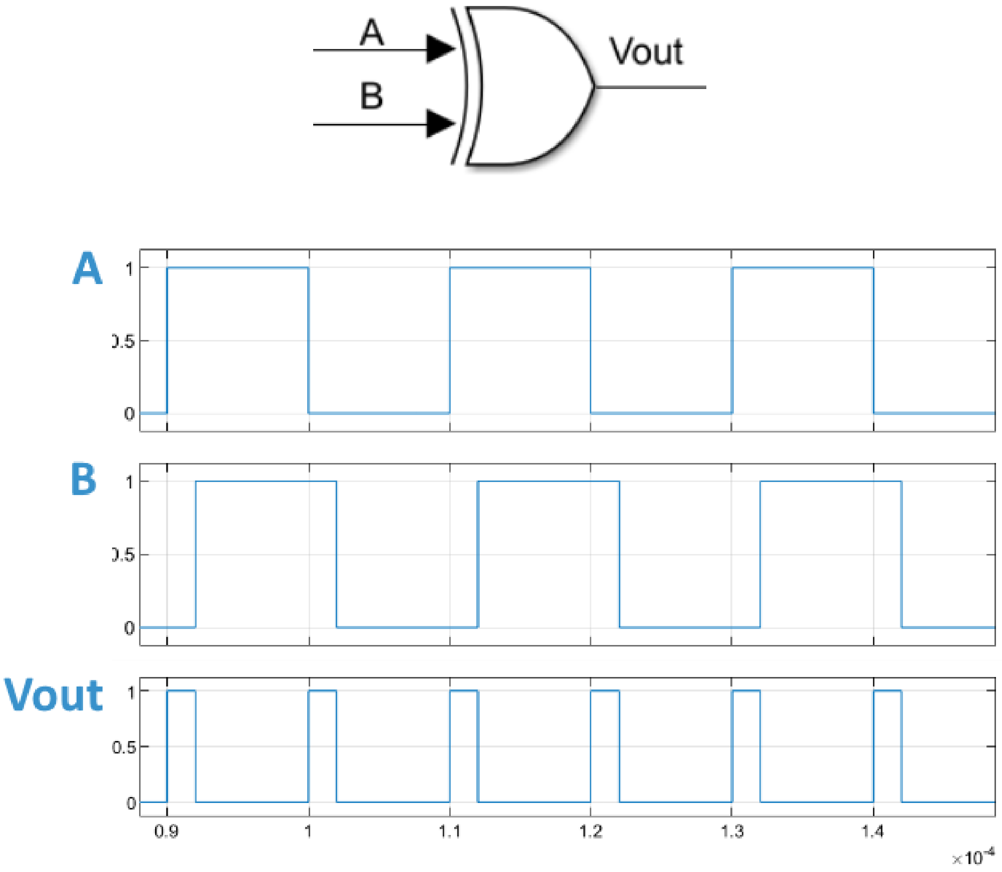


Figure 2. 2 An XOR gate used as a phase detector.

A phase detector operates on the assumption that both of its inputs are of the same (or very close to the same) frequency. When this is the case, the phase difference (and thus the PD pulse width) will stay the same. When they are different, however, the phase difference and PD pulse width will vary with time. This is seen in Figure 2.3, where two signals with different frequencies are fed into an XOR gate phase detector and the output pulse width changes over time.

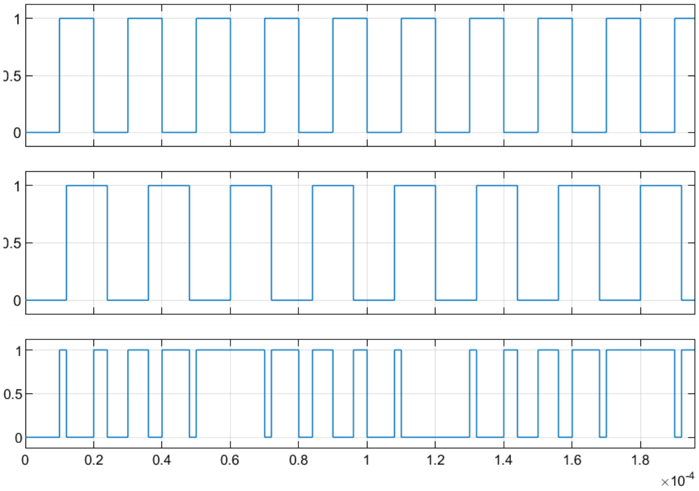


Figure 2. 3 An XOR gate phase detector with different frequency inputs

In a PLL-based frequency synthesizer, the output frequency (and thus the frequency of feedback signal that is compared to the reference signal) can vary greatly depending on the value of V_{cont}. This is especially true when the circuit is still in the process of locking. The solution to this problem is what is known as a phase frequency detector (PFD). While a phase detector can only detect when the inputs are out of phase, a phase frequency detector can also take frequency differences into account. A top-level implementation of a PFD is shown in Figure 2.4: it utilizes two resettable D flip-flops and an AND gate. When f_{ref} leads f_{fb}, UP will be 1; similarly, when f_{fb} leads f_{ref}, DOWN will be 1. As D is tied to VDD, Q will always output high when a rising edge occurs for that respective flip-flop; however, when both are high, the AND gate outputs 1 and both flip-flop outputs reset to 0 [3], [9], [10].

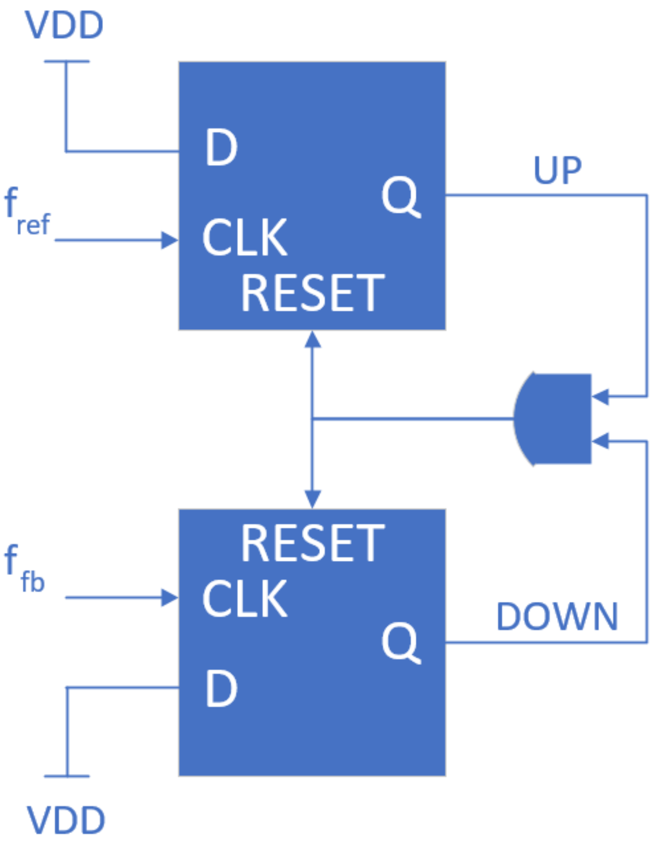


Figure 2. 4 A phase-frequency detector (PFD). Adapted from [3]

A phase detector's output is best described by logical statement ($A \oplus$ B), while a phase frequency detector is best described by the state diagram in Figure 2.5 [9]. There is never a scenario where both outputs are high (neglecting the propagation delay of the reset). The outputs also cannot both switch (e.g., from UP = 1 & DOWN = 0 to UP = 0 & DOWN = 1 or vice versa) at the same time; the PFD must first return to the neutral state where both outputs are 0. Additionally, by having the PFD be implemented with D flip-flops (an edge-triggered circuit), it is not required for the feedback signal to have a 50% duty cycle.

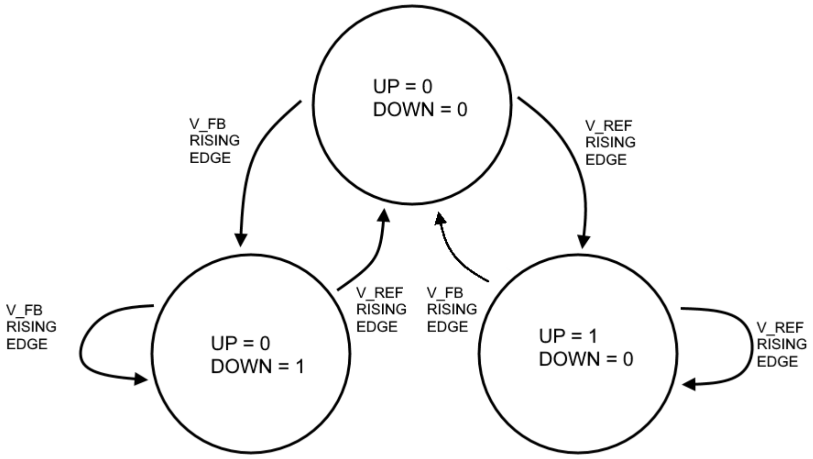


Figure 2. 5 State Diagram of a PFD. Adapted from [9]

2.3 Charge Pump

The purpose of the PFD having two outputs is to toggle the switches in the circuit's charge pump, as shown in Figure 2.6. A charge pump (in the context of a PLL) is, at its simplest, two current sources connected to the loop filter input through switches. Rather than directly connect the PFD to the loop filter (as is done with the XOR-based phase detector), the output of the loop filter V_{cont} is controlled by constant current sources. When UP is high, the top switch will turn on, allowing current to flow from VDD to the loop filter, increasing V_{cont} (as well as f_{out}). Similarly, when DOWN is high, the bottom switch turns out, allowing current to flow from the loop filter to ground, decreasing V_{out} and f_{out},.

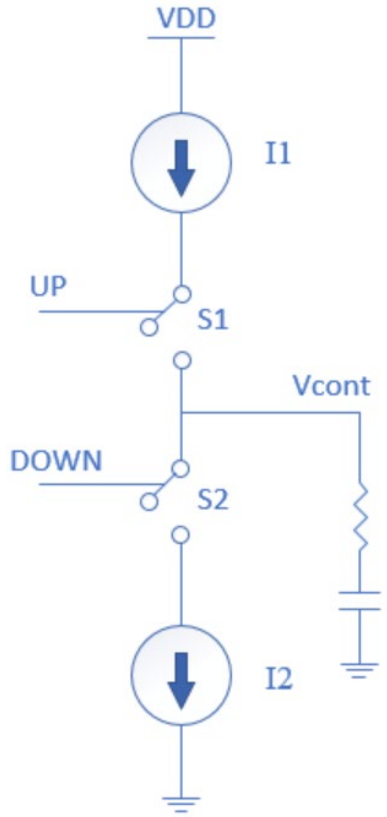


Figure 2. 6 Top level view of a charge pump. Adapted from [3]

2.4 Loop Filter

The output of the phase detector is a series of pulses at $\frac{1}{f_{REF}}$ second intervals. To smooth these pulses into a usable control signal V_{cont} to control the VCO, a low pass filter (also known as a loop filter within the context of PLLs) is used. The simplest low pass filter is a passive 1st order RC filter, shown in figure 2.7. Higher order filters (both passive and active varieties) are almost always used in modern PLL designs to further attenuate high-frequency content [3], [9], [11].

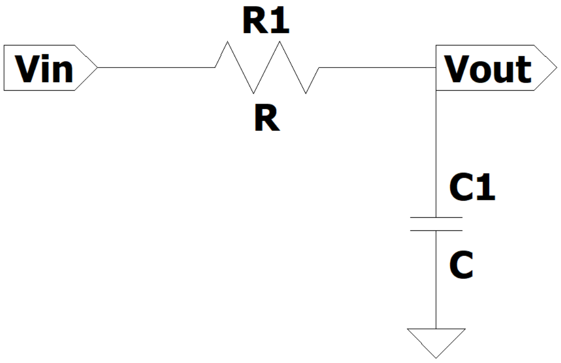


Figure 2. 7 A passive first order low pass filter

2.5 Voltage Controlled Oscillator

A PLL's VCO is typically built off a modified version of one of two oscillator types: ring oscillators and LC oscillators. Ring oscillators are built with a cascade of an odd number of inverters; the frequency of ring oscillators can be controlled by varying the gate delay. LC oscillators utilize an LC tank circuit's resonant frequency; this resonant frequency can be varied with a variable capacitor, also known as a varactor [3]. A ring VCO and LC VCO are shown in Figure 2.8a and 2.8b, respectively.

When sufficiently smoothed by the loop filter, the phase detector's pulse train becomes a constant control voltage that sets the output frequency of VCO. The tuning curve of a VCO may be slightly non-linear but is approximated as linear for ease of analysis. The output frequency f_{out} is proportional to the input control voltage V_{cont} multiplied by a factor K_{VCO} (defined in either Hz/V or rads/V) as shown in Equation 2.2:

$$f_{OUT} = \mathrm{f_o} + K_{VCO}V_{cont} \tag{2.2}$$

where f_o is the free-running frequency of the oscillator.

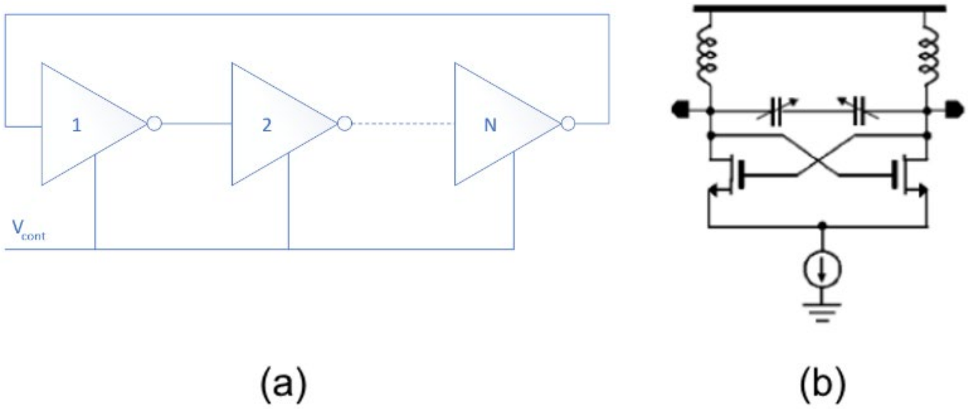


Figure 2. 8 (a) Ring VCO (b) LC VCO, adapted from [12], [13]

The VCO may also be viewed slightly differently in the context of correcting phase differences. In the case where $f_{REF} = Nf_{OUT}$ but $\phi_{REF} \neq \phi_{FB}$, the phase frequency detector and charge pump will output a pulse of current into the low-pass filter, slightly changing V_{cont} by ΔV_{cont}. One interpretation is that f_{OUT} slightly changes by a factor of $K_{VCO}\Delta V_{cont}$—another interpretation is that ϕ_{OUT} changes by the integral of this (as phase is the time integral of frequency). This "excess phase" ϕ_{ex} can in general be described by Equation 2.3 [3].

$$\phi_{ex} = K_{VCO}\int V_{cont}(t)dt \qquad (2.3)$$

2.6 Frequency Divider

The simplest frequency divider is realizable with only a D flip-flop, as shown in Figure 2.9. Connecting $\bar{Q}$ to D will cause Q to toggle between 1 and 0 every rising edge on the clock (tied to the input signal), thereby leading to an overall frequency division of 2. Multiple of these cells can be cascaded for higher division ratios; for N divide-by-2 frequency dividers cascaded in a row, this will result in an overall division ratio of 2^N. Slightly more complex circuitry is utilized for integer ratios that are not a power of 2, as described in Chapter 3 [8], [9], [14].

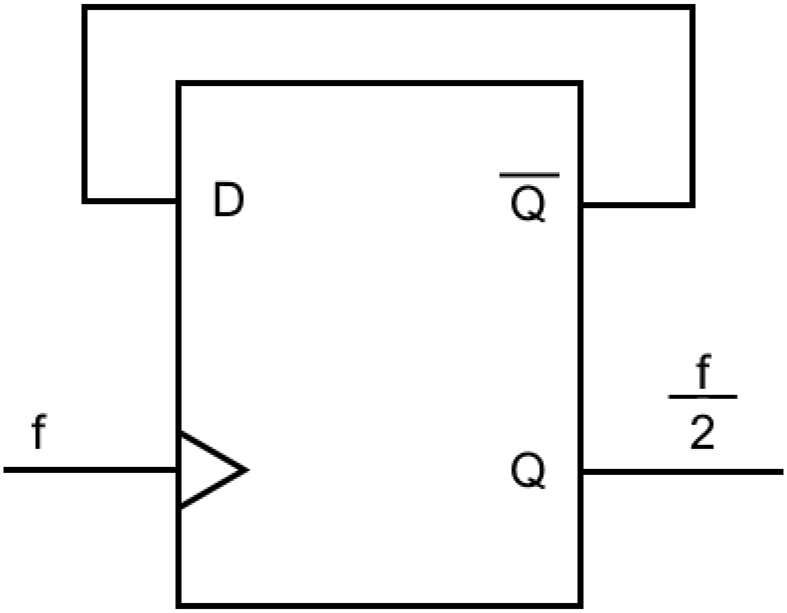


Figure 2. 9 D flip-flop-based divide-by-2 frequency divider

2.7 The Need for Fractional-N Frequency Synthesis

As discussed, many wireless devices must generate channels that are tightly spaced relative to the frequency band. Integer-N frequency synthesizers cannot have a reference frequency greater than the channel spacing, requiring large divide ratios. Not only would this require needlessly complex circuitry in the feedback path for the dividers, but it would necessitate a very narrow bandwidth; this would lead to long locking times and insufficient VCO phase noise suppression [3], [9]. The solution to this problem is to synthesize an output frequency that is a fractional multiple of the input frequency rather than an integer multiple, as laid out in Equation 2.4 (where N, M, and P are all integers). This is known as fractional-N frequency synthesis.

$$f_{OUT} = (N + \frac{M}{P})f_{IN} \tag{2.4}$$

In contrast to integer-N frequency synthesis where the divider is held at one ratio during operation, fractional-N frequency synthesis is achieved by having a divider toggle between its ratio (or “modulus”) such that the *average* ratio synthesizes the desired output frequency [3], [9], [15]. For example, if a frequency divider toggled between a ratio of 32 and 33 such that the fractional portion ($\frac{M}{P}$) is equal to 0.625, the output frequency would be 32.625 times the input frequency. One common toggleable divider is a divide-by-2/3 cell—as the name suggests, it can be toggled between dividing by 2 (like a normal D-flip flop in the configuration of Figure 2.9) or swallowing an extra pulse to divide by 3 [3], [14].

However, it is typically not sufficient to toggle the modulus with a simple, periodic pattern that averages out to the correct fractional value. This can introduce fractional spurs that fall within the loop bandwidth and are not attenuated by the loop filter, contributing to phase noise at the output. The solution is to sufficiently randomize the toggling of the modulus to reduce fractional spurs introduced into the loop [3]. This randomized toggling cannot be uniform, however, as a non-negligible portion will still fall within the loop bandwidth [3], [15]. A popular method, utilized by this thesis, is a delta sigma modulator (DSM, also called a sigma delta modulator).

2.8 Delta Sigma Modulator

Delta sigma modulators most commonly see use in analog-to-digital conversion (ADC) and digital-to-analog conversion (DAC), but are also particularly useful for toggling a PLL's divider [15], [16]. In general, a delta sigma modulator is an oversampling data converter that converts an input, either analog or a binary word, into a series of output pulses (that is only one or a few bits) by utilizing integration (the "sigma") and quantization (the "delta") [16]. A first order delta sigma modulator is shown in Figure 2.10. The only blocks are an integrator, a quantizer, and an adder/subtractor. For the purposes of frequency synthesis, the input to the DSM is a constant binary word equal or proportional to the fractional part of the total division ratio ($\frac{M}{P}$, following the convention of Equation 2.4). A delta-sigma modulator is preferable to other divider-toggling methods due to the high-pass nature of the DSM's frequency spectrum [15], [16].

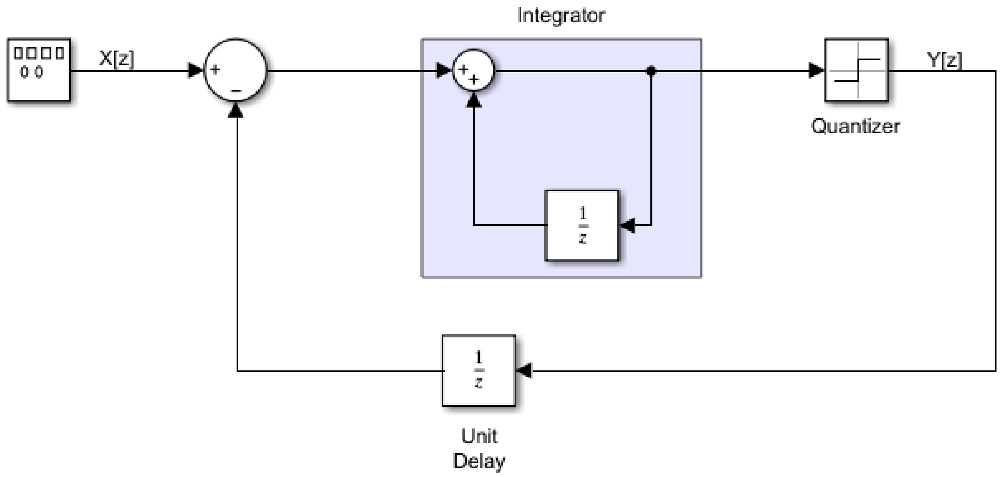


Figure 2. 10 A first order Delta-Sigma Modulator.

To analyze the frequency spectrum of a DSM, a linear z-domain model is utilized as shown in Figure 2.11. The quantizer is modeled as a source of uniform "quantization noise" E[z] added to the output of the integrator [9], [15]. The z-domain representation of the output DSM Y[z] is easily derived, as shown in Equation 2.5:

$$Y[z] = E[z] + \frac{1}{1-z^{-1}}(X[z] - Y[z]z^{-1}) = X[z] + (1 - z^{-1})E[z] \ (2.5)$$

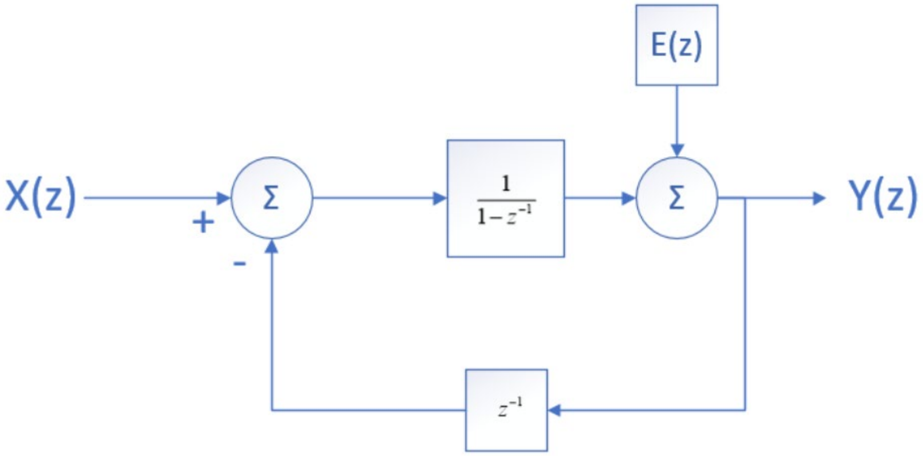


Figure 2. 11 z-domain model of 1st order DSM with uniform quantization noise. Adapted from [9]

Thus, the output frequency spectrum only contains the input (a constant) and the (roughly uniform) quantization noise multiplied by $(1 - z^{-1})$, which exhibits a high-pass behavior—this term is sometimes referred to as the noise transfer function (NTF) and becomes the primary focus of more complex DSM designs [15]–[17]. Rather than contribute a significant amount of noise within the bandwidth of the PLL, the DSM shapes the quantization noise, pushing it to a high enough frequency such that the (low pass) loop filter can sufficiently attenuate it. This analysis relies on the quantization noise being approximated as uniform—the accuracy of this approximation is input-dependent. For both analog and digital DSMs, the DC level of the input cannot be too close to the quantization levels. In the case of a single-bit quantizer where the output levels are 1 and -1, inputs with a DC level closer to 0 will result in the quantization noise being closer to uniform as shown in Figure 2.12. As the magnitude increases, however, this accuracy decreases. For example, an input relatively close to 1 will have an output of 1 most of the time and it becomes less accurate to say the DSM output is sufficiently random. If the input increases further, this overloads the quantizer and further degrades the quality of the system [16].

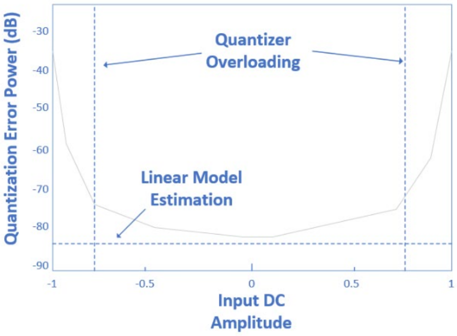


Figure 2. 12 Accuracy of DSM white noise assumption as a function of input amplitude. Adapted from [16].

Delta-sigma modulators are not limited to the first order architecture of Figure 2.11. Higher order DSMs are often utilized to achieve greater noise shaping as shown in Figure 2.13. Two observations are apparent: first, higher order DSMs have a lower noise floor—in the context of a PLL, this results in less in-band phase noise from the DSM. Second, the high-pass roll-off is steeper for higher order DSMs; in general, an L^{th} order DSM will exhibit $(2^L) \cdot 20 \frac{dB}{decade}$ roll-off [15].

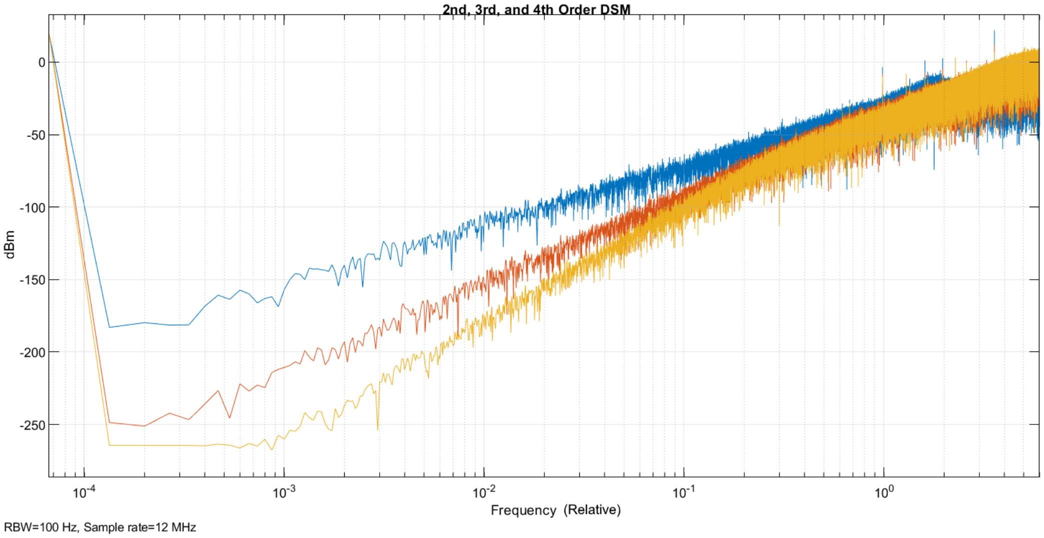


Figure 2. 13 Comparison of different order DSM spectrums.

Higher order DSMs fall into one of two categories: higher order single loop DSMs and MASH (Multi-stAge noise SHaping) DSMs. A higher order single loop DSM still only utilizes one quantizer, but can also employ the use of additional integrators, multi-level quantizers, distributed feedback, and/or feedforward[15]–[17]. The z-domain representation of a 2nd order single loop

DSM (utilized in this work) is shown in Figure 2.14. Its output Y[z] is derived similarly to Equation 2.5, as shown in Equation 2.6:

$$Y[z] = E[z] + \frac{1}{1-z^{-1}}(\frac{1}{1-z^{-1}}(X[z] - Y[z]z^{-1}) - Y[z]z^{-1}) =$$

$$Y[z](1 - z^{-1}) = (1 - z^{-1})E[z] + (\frac{1}{1-z^{-1}}(X[z] - Y[z]z^{-1}) - Y[z])$$

$$Y[z] = (1 - z^{-1})E[z] + \frac{1}{1-z^{-1}}(X[z] - Y[z]z^{-1})$$

$$Y[z](1 - z^{-1}) = (1 - z^{-1})^2E[z] + X[z] - Y[z]z^{-1}$$

$$Y[z] = (1 - z^{-1})^2E[z] + X[z] \qquad (2.6)$$

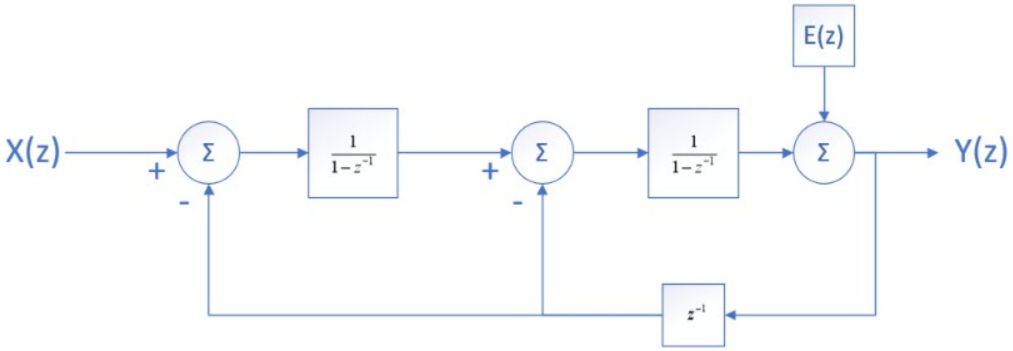
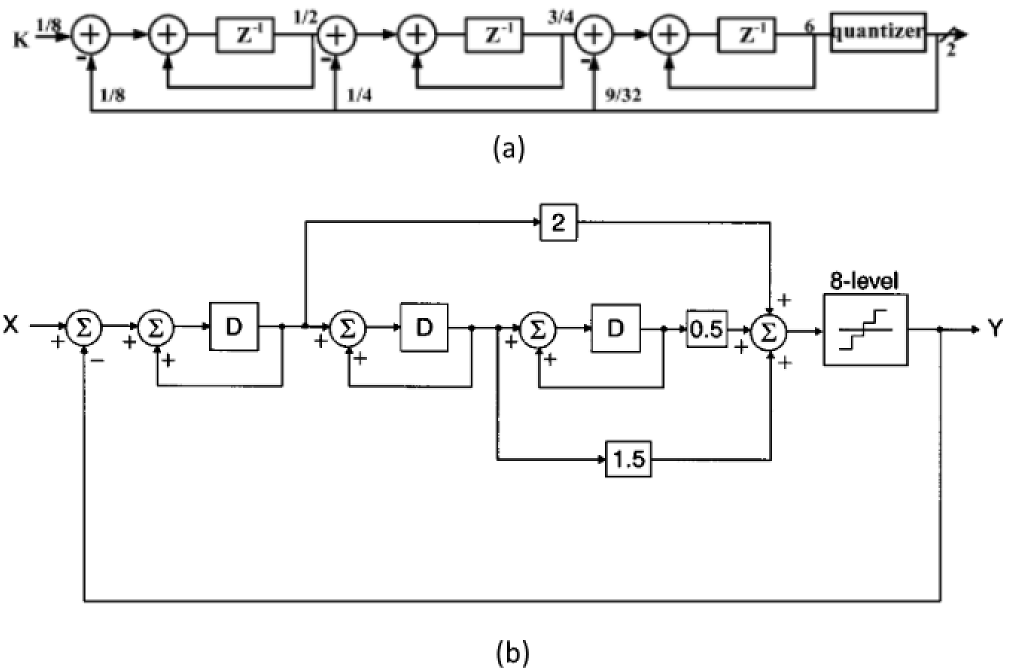


Figure 2. 14 z-domain representation of 2nd order single loop DSM. Adapted from [9]

In the case of a 2nd order single loop DSM, only an additional integrator and adder is required. For this topology, the noise transfer function $NTF_2(z) = (1 - z^{-1})^2$. If this design were iterated upon for an L^{th} order DSM, then the corresponding NTF would be of the form $NTF_L(z) = (1 - z^{-1})^L$. Unfortunately, single loop DSMs of this with an order greater than 2 quickly run into issues with overloading and stability unless additional modifications are made to alter the NTF [15], [16]. The 3rd order single loop DSM of Figure 2.15a utilizes distributed feedback and additional coefficients between stages to realize an NTF of $\frac{(1-z^{-1})^3}{1-1.3125z^{-1}+0.75z^{-2}-0.15625z^{-3}}$ [18]. The 3rd order single loop DSM of Figure 2.15b utilizes different elements—in this case, feedforward and a multi-level quantizer are used to realize an NTF of $\frac{(1-z^{-1})^3}{1-z^{-1}+0.5z^{-2}-0.1z^{-3}}$ [17].



(a)

(b)

Figure 2. 15 (a) A 3rd order single loop DSM w/ distributed feedback. (b) A 3rd order single loop DSM w/ feedforward and a multi-level quantizer [17], [18]

The alternative to single loop higher order DSMs are MASH DSMs [15], [16]. MASH DSMs achieve high order noise shaping by utilizing multiple lower order single loop DSMs in a cascade—typically only 1st and 2nd order due to their inherent stability for most inputs. A 1-1-1 MASH DSM is shown in Figure 2.16.

As the name suggests, it is composed of three 1st order DSM stages. The input to the first stage is the same, but each subsequent stage uses the (negative) quantization noise of the stage before it; the second stage uses $-E_1[z]$ as its, the third stage uses $-E_2[z]$, etc. The stages are then combined with $(1 - z^{-1})$ blocks (a digital differentiator) and added together. The output of a 1-1-1 MASH DSM's output Y[z] is derived as follows:

$$Y_1[z] = (1 - z^{-1})E_1[z] + X[z]$$

$$Y_2[z] = (1 - z^{-1})E_2[z] - E_1[z]$$

$$Y_3[z] = (1 - z^{-1})E_3[z] - E_2[z]$$

$$Y[z] = Y_1[z] + (1 - z^{-1})(Y_2[z] + (1 - z^{-1})Y_3[z])$$

$$Y[z] = X[z] + (1 - z^{-1})E_1[z] + (1 - z^{-1})((1 - z^{-1})E_2[z] - E_1[z] +$$

$$(1 - z^{-1})((1 - z^{-1})E_3[z] - E_2[z]))$$

$$Y[z] = X[z] + (1 - z^{-1})^3 E_3[z] \qquad (2.7)$$

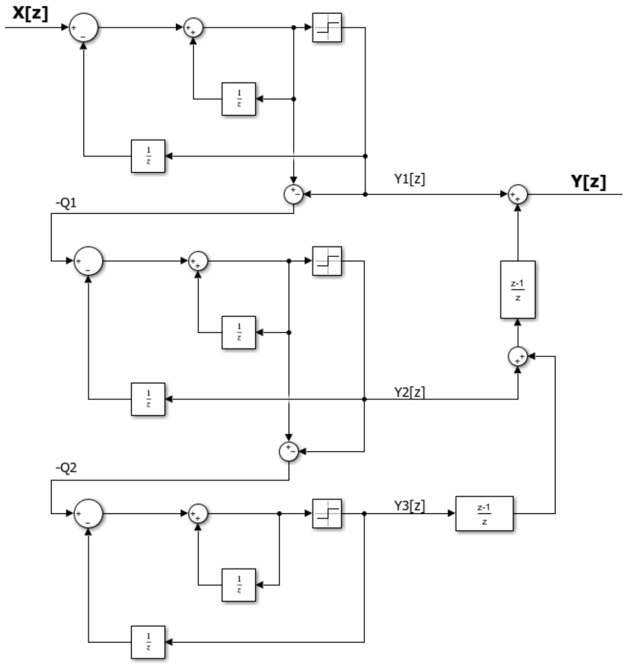


Figure 2. 16 A 1-1-1 MASH DSM

As each stage of the MASH DSM has a quantizer, a different term is used to represent the quantization noise of each. In the case of the 3-stage 1-1-1 MASH DSM, E_1, E_2, and E_3 refer to the quantization noise of the first, second, and third stages, respectively. While a higher order single loop can use a single-bit output or a multi-bit output, a MASH DSM must have multiple bits at its output—each stage's output bit is added together and all but the first are passed through at least one $(1 - z^{-1})$, so the output can not simply be a 0 or 1 [15], [16].

2.9 Literature Review

Much work has been done in the development and testing of PLL-based frequency synthesizers. Often these are designed with specific frequency bands or communication standard in mind (such as Bluetooth or Wi-Fi), and prioritizing some design aspect, such as power consumption, tuning range, or lock time/switching speed. As the publications examined in this section operate at different frequencies, both the total power consumption and the power consumption excluding the VCO is examined for comparison—the VCO typically consumes the most power in a PLL and this is frequency dependent.

A journal article by Wang and Luong in IEEE Transactions on Circuits and Systems concerns the design of a 0.8V supply 1.2GHz fractional-N frequency synthesizer in 0.18um CMOS technology [18]. This implementation used a 5-bit single-loop third-order delta sigma modulator, shown previously in Figure 2.15a. The DSM's NTF is designed to avoids phase noise increases in the intermediate frequency range by reducing noise gain at $\frac{f_{clk}}{2}$. The synthesizer has a total power consumption of 4.92mW—3.2mW from the VCO, and 1.72mW from the rest of the circuit. It achieved a settling time of 200us, a tuning range of 1.03-1.4GHz, a phase noise of -70dBc/Hz in-band, -104dBc/Hz at 200KHz offset, and -121dBc/Hz at 1MHz offset [18].

One publication by Lee et al. from Kwangwoon University covers the design of a 1V supply 2.4GHz fractional-N PLL in 65nm CMOS technology [19]. It utilized a 20-bit 1-1-1 MASH Delta Sigma Modulator for toggling its divider. The PLL achieved a locking time of 12us and had a total power consumption of 2.94mW; 1.12mW of this comes from the VCO and an output buffer, while the rest of the PLL consumes 1.82mW. The VCO demonstrated a tuning range of 1.9-2.7GHz; it demonstrated a phase noise (at 1MHz offset) of -110dBc for 2.7GHz and -120dBc at 1.9GHz [19].

A publication by Lee et al. from National Chung-Cheng University describes the design of a 1V supply 2.4GHz low-power fractional-N PLL in 0.18um CMOS technology [20]. A 5-bit third order single loop feed-forward delta sigma modulator with an inverse Chebyshev NTF was used to toggle its divider. The PLL had a tuning range of 2.136-2.53GHz, had a lock time of < 25us, and had a total power of 8.94mW. Phase noise of -126.85dBc/Hz was observed at 1MHz offset [20].

A publication by Vidojkovic et al. in IEEE Radio Frequency Integrated Circuits Symposium showcases a fully integrated 1.2V supply 1.7-2.5GHz fractional-N frequency synthesizer in 90nm CMOS [21]. A 1-1-1 MASH delta sigma modulator was utilized for toggling the divider. Total power consumption ranged from 1mW to 1.13mW depending on frequency. The VCO consumes 394-570uW, while the rest of the loop consumes 550-618uW. Lock time was approximately 40us, and phase noise was found to be -115dBc/Hz at 1MHz offset [21].

A publication by Yu and Kinget from Columbia University, published in ISSCC, discusses the design of a 0.65V supply 2.5GHz fractional-N frequency synthesizer in 90nm CMOS. 24-bit 1-1-1 MASH delta sigma modulator was utilized to toggle the circuit's current-mode logic (CML) modular frequency divider. 2mW was consumed by the VCO, 3mW was consumed by the load driving stage, and 4mW was consumed by the remainder of the loop; a phase noise of -111dBc/Hz was observed at 1MHz offset [22].

A publication by Paidimarri et al. from Massachusetts Institute of Technology covers the design of a 0.68V supply 2.4GHz integer-N frequency synthesizer in 65nm CMOS for use with BLE. A 12MHz crystal oscillator is divided down to 1MHz before the input of the PLL, allowing for the generation of BLE channels. The total power consumption was 680uW; 510uW of this came from the VCO and 170uW came from the remaining blocks of the PLL. A startup time of 130us was observed, and phase noise was -110dBc/Hz at 1MHz offset [23].

A paper by Abdelfattah et al. from McGill University describes the design of a 0.55V supply 1GHz integer-N frequency synthesizer in 65nm CMOS. A modified gate-switched low-voltage charge pump architecture is used to maximize the tuning range of the PLL. A forward-body-bias floating-input D flip flop architecture is used for a frequency divider that is both high speed and low voltage. Total power consumption was measured to be 3mW and phase noise was measured to be -106dBc/Hz at 1MHz offset [24].

A publication by Yan et al. from the Institute of Microelectronics of A*STAR discusses the design of a 1.36V supply 900MHz fractional-N frequency synthesizer in 0.13um CMOS. A CML divide-by-2 is used for generating output quadrature signals and a 21 bit 3rd order MASH DSM is used for toggling a TSPC divide-by-8/9 prescaler. Phase noise was measured to be -116dBc/Hz at 1MHz offset and total current consumption was 5.5mA (for a total power consumption of 7.48mW) [25].

Several key observations can be made about the power consumption and overall performance these examined works. Works such as [18] and [22] utilize a low voltage supply (0.8V and 0.65V, respectively) yet demonstrate considerably higher power consumption than higher supply works such as [19] and [21] (1V and 1.2V, respectively). Minimizing power consumption requires more than simply using lower voltage technology or operating on a smaller supply; architecture choices are also critical.

One important architecture choice is the implementation of the frequency divider in the feedback bath. Works such as [18], [22], and [25] utilize current-mode logic (CML), also known as source-coupled logic (SCL), for high speed dividers. CML-based flip-flops are capable of higher speed than TSPC-based flip-flops, but at the cost of noticeably higher power [26]–[28]. For this reason, CML frequency dividers are typically only utilized for frequency ranges higher that TSPC dividers cannot operate in. However, [18] is only a 1.2GHz PLL in 0.18um CMOS, [22] is a 2.5GHz PLL in 90nm CMOS, and [25] is a 900MHz PLL (with a 1.8GHz VCO) in 0.13um CMOS; multiple TSPC-based works demonstrate operation in similar or higher frequency ranges in similar or older technologies [26]–[28], so CML-based frequency dividers are typically unnecessary in this frequency range.

Additionally, the charge pump may also greatly contribute to power consumption. Despite operating at a significantly lower frequency than the divider, works such as [20] and [24] have a charge pump current of at least 500uA. While higher charge pump phase noise is inversely proportional to charge pump current, other works examined in this section such as [18], [23], and [25] achieve similar phase noise performances with a charge pump current that is less than 25uA.

This work will aim to minimize power consumption while attaining a reasonable lock time. As high-speed logic types such as CML and ratioed logic greatly increase power consumption, a frequency divider structure comprised only of TSPC-based flip flops is utilized instead to reduce divider power consumption. A low charge pump current is used to minimize power consumption from the charge pump. Comparison between works will attempt to examine both the total power consumption and the power consumption excluding the VCO, as the works in this section vary in output frequency and the VCO power consumption is typically the largest in a PLL and increases with frequency.

3 System Analysis and Circuit Design

3.1 PLL Linear Model

To analyze PLLs as a linear system, it is easier to work with the input and output in the phase domain (ϕ_{REF} & ϕ_{OUT}) rather than in the frequency (ω_{REF} & ω_{OUT}) or voltage (V_{REF} & V_{OUT}) domains, as shown in Figure 3.1 [29].

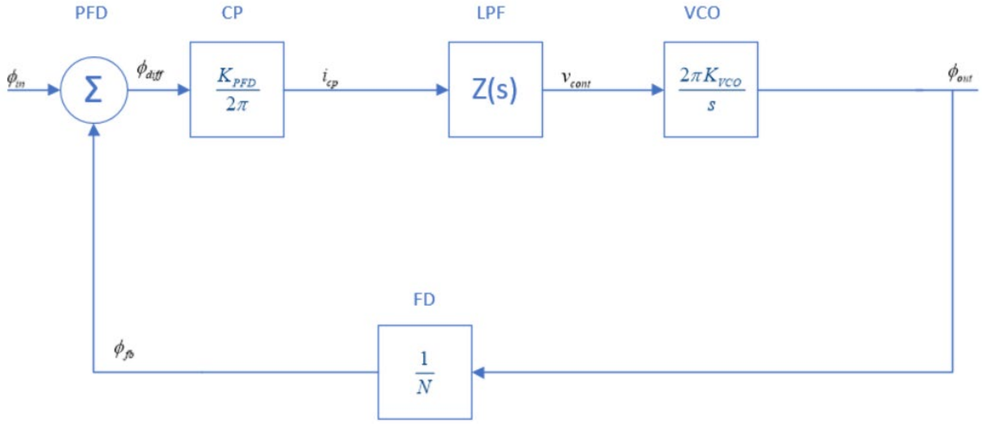


Figure 3. 1 Linear Model of a PLL. Adapted from [29].

The open loop transfer function $A_{OL}(s)$, feedback factor β, and closed-loop transfer function $A_{CL}(s)$ can be defined by Equation 3.1, Equation 3.2, and Equation 3.3 respectively:

$$A_{OL}(s) = \frac{K_{pfd}}{2\pi} Z(s) \frac{2\pi K_{vco}}{s} = \frac{K_{pfd} Z(s) K_{vco}}{s} \tag{3.1}$$

$$\beta = \frac{\phi_{FB}}{\phi_{OUT}} = \frac{1}{N} \tag{3.2}$$

$$A_{CL}(s) = \frac{\phi_{OUT}}{\phi_{REF}} = \frac{A_{OL}}{1+\beta A_{OL}(s)} \tag{3.3}$$

As the input ϕ_{REF} and output ϕ_{OUT} of this transfer function are in the phase domain, the s-domain representation is related to the frequency at which ϕ_{REF} changes, not the frequency of the VCO or the reference oscillator [3], [9]. The -3dB bandwidth ω_{-3dB} (also known as the loop bandwidth) is the frequency of phase modulation where the PLL starts to lose lock, defined by equation 3.4.

$$|A_{CL}(j\omega_{-3dB})| = \frac{\sqrt{2}}{2} \tag{3.4}$$

Two other important parameters for analyzing PLL loop dynamics are the unity-gain bandwidth ω_u and the phase margin PM as described by Equations 3.5 and 3.6, respectively.

$$|A_{OL}(j\omega_u) \cdot \beta| = 1 \tag{3.5}$$

$$PM = 180° - \angle A_{OL}(j\omega_u) \cdot \beta \tag{3.6}$$

Generally, the loop bandwidth affects lock time and spur levels. A higher loop bandwidth will be able to respond to higher frequency phase modulations and thus lock faster but will see higher spur levels. Even if lock time is less of a concern, bandwidth cannot be too low, as it must also suppress VCO phase noise. Phase margin also impacts lock time, but crucially it also impacts stability. A higher phase margin generally results in higher stability, but also a longer lock time [9], [30]. As most loop parameters are set as a constraint (i_{cp}) or potentially difficult to change (K_{VCO} for a given VCO), the main parameters that determine loop bandwidth and phase margin are the resistor and capacitor values of the loop filter [9].

The PFD and charge pump are modeled together (K_{pfd}) as having an output current i_{cp} proportional to the difference of its input phases—this is an approximation. As previously mentioned, the PFD outputs a pulse with a width proportional to ϕ_{diff}— this pulse is what controls the switches of the charge pump, but the current flowing into the charge pump is either $\pm I_{cs}$ (the value of the current sources) or 0. To model the entire PLL as a continuous-time system, however, this is approximated by averaging I_{cs} over a period modeling i_{cp} as a current proportional to the phase difference. This approximation holds true if the loop bandwidth is significantly lower than the reference frequency—the typical rule of thumb is $\omega_{-3dB} < \frac{\omega_{REF}}{10}$, but it is often lower (especially in fractional-N PLLs where fractional spurs occur due to the DSM's quantization noise and noise shaping) [3], [9], [29].

The s-domain model of Figure 3.1 denotes the loop filter transfer function as Z(s). For the third order passive filter (the type used in this work) shown in Figure 3.2b, it is simpler to express Z(s) in terms of the transfer function of a 2nd order loop filter shown in Figure 3.2a [30]. The 2nd order loop filter transfer function $Z_{2nd}(s)$ is defined in Equation 3.7 and the 3rd order loop filter transfer function is subsequently defined in Equation 3.8:

$$Z_{2nd}(s) = \frac{sC_2R_2+1}{s^2C_1C_2R_2+s(C_1+C_2)} \tag{3.7}$$

$$Z(s) = \frac{\frac{1}{C_3s}Z_{2nd}(s)}{Z_{2nd}(s)+R_3+\frac{1}{C_3s}} = \frac{Z_{2nd}(s)}{s(C_3Z_{2nd}(s)+C_3R_3+1)} \tag{3.8}$$

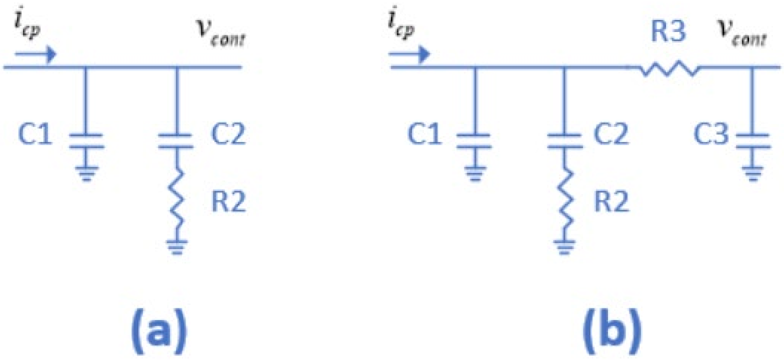


Figure 3. 2 (a) Second order loop filter. (b) Third order loop filter

The s-domain representation of the VCO can derived by taking the Laplace transform of ϕ_{OUT}, as shown in Equation 3.9 and Equation 3.10 [3], [9].

$$\phi_{OUT}(\mathrm{s}) = \mathcal{L}\{\phi_{\mathrm{OUT}}(\mathrm{t})\} = \mathcal{L}\left\{K_{VCO}\int V_{cont}(t)dt\right\} = K_{VCO}\frac{V_{cont}(s)}{s} \quad (3.9)$$

$$\frac{\phi_{OUT}(s)}{V_{cont}} = \frac{K_{VCO}(s)}{s} \quad (3.10)$$

3.2 Phase Noise

As a PLL is primarily analyzed in the phase domain, it is logical to extend this analysis to examining noise within the system. Ideally, the VCO (and thus PLL as a whole) will have an output voltage of the form

$$v_{out}(t) = V_0\cos(\omega_{\mathrm{out}}t) = V_0\cos\left((\omega_0 + K_{vco}V_{cont})t\right) \quad (3.11)$$

where V_0 is the amplitude of the VCO, ω_0 is the free-running frequency of the VCO (in rad/s). However, in a real system where noise is present, the output is

$$v_{out}(t) = V_0(1 + A(t))\cos((\omega_0 + K_{vco}V_{cont})t + \phi(t)) \quad (3.12)$$

where $A(t)$ is the variation in the output amplitude and $\phi(t)$ is the variation in output phase. In general, the amplitude variations $A(t)$ are less impactful on overall performance and can either easily be controlled or decay on their own over time [4], [29].

The phase variation $\phi(t)$, referred to as phase noise $\mathcal{L}(f)$, is very impactful on performance and is itself a figure of merit for PLLs and frequency synthesizers. In the ideal case of Equation 3.11, the output spectrum and waveforms look like that of Figure 3.3a—the output spectrum is a single impulse (for a sine wave) and the zero-crossing period is a constant T_1. When $\phi(t)$ is present as in Equation 3.12, however, the output phase and frequency is modulated, leading to a spectrum like that of Figure 3.3b. This manifests itself in the time-domain as zero crossings that are different periods (T_1, T_2, etc.) apart. The time-domain equivalent of phase noise is also known as jitter [3], [4].

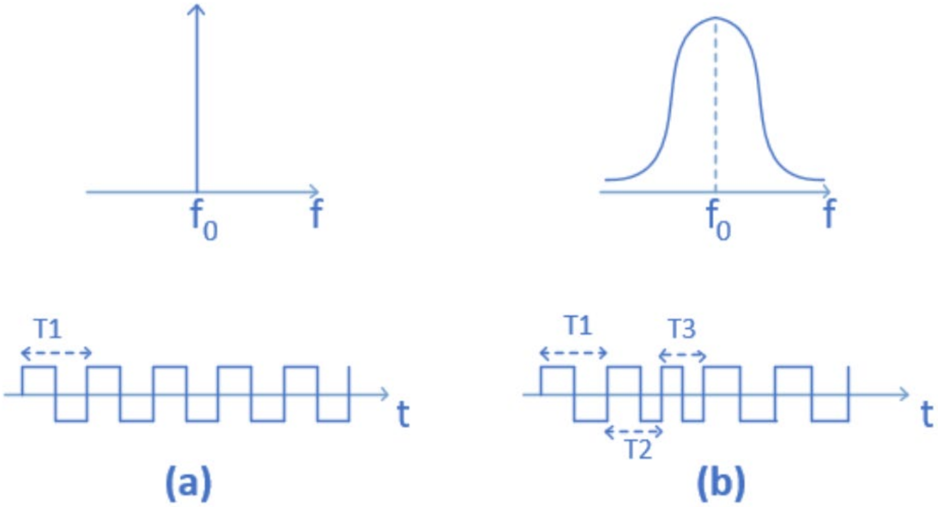


Figure 3. 3 (a) Ideal spectrum and waveform (b) Spectrum and waveform with phase noise. Adapted from [3]

Phase noise is formally defined as the ratio of noise power in a single sideband (normalized to a 1 Hz bandwidth) to the power of the carrier frequency (measured in dBc/Hz), as shown in Equation 3.13 and illustrated in Figure 3.4:

$$\mathcal{L}(\Delta f) = 10\log\left(\frac{P(f_0+\Delta f)}{P(f_0)}\right) \tag{3.13}$$

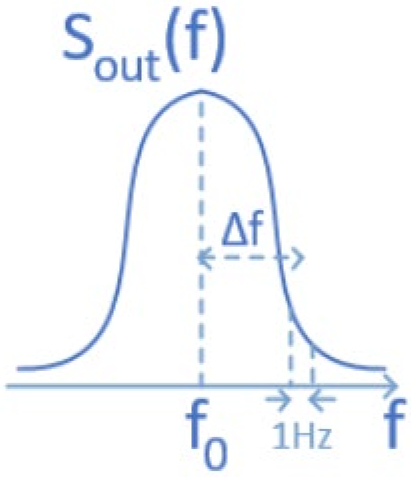


Figure 3. 4 Illustration of measurement of phase noise. Adapted from [3]

A robust model for modelling LC oscillator phase noise is known as Leeson's Model or Leeson's Equation. Leeson's Model views an oscillator as an amplifier with an RLC resonator in the feedback path as shown in Figure 3.5 [4].

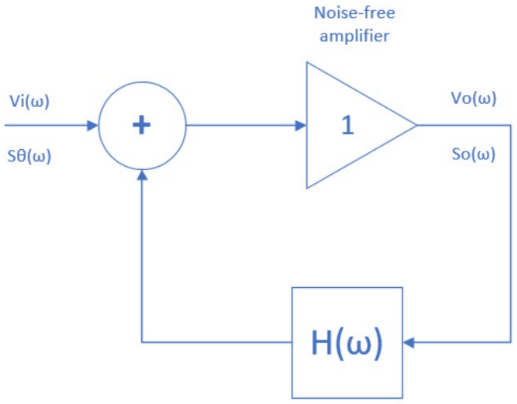


Figure 3. 5 Leeson Model of oscillator as a feedback amplifier. Adapted from [4]

The voltage transfer function can be expressed as

$$V_o(\omega) = \frac{V_i(\omega)}{1-H(\omega)} \tag{3.14}$$

and $H(\omega)$, modeled as a parallel RLC resonator, can be expressed as

$$H(\omega) = \frac{1}{1+jQ(\frac{\omega}{\omega_0}-\frac{\omega_0}{\omega})} = \frac{1}{1+2jQ*(\frac{\Delta\omega}{\omega_0})} \tag{3.15}$$

where Q is the loaded quality factor and ω_0 is the resonant frequency $\sqrt{(\frac{1}{LC})}$. Taking thermal noise and $\frac{1}{f}$ noise (also known as flicker noise) into account, the input power spectral density can be written as

$$S_\theta(\omega) = \frac{kTF}{P_0}(1 + \frac{K\omega_\alpha}{\Delta\omega}) \tag{3.16}$$

where k is Boltzmann's constant, T is the temperature, F is the noise factor, P_0 is the carrier power, K is a factor accounting for flicker noise, and ω_α is the flicker noise corner frequency [4], [31]. Additionally, the power spectral density of the input and output are related by

$$S_\phi(\omega) = \left|\frac{1}{1-H(\omega)}\right|^2 S_\theta(\omega) \tag{3.17}$$

Combining equations 3.14-3.17 and rearranging terms as done in [32] yields the modified Leeson's equation:

$$\mathcal{L}(\Delta f) = 10\log\left(\left(\left(\frac{f_0}{2Q\Delta f}\right)^2 + 1\right) * \left(\frac{FkT}{P}\right) * \left(\frac{f_\alpha}{\Delta f} + 1\right)\right) \tag{3.18}$$

The primary takeaway of Leeson's equation is that the power spectral density (PSD) of an oscillator's phase noise does not follow the same shape as the PSD of a traditional amplifier's noise (shown in Figure 3.6a). An amplifier's noise shows only $\frac{1}{f}$ noise and thermal noise, but an oscillator sees noise decreasing by $\frac{1}{f^3}$ and/or $\frac{1}{f^2}$. For a lower Q, the oscillator's phase noise will follow Figure 3.6b. For a higher Q, it will follow Figure 3.6c [4], [31].

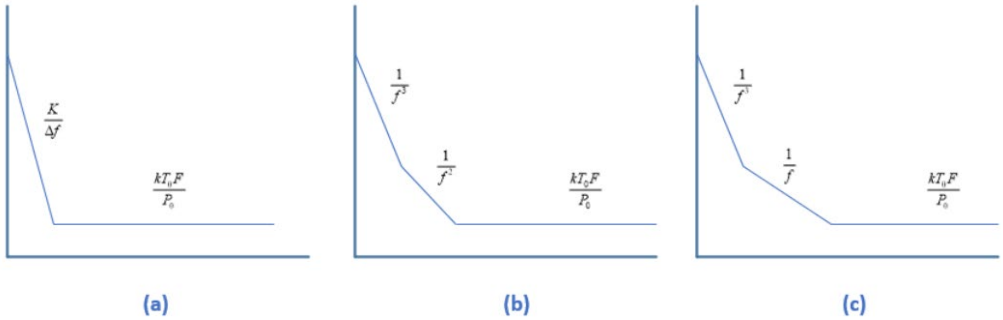


Figure 3. 6(a) PSD of amplifier noise, (b) PSD of oscillator phase noise with low Q, (c) with high Q. Adapted from [4]

Other notable phase noise contributions arise from the PFD, charge pump, loop filter, and supply. Phase noise is added from the PFD due to jitter as edges propagate through, causing slightly different pulse widths that turn the UP and DOWN switches on the charge pump on for different lengths [10]. This PFD phase noise rises with input frequency; phase noise rises by a factor of f_{ref} in the region where thermal noise dominates and by f_{ref}^2 when flicker noise dominates. Phase noise also occurs as a result of the charge pump—when both current sources are very briefly on before the reset propagates through (which takes several gate delays), both current sources are contributing uncorrelated noise [3], [29]. Supply noise can cause mismatches in the charge pump current through slight variations in V_{DD} as well as worsen VCO phase noise if it is particularly supply sensitive. Loop filter noise stems from thermal noise ($\bar{V}_n^2 = 4k_BTR$) associated with each resistor in the loop filter [3], [29].

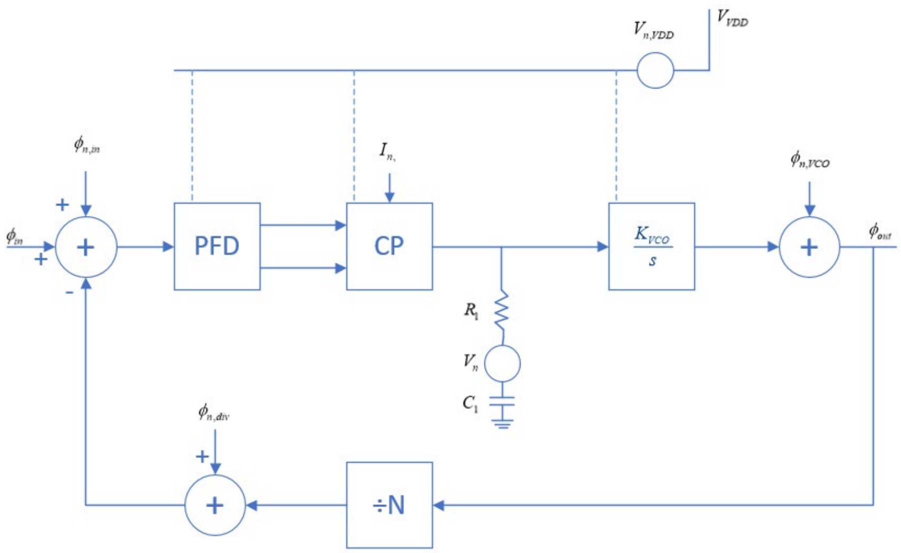


Figure 3. 7 Sources of phase noise in a PLL. Adapted from [3]

Examining phase noise in the entire PLL requires examining how phase noise propagates from stage-to-stage and affects the output; a figure outlining phase noise contributions to the PLL is shown in Figure 3.7. From the input to the output, any phase noise from the reference (which is typically approximated as uniform for a crystal oscillator) is shaped by the PLL's low-pass transfer function—for a divide ratio N, the input phase noise power is multiplied by N^2 (a dB increase of 20log(N)). Charge pump phase noise is similarly shaped when referred to the output, exhibiting the same poles and zeros as $\frac{\phi_{out}}{\phi_{ref}}$; the output-referred phase noise of the charge pump $\frac{\phi_{out}}{I_n}$ is also proportional to $\frac{1}{I_{cp}}$, so there is an inherent trade-off between charge pump power consumption and charge pump phase noise [3]. The VCO phase noise takes on a high-pass characteristic when referred to the PLL output [3], [29]. This can be shown in Equation 3.19 by examining the case where $\phi_{ref} = 0$ and neglecting all sources of phase noise except $\phi_{n,vco}$,

$$\phi_{out} = \phi_{n,vco} - \frac{\phi_{out} K_{PFD} Z(s) K_{VCO}}{Ns}$$

$$\phi_{out}\left(1 + \frac{K_{PFD} Z(s) K_{VCO}}{Ns}\right) = \phi_{out}\left(\frac{Ns + K_{PFD} Z(s) K_{VCO}}{Ns}\right) = \phi_{n,vco}$$

$$\frac{\phi_{out}}{\phi_{n,vco}} = \frac{Ns}{Ns + K_{PFD} Z(s) K_{VCO}} = \frac{1}{1 + \frac{K_{pfd} Z(s) K_{vco}}{Ns}} = \frac{1}{1 + A_{OL}(s)\beta} \qquad (3.19)$$

$A_{OL}(s)\beta$ is simply the loop gain of the PLL—low frequency phase modulation from the VCO is suppressed by the PLL but high frequency phase modulation is not, therefore the VCO’s phase noise has a high-pass behavior when referred to the PLL output. The combined phase noise contributions of the reference and VCO are shown in Figure 3.8—at low frequencies, reference and charge pump noise is multiplied by N^2 and can easily dominate, while at high frequencies the VCO phase noise typically dominates [1], [3].

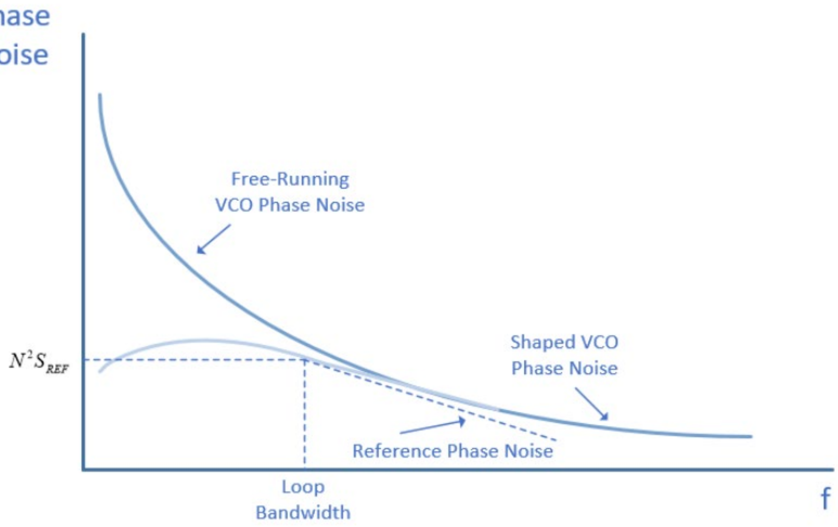


Figure 3. 8 Combined PLL phase noise contributions. Adapted from [1]

3.3 Loop Parameters and System-Level Simulations

System-level simulations were done using MATLAB and the Mixed Signal Blockset in Simulink. Table 3.1 shows the loop parameters for this work:

Loop Parameter	Value
f_{ref}	12MHz
N	100
K_{PFD}	10uA
K_{VCO}	$225\frac{MHz}{V}$
C_1	8.83pF
C_2	114.95pF
C_3	90pF
R_2	37.7kΩ
R_3	830Ω

Table 3. 1 Loop Parameters for this work

The open loop transfer response of the PLL is shown in Figure 3.9a and the closed loop transfer function is shown in Figure 3.9b.

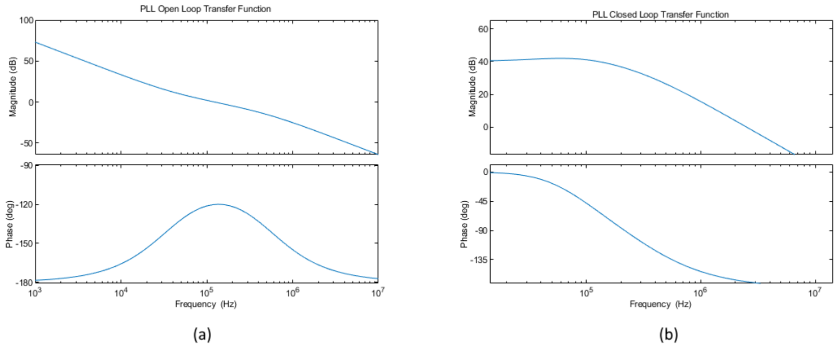


Figure 3. 9 (a) Open loop response of PLL. (b) Closed loop response of PLL

The loop bandwidth (ω_{3dB} of the closed loop response) was found to be 193kHz. The phase margin ($180° - \angle\frac{A_{OL}(j\omega_u)}{N}$) was found using the open loop response: the open loop unity bandwidth is 125kHz and the phase at this frequency is 120.1° degrees, therefore the phase margin is 59.9° (~60°).

A Simulink-based system-level representation of the 2^{nd} order DSM is shown in Figure 3.10a. Its output spectrum is shown in Figure 3.10b. As expected of a 2^{nd} order DSM, it exhibits a high-pass behavior with approximately 40dB/decade roll-off.

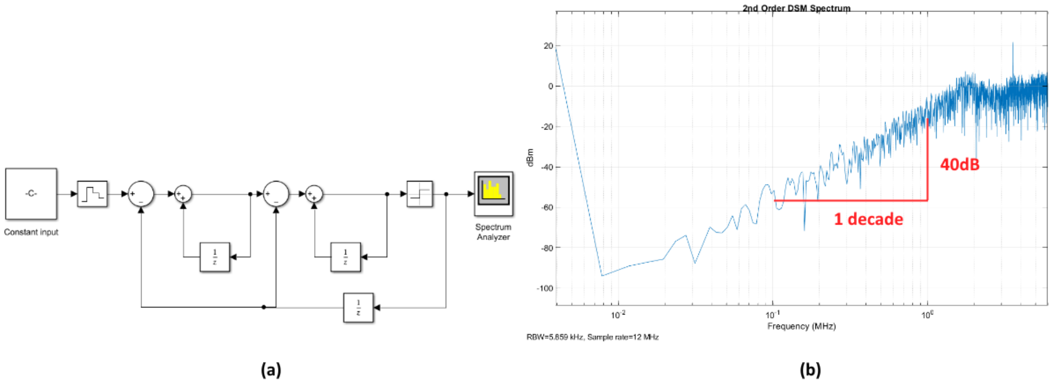


Figure 3. 10 (a) Simulink model of 2^{nd} order DSM. (b) Output spectrum of Simulink model

A Simulink-based system-level representation of this work's fractional-N PLL is shown in Figure 3.11. All PLL-specific blocks (the PFD + charge pump, loop filter, VCO, and dividers) are part of the Mixed Signal Blockset.

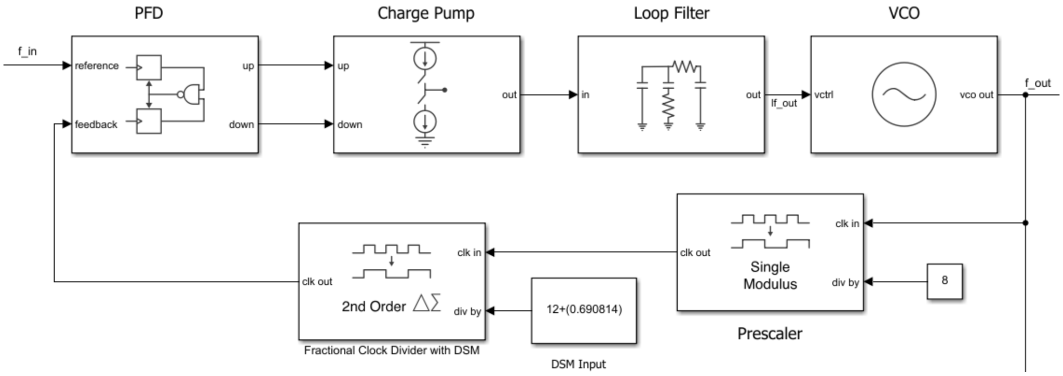


Figure 3. 11 Fractional-N PLL Simulink Model.

For examining the output spectrum, a unit step $u(t - t_{lock})$ is multiplied by the output signal so the output is only sampled after it has achieved lock. The transient response of the loop filter is shown in Figure 3.12a, showing a lock time of approximately 60us. The output spectrum is shown in Figure 3.12b.

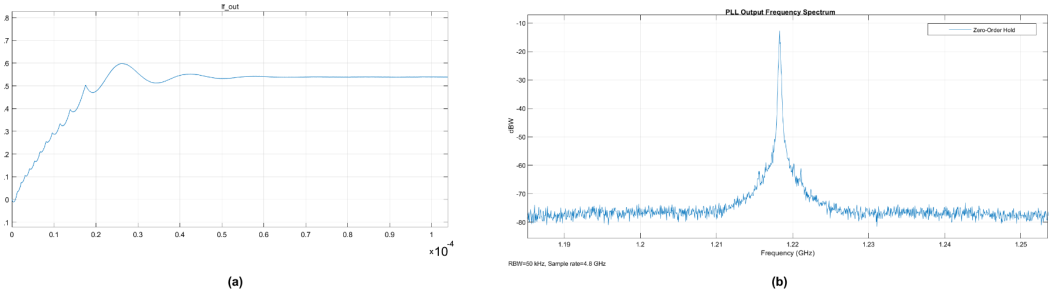


Figure 3. 12 (a) Transient response of loop filter in Simulink. (b) PLL output spectrum

3.4 TSPC D Flip-Flop Circuit Design

Multiple blocks of this work utilize D flip-flops, including the frequency divider, the delta-sigma modulator, and a modified version in the PFD. True single-phase clock (TSPC) D flip-flops are utilized in many modern frequency synthesizer designs. Their appeal stems from reliability at high speeds, lower power dissipation than other common architectures (CML/SCL, transmission gate based, NAND/NOR based, etc.), and only requiring a single clock [26], [33].

The TSPC D Flip-flop of Figure 3.13 operates as follows: in the pre-charge phase (when CLK is 0), node Y is pulled up to VDD. If D = 0 at this point, node X is also pulled up. When CLK = 1 and D = 0, node X stays high, pulling node Y down to 0, turning on PM11 and thus making $\bar{Q} = 1$ and $Q = 0$. Similarly, when D is 1, node X is pulled down. Then, when CLK = 1 and D = 1, Y stays high and pulls $\bar{Q}$ to 0, making $Q = 1$.

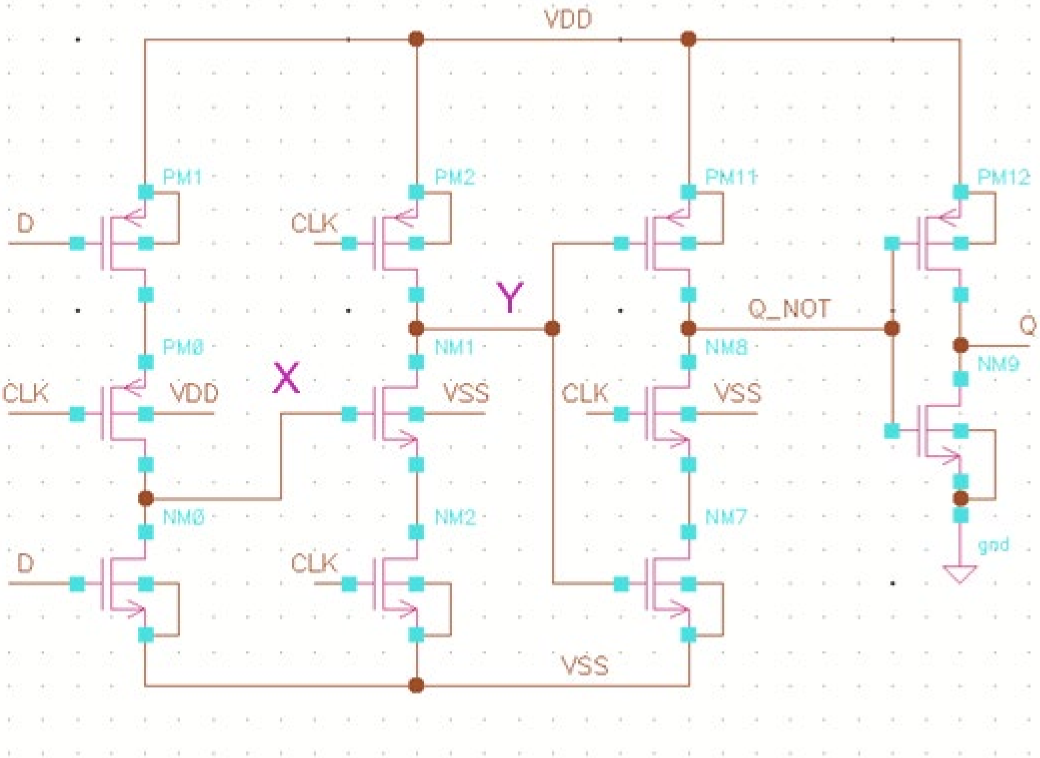


Figure 3. 13 TSPC D Flip-Flop Architecture

3.5 Phase Frequency Detector Circuit Design

As previously shown in Figure 2.4, D flip-flop-based PFDs can be implemented by tying D to VDD and having the two inputs (f_{ref} and f_{fb}) tied to the clocks. However, in the TSPC D flip-flop circuit of Figure 3.13, this renders several transistors redundant. With D tied to VDD, NM0 will always be on and PM1 will always be off—this means the gate of NM1 will always be 0. A reduced transistor (and reduced gate delay) equivalent TSPC PFD, adapted from [34], is used as shown in Figure 3.14.

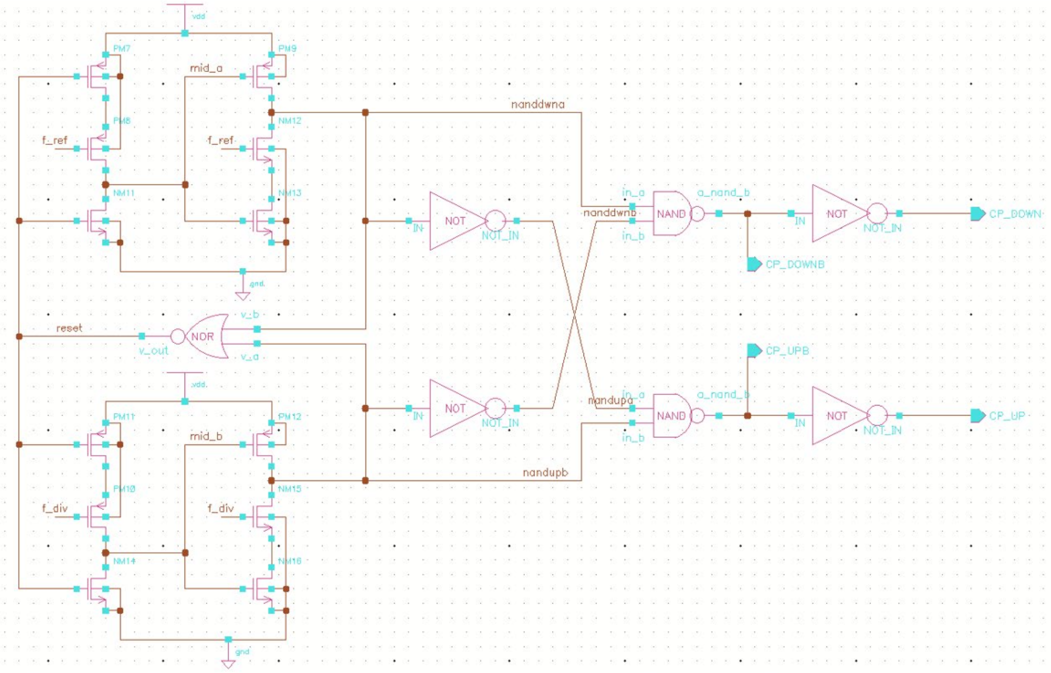


Figure 3. 14 TSPC PFD with complementary outputs and glitch removal. Adapted from [34], [35].

Extra logic is used to correct for small glitches that would normally occur at the output on some rising and falling edges—these normally occur when the PFD output changes state from one output being high to both being low. These small glitches cause the switches of the charge pump to briefly turn on, leading to changes at the loop filter output and causing small phase error. The output of the PFD without this glitch removal circuitry is shown in Figure 3.15a—these glitches can be seen on the falling edge of CP_{UP} (when CP_{DW} and $\overline{CP_{DW}}$ briefly spike) and the falling edge of CP_{DW} (when CP_{UP} and $\overline{CP_{UP}}$ briefly spike). The output with the glitch removal circuitry is shown in Figure 3.15b—CP_{UP} and $\overline{CP_{UP}}$ are prevented from switching on the falling edges of CP_{DW} and any small spike is <10mV, significantly less than the V_T of the transistor switches of the charge pump [35].

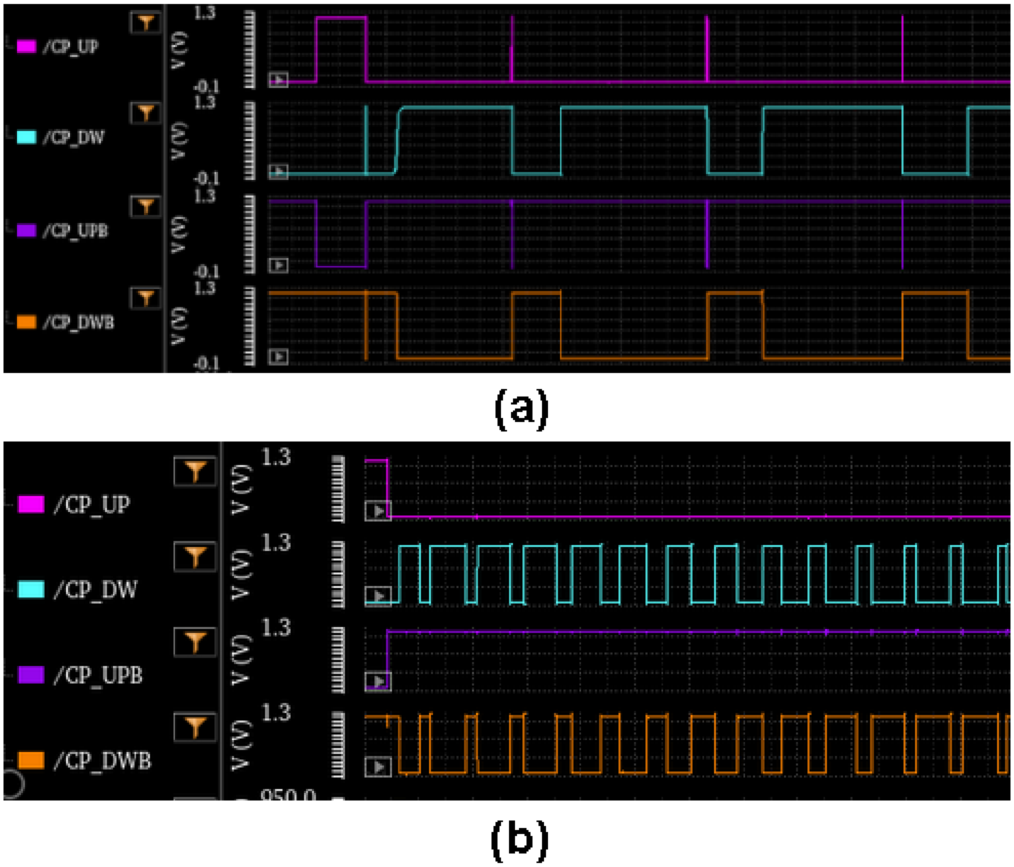


Figure 3. 15 (a) PFD output without glitch correction. (b) PFD output with glitch correction.

3.6 Charge Pump

The current steering charge pump in this work, shown in Figure 3.16, uses complementary inputs and only NMOS switches. A 10uA current source is mirrored to each path, consuming approximately 12uW per current source/mirror. The use of complementary inputs ensures that the current mirror transistors stay in saturation, rather than switch been cutoff and saturation and causing transient spikes. The use of NMOS switches avoids a timing mismatch that is inherent to traditional charge pump designs; when a PMOS switch is used for the UP current source, it must be turned on by $\overline{UP}$ rather than UP, which adds a delay due to the needed inverter. This delay is reduced by using an NMOS switch with a PMOS current mirror. Additionally, using only NMOS switches avoids any mismatch that would occur when using a mix of PMOS and NMOS switches [36].

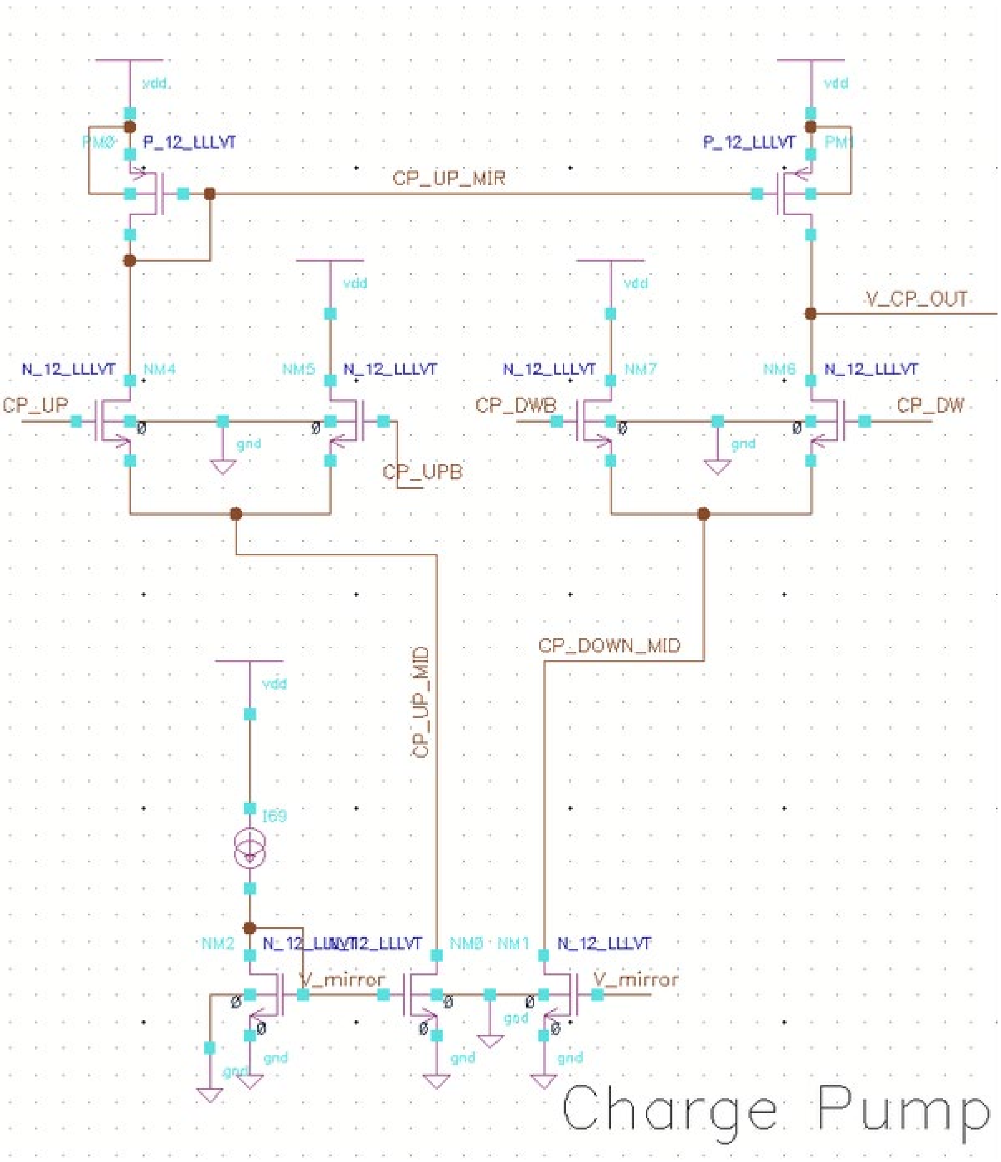


Figure 3. 16 Charge pump with NMOS switches and complementary inputs. Adapted from [36]

The switches of the charge pump were minimally sized to reduce switching time, but this is not the case for current mirror transistors. To a first order approximation, the drain current I_{Ds} of a MOSFET in saturation in only proportional to the gate-source voltage V_{GS} as shown in Equation 3.20:

$$I_{Ds} = \frac{1}{2}\mu C_{ox}\frac{W}{L}(V_{GS} - V_t)^2 \tag{3.20}$$

where μ is the carrier mobility, C_{ox} is the oxide capacitance per unit area, W is the transistor width, L is the length, and V_t is the threshold voltage. However, a second-order effect known as channel length modulation (CLM) occurs due to channel pinch-off. Pinch-off decreases the effective channel length by some amount ΔL as V_{DS} increases. As a result, CLM also makes I_{DS} dependent on the drain-source voltage V_{DS} as shown in equation 3.21:

$$I_D = \frac{1}{2}\mu C_{ox}\frac{W}{L}(V_{GS} - V_t)^2(1 + \lambda V_{dS}) \tag{3.21}$$

where λ is the CLM coefficient [8].

Modeling a PLL assumes the charge pump current is constant, regardless of V_{cont} and the voltage of the node preceding it in the loop filter. Channel length modulation will inevitably cause some variations in I_{cp}, but it is still possible to mitigate these variations. Two possible approaches to are to utilize a cascode current mirror or to simply increase the current mirror transistor width. A cascode structure would reduce variations in V_{DS} (and thus I_{DS}) but would also reduce voltage headroom due to the additional voltage drops required to keep all current mirror transistors in saturation. As this work operates on a 1.2V supply, this is not a practical solution to dealing with channel length modulation.

Instead, the length of the current mirror transistors was increased—for a fixed change in channel length ΔL, this results in a smaller % change to the effective length of the channel and thus a smaller % change to the drain current. The width was also increased to keep the same $\frac{W}{L}$. Increasing the length has diminishing returns, however. Figure 3.17 shows a plot of I_D vs V_{ds} for different transistor sizes (at multiples of the gate length from 1x to 5x). Scaling beyond 3x minimum length shows diminishing returns. If the current mirror transistors are too large, the gate-drain capacitance C_{gd} can create a clock feedthrough path that causes ripple at V_{cont} [3].

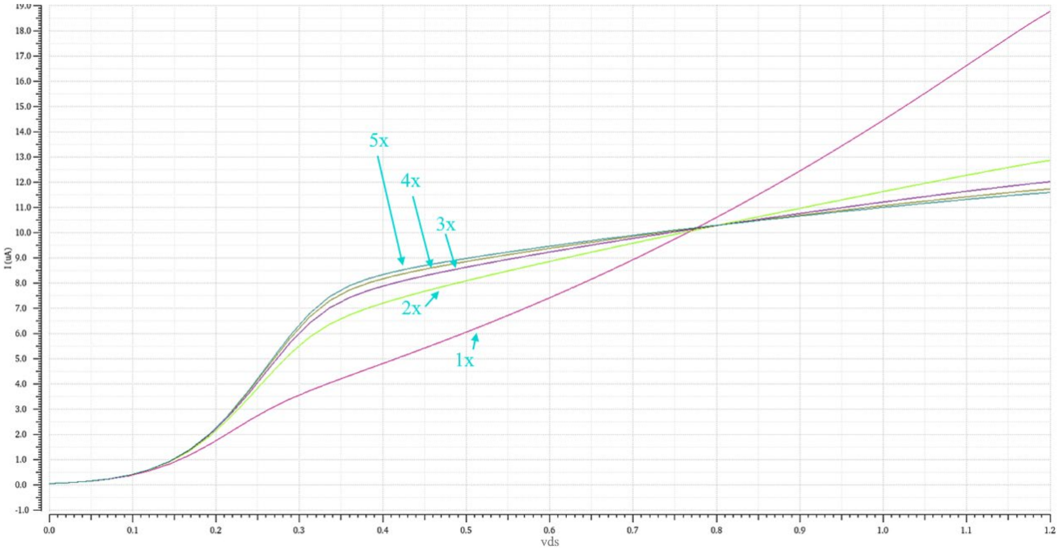


Figure 3. 17 Plot of V_{ds} vs I_d for increasingly large transistors

3.7 Loop Filter

To minimize total power consumption, a passive filter was utilized rather than an active one requiring op-amps. For this work, a third order passive RC filter was utilized as shown in Figure 3.18. Making C3 large was especially important to reduce ripple at V_{cont} from two factors—first, when the system is locked there are small changes in V_{cont} due to the charge pumps occasionally turning on to keep V_{cont} constant as C_3 discharges. Additionally, the VCO output itself can cause oscillations at V_{cont} due to the capacitance between the nodes from the varactor. Viewing the path from V_{VCO} through the varactor to the loop filter to ground as a voltage divider shows that increasing C_3 decreases Z_3, reducing the voltage drop across it and thus reducing ripple at V_{cont}.

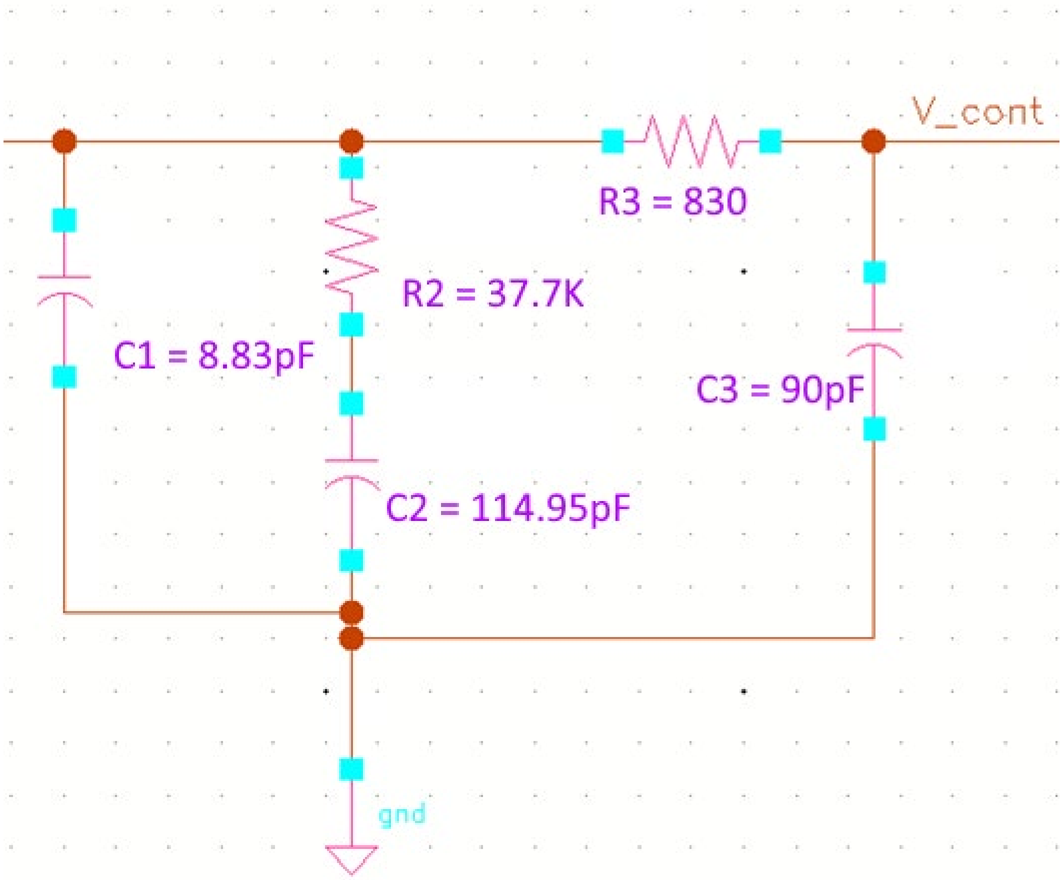


Figure 3. 18 Third order passive low pass filter

3.8 Voltage Controlled Oscillator

A 2.4GHz cross-coupled CMOS LC VCO was used in this work, as shown in Figure 3.19. There is a constant metal-oxide-metal (MOM) capacitor C_1 and a voltage-controlled capacitor known as a varactor C_{VAR}. In this VCO, L = 6.34257nH, C_1 = 23.0224fF, and $C_{VAR-MAX}$=792.4963fF. Taking these into account, the resonance frequency is calculable as $\omega_{RES} = \frac{1}{\sqrt{L(C_1+C_{VAR})}}$ [1], [3].

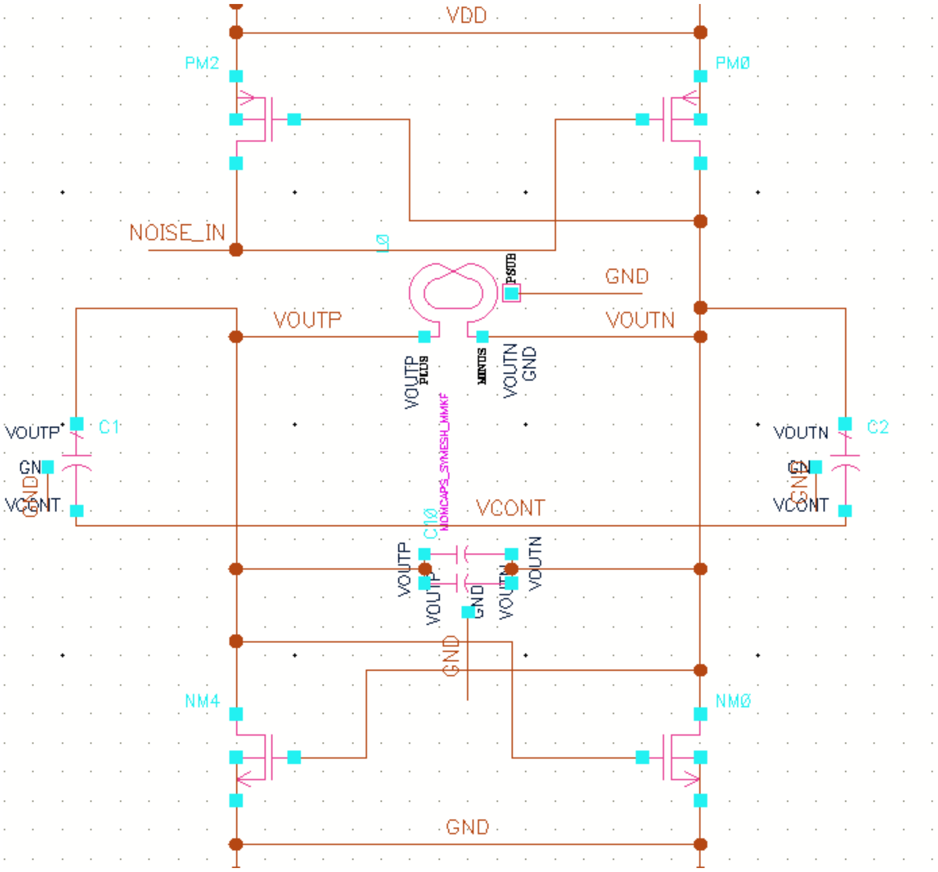
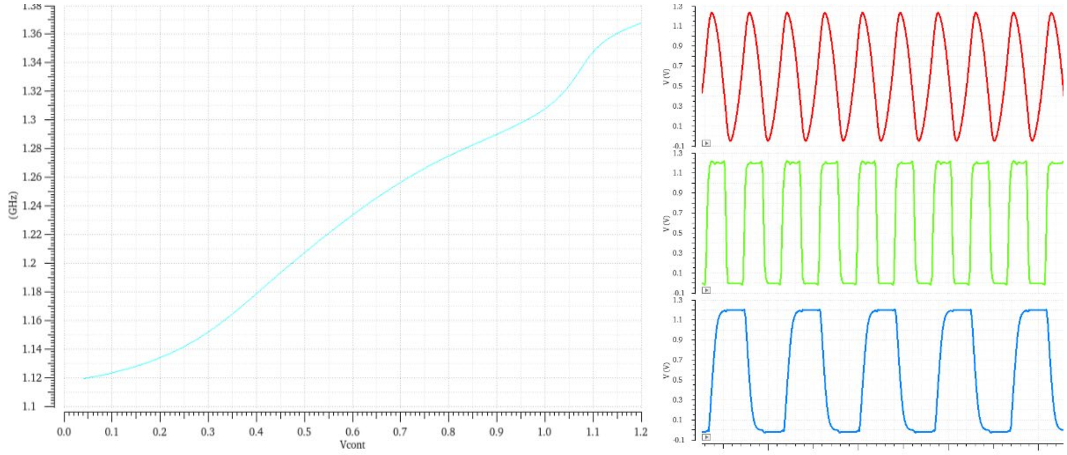


Figure 3. 19 LC Oscillator utilized in this work

As the feedback network (composed of the prescaler and toggleable frequency divider) of this PLL is digital in nature, the sinusoidal output of the LC oscillator is buffered through two digital inverters to produce a square wave. Additionally, as this system is designed with a 1.2GHz output in mind, an additionally divide-by-2 frequency divider is placed before the output. Taking this additional divider into account, the effective VCO tuning curve is shown in Figure 4.8a. As the tuning curve is not a perfect line, K_{VCO} is slightly dependent on V_{cont}; a linear fit in the center of the tuning curve gives an approximate K_{VCO} (used in this work's system analysis) as $225\frac{MHz}{V}$. The output waveforms at the output of the of the VCO, buffer, and first divider are shown in Figure 3.20b.



(a) (b)

Figure 3. 20 (a) VCO Tuning Curve (b) LC VCO output waveforms

3.9 Frequency Divider

The feedback path of this frequency synthesizer is comprised of two parts: a fixed divide-by-8 prescaler and a divide-by-12/13 toggleable frequency divider. Thus, the effective division ratio is divide-by 96/104. The divide-by-8 prescaler is implemented by simply cascading three divide-by-2 dividers together, as shown in Figure 3.21. The D flip-flops utilized in this prescaler are the same TSPC architecture shown in Figure 3.13.

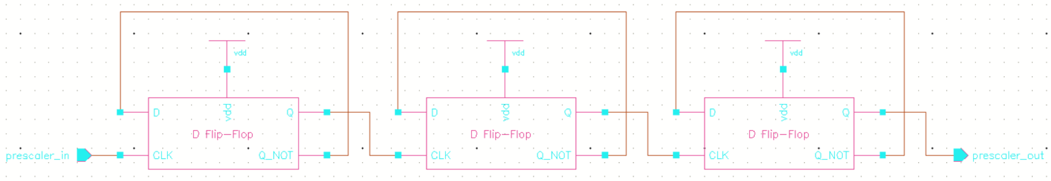


Figure 3. 21 Divide-by-8 prescaler

The toggleable frequency divider is implemented using a modular design composed of the divide-by-2/3 cells shown in Figure 3.22:

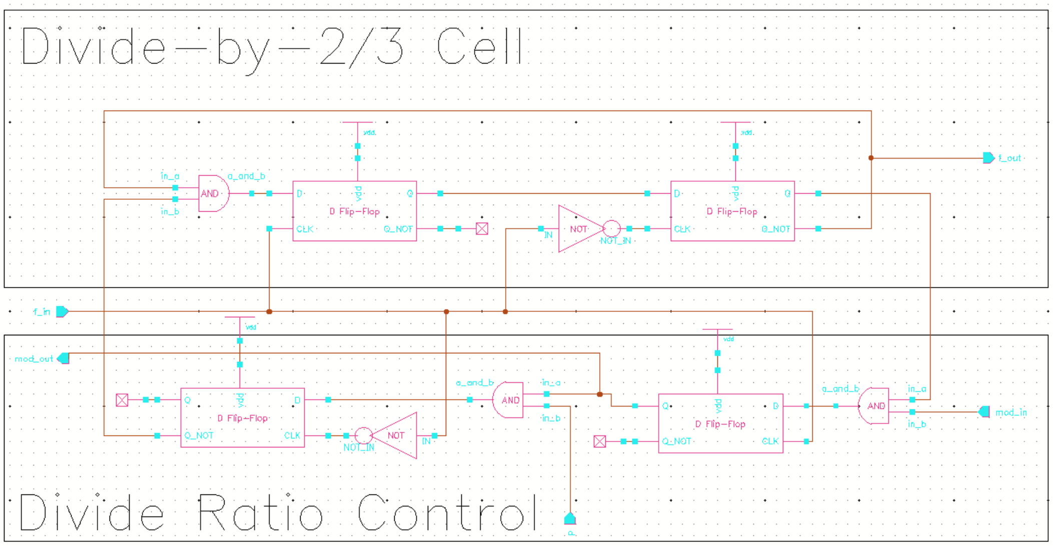


Figure 3. 22 Schematic of Divide-by-2/3 Cell. Adapted from [14]

The full modular frequency divider is shown in Figure 3.23, with the design adapted from [14]. For a circuit with n cells, the division ratio of each 2/3 cell is set by the control pins $p_0, p_1, \ldots p_i, \ldots p_{n-1}$. These control pins are either set to a constant value (1 or 0) or are toggled by the DSM, depending on the desired range of division ratios. The signal mod_i is passed from cell to cell and the control pin p_i is checked. If it is 0, the cell divides by 2; if it is 1, the cell swallows a pulse and the cell divides by 3. For a modular frequency divider with n cells, the total output period T_{out} is

$$T_{out} = T_{in}(2^n + 2^{n-1}p_{n-1} + 2^{n-2}p_{n-2} + \cdots + 2p_1 + p_0) \quad (3.22)$$

where T_{in} is the input period $(\frac{1}{f_{in}})$. This allows for a division range between 2^n and $2^{n+1} - 1$. Three 2/3 cells were used in this work, allowing for a division range of 8 to 15. However, due to the range of the VCO, the presence of the divide-by-8 prescaler, and the DSM only utilizing a single-bit output, two of these pins were held constant. To allow this divider to toggle between 12 and 13 (and thus allow the full division ratio to toggle between 96 and 104), p_2 is tied high, p_1 is tied low, and p_0 is controlled by the DSM [14].

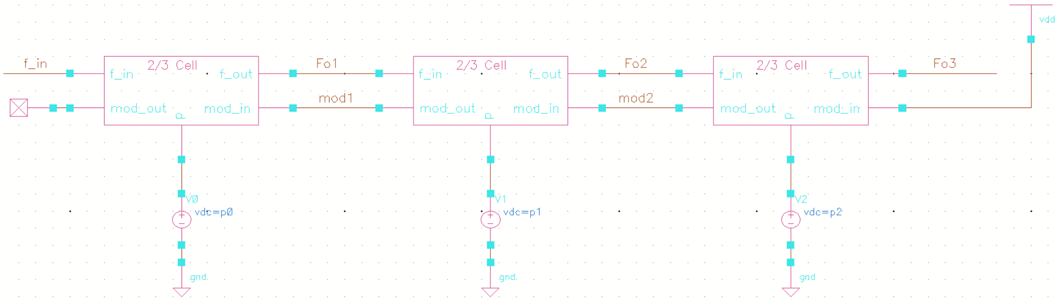


Figure 3. 23 Top level view of modular frequency divider. Adapted from [14]

3.10 Delta-Sigma Modulator

This work utilizes a 2nd order single loop DSM for toggling the frequency divider as shown in Figure 3.24. The input is a 16-bit binary word, but all blocks have a total of 20 bits: 1 bit is for a sign bit and 3 are to avoid incorrect outputs due to an overflow (above $2^{16} - 1$) or underflow (below $-(2^{16})$). An N-bit input theoretically allows for an output resolution of $\frac{f_{ref}}{2^N}$; in the case of this work, $\frac{12MHz}{2^{16}} = 181.1Hz$ resolution.

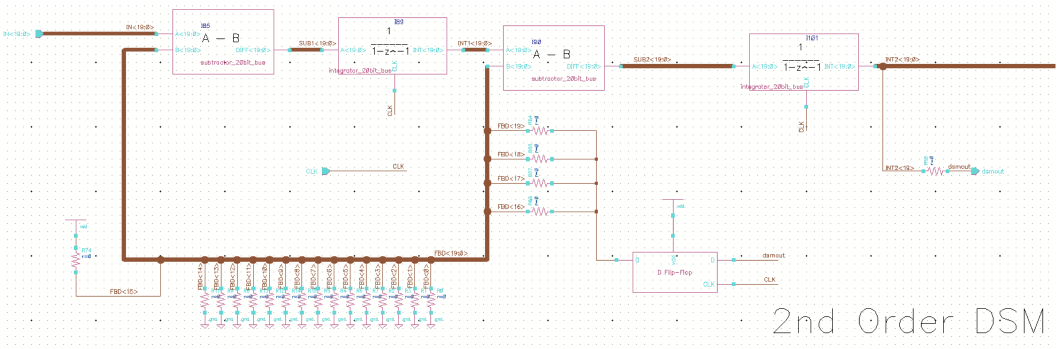


Figure 3. 24 Second order DSM schematic

The DSM and all operations within it are implemented by utilizing two blocks: delay cells and adders. Delay cells (z^{-1} in the z domain) are implemented with TSPC D flip-flops in parallel as shown in Figure 3.25. This figure shows only four bits for the sake of keeping the image readable, but the function extends to 20 bits as used in this work. The input (D) is simply a 20-bit word $X_0, X_1, \ldots X_{19}$, each flip-flop has its clock tied together, and the output (Q) is the output delayed by one clock cycle ($z^{-1}X_0, \ldots z^{-1}X_{19}$).

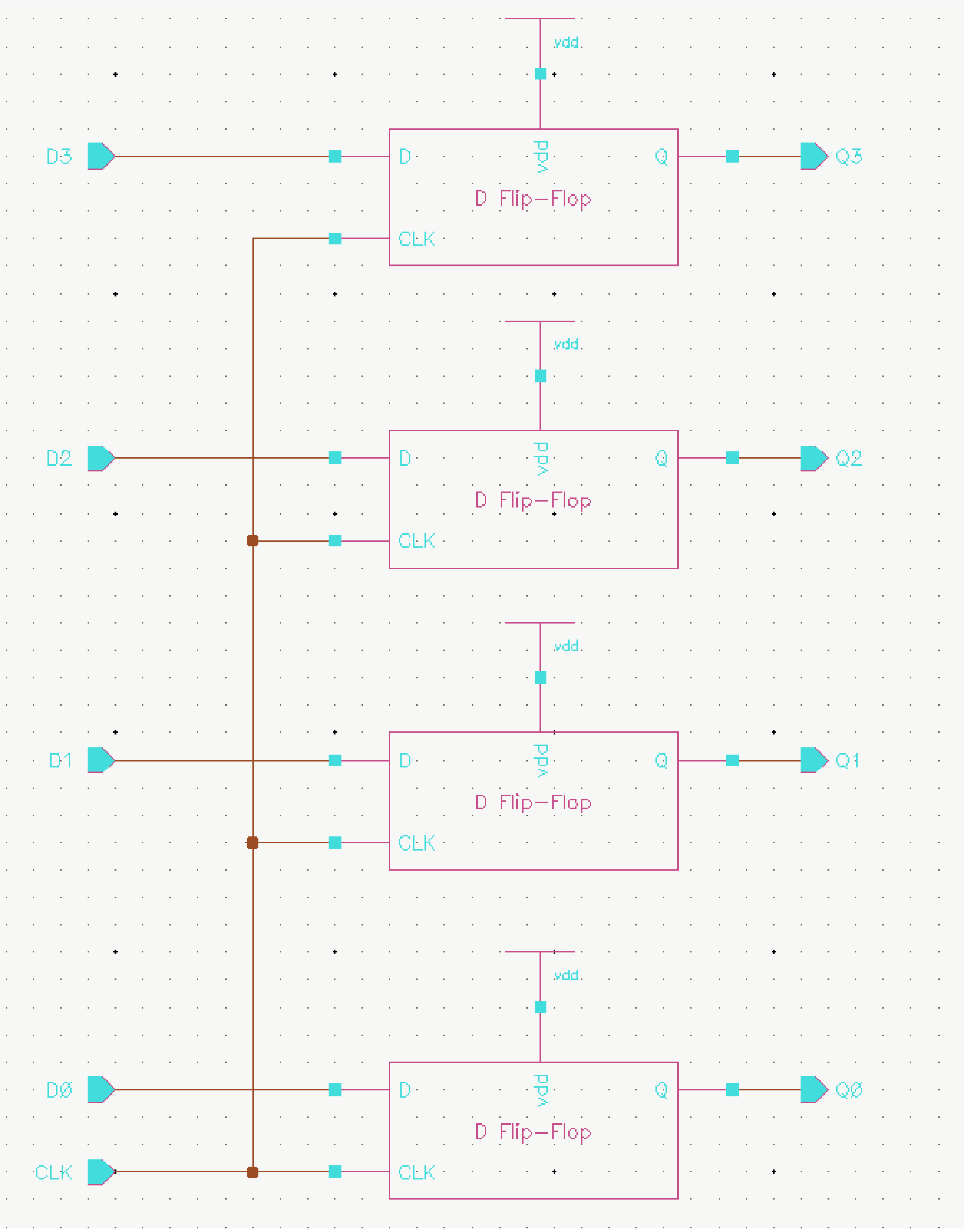


Figure 3. 25 Multi-bit delay cell using TSPC D flip-flops

For the adder cells, a ripple carry adder (RCA) architecture was utilized. Other architectures such as carry lookahead (CLA) and carry skip ahead (CSkA) are common in many modern digital circuit designs, but the RCA architecture was deemed sufficient in terms of speed and superior in terms of power consumption.

CLA and CSkA architectures (among others) achieve higher speeds, but this is not necessary when the DSM is operating at a fixed clock rate (as long as the propagation delay of the RCA is less than a clock period). Adder architectures such as CLA and CSkA also require additional logic, increasing the complexity and power consumption of the circuit [37]. The individual adder cell used in this work is a mirror adder cell, as shown in Figure 3.26a. As the carry out is inverted in this architecture, an inverter must be placed between $Cout_i$ and Cin_{i+1}, as shown in Figure 3.26b.

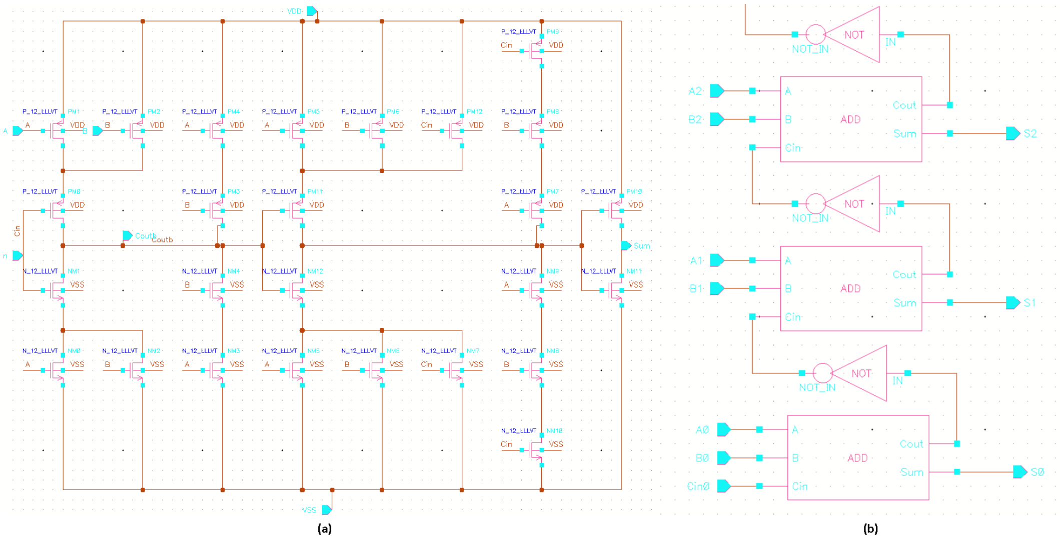


Figure 3. 26 (a) Mirror adder cell. (b) Ripple carry adder implementation

The integrator block ($\frac{1}{1-z^{-1}}$), shown in Figure 3.27, is implemented with an adder and a delay cell; the input is tied to one of the adder inputs (in this case, $A_0 - A_{19}$) and the output is tied to the input of the delay cell. The delay cell's output is tied to the other adder input ($B_0 - B_{19}$).

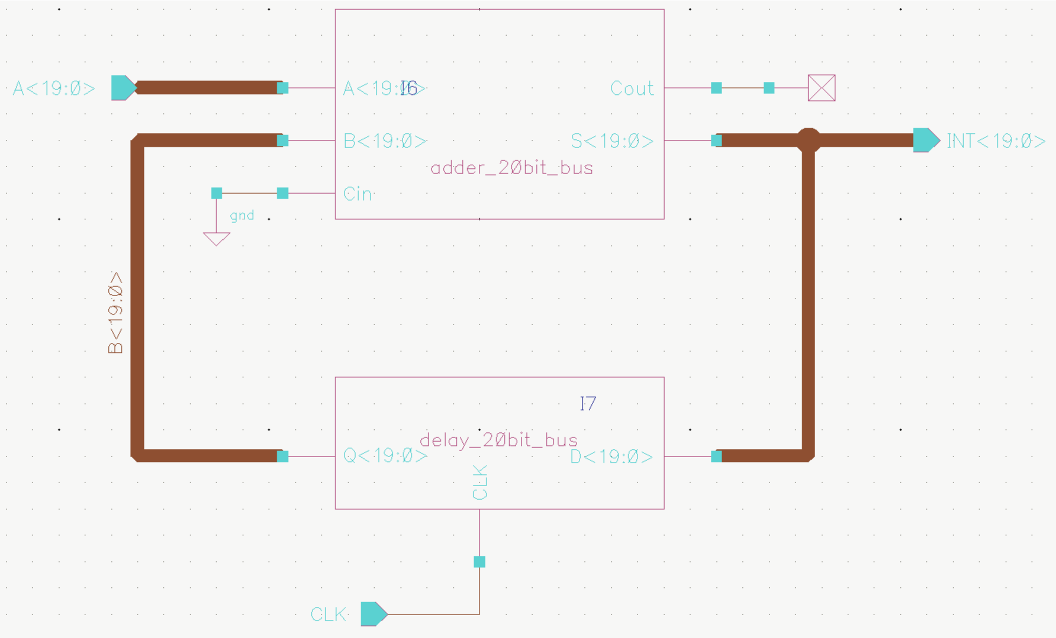


Figure 3. 27 Digital integrator block

No additional circuitry is required for the implementation of the quantizer. As the most significant bit (MSB) of a signed binary word is the sign bit, it is sufficient to simply use the sign bit itself ($INT2_{20}$ in the schematic) as the effective output of the quantizer. If the MSB of the last integrator is 0, then the output is ≥ 0; if the MSB is 1, then the output is < 0. For implementing a 20-bit subtractor, the existing 20-bit adder cell is modified by converting the subtrahend to its two's complement. The two's complement of a binary number is its bitwise complement + 1; this is easily realized in an adder by inverting every bit of the subtrahend and tying the adder's carry in to VDD.

The feedback of the DSM sends the two's complement representation of 1 or -1 to the subtrahend of the first subtractor cell. To feed back 1 (when the output sign bit $INT2_{20}$ is 0), the 20-bit word is 00001000000000000000. To feed back -1 (when the output sign bit is 1), the 20-bit word is 11111000000000000000. This is easily realized by tying the four most significant bits to $INT2_{20}z^{-1}$, setting the fifth bit to 1, and setting the remaining bits to 0. This is seen in Figure 3.24; the 0-ohm resistors are simply used in the schematic to connect each wire of the bus to other named nets without the simulator throwing warnings or errors.

The output spectrum of this work's 2nd order DSM is shown in Figure 3.28. At lower frequencies there is approximately a -70dB noise floor, and as expected for a 2nd order DSM, there is a $40\frac{dB}{decade}$ high-pass roll-off.

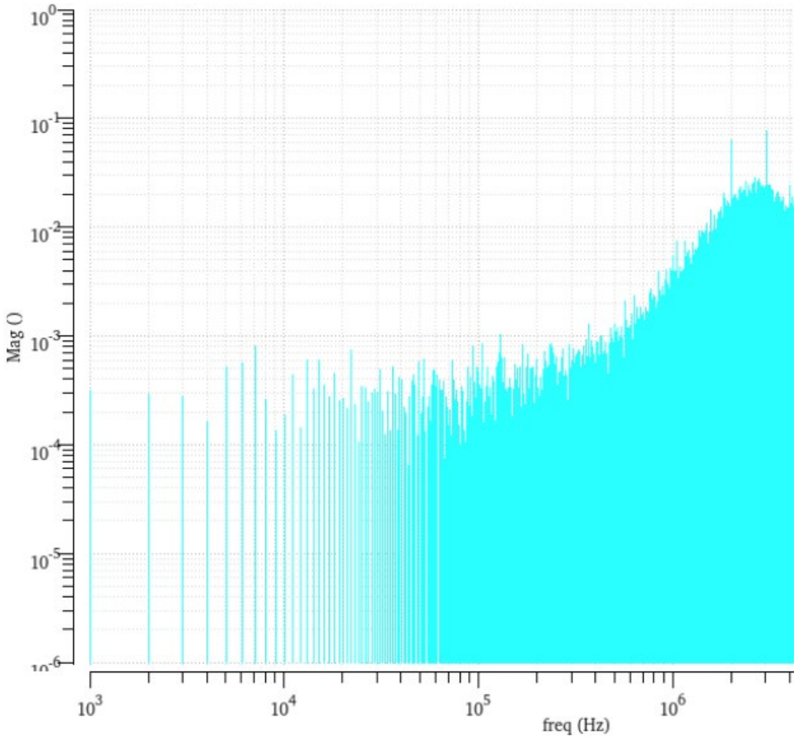


Figure 3. 28 Output spectrum of 2nd Order Delta Sigma Modulator

4 Simulation and Results

The full PLL was simulated in the typical, slow-slow (SS), and fast-fast (FF) process corners. Due to the time required to run the simulation, the slow-fast

(SF) and fast-slow (FS) corners were not run. This work achieved a lock time/switching speed of approximately 60us, as shown in the transient response of V_{cont} in Figure 4.1. Figure 4.1a depicts the SS corner, Figure 4.1b depicts the typical corner, and Figure 4.1c depicts the FF corner. The lock time (no initial condition) and switching speed (with an initial condition Even after locking, there is a small amount of ripple at V_{cont}; while increasing C_3 significantly reduces the ripple (as described in section 4.4), it does not completely eliminate it.

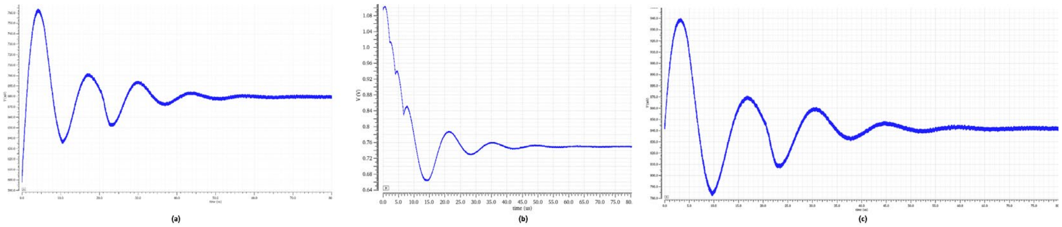
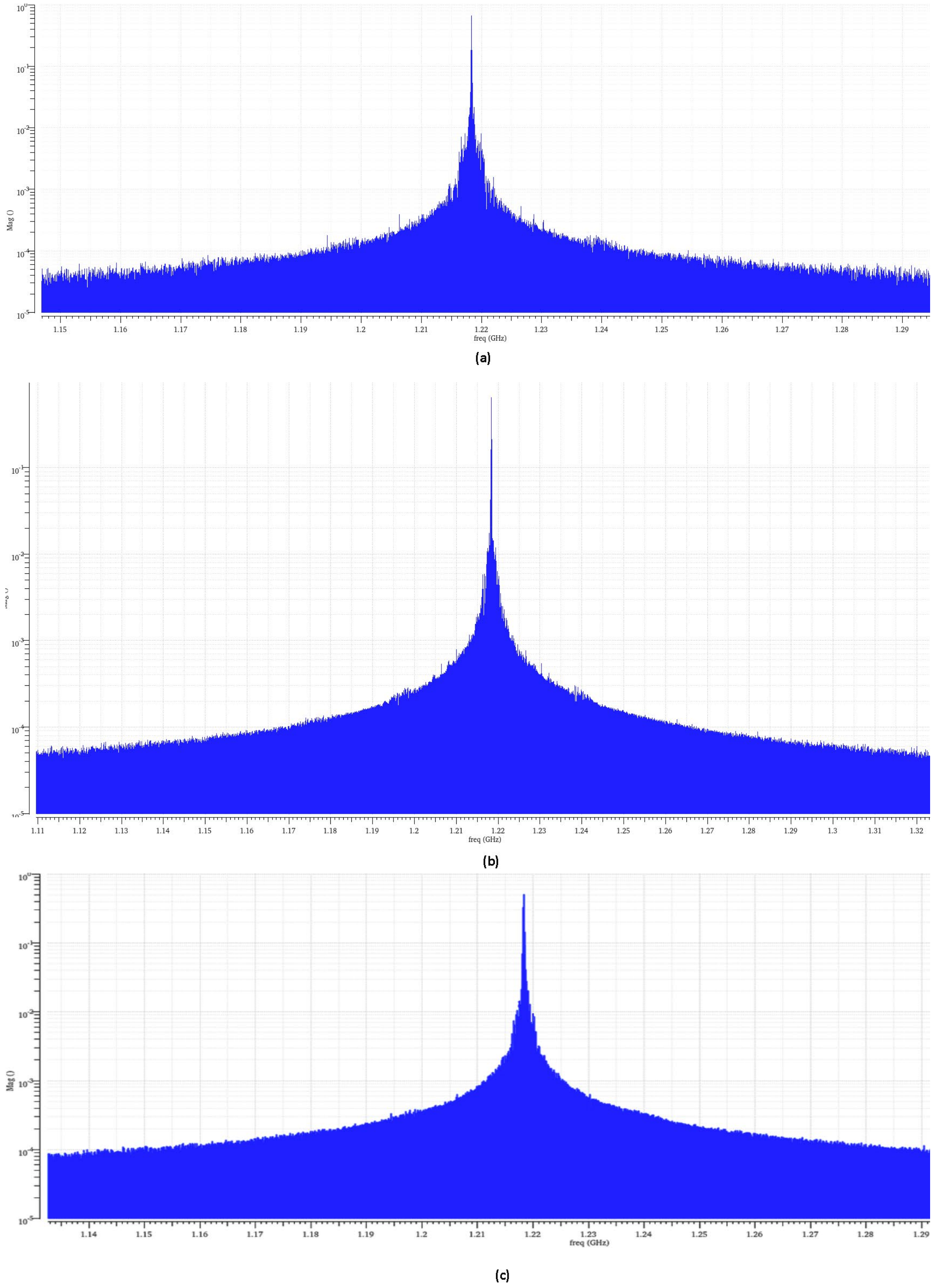


Figure 4. 1 (a) Transient response of V_{cont}, SS corner (b) Typical (c) FF

The output spectrums of the PLL for the SS, typical, and FF corner are shown in Figure 4.2a, Figure 4.2b, and Figure 4.2c, respectively. The center frequency of the output spectrum in each simulation is 1.2183 GHz—a more precise result would have required a significantly longer simulation due to the nature of the discrete Fourier transform (DFT) used to generate the output spectrum. The average output of the DSM was found to be 0.690814; therefore, the expected output frequency is $12MHz \cdot 8 \cdot (12 + 0.690814) = 1.281319GHz$—this is quite close to the output center frequency seen in the output spectrum.



(a)

(b)

(c)

Figure 4. 2 (a) Output spectrum of PLL, SS corner, (b) Typical (c) FF

The VCO was examined separately for examining out of-band phase noise due to normal phase noise measurement techniques not being applicable for fractional-N PLLs. Periodic steady state and harmonic balance analyses rely on a periodic output; due to the frequency divider toggling, the instantaneous output frequency is varying over time, even when the circuit is locked. The VCO phase noise plot is shown in Figure 4.3. At a 1MHz offset, the VCO has a phase noise of -120dBc/Hz.

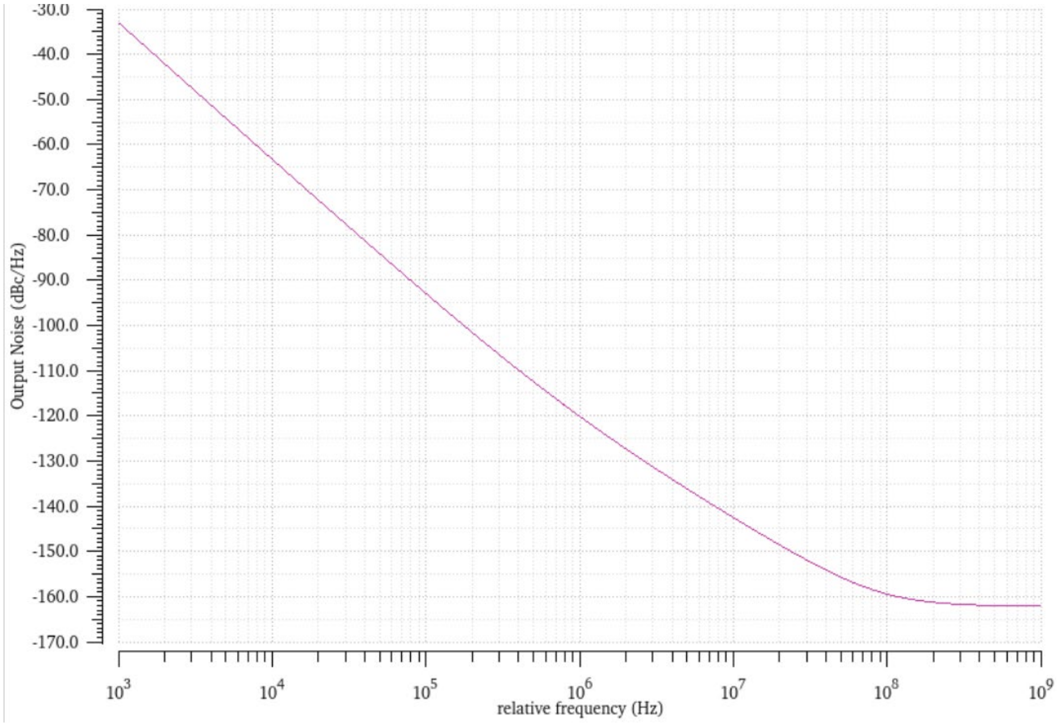


Figure 4. 3 Phase noise plot of VCO

The total power consumption of the entire frequency synthesizer was found to be 3.197mW. This was found by integrating the supply current over a time period, dividing by this period, and multiplying by the supply voltage as follows: $P_{total} = V_{DD} \int_{T_1}^{T_2} \frac{I_{supply}(t)dt}{T_2 - T_1}$. Excluding the VCO and the additional divide-by-2 frequency divider needed to generate a 1.2GHz output, the rest of the PLL consumed 633.7uW.

5 Conclusion and Future Work

A fractional-N PLL-based frequency-synthesizer was designed and simulated on a system level and transistor level. The open loop and closed loop responses, generated in MATLAB, show a loop bandwidth of approximately 193kHz and a phase margin of 60°. Simulink-level simulations of the DSM and PLL were run to verify functionality before transistor-level implementation. For the transistor-level implementation, multiple design choices were made to improve performance and reduce power consumption. The PFD and D flip-flops were implemented with TSPC logic. The PFD also has additional glitch-correction logic to avoid erroneous pulses turning the charge pump on when the PFD is changing states. The charge pump used an NMOS switch and current steering architecture, to avoid mismatch that may arise from a traditional charge pump with both NMOS and PMOS switches. The charge pump current was chosen to be 10uA and a 3rd order passive loop filter was used to minimize total power consumption. The frequency divider contains a divide-by-8 prescaler and a modular divider configured to toggle between 12 and 13. A 2nd order single loop delta sigma modulator was used for the toggling of this divider. Transistor-level simulations showed a lock time of 60us. For the chosen input to the DSM, a center frequency of 1.2183GHz was synthesized, which matches the expected frequency based on the average output of the DSM. Total power consumption was 3.197mW—excluding the VCO and additional divider needed for a 1.2GHz output, the rest of the PLL (divider, DSM, PFD, and charge pump) consumed 633.7uW.

A comparison of this work to those mentioned in Section 2.9 is shown in Table 5.1:

Work	Supply Voltage	Frequency	Lock Time	Total Power Consumption	Non-VCO Power	Phase Noise @ 1MHz
[18]	0.8V	1.2GHz	200us	4.92mW	1.72mW	-121dBc/Hz
[19]	1V	2.4GHz	12us	2.94mW	1.82mW	-120dBc/Hz (1.9GHz)
[20]	1V	2.4GHz	25us	8.94mW	N/A	-126.85dBc/Hz
[21]	1.2V	1.7-2.5GHz	40us	1mW-1.13mW	550-618uW	-115dBc/Hz
[22]	0.65V	2.5GHz	N/A	6mW	4mW	-111dBc/Hz
[23]	0.68V	2.4GHz	130us	680uW	170uW	-110dBc/Hz
[24]	0.55V	1GHz	N/A	3mW	N/A	-106dBc/Hz
[25]	1.36V	900MHz	N/A	7.48mW	N/A	-116dBc/Hz
This work	1.2V	1.2GHz	60us	3.197mW	633.7uW	-120dBc/Hz

Table 5. 1 Comparison with other works

This work was designed with for 1.2GHz output range but utilized an existing 2.4GHz VCO—this necessitated the additional frequency divider and is a part of the reason why the power consumption of the loop excluding the VCO is also examined. Future improvements on this design will utilize a 1.2GHz VCO to avoid unnecessary power consumption.

Additionally, this work may be improved upon by utilizing a higher order delta-sigma modulator for the toggling of the divider. A 2nd order DSM was used for this work due to its inherent stability and the simplicity of a 1-bit output, but there is a trade-off of inferior noise shaping compared to higher order DSMs. If a higher order single loop DSM is chosen, its NTF must be modified to avoid overloading. If a MASH DSM is chosen, stability within each individual loop is not a concern, but additional output bits are needed and the multi-bit output must be capable of toggling the divider without overflow or underflow to maintain the proper divide ratio [15], [16].

Besides increasing the order, other techniques may be examined to improve the performance of the DSM and thus the PLL. As a DSM has a finite number of bits and output levels, it is a finite state machine—after some amount of cycles, the DSM will repeat its series of outputs in a cycle. If this cycle is short, then the quantization noise will appear more prominently at a few tones, which will in turn appear within the PLL as phase noise—to avoid this, there are techniques to maximize cycle lengths in DSMs. Setting initial conditions on the integrators to an odd value can ensure a lower limit on the number of states before the DSM's cycle repeats. [15].

References

[1] B. Razavi, *RF Microelectronics*, 2nd ed. Prentice Hall, 2012.

[2] M. Woolley, “Understanding Reliability in Bluetooth® Technology,” no. December, pp. 1–43, 2020.

[3] B. Razavi, *Design of CMOS Phase-Locked Loops*. 2020.

[4] Pozar David M, “Microwave_and_Rf_Design_of_Wirele.” pp. 1–379, 2001.

[5] A. V. Chenakin, V. N. Kochemasov, and A. V. Pestryakov, “Current State and Development Trends of Microwave Frequency Synthesizers,” *2018 Syst. Signal Synchronization, Gener. Process. Telecommun. SYNCHROINFO 2018*, vol. 1, 2018, doi: 10.1109/SYNCHROINFO.2018.8457033.

[6] A. Chenakin, “Frequency synthesis: Current solutions and new trends,” *Microw. J.*, vol. 50, no. 5, pp. 256–266, 2007.

[7] J. Vankka, “Direct Digital Synthesizers: Theory, Design and Applications,” Helsinki University of Technology, 2000.

[8] B. Razavi, *Design of Analog CMOS Integrated Circuits 2nd edition*. 2017.

[9] D. Banerjee, *PLL Performance, Simulation and Design, Fourth Edition*. 2006.

[10] A. Homayoun and B. Razavi, “Analysis of phase noise in phase/frequency

detectors," *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 60, no. 3, pp. 529–539, 2013, doi: 10.1109/TCSI.2012.2215792.

[11] W. O. Keese, "An Analysis and Performance Evaluation of a Passive Filter Design Technique for Charge Pump Phase-Locked Loops," p. National Semiconductor Application Note 1001, 1996.

[12] S. Hathwalia and N. Grover, "A Comparative Study of Ring VCO & LC-VCO based PLL," *J. Eng. Res. Appl.*, vol. 9, no. 1, pp. 24–29, 2019, doi: 10.9790/9622.

[13] M. Hossain and A. C. Carusone, "CMOS oscillators for clock distribution and injection-locked deskew," *IEEE J. Solid-State Circuits*, vol. 44, no. 8, pp. 2138–2153, 2009, doi: 10.1109/JSSC.2009.2022917.

[14] C. S. Vaucher, I. Ferencic, M. Locher, S. Sedvallson, U. Voegeli, and Z. Wang, "A family of low-power truly modular programmable dividers in standard 0.35μm CMOS technology," *Phase-Locking High-Performance Syst. From Devices to Archit.*, vol. 35, no. 7, pp. 346–352, 2003, doi: 10.1109/9780470545492.ch40.

[15] K. Hosseini and M. P. Kenndy, *Minimizing Spurious Tones in Digital Delta-Sigma Modulators*. Springer.

[16] J. M. de la Rosa and R. del Río, *CMOS Sigma-Delta Converters: Practical Design Guide*. 2013.

[17] W. Rhee, B.-S. Song, and A. Ali, "A 1.1-GHz CMOS fractional-N

frequency synthesizer with a 3-b third-order Delta Sigma modulator," *IEEE J. Solid-State Circuits*, vol. 35, no. 10, pp. 1453–1460, 2000.

[18] W. Wang, H. C. Luong, and S. Member, "A 0.8-V 4.9mW 1.2GHz CMOS Fractional-N Frequency Synthesizer for UHF RFID Readers," *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 55, no. 9, pp. 2505–2513, 2008.

[19] Y. Lee, S. Kim, and H. Shin, "Design of a 1-V 3-mW 2.4-GHz fractional-N PLL synthesizer in 65nm CMOS," *Proc. - Int. SoC Des. Conf. 2017, ISOCC 2017*, pp. 230–231, 2018, doi: 10.1109/ISOCC.2017.8368867.

[20] S. Y. Lee, C. H. Cheng, M. F. Huang, and S. C. Lee, "A 1-V 2.4-GHz low-power fractional-N frequency synthesizer with sigma-delta modulator controller," *Proc. - IEEE Int. Symp. Circuits Syst.*, pp. 2811–2814, 2005, doi: 10.1109/ISCAS.2005.1465211.

[21] M. Vidojkovic, Y. Liu, X. Huang, K. Imamura, and G. Dolmans, "RMO3C-4 A Fully Integrated 1 . 7-2 . 5GHz 1mW Fractional-N PLL for WBAN," *Symp. A Q. J. Mod. Foreign Lit.*, pp. 185–188, 2012.

[22] S. Yu and P. Kinget, "A 0.65V 2.5GHz Fractional-N Frequency Synthesizer in 90nm CMOS," *ISSCC*, pp. 316–318, 2007.

[23] A. Paidimarri, N. Ickes, and A. Chandrakasan, "A 0.68V 0.68mW 2.4GHz PLL for Ultra-low Power RF Systems," *IEEE Radio Freq. Integr. Circuits Symp.*, pp. 397–400, 2015.

[24] O. Abdelfattah, I. Shih, G. Roberts, and Y. C. Shih, "A 0.55-V 1-GHz

frequency synthesizer PLL for ultra-low-voltage ultra-low-power applications," *2015 IEEE 6th Lat. Am. Symp. Circuits Syst. LASCAS 2015 - Conf. Proc.*, 2015, doi: 10.1109/LASCAS.2015.7250428.

[25] D. L. Yan, A. Bansal, M. K. Raja, A. Yoshida, and M. Je, "A Low Power , 900MHz sizer with Quadrat ture outputs in 0 . 13um CMOS S," *IEEE MTTS IMWS-BIO*, pp. 9–11, 2013.

[26] Z. Deng and A. M. Niknejad, "The Speed – Power Trade-Off in the Design of CMOS True-Single-Phase-Clock Dividers," vol. 45, no. 11, pp. 2457–2465, 2010.

[27] W.-H. Chen, "High-Speed Low-Power True Single-Phase Clock," *IEEE Trans. Circuits Syst. II Express Briefs*, vol. 58, no. 3, pp. 144–148, 2011.

[28] X. Li, J. Gao, Z. Chen, and X. Wang, "High-Speed Wide-Range True-Single-Phase-Clock CMOS Dual Modulus Prescaler," *Electronics*, vol. 9, no. 5, pp. 17–24, 2020, doi: 10.3390/electronics9050725.

[29] K. S. Kundert, "Predicting the Phase Noise of PLL-Based Frequency Synthesizers," *Phase-Locking High-Performance Syst. From Devices to Archit.*, pp. 46–69, 2003, doi: 10.1109/9780470545492.ch5.

[30] B. Zakia and K. Samir, "The impact of PLL loop bandwidth on Frequency Synthesizer'S performances for LTE/ LTE-Advanced mobile communications," *2015 4th Int. Conf. Electr. Eng. ICEE 2015*, pp. 28–31, 2016, doi: 10.1109/INTEE.2015.7416708.

[31] D. B. Leeson, “A simple model of feedback oscillator noise spectrum,” *Integr. Circuits Wirel. Commun.*, pp. 429–430, 1998, doi: 10.1109/9780470544952.ch6.

[32] X. Huang, F. Tan, W. Wei, and W. Fu, “A revisit to phase noise model of Leeson,” *Proc. IEEE Int. Freq. Control Symp. Expo.*, pp. 238–241, 2007, doi: 10.1109/FREQ.2007.4319072.

[33] S. T. Oskuii, “Comparative study on low-power high-performance flip-flop,” 2003.

[34] P. Y. Deng and J. F. Kiang, “A 5-GHz CMOS frequency synthesizer with an injection-locked frequency divider and differential switched capacitors,” *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 56, no. 2, pp. 320–326, 2009, doi: 10.1109/TCSI.2008.2001761.

[35] U. Low and J. Pll, “A 960-Mb / s / pin Interface for Skew-Tolerant Bus,” vol. 32, no. 5, pp. 691–700, 1997.

[36] W. Rhee, “Design of high-performance CMOS charge pumps in phase-locked loops,” *Proc. - IEEE Int. Symp. Circuits Syst.*, vol. 2, pp. 545–548, 1999, doi: 10.1109/iscas.1999.780807.

[37] J. Kaur; and L. Sood, “Comparison Between Various Types of Adder Topologies,” *Int. J. Comput. Sci. Technol.*, vol. 6, no. 1, pp. 62–66, 2015.