Fractional-N PLL-Based Frequency Synthesis through Sigma-Delta Modulation of the Reference Clock Frequency

by

Matthew Mikkelsen, **B.Sc.**

A thesis submitted to the
Faculty of Graduate and Postdoctoral Affairs
in partial fulfillment of the requirements for the degree of

Master of Applied Science in Electrical Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering
Department of Electrical Engineering
Carleton University
Ottawa, Ontario
January, 2021

Abstract

Used in optical modems amongst a variety of other electronics applications, Integer-N PLL-based frequency synthesizers effectively multiply the frequency of a reference clock by an integer value to synthesize a higher frequency. When a synthesizer is packaged and its reference frequency should be generated externally, the solution is often to use another synthesizer system with another reference frequency and an oscillator that can tune to a valid reference frequency; one that divides evenly into the desired frequency. The proposed integrated-circuit solution divides a high-frequency clock already existing on the ASIC by a floating-point divisor to generate the packaged synthesizer's reference clock through the use of a sigma-delta divider and without requiring an additional oscillator. This thesis will detail the design considerations needed to allow tuning to a set of desired frequencies with a frequency error of less than 1 PPM and jitter less than 500 fs at the synthesizer output. Furthermore, it will discuss the simulation and measurement of the resulting design implemented in TSMC's 7 nm FinFET technology.

I would like to dedicate this thesis to Alice Fernandes, who will never read this document, but will know that the long days and nights I spent at the office were not for nothing

Acknowledgments

First, I would like to thank Professor Calvin Plett for navigating me through my degree to completion and even coming out of retirement to do so. I appreciate you doing so and wish you the best.

I would like to thank the Department of Electronics at Carleton University for giving me the opportunity to undertake this program and for adapting their processes to accommodate the necessary safety precautions and additional support needed due to the COVID-19 pandemic.

I would also like to thank the team at Ciena who made this thesis possible:

Robert Gibbins - For helping plan logistics and decoding the control signals.

Young Cho - For the design and layout of the programmable frequency divider.

Mahdi Parvizi - For the design of the full-adder.

Shawn Fitzpatrick - For the synthesis and layout routing of several blocks.

Soheyl Ziabakhsh Shalmani - For introducing me to the methodology of how to measure the signal-to-noise ratio for sigma-delta modulators

Mohammad Honarparvar - For the late nights of working on sigma-delta DACs and ADCs, the initial design and passing on knowledge of sigma-delta modulators

Douglas McPherson - For the excellent Visio template.

Abidi Mouadh and Abderaouf Kouhoul - For all of the layout integration effort.

Ross Dickson, Marko Antonic, and Vincent Fifer - For all lab work done to make these measurements possible.

Tingjun Wen - For developing a clever way to verify the layout of digital circuits, and helping me debug the implementation of the MASH 111 structure and frequency divider glitches

Sadok Aouini and Naim Ben-Hamida - For coming up with this topic and mentoring me throughout this process

There were quite a few names to list here and if your name was not listed, thank you for your contribution to this project

Table of Contents

List of Acronyms

ADC Analog-to-Digital Converter.

AMS Analog Mixed-Signal.

CIFB Cascaded Integrator FeedBack.

CML Current-Mode Logic.

CMOS Complimentary Metal Oxide Semiconductor.

DAC Digital-to-Analog Converter.

DC Direct Current.

ENOB Effective Number of Bits.

ESD ElectroStatic Discharge.

FCW Frequency Control Word.

FFT Fast-Fourier Transform.

FSR Full-Scale Range.

IBN In-Band Noise.

LPF Low-Pass Filter.

LSB Least-Significant Bit.

MMD Multi-Modulus Divider.

MSA Maximum Stable Amplitude.

NCO Numerically-Controlled Oscillator.

NTF Noise Transfer Function.

OBG Out-of-Band Gain.

OSR Over-Sampling Ratio.

PDF Probability Density Function.

PFD Phase-Frequency Detector.

PLL Phase-Locked Loop.

PLS Post-Layout Simulation.

PPM Parts Per Million.

PSD Power Spectral Density.

RMS Root Mean Squared.

SDM Sigma-Delta Modulator.

SNR Signal-to-Noise Ratio.

SQNR Signal to Quantization Noise Ratio.

SSB Single-Side Band.

STF Signal Transfer Function.

TSMC Taiwan Semiconductor Manufacturing Company.

VCO Voltage-Controlled Oscillator.

Chapter 1

Introduction

The world has progressed into a largely data-driven society, where various types of data are transported over hundreds of kilometers and accessible at one's fingertips within fractions of a second. Achievable data rates within optical telecommunication systems installed in the data centres which transport this data continuously increase to meet the consumer demands. When modulating and demodulating high-speed data to and from an optical fibre, high-frequency low-jitter synthesizers are used to clock optical modems when converting between electrical signals and light. One common type of synthesizer effectively multiplies a low-jitter low-frequency reference frequency by a factor greater than or equal to one through the use of a PLL. A method of achieving non-integer multiples of this reference frequency involves averaging different integer frequency multipliers over time. While such a method can involve reconfiguring the PLL to support it, it would be convenient to be able to achieve non-integer multiplication without modification to the PLL design.

A state-of-the-art method for synthesizing non-integer multiples of a reference frequency is using what is known as a fractional-N PLL. The architecture of this is similar to that of a PLL capable of integer multiplication, known as an integer-N PLL, with the addition of circuitry for modulating between integer frequency multipliers over time. Prior to the existence of fractional-N PLL-based synthesizers, the reference frequency to an integer-N PLL needed to be divided prior to being multiplied by the PLL to achieve higher frequency tuning resolution. The fractional-N PLL can achieve high frequency resolution without requiring large integer multipliers, which are generally undesirable as they also multiply the frequency noise. To avoid adding undesired tones at the rate of modulation, a circuit known as the Sigma-Delta Modulator (SDM) is often used to add pseudo-randomness to the modulation pattern.

1.1 Motivation

The motivation for this work was to obtain the frequency resolution possible with a fractional-N PLL-based synthesizer without having to redesign an existing integer-N PLL. An integer-N PLL requires its reference clock period to be a positive integer multiple of the period to be synthesized. This restricts the reference frequencies that can be used, sometimes requiring an additional synthesizer to generate such a frequency. A fractional-N PLL relieves this restriction, since the target frequency can be a non-integer multiple of the reference frequency. However, a fractional-N PLL uses frequency modulation within the feedback of its structure; requiring modification of an existing integer-N PLL design. Theoretically, this frequency modulation could be moved from the feedback of the PLL to the reference clock path. A sigma-delta divider can be used to generate this frequency modulated reference clock. This solution is appealing since it would not require modification to an existing integer-N PLL and would require smaller area, lower complexity, and less power than an additional synthesizer for reference generation.

1.2 Objective

The objective of this thesis is to design, implement, and demonstrate the operation of an integrated sigma-delta divider circuit in Taiwan Semiconductor Manufacturing Company (TSMC)'s 7 nm FinFET technology that generates the reference clock for a separately packaged PLL-based synthesizer. It should be capable of replacing the function of a Numerically-Controlled Oscillator (NCO) for reference clock generation, and should be able to tune the synthesizer to a set of desired frequencies with a frequency error less than 1 Part Per Million (PPM). When driven by the sigma-delta divider, the synthesizer output frequency should have less than 500 fs of Root Mean Squared (RMS) jitter, integrated from a 500 kHz to 10 MHz frequency offset.

1.3 Contributions

The contributions of this work are the analysis, design, and demonstration of using a sigma-delta divider in the reference clock path of a PLL to effectively implement

fractional-N synthesis. To the author's knowledge, this is a novel method of fractional-N synthesis and protection of this intellectual property has been published [1].

1.4 Thesis Organization

This thesis consists of five chapters. Chapter 1 introduces the thesis topic, including the motivation for the work and the objective. Chapter 2 provides background knowledge on the analysis and performance indicators of sigma-delta converters and PLL-based synthesizers. Chapter 3 builds on these analyses to explain how and why the sigma-delta divider system works and the considerations that were taken during the design and implementation process. Chapter 4 presents simulations of a mixture of transistor level and Simulink models to demonstrate the operation and compares them to measurements of the integrated circuit implemented in TSMC's 7 nm FinFET technology. Chapter 5 discusses conclusions on whether the objective was met and identifies areas of improvement to be executed in future work.

Chapter 2

Background Theory

There are several phenomena and equations the reader should be aware of to better understand the content of this thesis. This chapter will present such concepts and detail related relevant information, diagrams, and expressions. The specific topics covered herein include phase coherency, sigma-delta converters, and PLL-based synthesizer theory.

2.1 Phase Coherency

An important principle that should be considered when dealing with sampled systems such as a sigma-delta divider is phase coherency. Depending on the desired upsampling or decimation ratio, k_2 sampling periods should be the equivalent length of k_1 data periods such that the frequency of data f_{data} and sampling rate f_s are sub-harmonically related to remain phase coherent [2]:

$$f_{data} = k_1 \frac{f_s}{k_2} \tag{2.1}$$

If this relationship is not held, the phase relationship between the data and the sampling clock can drift, causing the desired data to be lost.

2.2 Sigma-Delta Converters

A sigma-delta converter is a type of oversampling data converter. Data converters come in the form of Digital-to-Analog Converter (DAC)s or Analog-to-Digital Converter (ADC)s. A typical sigma-delta converter consists of SDM connected to a

low-pass filter. For a sigma-delta ADC, the sigma-delta is an analog structure driving a digital filter, while a sigma-delta DAC has a digital sigma-delta connected to an analog filter. Herein, SDM theory will be presented, including oversampling and quantization noise, the SDM structure and its noise shaping capability, and metrics to quantify its operation range and resolution.

2.2.1 Oversampling and Quantization Noise

In a discrete-time sampled system, the sampling rate f_s must be at least twice the bandwidth of the input signal f_{BW} [3], referred to as the Nyquist rate f_N, to be able to reconstruct the original signal. For every doubling of f_s in excess to f_N, the Signal-to-Noise Ratio (SNR), which is the signal power divided by the noise power, of the system improves by 3 dB [4]. The ratio of f_s to f_N is referred to as the Over-Sampling Ratio (OSR), defined as:

$$OSR = \frac{f_s}{f_N} = \frac{f_s}{2f_{BW}} \tag{2.2}$$

In a discrete-time sigma-delta converter, a quantizer takes in either an analog voltage input or a digital input, sampled at a rate f_s, and discretizes it into one of $2^Q - 1$ possible digital values, where Q is the number of quantizer bits. Then, these digital values represent quantization levels that are equally spaced by an interval Δ_Q within the Full-Scale Range (FSR) of the input signal. Assuming the analog samples do not happen to consistently coincide with the voltage thresholds of quantization levels for ADCs and that the number of quantizer bits is less than the number of input bits for DACs, the quantized representation of each sample will have some quantization error e_Q. The bounds on e_Q due to quantization as a function of Δ_Q are expressed as [4]:

$$e_Q \leq \left| \frac{\Delta_Q}{2} \right| \tag{2.3}$$

Assuming that the input to the quantizer can be modeled as a pseudo-random sequence, e_Q will vary from sample to sample in a pseudo-random manner [4]. Since e_Q would be uncorrelated between samples, it can be useful to approximate it as if it were additive white noise. For this reason, e_Q is often referred to as quantization noise. Given (2.3), the Probability Density Function (PDF) of quantization noise, specified as $PDF(e_Q)$, is [4]:

$$PDF(e_Q) = \begin{cases} 1/\Delta_Q & -\frac{\Delta_Q}{2} \leq e_Q \leq \frac{\Delta_Q}{2} \\ 0 & otherwise \end{cases} \tag{2.4}$$

Using 2.4 the calculation of the RMS quantization error $e_{Q_{rms}}$ can be simplified as shown [4]:

$$e_{Q_{rms}} = \sqrt{\int_{-\infty}^{\infty} PDF(e_Q) {e_Q}^2 de_Q} = \sqrt{\frac{1}{\Delta_Q} \int_{-\Delta_Q/2}^{\Delta_Q/2} {e_Q}^2 de_Q} = \frac{\Delta_Q}{\sqrt{12}} \tag{2.5}$$

For the real frequency spectrum of the sampled system, the bounds of the Power Spectral Density (PSD) are between 0 and f_N [4]. Knowing this and using 2.5, the expression for the PSD of the quantization noise is denoted by PSD_{QN} is [4]:

$$PSD_{QN}(f) = \frac{{e_{Q_{rms}}}^2}{f_N} = \frac{{\Delta_Q}^2}{6 f_s} \tag{2.6}$$

2.2.2 Noise Shaping

The z-domain expression for the forward Euler model of a discrete-time integrator, often referred to as an accumulator, is given by [4]:

$$H(z) = \frac{z^{-1}}{1 - z^{-1}} \tag{2.7}$$

Figure 1 demonstrates the structure of a first-order SDM [4]. The output of the accumulator is quantized to 2^Q levels with a quantizer to produce output y_i. This output is fed back to be subtracted from the input x_i. The delta between y_i and x_i is then added to the previous value of the delay element within the accumulator; hence the name sigma-delta modulator. An adder input with error e_i, is used to model the quantization error of the quantizer; e_{Qrms} from (2.5). Analyzing the SDM system in the z-domain, an expression for the output Y(z) can be found given the inputs X(z) and E(z) [4]:

$$\begin{aligned} Y(z) &= \frac{H(z)}{1 + H(z)} X(z) + \frac{1}{1 + H(z)} E(z) \\ Y(z) &= z^{-1} X(z) + (1 - z^{-1}) E(z) \end{aligned} \tag{2.8}$$

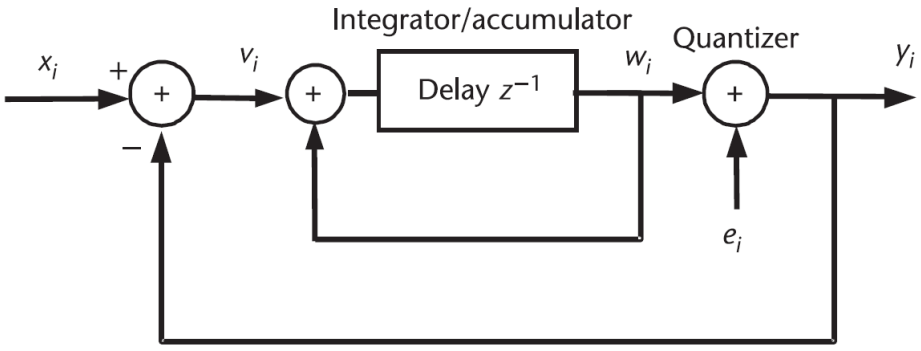

Figure 1: A system-level diagram of a first-order SDM [4]

In (2.8), the term z^{-1} represents what is known as the Signal Transfer Function (STF) [4], while the term $(1 - z^{-1})$ represents the Noise Transfer Function (NTF). In general, the magnitude of the STF of an SDM is designed to be unity at low frequencies within the operation band. The magnitude of the NTF exhibits a high-pass transfer and yields what is known as the noise shaping effect. To take advantage of the noise shaping effect, a sigma-delta converter places an appropriately designed low-pass filter, with a cutoff frequency at the bandwidth of interest, at the output of an SDM to negate the noise shaping out of band. Visualized in Figure 2, this is advantageous because without noise shaping, the quantization noise would instead be flat at a level of ${\Delta_Q}^2/6f_s$ from (2.6); leaving a higher density of quantization noise in-band without noise shaping than with noise shaping.

To achieve a steeper incline of noise shaping, the order of the SDM can be increased. The order of the SDM is dictated by the number of cascaded integrators/accumulators in series prior to the quantizer. In general, the frequency noise shaping of an SDM can be said to be $20 \times m$ dB/decade, where m is the order of the SDM [4]. In a sigma-delta converter, the order of the filter must be greater or equal to the order of the modulator to take advantage of lower in-band quantization noise.

2.2.3 SQNR and ENOB

A metric often used to quantify the resolution of a sigma-delta converter is its Signal to Quantization Noise Ratio (SQNR). SQNR is the SNR specifically for quantization noise. Since SNR is the ratio of signal power to noise power, the calculation should be completed as if the sigma-delta converter is an ADC irrespective of whether it

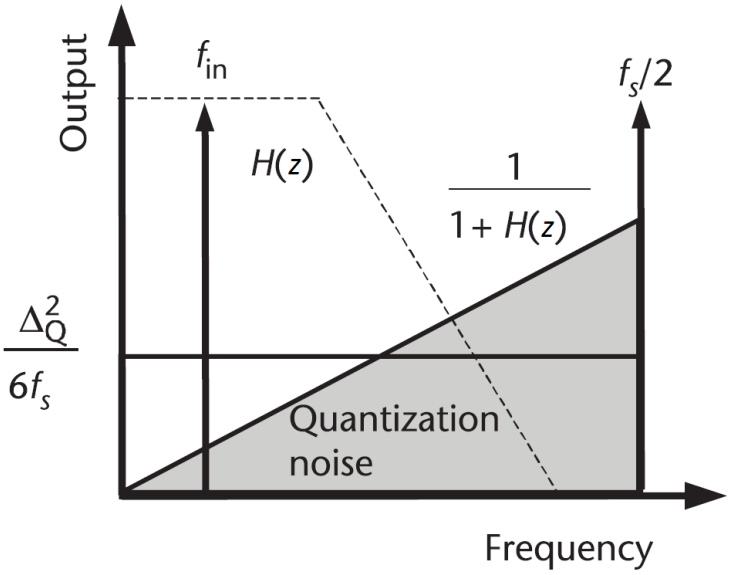

Figure 2: Demonstration of the benefit of the high-pass NTF for systems using SDMs [4]

is actually a DAC to treat the input signal and the quantization noise as voltages. Assuming quantization noise can be modeled as white-Gaussian noise, the quantization noise power $P_{n,QN}$ can be calculated by integrating the PSD_{QN} of (2.6) over the bandwidth f_{BW} and substituting (2.2) [4]:

$$P_{n,QN} = \int_0^{f_{BW}} PSD_{QN}(f)df = \frac{f_{BW}{\Delta_Q}^2}{6f_s} = \frac{{\Delta_Q}^2}{12OSR} \tag{2.9}$$

It is worth commenting on the differences in the meaning of Δ_Q between a digital SDM and analog SDM. For an analog SDM, Δ_Q is the amount of voltage per Least-Significant Bit (LSB) of the quantizer, and the square relationship holds for power. For a digital SDM, an assumption could be that since SQNR is measured in band, the power is analyzed as if there were an analog filter placed at its output. To keep the measurements relative between digital and analog domains, the input signals and noise signals should be expressed in dB with respect to the FSR. Thus, Δ_Q can be defined as the ratio of the input FSR, dictated by the number of input bits to the SDM, to the quantizer FSR.

To ease the ability to distinguish the signal from the noise when measuring in the frequency domain via a Fast-Fourier Transform (FFT), a sine wave is often used as

the input signal due to its spectral purity. When measuring the SNR of a sigma-delta converter, this is the signal that is compared to the noise. While sine waves do not have harmonics themselves, spurious tones, at odd multiples of the input frequency [5], arise from the quantizer error and are noise shaped by the sigma-delta. To avoid having a spurious tone landing in band, the frequency of the input sine wave should be selected according to the following relation, where P is the number of FFT points to be used and bin_{prime} is a prime number corresponding to the FFT bin number at which the fundamental tone f_{in} will be situated:

$$f_{in} = \frac{bin_{prime} \cdot f_s}{P} \tag{2.10}$$

Since the quantization noise will only be integrated across the bandwidth for the measurement, P should be selected to be sufficiently large to provide the number of frequency bins required for the desired accuracy of the SQNR measurement. If the odd harmonics are problematic, the prime number should be set such that f_{in} is greater than one third of the bandwidth but less than the bandwidth, such that the first spurious tone at the third harmonic is out of band but f_{in} is in band.

An expression for SQNR can be made by first approximating the signal power. The signal amplitude projected to the output of the system can be represented by allowing $(2^Q - 1)\Delta_Q$ to represent the FSR and A to represent the sinusoid input amplitude with respect to half the FSR (or equivalently, the peak-to-peak amplitude with respect to the FSR). Squaring and dividing by 8 to convert from peak to peak to RMS converts the signal amplitude into signal power, normalized to $1\,\Omega$. Given the OSR, the signal power can be divided by the quantization noise power to give the SQNR of an oversampled system as shown in the following expression [4]:

$$SQNR = \frac{P_{signal}}{P_{n,QN}} = \frac{\frac{1}{8}A^2(2^Q - 1)^2{\Delta_Q}^2}{P_{n,QN}} \approx \frac{3}{2} \cdot 2^{2Q} \cdot A^2 \cdot OSR \tag{2.11}$$

Given the order of the SDM m, an expression for the SQNR with shaped in-band quantization noise is [4]

$$SQNR = \frac{\frac{1}{8}A^2(2^Q - 1)^2{\Delta_Q}^2}{\frac{{\Delta_Q}^2}{12}\frac{\pi^{2m}}{2m+1}\left(\frac{1}{OSR}\right)^{2m+1}} \approx \frac{3}{2} \cdot 2^{2Q} \cdot A^2 \cdot OSR^{2m+1} \cdot \left(\frac{\pi^{2m}}{2m+1}\right)^{-1} \tag{2.12}$$

The lower levels of in-band quantization noise due to oversampling and noise

shaping correspond to increased resolution and SQNR. Since the SQNR is measured in-band, it can be related with the Effective Number of Bits (ENOB) of the modulator. Knowing that a doubling of amplitude resolution would require an additional binary bit to resolve and corresponds to an increase of 6 dB in SQNR [4], an expression for ENOB can be found by converting (2.12) to dB and dividing by 6 dB [4]:

$$\begin{aligned} ENOB &= \frac{10\log_{10}(SQNR)}{6\,\mathrm{dB}} \\ ENOB &\approx Q + \frac{2m+1}{2}\log_2(OSR) - 1.67\log_{10}\left(\frac{\pi^{2m}}{A^2(2m+1)}\right) \end{aligned} \tag{2.13}$$

Typically, the ENOB is desired to be characterized at the maximum achievable SQNR, known as the dynamic range of the SDM [4]. The dynamic range is determined by repeatedly measuring the SQNR while increasing the value of A. Shown in Figure 3, it can be seen that the value of A corresponding to the dynamic range is not necessarily 0 dB as you would expect from an ideal modulator. The SQNR can saturate prior to the FSR, limiting the dynamic range. The value of A corresponding to the SQNR being saturated by 6 dB from the dynamic range is known as the overload level [4] or Maximum Stable Amplitude (MSA) [6], where the latter is used throughout this thesis.

2.3 Multi-Modulus Dividers

A Multi-Modulus Divider (MMD) is a type of programmable frequency divider that is modular in architecture and is able to divide by positive binary integers greater than 2. They generally consist of cascaded divider stages that can each be programmed to divide by either 2 or 3, and the frequency signal is decreased as it is passed from stage to stage. An additional signal is passed between the consecutive stages, namely, the modulus signal. Multiple architectures employing these cascaded divide by 2 or 3 cells have been proposed over recent years, and two will be introduced herein.

2.3.1 Divide by 2 or 3 Cells

A typical dual-modulus divide by 2 or 3 structure is shown in Figure 4. The input sign clocks the D-latches with alternating consecutive polarities in a loop. When

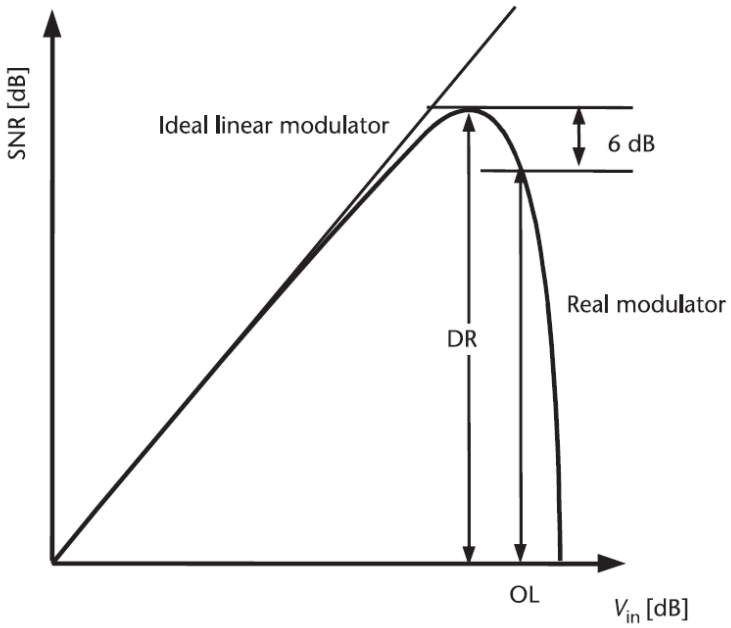

Figure 3: Depiction of a typical dynamic range measurement, where DR is Dynamic Range, and OL is the overload level, also known as the MSA [4]

Mod_{in} is low, the frequency of F will be half that of the input signal and Mod_{out} will go to zero once the input signal goes high. When Mod_{in} is high, the frequency of F will be one third of the input if the programmable input R is high but one half if R is low. Mod_{out} will also track F if Mod_{in} is high but delayed by half of one input clock cycle.

2.3.2 Generic MMD Architecture

A generic MMD architecture is depicted in in Figure 5. One encoded bit is provided to input R of each divider stage, which, if enabled at the same time as the stage's input modulus signal, enables the divide by 3; otherwise it divides by 2. The final divider stage always has its modulus input tied high. Since the modulus signal ripples down the stages, the MMD is able to be programmed to divide within a range of adjacent integers. The step size S between programmable divisor values is configurable by adding an additional divider by S stage prior to the first divide by 2 or 3 stage. The range of programmable integers is increased by the step size for every additional divide by 2 or 3 stage. In the particular architecture depicted in Figure 5, the lower

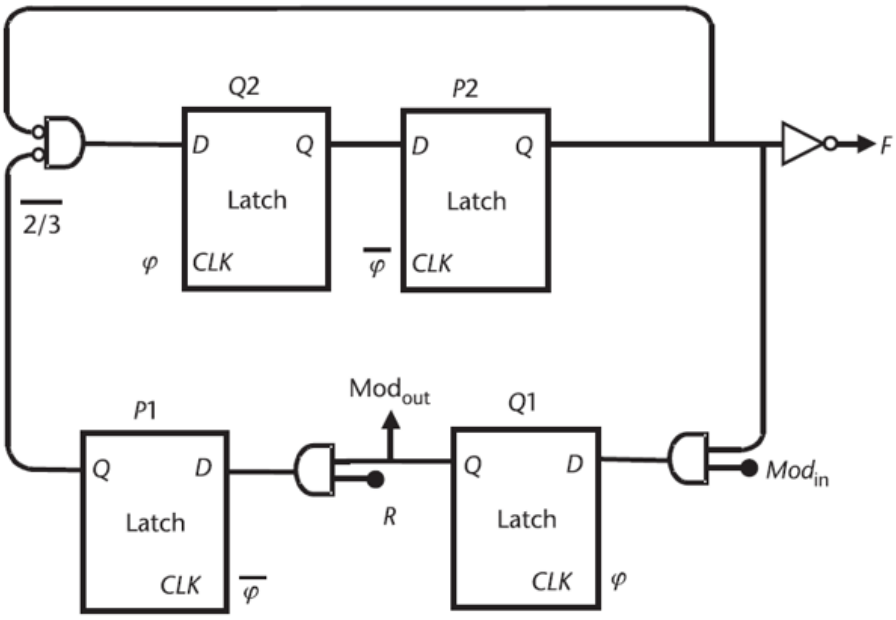

Figure 4: Typical dual-modulus divide by 2 or 3 cell. The input clock signal is represented by φ, where the overline represents its complement [4]

bound of programmable divisors also increases by a factor of 2 for every additional stage. The range of divisors can also be shifted by replacing the final divide by 2 or 3 stage with a divide by P or $P + 1$ stage. For k stages, the minimum divisor N_{min} and max divisor N_{max} can be found using the following relations:

$$\begin{aligned} N_{min} &= S{\cdot}P \cdot 2^{k-1} \\ N_{max} &= S(N_{min} + 2^k - 1) \end{aligned} \tag{2.14}$$

2.3.3 Truly Modular MMD Architecture

The "truly modular" [7] MMD architecture has a few differences from the generic architecture. Shown in Figure 6, logic is added between divide by 2 or 3 stages to extend the divider range. If this additional circuitry is placed between every divide by 2 or 3 stage, N_{min} is 2, regardless of the number of stages. Assuming this is true, all stages are divide by 2 or 3, and the desired step size is 1, N_{min} and N_{max} for this architecture are as follows:

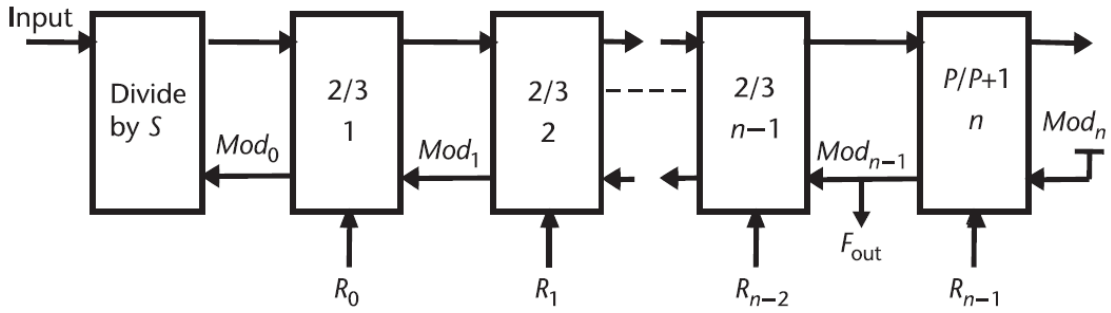

Figure 5: Generic MMD architecture. The final output clock can be tapped at the Mod_{out} or clock output of the final $P/P+1$ stage [4]

$$\begin{aligned} N_{min} &= 2 \\ N_{max} &= 2^{k+1} - 1 \end{aligned} \tag{2.15}$$

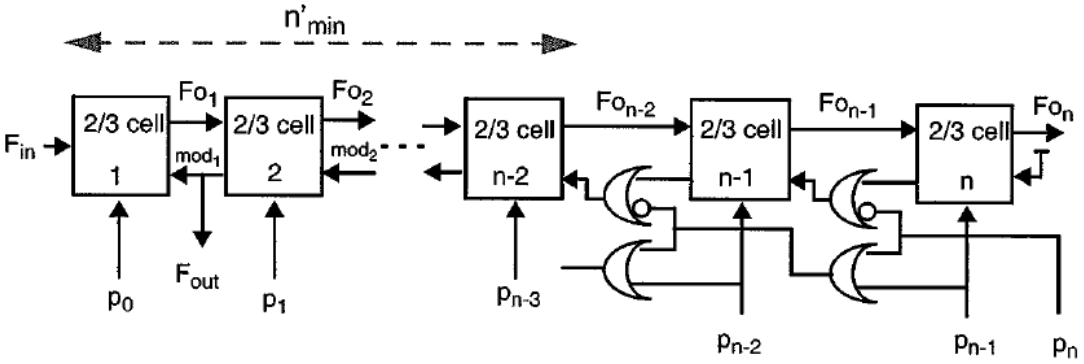

Figure 6: A truly modular MMD, where additional logic is placed between the divide by 2 or 3 cells to extend the programmable divisor range [7]

The N_{max} is observed to also be extended by nearly a factor of 2 when compared to the generic MMD architecture since it is able to allow the programmable divisor binary code word to be 1 bit larger than the number of divide by 2 or 3 stages. However, the division range-extension circuitry does come at a cost of a potentially narrow duty cycle. The extension of N_{min} operates by disabling unnecessary stages for the programmed value by tying their modulus input low. For this to operate correctly, the output of the MMD must be the modulus output of the first stage rather than the output stage. This causes the pulse width of the output signal to

always be equal to the input clock period. While the generic architecture will never have a duty cycle less than 25%, this "truly modular" architecture duty cycle decreases as the programmed divisor is increased.

2.4 PLL-based Synthesizers

2.4.1 System-Level Analysis of Integer-N PLL Synthesizers

The general structure of a charge pump PLL is depicted in Figure 7. Using a Phase-Frequency Detector (PFD), the PLL compares a reference clock to a tunable oscillator, namely, a Voltage-Controlled Oscillator (VCO), and continuously adjusts the frequency of the oscillator until its phase reaches a fixed offset from the phase of the reference clock using negative feedback [4]. The offset is determined by the phase resolution of the PFD and charge pump circuit. The PFD sends a signal to the charge pump indicating whether the feedback phase was either "early" or "late" by comparing the rising edges of the reference and feedback signals. These signals enable a current supply within the charge pump to either source or sink current for the duration the signal is asserted. The charge pump drives the loop filter, which plays the role of converting the current to the voltage while also controlling the dynamics of the loop. If the feedback phase arrives sooner than the reference, the accumulated charge on the filter is decreased, while if the feedback arrives after, the accumulated charge is increased. This corresponds to either an increase or decrease in voltage input to the VCO, and raises or lowers its output frequency. In a PLL-based frequency synthesizer, a frequency divider is placed in the feedback of the PLL. If the bandwidth of the loop is set appropriately, the divided down frequency of the VCO is aligned with the reference clock frequency once the loop settles since the phase will track the reference for every cycle. Thus, once the feedback frequency is locked to the reference frequency f_{ref}, an expression for the frequency output from the VCO f_{out} can be made given integer feedback divisor N:

$$f_{out} = N f_{ref} \tag{2.16}$$

To anticipate the performance of a PLL, the expected phase noise and related jitter can be modeled. To characterize the phase noise, the phase transfer function of

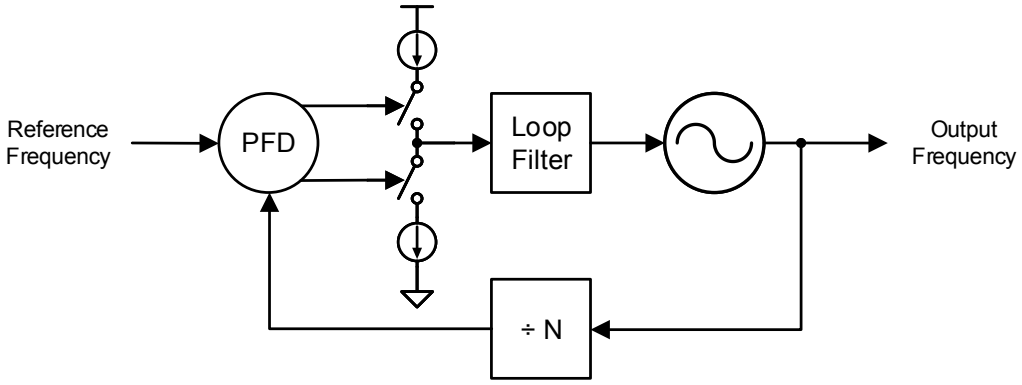

Figure 7: A typical representation of a integer-N charge-pump PLL-based synthesizer

the PLL can be used. To derive the phase transfer function, the Laplace domain can be used with a corresponding linear model for each loop component [4]. A diagram demonstrating such a model is available in Figure 8. Here, K_{VCO} represents the voltage to frequency transfer of the VCO, known as the VCO gain with units Hz/V, while $F(s)$ represents the Laplace domain transfer function of the loop filter. The phase detector and charge pump are modeled by subtracting the feedback phase from the reference phase input, $\Phi_{noise,ref}(s)$, and multiplying the result by gain factor K_{phase}. To convert from phase to current, K_{phase} is a function of the Direct Current (DC) current I_{CP} that is either sunk or sourced by the charge pump and the phase range of the PFD [4]:

$$K_{phase} = \frac{I_{CP}}{2\pi} \tag{2.17}$$

Knowing that angular frequency $\omega = 2\pi f$ for a given frequency f, and that it is equal to the derivative of phase [4], the output phase term $\Phi_{noise,out}(s)$ is made by integrating the VCO frequency output by multiplying the output of K_{VCO} by $2\pi/s$. This integration term also performs the needed frequency to phase conversion for the feedback and reference phase comparison, which can occur prior to the block dividing the frequency by N since all loop components are modeled to be linear. Assuming the noise contribution of the VCO $\Phi_{noise,VCO}(s)$ is zero for the time being, an expression for the phase transfer function from the reference input to PLL output is made [4]:

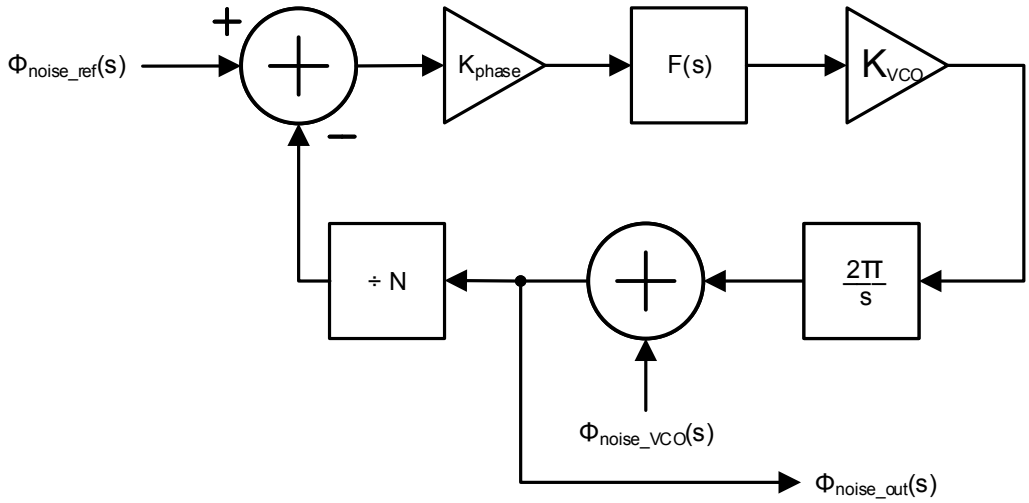

Figure 8: System diagram of the phase transfer of an integer-N charge-pump PLL-based synthesizer given linear models of each of the PLL loop components in the Laplace domain

$$\frac{\Phi_{noise,out}(s)}{\Phi_{noise,ref}(s)} = \frac{F(s)2\pi K_{VCO}K_{phase}}{s + \frac{F(s)2\pi K_{VCO}K_{phase}}{N}} \tag{2.18}$$

This transfer function can be used to determine the phase noise contribution of each loop component except the VCO at the output of the PLL [4]. This can be accomplished by first projecting back their contributions to an input of the adder by dividing by the gain between the adder and the component under analysis, then multiplying the projected phase noise by (2.18). While this method is not used directly in this thesis, it is relevant to understand that the phase noise contribution of each loop component other than the VCO is effectively low-pass filtered by the loop for discussion. Another important observation is noticing that as the offset frequency, and inherently s, approaches zero, the low-pass phase transfer function approaches N. By taking $\Phi_{noise,ref}(s)$ to be zero instead and considering the phase noise contribution of $\Phi_{noise,VCO}(s)$ only, it can be seen in the following expression that $\Phi_{noise,VCO}(s)$ is high-pass filtered by the loop [4]:

$$\frac{\Phi_{noise,out}(s)}{\Phi_{noise,ref}(s)} = \frac{s}{s + \frac{F(s)2\pi K_{VCO}K_{phase}}{N}} \tag{2.19}$$

Once the phase transfer function is applied to a specified phase noise contributor to obtain the phase noise as a function of offset frequency f from the carrier frequency

f_c, its corresponding RMS integrated phase noise, referred to as the integrated phase jitter [4], can be found. The resulting phase noise function $\Phi(f)$ can be squared then divided by two to obtain the Single-Side Band (SSB) phase noise power,then integrated over a specified bandwidth between f_1 and f_2 and square rooted to obtain the RMS jitter. Then, dividing by $2\pi f_c$, where f_c is the carrier frequency for the phase noise measurement, an expression for the integrated RMS jitter in time $\tau_{jitter,rms}$ is found [4]:

$$\tau_{jitter,rms} = \frac{1}{2\pi f_c}\sqrt{\int_{f_1}^{f_2} \Phi(f)^2 df} = \frac{\sqrt{2}}{2\pi f_c}\sqrt{\int_{f_1}^{f_2} \frac{\Phi(f)^2}{2} df} \tag{2.20}$$

It was mentioned that the loop filter can be used to control the settling dynamics of the loop and convert the charge pump current to the control voltage v_c of the VCO, however, the filter can additionally be used to set the order of the PLL. Passive loop filters are common such as the third order filter shown in Figure 9. The highest order passive filter possible without having a series resistance between the charge pump and VCO is second order [4], where additional poles can be added for higher order through additional resistor and capacitor pairs. If the components are appropriately sized for stability, the following expressions can be used to solve for $F(s)$ of a third order loop filter [4]:

$$\begin{aligned} \tau_1 &= R_1 \cdot C_1 \\ \tau_2 &= R_1 \cdot C_1 \cdot C_2/(C_1 + C_2 + C_3) \\ \tau_3 &= R_3 \cdot C_3 \\ F(s) &= \frac{1 + s\tau_1}{s(C_1 + C_2 + C_3)(1 + s\tau_2)(1 + s\tau_3)} \end{aligned} \tag{2.21}$$

Due to the integration term of converting frequency to phase, the order of the phase transfer function for the generic structure shown in Figure 8 is technically one order higher than the order of the loop filter. However, since τ_1 form a zero, the closed-loop out-of-band attenuation of the phase transfer function can be observed to fall at $20 \times n$ [4], where n is the order of the loop filter.

To make an estimate of the settling behaviour of the loop, the effect of the zero due to τ_1 can be considered on its own, ignoring the effect of the higher frequency

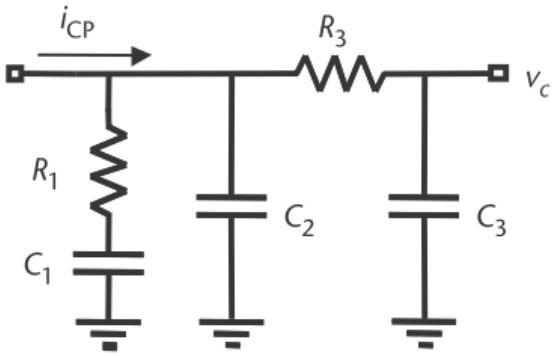

Figure 9: A passive third-order low pass loop filter for a PLL [4]

poles [4]. This makes the phase transfer function second-order, and loop dynamic parameters, namely ω_n and ζ, can be solved for [4]:

$$\omega_n = \sqrt{\frac{2\pi K_{phase} K_{VCO}}{N C_1}} \tag{2.22}$$

$$\zeta = \frac{R_1}{2}\sqrt{\frac{2\pi K_{phase} K_{VCO} C_1}{N}} \tag{2.23}$$

After computing the loop dynamic parameters, the angular corner frequency ω_{3dB} can be estimated through the following expression [4]:

$$\omega_{3dB} = \omega_n \sqrt{1 + 2\zeta^2 + \sqrt{4\zeta^4 + 4\zeta^2 + 2}} \tag{2.24}$$

2.4.2 System-Level Analysis of Fractional-N PLL Synthesizers

A fractional-N PLL-based synthesizer is very similar to an integer-N PLL-based synthesizer with one key difference; f_{out} can be a non-integer multiple of f_{ref}. This is accomplished by modulating the integer divisor of the programmable divider placed in the loop [4], as shown in Figure 10. The programmable divisor is modulated in such a way that the time-averaged value of the divisor is the desired non-integer multiple of f_{ref}. This modulation can be achieved using a SDM input with a value that will be referred to as the Frequency Control Word (FCW). By adding the SDM output with an integer value N as shown in Figure 10, the integer portion of the fractional division can be programmed. Assuming the bandwidth of the loop filter is sized to

effectively filter the shaped quantization noise of the SDM, an expression for f_{out} can be found given the FSR of the FCW [4]:

$$f_{out} = \left(N + \frac{FCW}{FSR}\right) f_{ref} \tag{2.25}$$

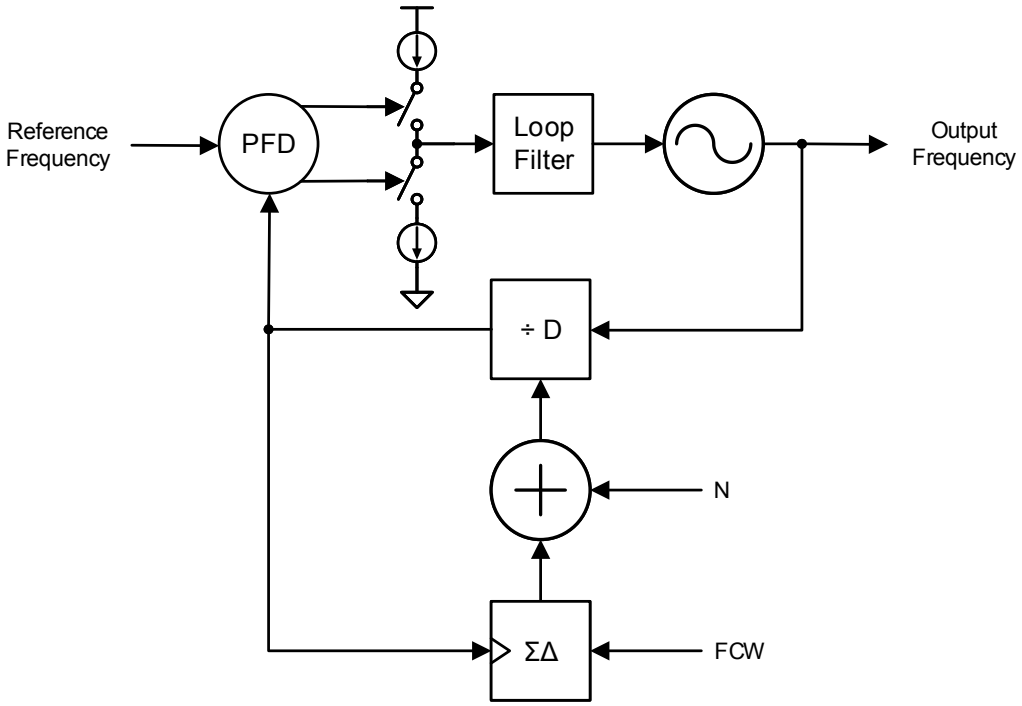

Figure 10: A typical representation of a fractional-N charge-pump PLL-based synthe-sizer

A similar phase transfer model to Figure 8 is made for the fractional-N PLL as shown in Figure 11. By setting all other phase noise contributors to 0 and considering only the contribution of $\Phi_{noise,\Sigma\Delta}(s)$, the phase transfer function to project the phase noise of the SDM to the output of the PLL can be found [4]:

$$\frac{\Phi_{noise,out}(s)}{\Phi_{noise,\Sigma\Delta}(s)} = \frac{\frac{F(s)2\pi K_{VCO}K_{phase}}{N}}{s + \frac{F(s)2\pi K_{VCO}K_{phase}}{N}} \tag{2.26}$$

For simplicity, K_{phase} is often modeled to be linear, even for fractional-N PLLs.

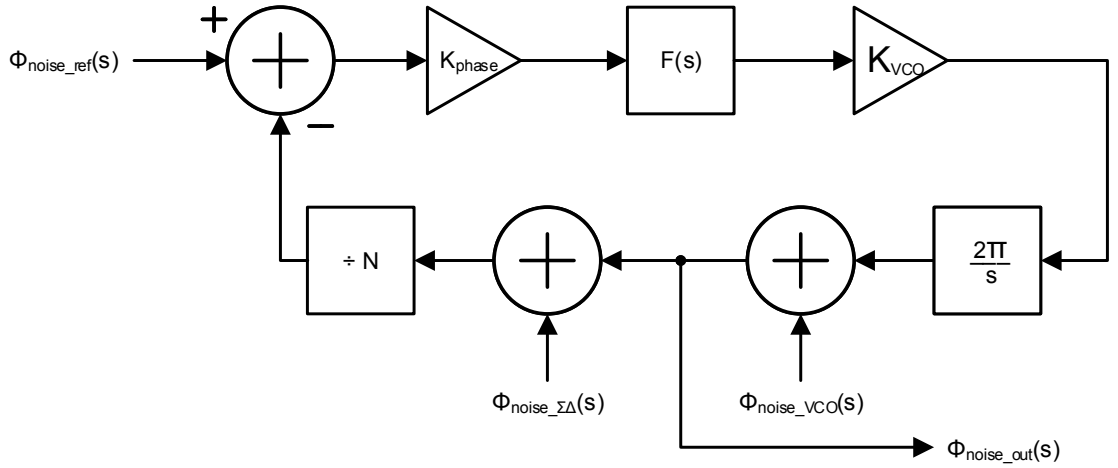

Figure 11: System diagram of the phase transfer of an fractional-N charge-pump PLL-based synthesizer given linear models of each of the PLL loop components in the Laplace domain

However, mismatches between up and down currents within the charge pump create a non-linear transfer characteristic from phase error to current [8]. While this non-linearity is less of a concern in integer-N synthesis since the phase error will settle to a value, it is an important consideration in fractional-N synthesis since the phase error will effectively be constantly modulated by the SDM even when the loop is locked. Thus, while approximating K_{phase} to be linear is an effective assumption for approximating the loop bandwidth, its non-linear transfer should be considered when quantifying jitter performance.

Another contribution to jitter that should be considered is spurious tones, often referred to as spurs [4]. Reference spurs, occuring at integer multiples of the reference, are typically not as much of a concern as what are known as fractional spurs, since the loop bandwidth should be set to filter out the quantization noise of the SDM. This will force reference spurs to be out of band, while fractional spurs can land in band. The frequency location of fractional spurs are at integer multiples for a given arbitrary integer k according to the following relation [4]:

$$f_{spur} = k \cdot f_{ref} \frac{FCW}{FSR} \tag{2.27}$$

2.5 Chapter Summary

Background theory which the reader should be aware of was presented. The importance of phase coherency in sampled systems was highlighted along with the criteria to maintain it. Sigma-delta converter theory was presented, including the concept of oversampling and quantization noise in data converters. The structure of an SDM was shown and the concepts of an STF and NTF were introduced, leading into the noise shaping benefit of SDMs. Metrics to quantify SDM resolution and operation ranges were shown including SQNR, ENOB, MSA, and dynamic range. The basics of the MMD dividers were shown, including the structure of the dual-modulus divide by 2 or 3 cell and two common MMD architectures. Theory was discussed for both Integer-N and Fractional-N types of PLLs. This began with their general structures, and proceeded into linear models describing their phase transfer and their corresponding phase transfer functions. The calculation for RMS jitter through integration of the SSB phase noise was shown. Third order loop filters were presented, along with a method for estimating the loop dynamics and loop bandwidth. Lastly, fractional spurs were mentioned, along with their theoretical location. In the following chapter, the theory presented will be used in the design and analysis of the sigma-delta divider.

Chapter 3

Design and Analysis

To develop the sigma-delta divider system to meet the objective, the system was analyzed and contrasted to existing theory and design methodology detailed in Chapter 2. These analyses lead to several important design considerations, leading to the development of system specifications for the sigma-delta divider, and finally its implementation. First, equations and models describing the operations of the sigma-delta divider converting an integer-N PLL into a fractional-N synthesizer will be uncovered by relating them to systems which have similar functions. Next, this chapter will detail the design considerations accounted for to meet the specifications and their corresponding performance trade-offs. Following this, a discussion on how the models, equations, and design considerations were combined to derive a set of system specifications is presented. Lastly, the selected architectures for each block of the system will be revealed along with the considerations, analysis, and calculations required for the implementation.

3.1 Concept Analysis

A sigma-delta divider contains two main blocks: the SDM and the programmable frequency divider. Shown in Figure 12, a SDM is input with an FCW and drives the control bus of a programmable frequency divider. The divider divides the frequency of a high-speed clock down by a divisor determined by the value of the control bus. The two blocks collaborate to have the FCW set the average frequency of the frequency modulated divider output. The resulting clock, which provides the modulated reference frequency for the synthesizer, is used to clock the SDM to maintain

system coherency. The PLL-based synthesizer outputs a single frequency corresponding to an integer multiple of the average frequency of the modulated reference clock. This is possible because the PLL itself acts as a frequency domain low-pass filter for sigma-delta modulated frequency. The FCW of the sigma-delta divider effectively tunes the output frequency of the synthesizer and can have finer frequency tunability than synthesis using solely integer multiplication of a fixed reference frequency. The abstraction of the sigma-delta divider input to the PLL-based synthesizer resembles that of an NCO solution. Thus, it can be said the sigma-delta divider fulfills the function of an NCO in this application.

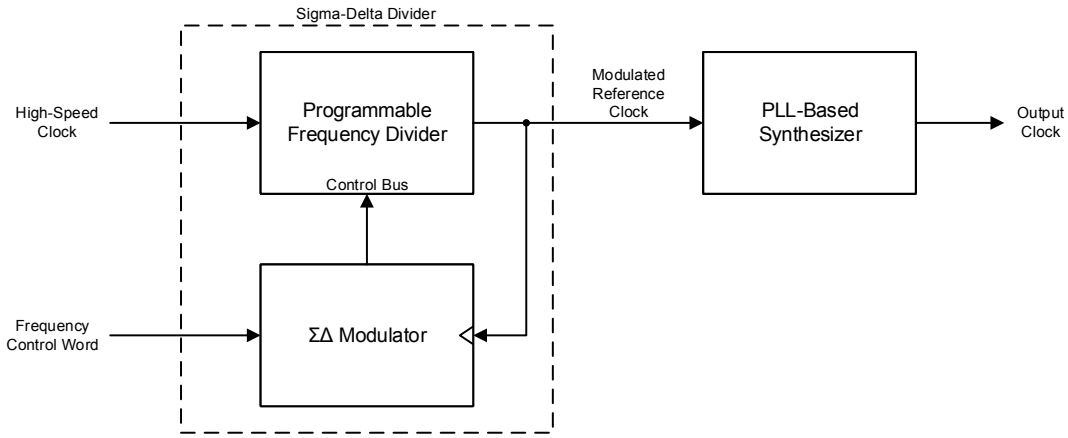

Figure 12: Simplified high-level view of the sigma-delta divider system driving the reference frequency of the PLL-based synthesizer

To help determine the requirements for the sigma-delta divider for modulating the reference clock of a PLL, it is important to develop an understanding of how the system functions. To achieve this, an analogy is made between the sigma-delta divider driving a PLL-based integer-N synthesizer and a sigma-delta converter when both are provided with a static input. Following this, an analysis of the resolution and phase noise for the proposed synthesizer method is compared to a conventional fractional-N PLL-based synthesizer. For both these analyses, the SDMs in all systems will be digital circuits with single-bit quantizers for simplicity.

3.1.1 Sigma-Delta Converter Analogy

From a top-level perspective, both the objective of this thesis and the analogous system are comprised of two main blocks: one performing sigma-delta modulation and the other executing a Low-Pass Filter (LPF) function. Since all SDMs are digital in this analogy, the sigma-delta converter in System 1 of Figure 13 is a sigma-delta DAC comprised of a SDM and an analog LPF. System 2 depicts the objective of the thesis and is comprised of the sigma-delta divider and a PLL-based integer-N synthesizer. It will be shown that the operating mechanisms for a sigma-delta DAC analyzed in the amplitude domain are analogous to the sigma-delta divider driving a PLL-based integer-N synthesizer analyzed in the frequency domain.

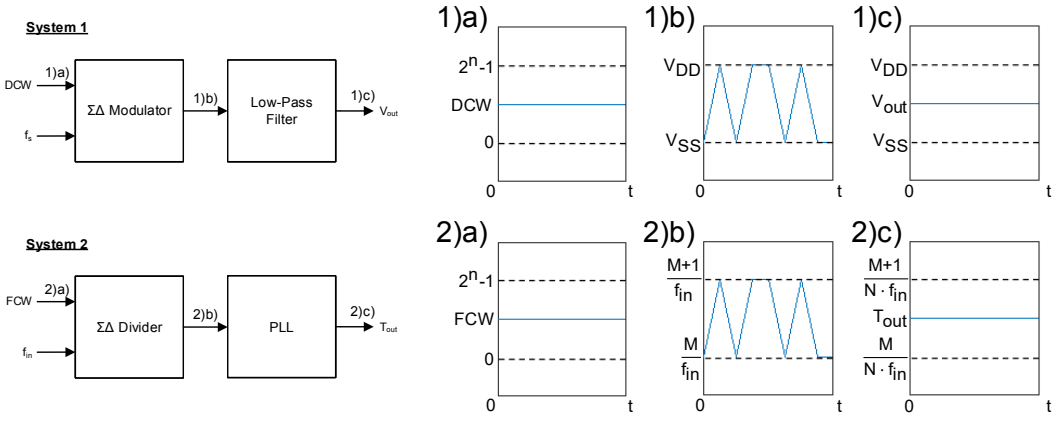

Figure 13: Two analogous sigma-delta modulated systems along with mock inputs and outputs for each. System 1 is a sigma-delta DAC and System 2 is a sigma-delta divider driving a PLL-based synthesizer

Shown in waveforms 1)a) and 2)a) of Figure 13, an arbitrary unsigned digital DC input is provided to both the SDM and sigma-delta divider inputs. These inputs, namely, the Digital Code Word DCW and FCW, are bounded between 0 and $2^n - 1$, where n is the number of bits in each word. In 1)b), the output of the SDM of System 1 is seen to toggle at a high frequency (to achieve the noise shaping effect) between two analog voltages: V_{DD}, corresponding to logic 1, and V_{SS}, corresponding to logic 0. Assuming the value of DCW is within the dynamic range of the SDM, the time-averaged pulse density of this output should be the equivalent to the ratio of the value of the DCW to its digital FSR. If V_{SS} is common between the blocks,

provided the gain of the LPF A_V of System 1, its output V_{out} is a DC analog voltage can be described by the following relation:

$$V_{out} = A_V \left(\frac{DCW}{FSR}(V_{DD} - V_{SS}) + V_{SS} \right) \tag{3.1}$$

This expression is certainly true if A_V is unity. A value for A_V greater than 1 has the potential to saturate the filter depending on the SDM input amplitude if it uses the same voltage rails, and could limit the dynamic range of the sigma-delta converter. The ratio of the dynamic range of the LPF to the analog FSR should be greater than or equal to the ratio of the dynamic range of the SDM to the digital FSR as to not hinder the dynamic range of the system.

For System 2, assume the same digital code is applied to FCW as DCW in System 1, shown in 2)a). The sigma-delta divider output of 2)b) would then toggle with the same time-averaged pulse density as 1)b), although it would toggle between two clock periods instead of two voltage levels. Assuming a programmable divisor step size of 1 for the programmable divider nominally set to a divisor of M, the sigma-delta divider output would toggle between frequency values of f_{in}/M and $f_{in}/(M+1)$, where f_{in} is the frequency of the clock input. From (2.16), the frequency relation between the reference frequency and output frequency of an Integer-N PLL-based synthesizer is a factor of N, where N is the value of the feedback divisor. Given N, the output period T_{out} of the PLL-based synthesizer follows this equation:

$$T_{out} = \frac{1}{N f_{in}} \left(\frac{FCW}{FSR}((M+1) - M) + M \right) \tag{3.2}$$

Comparing (3.1) and (3.2), it is apparent that the Systems 1 and 2 are analogous, with the resulting time-averaged divisor of System 2 determined through the similar mechanism as the time-averaged voltage of System 1. Taking the reciprocal and simplifying (3.2), the output frequency f_{out} depicted in 2)c) of Figure 13 can be found using:

$$f_{out} = \frac{N f_{in}}{M + FCW/FSR} \tag{3.3}$$

Similar to the dynamic range condition for (3.1), (3.3) is true only if the frequency tuning range of the PLL-based synthesizer and the dynamic range of the sigma-delta divider support it. To maximize the dynamic range of the system, the VCO within

the PLL should be capable of tuning wider than the sigma-delta divider dynamic range multiplied by the feedback divisor.

While this analogy aids in developing a fundamental understanding, other design considerations are better explored through comparisons with systems that more closely resemble the system being analyzed or through other types of modeling. In the following section, the sigma-delta divider driving an integer-N PLL will be compared to a fractional-N PLL to utilize existing analyses. Afterwards, these analyses will be elaborated and other models will be analyzed to explore important design considerations and develop design specifications.

3.1.2 Comparison with a Fractional-N PLL-Based Synthesizer

To determine the design trade-offs and considerations to meet the performance specifications of the system, the proposed system was compared with the conventional fractional-N PLL-based synthesizer. Since using a sigma-delta divider in the reference path inherently divides down the frequency, the fractional-N PLL system to be compared with also includes a frequency divider in the reference path without a SDM. First, equations for the change in resolution when changing the LSB of the FCW for both synthesizer methodologies will be derived and compared. Afterwards, the phase transfer functions of each system will be compared to see the effect each system will have on the phase noise.

Frequency Control Word Resolution Comparison

The sigma-delta divider driving an integer-N PLL is shown in Figure 14. In this figure, the adder is shown as a separate block, adding an integer M to the output of the SDM to form a fractional divisor D. Expanding on (3.3), and letting D_M represent D with M as an addend and FSR represent the digital FSR of the SDM:

$$f_{out} = \frac{Nf_{in}}{M + FCW/FSR} = \frac{Nf_{in}}{D_M} \tag{3.4}$$

If the LSB of the FCW were to be changed in either direction, the equation representing the new output frequency $f_{out\Delta LSB}$ would be:

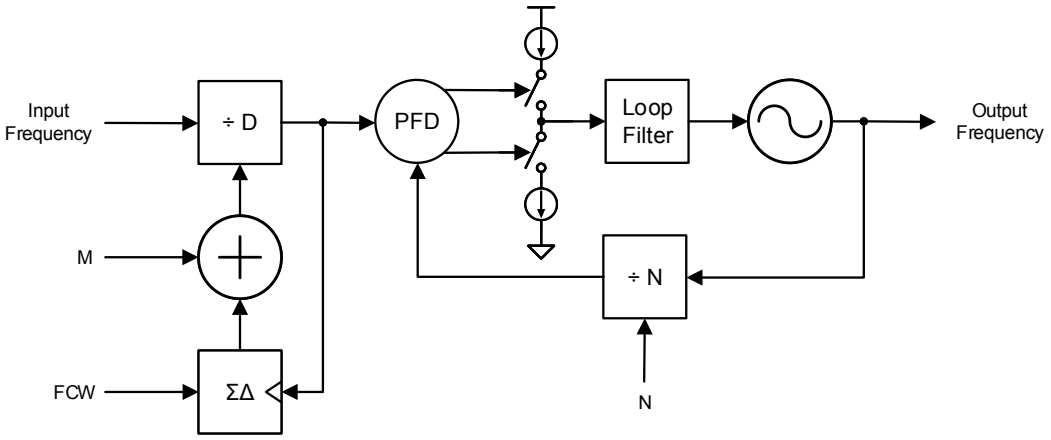

Figure 14: System diagram of the sigma-delta divider driving the an integer-N PLL-based frequency synthesizer

$$f_{out\Delta LSB} = \frac{Nf_{in}}{M + (FCW \pm 1)/FSR} = \frac{Nf_{in}}{D_M \pm 1/FSR} \tag{3.5}$$

Using (3.4) and (3.5) together, the frequency difference between $f_{out\Delta LSB}$ and f_{out} in PPM, PPM/LSB_{sddref}, can be found for when a sigma-delta divider is used in the reference path of PLL:

$$PPM/LSB_{ref} = \left| \frac{f_{out\Delta LSB} - f_{out}}{f_{out}} \right| \times 10^6 = \left| \frac{D_M}{D_M \pm 1/FSR} - 1 \right| \times 10^6 = \left| \frac{10^6}{D_M FSR \pm 1} \right| \tag{3.6}$$

As mentioned previously, a divider was placed in the reference path in the Fractional-N PLL comparison system as shown in Figure 10. Expanding upon (2.25) with D_N representing the sum of the integer divisor N and the SDM output:

$$f_{out} = \frac{(N + FCW/FSR)f_{in}}{M} = \frac{D_N f_{in}}{M} \tag{3.7}$$

Similar to (3.5), the equation representing the output frequency with the LSB of the FCW changed in either direction, $f_{out\Delta LSB}$, becomes:

$$f_{out\Delta LSB} = \frac{(N + (FCW \pm 1)/FSR)f_{in}}{M} = \frac{(D_N \pm 1/FSR)f_{in}}{M} \tag{3.8}$$

Then, the PPM frequency difference per LSB can be found for the case of the

sigma-delta divider located in the feedback path of a PLL:

$$PPM/LSB_{fb} = \left| \frac{f_{out\Delta LSB} - f_{out}}{f_{out}} \right| \times 10^6 = \left| \frac{\pm 1/FSR}{D_N} \right| \times 10^6 = \frac{10^6}{D_N FSR} \tag{3.9}$$

Since FCW must be less than the FSR, it becomes clear that for high resolution, either/both the integer portion of D (i.e. M for D_M or N for D_N) or/and the FSR should be maximal. Increasing D too much will require a large feedback divisor in the PLL, increasing the in-band phase noise of the system. However, apart from the increased chip area, power, and complexity of keeping the modulator stable, increasing the number of bits in the SDM, and thus its FSR, comes at no performance cost.

Comparing (3.6) and (3.9), it becomes clear that the FCW frequency tuning is linear for a conventional Fractional-N PLL, but non-linear when using a sigma-delta divider in the reference path. Conversely, if the reciprocal is take of (3.4) and (3.7) to express the relation for determining the output period of the system, tuning FCW has a linear effect on the period when the sigma-delta divider is in the reference path but non-linear when in the feedback path. Tuning using FCW is monotonic in both scenarios, so using either system in a frequency or period tuning control loop is possible. Furthermore, as the larger product of FSR and D_M of (3.6) becomes larger, the ± 1 term of the denominator becomes increasingly dominated; thus the relation approaches linearity.

To determine the conditions where each synthesizer methodology is advantageous for frequency tuning resolution, it was desired to find the conditions to make them equal. Assuming FSR is set sufficiently large, D_M should be set equal to D_N to make their PPM/LSB frequency resolution equal in (3.6) and (3.9). Let the variables related to the synthesizer with the sigma-delta divider in the reference path have a subscript of 1, and with the sigma-delta divider in the feedback have a subscript of 2. Then, rearranging (3.4) for D_M and (3.7) for D_N and setting them equal, the condition for equal resolution becomes:

$$\frac{N_1 f_{in1}}{f_{out1}} = \frac{M_2 f_{out2}}{f_{in2}} \tag{3.10}$$

Since a large fractional divisor allows higher tunable frequency resolution, whether it be D_M or D_N, setting the numerator of both sides of (3.10) greater than their denominators is desirable. This means it is beneficial if f_{in} is greater than f_{out} when

the sigma-delta divider is in the reference path, and f_{in} less than f_{out} is preferable if it is situated in the feedback. Figure 15 depicts the situation clearly, plotting both (3.6) and (3.9) in both a) and b), followed by the difference between them in c). The number of input bits was set to 16 and N_1 was set equal to M_2 for a fair comparison. In a), f_{in} is set larger than f_{out}, and the PPM/LSB is lower when the sigma-delta divider is in the reference path. In b), f_{in} is set lower than f_{out}, and the PPM/LSB is lower when the sigma-delta divider is in the feedback path. In c), (3.10) is satisfied by setting $f_{in1} = f_{out2}$ and $f_{out1} = f_{in2}$, and the difference between the PPM/LSB of each synthesizer method is smaller than 1 part per billion and is negligible. For all three plots, the PPM/LSB decreases as $N1 = M2$ increases.

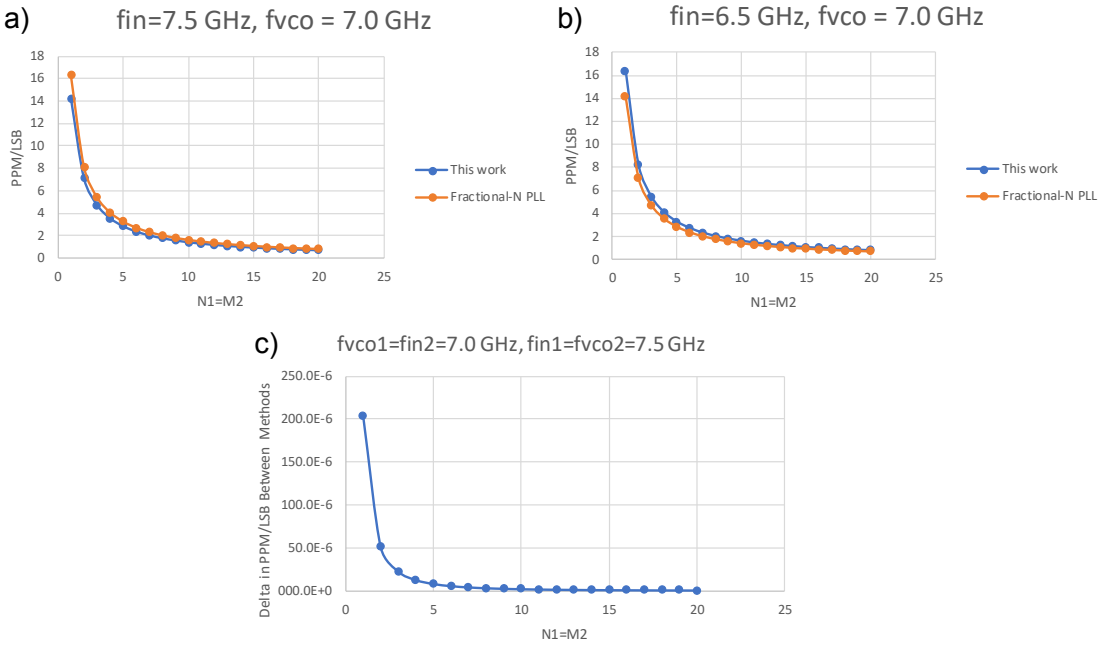

Figure 15: a) PPM/LSB of both synthesizers when $f_{in} > f_{VCO}$ b) PPM/LSB of both synthesizers when $f_{in} < f_{VCO}$ c) Difference between the PPM/LSB for each synthesizer when $f_{in1} = f_{VCO2} > f_{in2} = f_{VCO1}$

To summarize, setting the number of input bits to the SDM and fractional divisor D sufficiently large allows sub 1 PPM/LSB resolution in either methodology. Since a larger value for D gives better frequency resolution for both synthesizer methods, certain scenarios present slight advantages to using one method over the other. While it does not fully apply to the objective of this thesis since a sigma-delta divider is

replacing the function of an intermediate synthesizer in a cascaded synthesizer system, they are worth commenting for the design of a fractional-N synthesizer in other scenarios. In general, from a frequency resolution standpoint, if selecting between placing a sigma-delta divider in the feedback or reference path of the synthesizer and the desired f_{out} is greater than an available clock frequency f_{in}, then it is most beneficial to have the sigma-delta divider in the feedback path. If the desired f_{out} is less than f_{in}, then it is most beneficial to have the sigma-delta divider in the reference path. While the resolution could also be increased by setting N_1 or M_2 to a larger value, this also has the effect of increasing the in-band phase noise transfer function for all phase noise contributors except for the SDM and the VCO. The phase noise contributions of the SDM will be explored for both synthesizer methods in the subsequent section.

Phase Noise Contribution Comparison

To determine the effect of placing the sigma-delta divider in the reference path of PLL on the phase noise, its phase noise interaction with other loop components was determined and compared to that of a conventional fractional-N PLL. This was accomplished using a frequency domain model and analysis similar to that presented in Chapter 2.

Shown in Figure 16, a Laplace domain model of a PLL with two potential phase noise contributor inputs and one phase noise output is shown. If analyzing the sigma-delta divider in the reference path, $\Phi_{noise\_in}(s)$ is set to the phase noise contribution of the SDM $\Phi_{\Sigma\Delta}(s)$ and $\Phi_{noise\_fb}(s)$ is set to 0, while the opposite is true if the sigma-delta divider is in the feedback path. Letting $K = K_{phase}F(s)K_{VCO}$ results in two similar phase noise transfer functions for the SDM for each scenario, shown in (3.11) and (3.12).

$$\Phi_{noise\_out}\left(1+\frac{K}{sN}\right) = -\frac{K}{sN}\Phi_{noise\_fb} \Rightarrow \frac{\Phi_{noise\_out}}{\Phi_{noise\_fb}} = -\frac{\frac{K}{sN}}{1+\frac{K}{sN}} \tag{3.11}$$

$$\Phi_{noise\_out}\left(1+\frac{K}{sN}\right) = \frac{K}{sM}\Phi_{noise\_in} \Rightarrow \frac{\Phi_{noise\_out}}{\Phi_{noise\_in}} = \frac{\frac{K}{sM}}{1+\frac{K}{sN}} \tag{3.12}$$

Since quantization can be modeled as noise and the final stage of an SDM is a

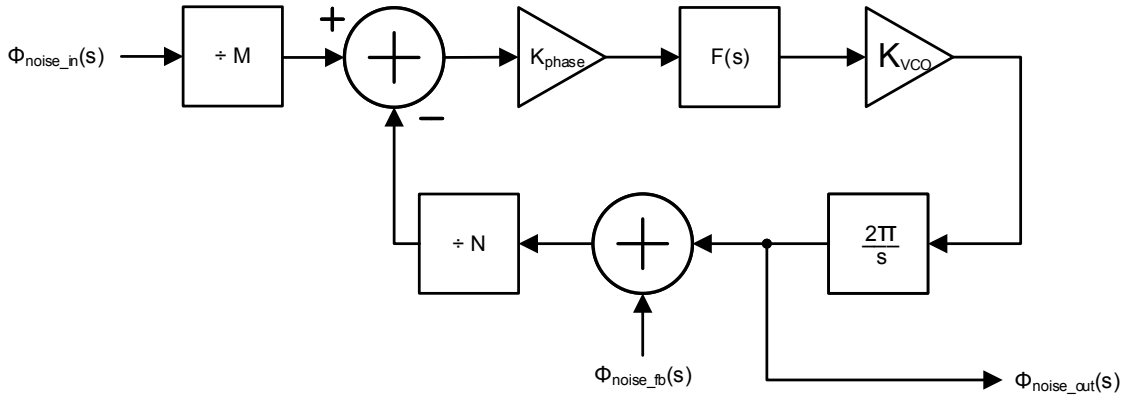

Figure 16: System diagram of the phase transfer of an integer-N charge-pump PLL-based synthesizer given linear models of each of the PLL loop components in the Laplace domain; contrasting specifically the reference and feedback phase inputs

quantizer, the difference in sign between (3.11) and (3.12) does not effect the magnitude of the phase noise transfer, but instead dictates the direction of the phase excursion. This can be visualized by comparing Figure 10 and 14 and assuming the SDM has a single-bit quantizer. When the SDM output is high, the phase error Φ_e will be positive if the sigma-delta divider is in the feedback path but negative if it is in the reference path. The opposite is true when the SDM is low. Since the phase detector's "late" output is high when the phase error is positive (lagging) and "early" output is high when negative (leading), the charge pump will effectively adjust the charge input to the filter to match the direction of the phase error. The direction of the voltage variations at the filter output dictates the direction of the phase error at the VCO output transferred through the loop from the SDM.

The difference between the transfer functions expressed in (3.11) and (3.12) that has not been discussed is the divisors of the numerators. It can be seen by analyzing Figure 16 that when the sigma-delta divider is in the feedback path, the phase-noise contribution of the SDM is divided by a factor of N, but when it is in the reference path the phase noise is divided by a factor of M. The difference is only relevant if M is not equal to N, since if they were equal, the phase noise performance of either synthesizer method is the same. As mentioned in Chapter 2, the phase noise transfer function of all other contributors other than the VCO has a low-pass characteristic with the in-band noise multiplied by a factor of N. Thus, placing the sigma-delta divider in the reference path can be advantageous if M is set greater than N, since it

is not constrained to trade-off between raising the in-band noise and attenuating the sigma-delta phase noise contribution. However, observing (3.3), it is realized this can only be achieved if f_{in} is greater than f_{out}. If f_{in} is synthesized as it is in this design scenario, then requiring a larger feedback divide ratio would need to be used in this synthesizer to increase the frequency. It is important to consider that the in-band phase noise of such a synthesizer can increase proportionally if its specified frequency increases. If the resulting jitter is not mitigated, the phase noise benefit of having a high frequency f_{in} becomes moot, since the phase noise savings are just shifted to the input. Thus, using a high frequency synthesizer to generate f_{in} can be beneficial in terms of phase noise only if the phase noise of the previous stage can be optimized sufficiently.

3.2 Design Considerations

Nearly every design has trade-offs that should be considered. A discussion of the reasoning used during the design process, which takes into account the theory presented in the background section along with some additional cited sources, is detailed herein.

3.2.1 Divider Step Size

Until now, all equations for calculating the output frequency of a sigma-delta divider have assumed that the divider step size has been set to 1, and for good reason. Herein, it will be shown that increasing the divider step size will effectively increase the dynamic phase excursion output of the phase detector once the PLL is locked, thus increasing the phase-noise contribution of the sigma-delta divider. Afterwards, a method for compensating for SDM stability will be presented, which involves increasing the divider step size, creating a potential trade-off.

The divider step size for a sigma-delta divider is the integer factor the instantaneous value of the divisor increases by per LSB of the SDM quantizer. For a single-bit quantizer, the peak-to-peak frequency excursion Δf due to divider step size $\Delta step$ is found by taking the difference between the instantaneous minimum and maximum frequencies output from the sigma-delta divider, f_{min} and f_{max} respectively. For large values of divisor M, an approximation can be made:

$$\Delta f = f_{max} - f_{min} = \frac{f_{in}}{M} - \frac{f_{in}}{M + \Delta step} = \frac{f_{in}\Delta step}{M(M + \Delta step)} \approx \frac{f_{in}\Delta step}{M^2} \tag{3.13}$$

Thus, if the $\Delta step$ were increased from 1 to 2, it can be seen that Δf would approximately double. For a fixed FCW, this would also double both the positive and negative normalized frequency excursions of (3.16) and (3.17), effectively doubling both $F(z)$ and $\Phi(z)$. To keep the phase noise minimal, it is typical to see $\Delta step$ set to 1 in most sigma-delta dividers. However, when the MSA is less than the FSR of the input, $\Delta step$ can be increased to minimize gaps in the frequency tuning at the expense of phase noise performance.

If the Out-of-Band Gain (OBG) of an SDM is increased, the number of bits within a quantizer can be increased to prevent it from being overloaded and causing the SDM to go unstable. For SDMs that are conditionally stable, the quantizer becomes overloaded after the input amplitude of a signal, biased at half the input FSR, surpasses the MSA. For SDMs with a single-bit quantizer, the MSA can never reach the FSR, since it is not possible to have the time-averaged value of an SDM output correspond linearly to a maximum or a minimum input while it toggles between the two. This can be realized intuitively through a simple example. If the FCW was set to zero, $\overline{f_{ref}}$ of (3.14) would evaluate to f_{max}, but the sigma-delta divider output would have to toggle between both f_{max} and f_{min} to achieve the desired noise shaping, making an average value of f_{max} not possible. A similar situation would occur if the FCW was set to the FSR. Thus, if $\Delta step$ is 1, the FCW should not exceed the MSA as $\overline{f_{ref}}$ no longer follows (3.14).

The plot on the left of Figure 17 demonstrates the achievable fractional divisors of a sigma-delta divider provided a programmable divisor M, with a $\Delta step$ set to 1, and containing a single-bit-quantizer SDM with the MSA being half of the FSR. The highlighted region represents the tuning range where (3.14) is valid, and thus the valid settings within the tuning range corresponds to the ratio of the MSA to the FSR for any value of M. However, if $\Delta step$ is set to 2 for the otherwise same sigma-delta divider as shown on the right of Figure 17, the range of achievable fractional divisors doubles. Instead of having valid fractional divisors from $M + 0.25$ to $M + 0.75$, this expanded range covers from $M + 0.5$ to $M + 1.5$. The doubling of the valid range of fractional divisors from 0.5 to 1 allows a continuous tuning range if M is incremented or decremented. Thus, gaps in achievable fractional divisors for single-bit quantizer

sigma-delta dividers can be mitigated by setting $\Delta step$ to 2 so long as the MSA is greater than half the FSR.

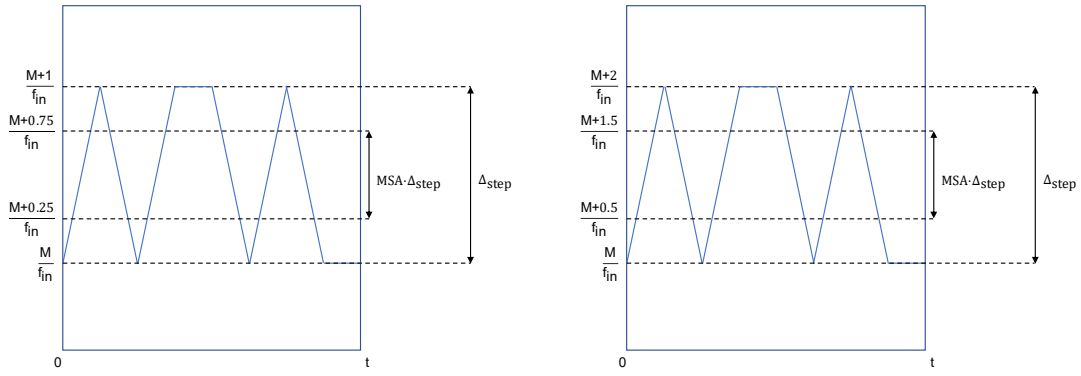

Figure 17: Achievable fractional divisors if the MSA of the SDM is half of the FSR with a divider step size of 1 (left) and 2 (right)

3.2.2 Phase Coherency

To ensure the average value of the sigma-delta modulated frequency output of the sigma-delta divider is correct, it is important to set the sampling clock of the SDM to be phase coherent with the modulated clock. If the phase between the two clocks is not coherent, it can drift over time. This phase drift can result in periods intermittently having two or no samples, depending on the direction of the drift. Missing or double-counting periods can lead to skewing the average value of the frequency.

As expressed previously, when the sigma-delta divider drives an integer-N PLL, the PLL acts like a frequency averager by locking to a multiple of the average frequency of its reference input. By removing the frequency multiplication effect of the feedback divider from (3.4), the average of the reference clock generated by the sigma-delta $\overline{f_{ref}}$ is:

$$\overline{f_{ref}} = \frac{f_{in}}{D_M} = \frac{f_{in}}{M + FCW/FSR} \tag{3.14}$$

To maintain coherency within a sampled system, the sampling rate should be an integer multiple of the data rate, as shown previously in (2.1). Since the quantization noise shaping of the SDM relies on oversampling to operate correctly, by

ensuring the output frequency of the divider is phase coherent with the sampling of the sigma-delta, it allows phase noise shaping about $\overline{f_{ref}}$. Then, to be phase coherent between the output frequency of the programmable divider and the oversampling rate of the SDM, the instantaneous frequency of the sampling clock of the SDM should be an integer multiple of the instantaneous frequency of the divider output. It has already been established that to minimize the phase contribution of the sigma-delta divider, the divider step size should be set to 1 whenever possible. Allowing the divider step size to potentially be 1 means the lowest common multiple between the divider's two instantaneous output frequencies is its input frequency, f_{in}. Thus, the options are to use a multiple of f_{in} as the sample clock to the SDM or to use the divider's instantaneous output frequency f_{ref}. However, changing the control signal to the programmable divider more than once per period does not yield the benefits of oversampling, since a programmed period should not be allowed to change until it completes. The programmable divider would be required to effectively decimate by the same ratio at which the period is oversampled to preserve the value of the period. Thus, clocking the SDM at any rate faster than f_{ref} needlessly creates a more difficult timing criterion between the SDM and divider and increases the required sample rate of the SDM design. To avoid these unnecessary constraints of the design, it was opted to use the output of the divider to clock the SDM.

3.2.3 Order and Phase-Noise Shaping of the Sigma-Delta Modulator

When selecting the order for the sigma-delta, ultimately it should meet the same criteria as the selection of the order for a fractional-N PLL mentioned in Chapter 2. To understand why this criteria holds, a frequency-domain model of the sigma-delta divider was constructed to be interfaced with the model of the integer-N PLL it drives. Then, the model was analyzed to compare the effect of the order of the SDM on the rate of change of the phase noise to that of the loop filter within the integer-N PLL.

To ensure all phase noise shaping is attenuated at the output of the synthesizer, it is important to understand the relation of the order of the SDM and loop filter to the phase noise. First, it should be noticed that the order of the PLL frequency-domain model, observable in Figure 16, is one higher than the order of the loop filter due to the integrator, and that K_{VCO} has units of radians/(volts×seconds).

These operations are necessary because the phase detector takes the difference in phase between its two inputs, so the frequency output from the VCO needs to be converted to phase. Similarly, to model the conversion of frequency noise output from the sigma-delta divider to phase noise, a factor of 2π and an integrator should be included. A discrete time integrator can be used as shown in , with the update rate being the average frequency of the sample clock of the SDM $\overline{f_{ref}}$. Using the forward Euler discrete-time integration approximation [9] instead of backwards Euler [4], the conversion from frequency noise $F(z)$ to phase noise $\Phi(z)$ is modeled as:

$$\Phi(z) = \frac{2\pi}{\overline{f_{ref}}} \frac{z^{-1}}{1 - z^{-1}} F(z) \tag{3.15}$$

To model the frequency noise $F(z)$ in a simulation, one can combine the STF and NTF of the SDM if the quantization noise can be considered white, or simply use a model of the SDM itself [9]. For simplicity, an SDM with a single-bit quantizer and a programmable divider with a step size of 1 were considered. When the output of the SDM is low, the sigma-delta divider instantaneous output frequency is at its maximum; when high, it is at its minimum. The differences between the instantaneous maximum and minimum frequencies, f_{max} and f_{min} respectively, subtracted by $\overline{f_{ref}}$, each correspond to the positive and negative frequency excursions of the frequency noise. Bringing $\overline{f_{ref}}$ to the input of the integrator since it is a linear operation, the two possible inputs to the integrator are approximated as follows, assuming the M is sufficiently large:

$$\frac{1}{\overline{f_{ref}}} F(z) \mid_{\Sigma\Delta=0} = \frac{1}{\overline{f_{ref}}} (f_{max} - \overline{f_{ref}}) = \frac{M + FCW/FSR}{M} - 1 = \frac{FCW}{FSR} \frac{1}{M} \tag{3.16}$$

$$\begin{aligned} \frac{1}{\overline{f_{ref}}} F(z) \mid_{\Sigma\Delta=1} &= \frac{1}{\overline{f_{ref}}} (f_{min} - \overline{f_{ref}}) = \frac{M + FCW/FSR}{M+1} - 1 \\ &= \frac{M + 1 + FCW/FSR - 1}{M+1} - 1 \approx \left(1 - \frac{FCW}{FSR}\right) \frac{1}{M} \end{aligned} \tag{3.17}$$

Since the factor of $1/M$ is already taken into account in the phase noise transfer characteristic of the integer-N PLL, as shown in (3.12) and visualized in Figure 16, the frequency noise normalized to $\overline{f_{ref}} \times M$ can be approximated by the SDM output

for a given FCW subtracted by the ratio of the FCW to its FSR to remove the DC component [8]. Thus, a model for generating SDM phase noise is made as shown in Figure 18.

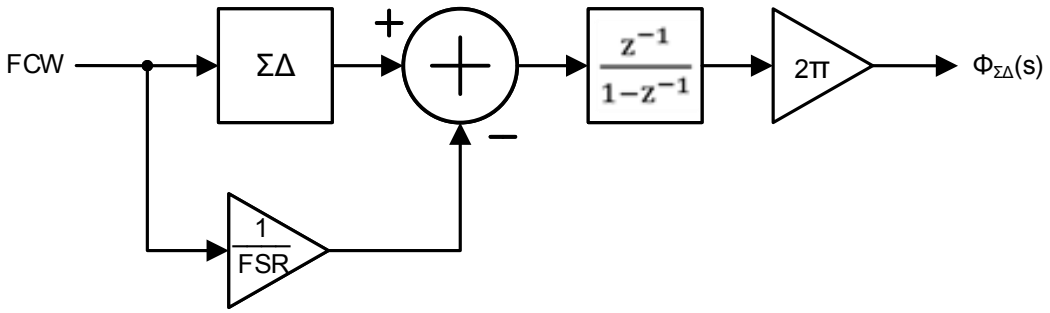

Figure 18: Linear model approximating the phase noise due to SDM

While an SDM of order m has frequency-noise shaping of $20{\times}m$ dB/decade according to its NTF, the denominator of the integration term when converting to phase reduces the effective order by one, causing the phase-noise to increase instead at $20(m-1)$ dB/decade. To at least cancel the shaped phase noise, the loop filter order p should be greater than or equal to m-1, but to attenuate it, p should be greater or equal to m [4]. Considering the design scenario is to create a sigma-delta divider that achieves fractional-N synthesis at the output of an otherwise integer-N PLL, this limits m to be at most one order greater than p.

3.2.4 Quantizer Bits Effect on Charge Pump Non-Linearities and Maximum Stable Amplitude

Just as the non-linear transfer of phase error from the output of the PFD to current at the output of the charge pump is a concern in conventional fractional-N PLLs, it is a concern when achieving fractional-N synthesis using sigma-delta dividers in the reference path of an integer-N PLL. As mentioned in Chapter 2, this non-linearity can increase in-band noise. Since this non-linearity can only be properly mitigated through design techniques within the PLL itself, the only technique that can be applied to the sigma-delta divider is to make it less susceptible to the non-linear

transfer characteristic. Depending on the nature of the non-linearity, this can be accomplished by minimizing the number of quantizer bits in the SDM. However, reducing the number of quantizer bits constrains the stability of the SDM, often limiting its input MSA.

When designing an integer-N PLL, it is possible for the non-linear transfer characteristic of the PFD and charge pump to be neglected since the current mismatch in the charge pump is typically only a concern over a small phase error range near zero that cause reference spurs. Unfortunately, the non-linearity is often the most drastic near this zero crossing, [8]. This especially poses a problem when converting an integer-N PLL into a fractional-N PLL, since the constant toggling of the SDM means a wider range of phase error is covered when the loop is locked than the presumably static phase error of its integer-N PLL counterpart. This non-linear characteristic, if not mitigated, can raise the in-band noise of a fractional-N PLL [8].

Several techniques for compensating for charge pump mismatches have been employed in prior art, including locking at a phase offset [8] and mismatch correction [10]. If compensation is not applied and the non-linearity spans the dynamic phase error region of a locked fractional-N PLL, the next option for reducing the impact of the non-linearity would be to minimize the dynamic phase excursion. As mentioned previously, this can be accomplished by minimizing the divider step size and the number of quantizer bits in the SDM. The reason this can help can be visualized using Figure 19, where a sketch of a typical PFD and charge pump non-linearity can be observed, and Φ_e represents the phase error output of the PFD. If locking at a zero phase offset, as the SDM toggles, the dynamic phase excursion will vary the charge pump current I_{CP}. When comparing the blue shaded region to the green, it can be seen that reducing the dynamic phase excursion would also reduce the amount I_{CP} is changed. Locking at a static phase offset reduces the change in I_{CP}, which in turn reduces the in-band noise increase due to the non-linearity. By the same logic, lowering the dynamic phase excursion in this scenario would also aid in reducing in-band noise. Thus, to minimize the effect of such a non-linearity, a single-bit quantizer should be used in the SDM design if possible, and the divider step size should be set as low as possible such that the SDM is stable given the required FCW input for the desired tuning frequencies.

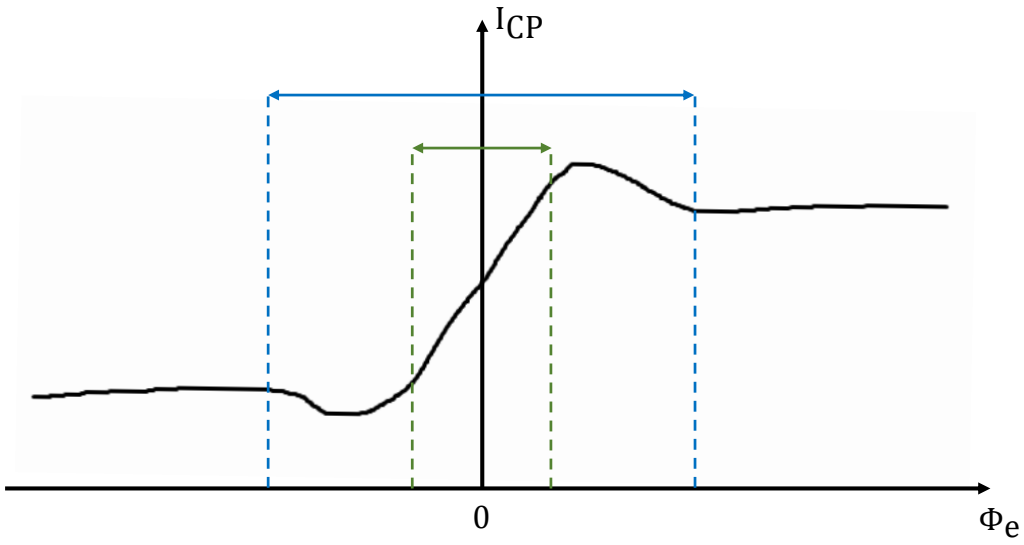

Figure 19: Sketch of how a smaller dynamic phase excursion, marked in green, can experience less of a non-linear phase transfer, represented in blue

3.3 System Specifications

To determine the specifications required by the system, the design considerations from the previous section should be taken into account to allow the system to tune to the required frequencies while meeting the desired performance. The specifications for integer-N PLL that the system will be driving were obtained from the designers, however to preserve confidentiality of the PLL design, the component values for the loop filter, charge pump, and VCO were omitted from this thesis, with the general structure of the PLL depicted in Figure 20.

Two independent sigma-delta divider systems were required; each one for generating a reference clock to a PLL. One PLL was used for sampling data, herein referred to as the data clock, and the other was for chip communication. The desired tuning frequency of the PLL for the data clock was variable around 14 GHz. For this clock, it was specified that $\tau_{jitter,rms}$ be less than 500 fs when integrated from 500 kHz to 10 MHz and the frequency tuning resolution less than 1 PPM/LSB. The reference clock frequency provided to a separate PLL for the communication clock should be 156 MHz on average.The specifications for the communication clock PLL were not provided. Thus, the system was designed to meet the specifications provided for the

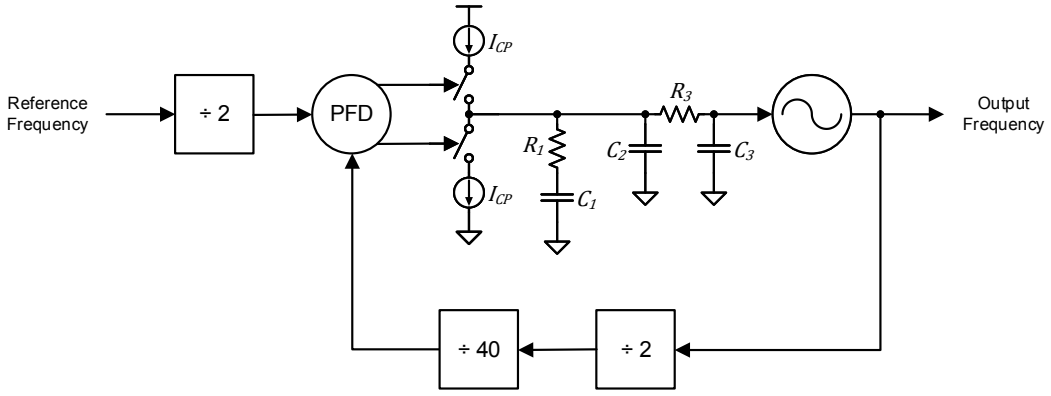

Figure 20: Data clock PLL structure

data clock PLL with the notion that any adjustments could be made to meet the requirements of the other design.

For an input clock, a 7.1568 GHz clock was available. Higher multiples of this frequency were also available, but ultimately decided against due to the power consumption of buffering and routing such a high frequency signal from its source. The clock source is a PLL-based synthesizer which uses a jitter-reduction scheme [11]. While it is primarily used for a separate function and not a part of this work, buffering and dividing down this high-frequency clock source allowed for the input clock to be low-jitter. This is worth mentioning because it means utilizing a high frequency clock at the sigma-delta divider input did not proportionally come at the expense of jitter. Due to the high-pass phase transfer of the jitter-reduction scheme, the lower end of the integration range of the phase noise specifcation was able to be out of band of the PLL used to filter the sigma-delta divider output in this work.

3.3.1 Required Fractional Divisors by Sigma-Delta Divider

To determine the design constraints for the sigma-delta divider, the fractional divisors D_M to produce $\overline{f_{ref}}$ given f_{in} were required. Knowing the feedback divider is 40 with a prescaled divide ratio of 2 as shown in Figure 20, simply rearranging (2.16) and applying the effective divide ratio of 80 means the required average reference frequency $\overline{f_{ref}}$ for the data clock is around 350 MHz. Then, rearranging (3.4) for D_M, the required fractional divisor D_1 for producing $\overline{f_{ref1}} = 350$ MHz and D_2 for

$\overline{f_{ref2}} = 156$ MHz are found:

$$\begin{aligned} D_1 &= \frac{f_{in}}{\overline{f_{ref1}}} = \frac{7.1568\,\text{GHz}}{350\,\text{MHz}} = 20.429 \\ D_2 &= \frac{f_{in}}{\overline{f_{ref2}}} = \frac{7.1568\,\text{GHz}}{156\,\text{MHz}} = 45.833 \end{aligned} \tag{3.18}$$

3.3.2 Order of Sigma-Delta Modulator

The specification for the order of the SDM is limited by the order of the loop filter of the integer-N PLL. As mentioned previously, the loop filter order must be at least one order greater than the SDM for a charge-pump based PLL. Due to the three capacitors within the loop filter, the PLL can be identified as a Type III PLL. Thus, in order for the PLL to be capable of attenuated the phase noise shaping, the order of the SDM must be less than or equal to 4.

3.3.3 Number of Sigma-Delta Modulator Input Bits

The number of input bits n is used to determine the input FSR to the SDM . As shown in (3.6), the FSR limits PPM/LSB along with D_M. Since D_M is fixed for each desired frequency, the limiting one should be substituted to allow the same SDM to be used in both applications. Substituting the limiting variables and 2^n for FSR into (3.6), the specification for n to give a frequency resolution in PPM/LSB less than 1 can be found by rearranging the inequality as follows:

$$\begin{aligned} 1 &> PPM/LSB \\ 1 &> \left| \frac{10^6}{D_M 2^n - 1} \right| \\ 2^n(20.42857) &> 10^6 + 1 \\ n &> \log_2 \left(\frac{10^6 + 1}{20.42857} \right) \\ n &> 15.579 \end{aligned} \tag{3.19}$$

Thus, to get less than 1 PPM/LSB, n should be set to at least 16 bits. To avoid unnecessary complexity and high power consumption, 16 bits were selected for the SDM input width specification.

3.3.4 Number of Sigma-Delta Modulator Quantizer Bits

As previously mentioned, if a substantial non-linear response of the phase detector and charge pump combination is present in the loop, the number of sigma-delta quantizer bits should be minimized to avoid raising the in-band phase noise. However, if the required fractional divisor D requires the SDM input to be outside the MSA for a single-bit quantizer, the divider step size Δ_{step} can be doubled to compensate, so long as the MSA is at least 0.5 the FSR. Since the result of (3.18) showed that fraction portion of the D_1 is within 0.25 to 0.75, its corresponding SDM input will be stable. However, the fraction portion of D_2 is greater than 0.75, meaning compensation should be used if a single-bit quantizer SDM is used. The doubling of Δ_{step} corresponds to the doubling of quantization interval Δ_Q; thus the in-band quantization noise would also double according to (2.6). This presents a trade-off depending on the extent of the non-linear response. However, since D_1 corresponds to frequency with the jitter requirement, it was opted to begin the design with a single-bit quantizer SDM and qualify it in simulation against a conventional third-order multi-bit MASH 111. The results of this simulation can be found in Chapter 4.

3.3.5 Oversampling Ratio

The OSR of the SDM, $OSR_{\Sigma\Delta}$, should be selected such that it allows the ENOB to be greater than the 16 bits required for the desired frequency resolution. To yield the benefits of noise shaping, $OSR_{\Sigma\Delta}$ should also be less than the OSR of the PLL itself OSR_{PLL}; limited by the PLL bandwidth and its lowest required reference frequency. It has already been established that an SDM with a single-bit quantizer cannot be stable with FSR input power. However, assuming the MSA will meet the specification of at half the FSR, 16 bits and 1 were substituted for $ENOB$ and A respectively into (2.13) and rearranged for $OSR_{\Sigma\Delta}$ to find the minimum bound. It is worth noting that the same result would be achieved by substituting 15 bits and 0.5 for $ENOB$ and A instead.

$$\begin{aligned} 16 &< 1 + \frac{2 \cdot 3 + 1}{2} \log_2(OSR_{\Sigma\Delta}) - 1.66 \log_{10}\left(\frac{\pi^{2 \cdot 3}}{1 \cdot 2 \cdot 3 + 1}\right) \\ \log_2(OSR_{\Sigma\Delta}) &> \frac{30}{7} + \frac{3.32}{7} \log_{10}\left(\frac{\pi^6}{7}\right) \\ OSR_{\Sigma\Delta} &> 2^{5.29964} \\ OSR_{\Sigma\Delta} &> 39.38687 \end{aligned} \tag{3.20}$$

To determine the upper bound on the OSR specification of the SDM, the loop bandwidth of the PLL was approximated to determine the OSR of the PLL. While the calculation for ω_{3dB} of (2.24) is for a second-order system and only includes the contribution of C_2 and R_2, the two additional high-frequency poles due to the inclusion of C_1, C_3, and R_3 will have minimal effect on ω_{3dB}. Plugging into (2.23) and (2.22), and substituting their results into (2.24), ω_{3dB} can be found. However, to preserve the confidentiality of the PLL design, only the result of this calculation is shown:

$$\omega_{3dB} = 1.603\,96\,\text{Mrads/sec} \tag{3.21}$$

To ensure the shaped quantization noise is filtered sufficiently to restrain the In-Band Noise (IBN) from reducing into the ENOB, the bandwidth of the SDM, $f_{BW_{\Sigma\Delta}}$, should be less than the loop bandwidth of the PLL while accounting for any reductions due to frequency division. Since the average reference to the PLL, which is its sampling rate, is half of $\overline{f_{ref}}$ output from the sigma-delta divider due to the frequency divider at the PLL shown in Figure (20), $f_{BW_{\Sigma\Delta}}$ should be at least twice the PLL loop bandwidth. Using this relationship, the upper bound on $OSR_{\Sigma\Delta}$ can be solved, where ω_{3dB} was set to $2\pi \times 300\,\text{kHz}$ to allow design margin:

$$\begin{aligned}
\frac{\omega_{3dB}}{2\pi} &< \frac{f_{BW_{\Sigma\Delta}}}{2} \\
\frac{\overline{f_{ref}}}{2 \cdot f_{BW_{\Sigma\Delta}}} &< \frac{\pi\overline{f_{ref}}}{2 \cdot \omega_{3dB}} \\
OSR_{\Sigma\Delta} &< \frac{OSR_{PLL}}{2} \\
OSR_{\Sigma\Delta} &< \frac{350\,\text{MHz}}{4 \cdot 300\,\text{kHz}} \\
OSR_{\Sigma\Delta} &< 291.67
\end{aligned} \tag{3.22}$$

3.3.6 Divider Step Size

Since the requirement on the upper bound of $OSR_{\Sigma\Delta}$ is dependent on $\overline{f_{ref}}$, one could opt to design two separate SDMs, each optimized for their respective application. For simplicity, however, the design of the SDM was done with $\overline{f_{ref1}}$ in mind and re-used for $\overline{f_{ref2}}$. However, since the fraction portion of D_2 is not within the range of 0.25 to 0.75 specified earlier while D_1 is within the range, it was opted to select Δ_{step} to be 1 for D_1 and 2 for D_2 to relax the MSA requirements of the SDM.

3.4 Implementation

Specific challenges overcome while implementing the design to meet the specifications set in the previous section are discussed here. The sigma-delta modulator implementation is covered in depth. The other two components of the sigma-delta divider system, namely the programmable frequency divider and the adder, were not implemented by the author, and as a such, only a description of their specifications are listed. Finally, the system level implementation is presented, with comments on feedback timing and divider step size.

3.4.1 Sigma-Delta Modulator

While sigma-delta converters come in the form of either DAC or ADC, it was opted to use a digital SDM for this application. Using a digital was simpler since the design could be coded in RTL, synthesized, then placed and routed. It also avoids the

complexities of implementing an analog SDM, such as generating a stable voltage reference, or requiring a DAC for firmware to communicate with the circuit for frequency tuning. The SDM design used was actually a sigma-delta ADC that was converted into a sigma-delta DAC. The original intent of this conversion was to obtain a digital sigma-delta that could be used independently for a 16-bit sigma-delta DAC application. Once this SDM was implemented, it was compared to a digital MASH 111 architecture, available in [4]. If its performance functioning as an NCO was superior to the MASH 111 architecture, it would be used, otherwise, the MASH 111 would be implemented. This comparison is made in Chapter 4. The bandwidth for the sigma-delta DAC application was specified to be 5 MHz. This was a much wider bandwidth requirement than for the sigma-delta divider application, and thus the measurements and characterization were completed for the sigma-delta DAC specifications since it limited the constraints for synthesis.

The implemented sigma-delta architecture depicted in Figure 21 and the corresponding coefficient values available in Table 1 produce a design that can achieve an OSR of 160. This is a single-bit quantizer variant of the Cascaded Integrator FeedBack (CIFB) architecture, available [12], with a feedback resonator coefficient omitted. The modulator is third-order with an OBG of 1.5, which coheres with the rule of thumb to set the OBG no greater than 1.5 for stability [13]. The coefficients were then quantized to be achievable as summations of 2^k, where k is an arbitrary integer, also available in Table 1.

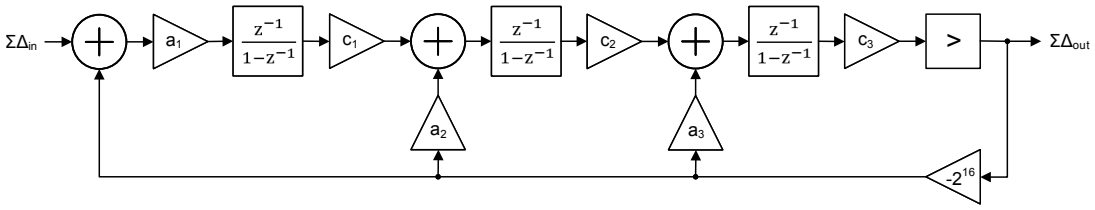

Figure 21: Z-domain CIFB SDM architecture

Assuming the quantization noise is injected solely at the quantizer, denoted by a "greater than" symbol in Figure 21, the NTF can be determined through a z-domain loop analysis. Then, using the forward Euler approximation of a discrete-time integrator, the continuous-time Laplace domain representation can be found.

Table 1: SDM coefficients

Coefficient	Floating Point Value	Quantized Approximation Value
a_1	0.0313	1/32
a_2	0.0256	1/64+1/128
a_3	0.071	1/16+1/128
c_1	0.125	1/8
c_2	1	1
c_3	11.2596	8+2+1

$$\begin{aligned} \alpha &= a_1 c_1 c_2 c_3 \\ \beta &= a_2 c_2 c_3 \\ \gamma &= a_3 c_3 \\ NTF(z) &= \frac{1}{1 + \frac{\alpha}{(z-1)^3} + \frac{\beta}{(z-1)^2} + \frac{\gamma}{z-1}} \\ NTF(z) &= \frac{(z-1)^3}{(z-1)^3 + \gamma(z-1)^2 + \beta(z-1) + \alpha} \end{aligned} \tag{3.23}$$

Substituting $z = e^{j\omega/f_s}$ and solving the NTF for its three poles and three zeroes using MATLAB yields the pole-zero plot shown in Figure 22. For both with and without quantization of the SDM coefficients, available in Table 1, the poles of the NTF are within the unit circle and thus have a stable response. It is worth noting that this stability is independent of the effects of the quantizer, which will limit the MSA and was analyzed by testing the dynamic range of the system.

The NTF of the system can also be used to estimate the jitter of the system. First, one can apply the approximation for the quantization noise from (2.5) into the NTF to obtain the frequency noise power, $F(z)^2$ by also recognizing this error power is spread over the sampling bandwidth f_{ref} [4]:

$$\begin{aligned} F(z)^2 &= {e_{Q_{rms}}}^2 NTF(z)^2 \overline{f_{ref}} \\ F(z)^2 &= \frac{1}{12} NTF(z)^2 \overline{f_{ref}} \end{aligned} \tag{3.24}$$

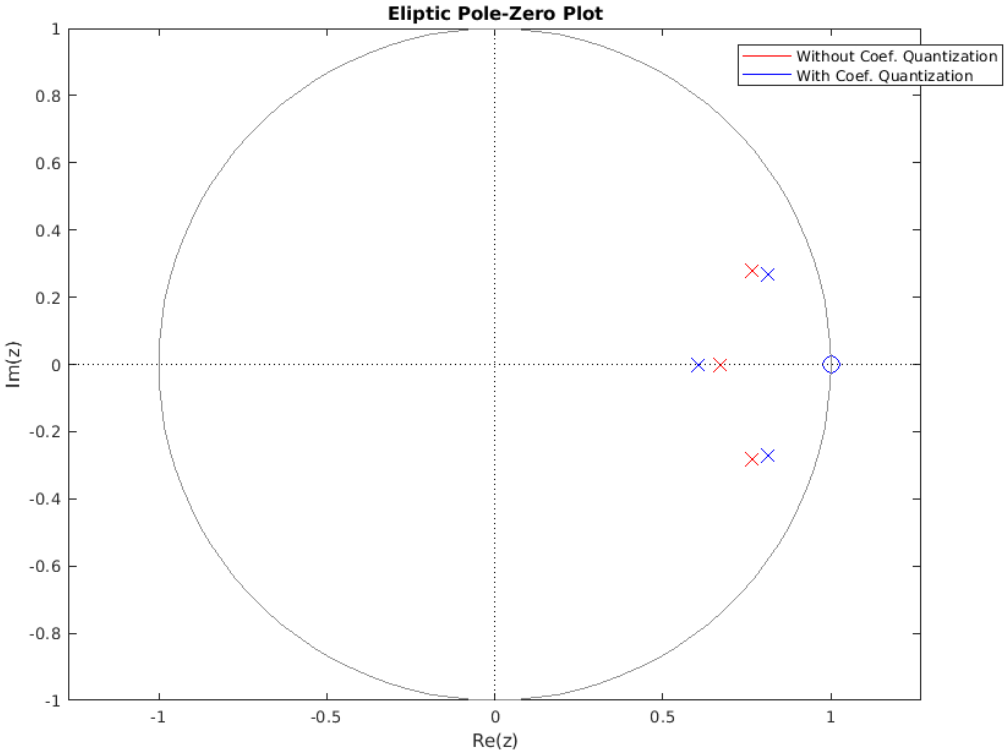

Figure 22: Pole-Zero plot for the SDM NTF. Circles represent zeroes while poles are denoted with an X

Recognizing that the SSB phase noise PSD is $\Phi(z)^2/2$ [4] and substituting (3.24) into (3.15), a z-domain expression for the phase noise PSD is made:

$$\begin{aligned} \frac{\Phi(z)^2}{2} &= \frac{2^2\pi^2}{2\overline{f_{ref}}^2}\left(\frac{z^{-1}}{1-z^{-1}}\right)^2 F(z)^2 \\ \frac{\Phi(z)^2}{2} &= \frac{\pi^2}{6\overline{f_{ref}}}\left(\frac{(z-1)^2}{(z-1)^3+\gamma(z-1)^2+\beta(z-1)+\alpha}\right)^2 \end{aligned} \tag{3.25}$$

Substituting $z = e^{j\omega/f_s}$ where $\omega = 2\pi f$ for offset frequency f, an estimate for the phase noise profile at the output of the SDM can be made. Shown in Figure 23, this "raw" or unfiltered phase noise profile was plotted from 10 kHz up to $f_N = f_s/2 = 175$ KHz. From this approximation, the expected 40 dB/decade of noise shaping can be observed, which ends prior to f_N due to the OBG being near 1.5. To estimate the phase noise contribution the sigma-delta divider has on the output

of the PLL, the phase transfer function of (3.12) can be applied, where the effective values $M = 2 \times 20 = 40$ and $N = 2 \times 40 = 80$ can be substituted. A plot of the phase transfer function is available in Figure 24. The corner frequency of this phase transfer is observed to be around 300 kHz, which is in proximity of the estimate found earlier. The out-of-band attenuation falls at a rate of -60 dB/decade as expected since $F(s)$ is third order. The DC gain of the phase transfer function can be solved to be $N/M = 2$ by substituting $\omega = 2\pi f = 0$ into (3.12), corresponding to a DC gain of 6 dB. Applying this phase transfer function to the raw SDM phase noise, the estimated profile of the phase noise of the sigma-delta divider filtered by the PLL is shown in Figure 23. To estimate the jitter, trapzoidal integration was applied from 500 kHz to 10 MHz using MATLAB to solve (2.20) for $\tau_{jitter,rms}$. Thus, the estimated RMS jitter contribution for the sigma-delta divider projected to the output of the PLL over the specified bandwidth was 365.8 fs.

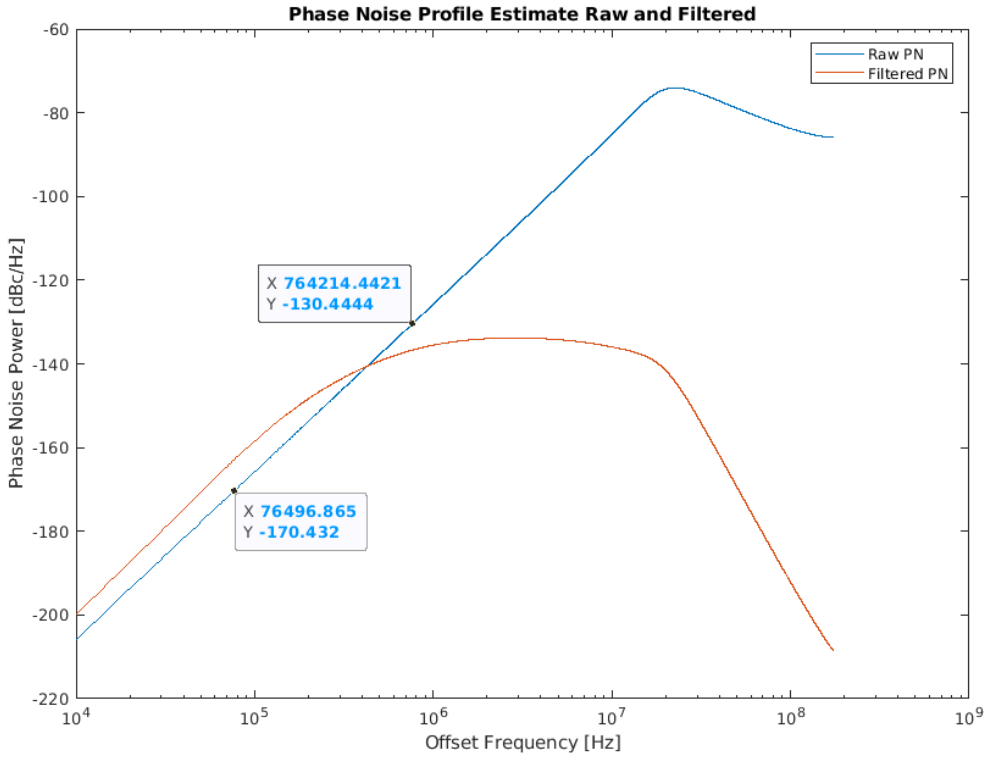

Figure 23: Estimated phase noise profile of the SDM before being filtered (raw, in blue) and after filtering (in red)

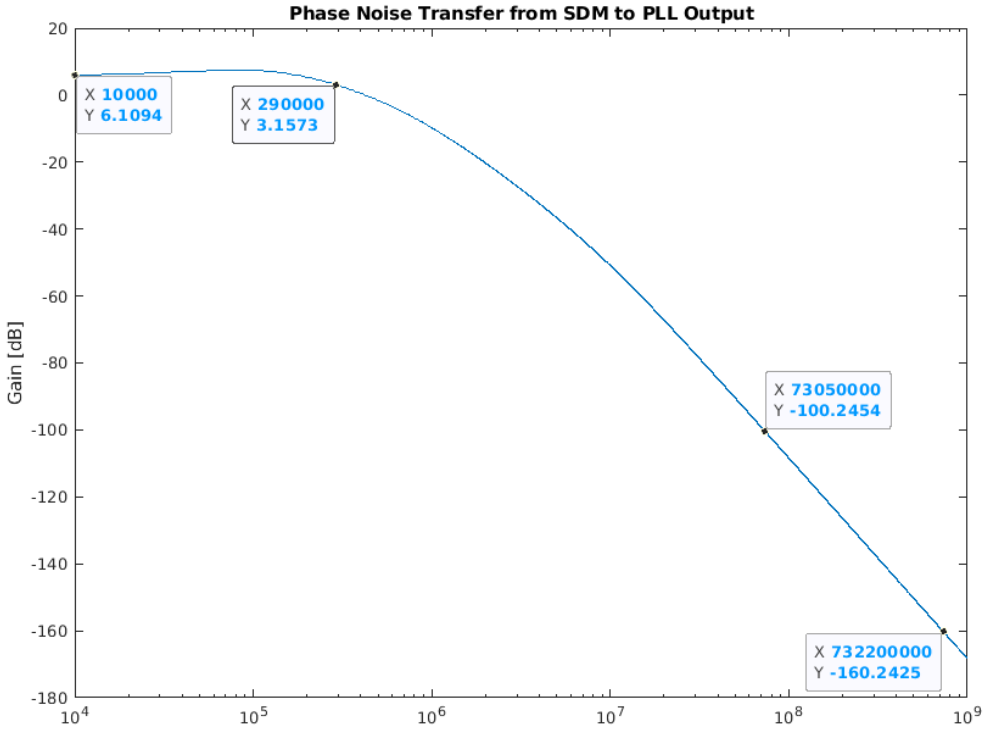

Figure 24: Phase noise transfer profile of the PLL

Ignoring the quantization error input for the time being and considering the SDM input only, the STF can be determined using a similar analysis and assumptions as for the NTF. After approximating the STF into the Laplace equivalent in steady state using the forward Euler discrete-time integration, the DC gain of the loop can be approximated by substituting $\omega = 2\pi f = 0$

$$\begin{aligned} STF(z) &= \frac{\frac{\alpha}{(z-1)^3}}{1 + \frac{\alpha}{(z-1)^3} + \frac{\beta}{(z-1)^2} + \frac{\gamma}{z-1}} \\ \frac{1}{s} &\approx \frac{1}{f_s(z-1)} \\ STF(s) &\approx \frac{\alpha {f_s}^3}{s^3 + \gamma f_s s^2 + \beta {f_s}^2 s + \alpha {f_s}^3} \\ |STF(\omega = 0)| &\approx \left| \frac{\alpha {f_s}^3}{\alpha {f_s}^3} \right| = 1 \end{aligned} \tag{3.26}$$

Since the FCW of the sigma-delta divider is nearly static, its rate of change is at or near DC. A DC gain of 1 is thus desirable and it indicates the loop should not introduce a frequency offset greater than its PPM/LSB resolution by design. A visual representation of the STF and NTF plotted against frequency was generated by setting the sample rate f_s to 1.6 GHz and is available in Figure 25. The in-band gain is observed to match the calculated DC gain with a linear value of 1, and the OBG aligns closely to the target of 1.5 with a linear value of 1.487. The third-order frequency-noise shaping is also observed in the NTF, having a slope of 60 dB/decade. Note that it is known that f_s should be set to 1.6 GHz for the sigma-delta DAC application by rearranging the relation (2.2) for f_s, where f_{BW} is the bandwidth requirement of the SDM.

$$\begin{aligned} OSR_{\Sigma\Delta} &= \frac{f_s}{2 \cdot f_{BW}} \\ f_s &= OSR_{\Sigma\Delta} \cdot 2 \cdot f_{BW} \\ f_s &= 160 \cdot 2 \cdot 5\,\text{MHz} \\ f_s &= 1.6\,\text{GHz} \end{aligned} \tag{3.27}$$

To determine the MSA and dynamic range of the modulator, an SQNR measurement using a Hanning windowed FFT was repeated several times while increasing the amplitude of the sinusoid input to SDM as described in Chapter 2. The model used for these measurements used floating point values and accumulators with infinite range for the time being. The input frequency used was calculated by selecting P to be 2^{14} points and selecting 11 as the prime bin number bin_{prime}:

$$\begin{aligned} f_{in} &= \frac{bin_{prime} \cdot f_s}{P} \\ f_{in} &= \frac{11 \cdot 1.6\,\text{GHz}}{2^{14}} \\ f_{in} &= 1.074\,221\,875\,\text{MHz} \end{aligned} \tag{3.28}$$

Observable in Figure 26, the MSA of the modulator is approximately -3 dB of the FSR; any amplitude higher and the modulator ceases to operate correctly. The optimal dynamic range is 108.7 dB at an amplitude of about -6 dB of the FSR;

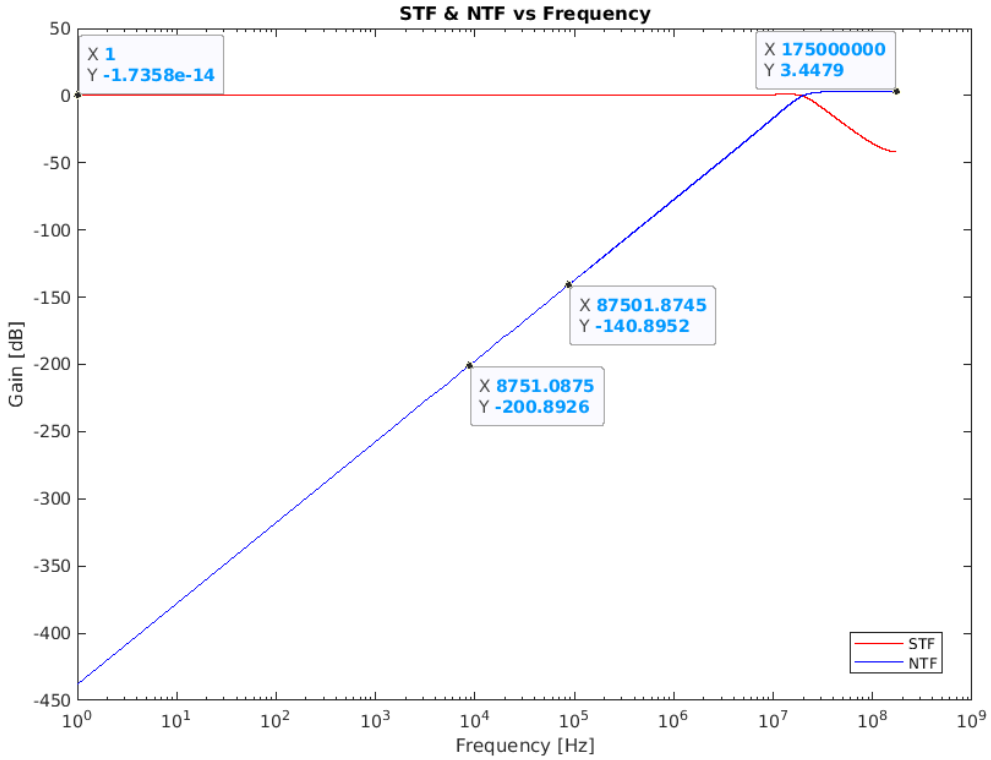

Figure 25: SDM STF in red, NTF in blue

indicating the final point where the loop acts as a linear modulator. The reason it is the optimal dynamic range is that the real-life implementation of the modulator gain elements (for the coefficients) and accumulators will have a finite number of bits.

The sizing of these binary gain and accumulator components along with their resolution trade-offs are discussed herein, but first an important note on fractional spurs should be noted now that the MSA of the modulator design has been determined. Since fractional spurs are a function of the FCW fed to the sigma-delta divider, and the MSA limits the range of usable FCW values for a stable modulator as shown in (2.27), the MSA inherently limits the possible locations of fractional spurs. Given that the FCW changes at DC, the closest in spur would occur when the FCW is at its minimum possible value. A sinusoidal input with amplitude 0 would correspond to an FCW of half the FSR, thus the lowest FCW value can be found be rounding up the difference between half the FSR and half the FSR multiplied with the MSA:

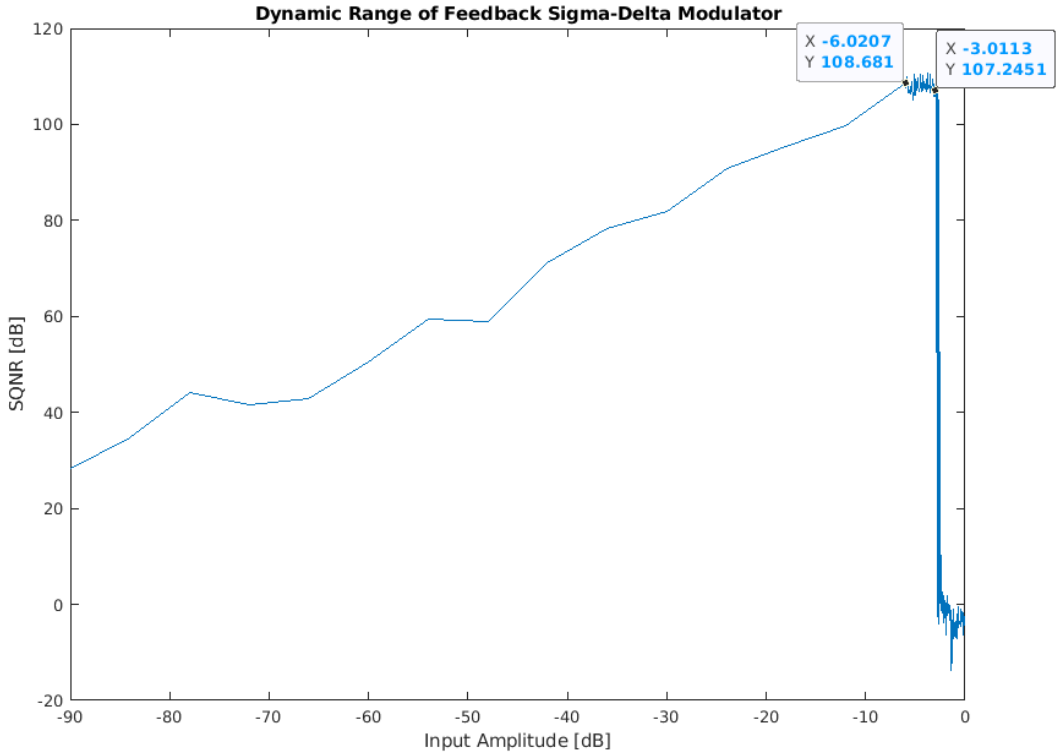

Figure 26: Plot of multiple SQNR as a function of increasing input amplitude, taken with respect to the FSR. Peak value represents the dynamic range, amplitude value prior to drop-off is an approximate MSA

$$\begin{aligned} FCW_{min} &= \frac{FSR}{2} - \frac{FSR}{2} MSA \\ FCW_{min} &= 2^{15}(1 - \frac{1}{\sqrt{2}}) \\ FCW_{min} &= 9597.525 \end{aligned} \tag{3.29}$$

Despite likely never using values of FCW as low as FCW_{min}, calculating the closest-in spur frequency for a given output frequency indicates whether fractional spurs should be of concern. Substituting 9598 in for FCW in (2.27), dividing f_{ref} by 2 to account for the reference divider shown in Figure 20, and setting arbitrary integer k to 1 for the lowest integer multiple, the lowest fractional spur frequency for both f_{ref1} and f_{ref2} are found to be out of band:

$$
\begin{aligned}
f_{spur1} &= \frac{f_{ref1}}{2}\frac{FCW}{FSR} \\
f_{spur1} &= \frac{350\,\text{MHz}}{2}\frac{9598}{2^{16}} \\
f_{spur1} &= 25.6295\,\text{MHz} \\
f_{spur2} &= \frac{f_{ref2}}{2}\frac{FCW}{FSR} \\
f_{spur2} &= \frac{156\,\text{MHz}}{2}\frac{9598}{2^{16}} \\
f_{spur2} &= 11.4235\,\text{MHz}
\end{aligned}
\tag{3.30}
$$

To approach the real life implementation, the model was adjusted to have finite-sized accumulators. To allow the quantization noise be shaped by the loop to match the NTF as much as possible, the quantization error could be introduced only at the quantizer. However, it is possible to introduce quantization at each integration stage, such as this implementation of the MASH-111 architecture [4]. To maintain the NTF of the selected design, quantization should ideally not hinder the resolution until the quantizer in the final stage. Therefore, the number of bits in each accumulator should be sized to accommodate its internal rate of growth as well as the rate of growth of its input without saturation. To not introduce excessive quantization, the in-band quantization noise must remain lower than the Nyquist rate equivalent SQNR dictated by the number of input bits to the SDM n. To convert n to its corresponding Nyquist rate SQNR in dB, n can be substituted for Q of (2.11) with OSR set to 1, then the expression can be converted to dB. The input to an accumulator can be modeled as an m^{th}-order SDM, where m would be set to the number of accumulator stages prior and Q representing the minimum number of bits required. Realizing that the in-band noise level with respect to a signal with amplitude A is the reciprocal to its SQNR and can be projected to the input of an accumulator by multiplying by the forward gain G, the minimum value for Q can be solved by plugging in (2.12) and (2.13) into the inequality relating the SQNR to its Nyquist rate resolution:

$$\begin{aligned} SQNR \cdot G^2 &\geq 10^{(n*6.02+1.76)/10} \\ 2^{2Q} &\geq \frac{2 \cdot 10^{(n*6.02+1.76)/10} \cdot \pi^{2m}}{3 \cdot (2m+1) \cdot OSR^{2m+1} \cdot G^2 \cdot A^2} \\ Q &\geq 0.5 \log_2 \left(\frac{2 \cdot 10^{9.808} \cdot \pi^{2m}}{3 \cdot (2m+1) \cdot 160^{2m+1} \cdot G^2 \cdot A^2} \right) \end{aligned} \tag{3.31}$$

To allow some margin on the specification for the MSA to exceed 0.5, the value MSA of the modulator, approximately $1/\sqrt{2}$ from Figure 26 with a model with unlimited number of bits, was substituted for A. For each of the three accumulators, the minimum number of bits Q_{acc1}, Q_{acc2}, and Q_{acc3} are solved by substituting the number of previous accumulators for m and the product of all forward gain coefficients prior to the accumulator for G, following the methodology of [12]:

$$\begin{aligned} Q_{acc1} &\geq 0.5 \log_2 \left(\frac{2 \cdot 10^{9.808} \cdot \pi^{2 \cdot 0} \cdot \sqrt{2}^2}{3 \cdot (2 \cdot 0 + 1) \cdot 160^{2 \cdot 0 + 1} \cdot {a_1}^2} \right) \\ Q_{acc1} &\geq 17.84 \\ Q_{acc2} &\geq 0.5 \log_2 \left(\frac{2 \cdot 10^{9.808} \cdot \pi^{2 \cdot 1} \cdot \sqrt{2}^2}{3 \cdot (2 \cdot 1 + 1) \cdot 160^{2 \cdot 1 + 1} \cdot {a_1}^2 \cdot {c_1}^2} \right) \\ Q_{acc2} &\geq 14.37 \\ Q_{acc3} &\geq 0.5 \log_2 \left(\frac{2 \cdot 10^{9.808} \cdot \pi^{2 \cdot 2} \cdot \sqrt{2}^2}{3 \cdot (2 \cdot 2 + 1) \cdot 160^{2 \cdot 2 + 1} \cdot {a_1}^2 \cdot {c_1}^2 \cdot {c_2}^2} \right) \\ Q_{acc3} &\geq 8.336 \end{aligned} \tag{3.32}$$

After rounding the results of (3.32) up to their respective nearest integer values, the model of the design was improved to more closely imitate a physical implementation, specifically the SDM coefficients and binary accumulators. Previously, the accumulators were modeled to account for a roll-over at 2^Q using the modulus operator, however, floating point values were used to model the accumulated result instead of integer values. By plotting the SQNR, an issue was uncovered with the way the gain coefficients were realized once integers were used to represent the binary values instead. Since the gain coefficients were quantized to be summations of 2^k, where k is an arbitrary integer, the simplest way to implement each gain coefficient was to shift

each binary word to the left k times for each 2^k and sum the result of each multiplication. The problem was observed for factors where k is negative, or put in another way, where the gain is less than 1. When k is negative, the bits should instead be shifted to the right, and since no bits were encoded for the fractional portion of the floating point value, the result is effectively truncated to an integer. This truncation injects undesired quantization noise into the loop not characterized by the NTF, observable in the blue trace of Figure 28.

To overcome the issue of truncating the binary value when k is negative, it was reasoned that the inputs to the coefficients could simply be scaled up by the reciprocal of the coefficient. Then, the problem coefficients were identified. Coefficients c_2 and c_3 of Table 1 were both greater than 1 and thus k is positive and cannot lead to truncation. Coefficients a_2 and a_3 are less than 1, however they are input with either 0 or the negative FSR. The magnitude of k for all values of 2^k for both coefficients are less than the 16 bits of the FSR, thus the resulting output will always be an integer and will not be subject to truncation. Coefficients a_1 and c_1 are less than 1 and are input with the difference between the feedforward and feedback paths, and are of potential concern here.

To scale up the inputs to the coefficients a_1 and c_1, the input and feedback paths were scaled up by the reciprocal of the coefficient. First, scaling was completed for the quantization effect of a_1. It was realized that scaling the input and feedback path to the first adder would cancel the effect of a_1. Thus, scaling up the input and feedback by $1/a_1$ is equivalent to removing a_1 and scaling a_2 and a_3 by $1/a_1$, as shown in Figure 27. The same technique could be applied for additional scaling of $1/c_1$. The improved SQNR yielded by applying this technique can be observed in Figure 28. Prior scaling, the SQNR was modeled to be 84.23 dB. After scaling for the quantization of a_1, then for both a_1 and c_1, the SQNR was found to be 106.9 dB and 108.7 dB, respectively. Note that the accumulators were given infinite range once again to perform this measurement since the coefficient scaling would require the accumulators to be sized differently than the ideal sizes depicted in (3.32).

Since scaling for only a_1 provided sufficient SQNR for the applications of SDM, the scaling for c_1 was not implemented as it would require larger accumulators for the excessive resolution. Through iteration, the final accumulator sizes used for this

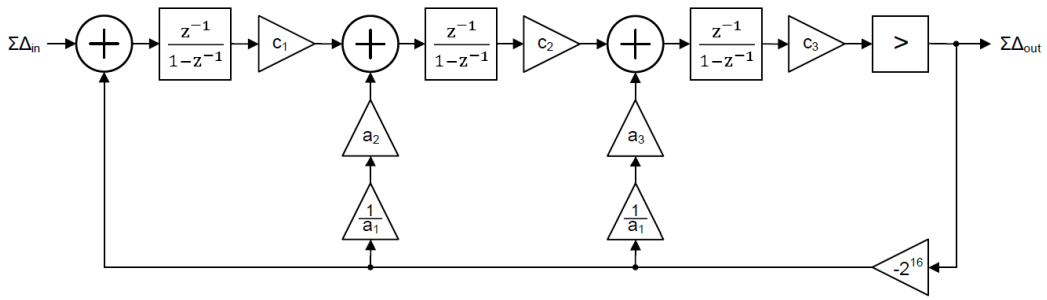

Figure 27: Z-domain CIFB SDM architecture with effective $1/a_1$ coefficient scaling of the feedback and input

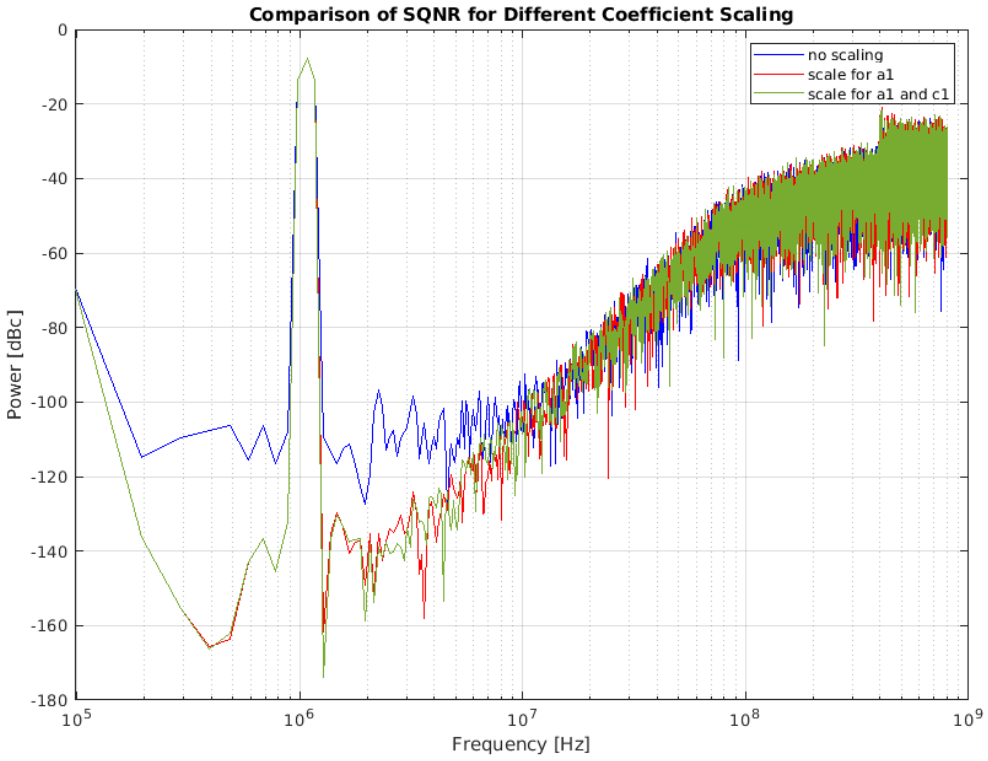

Figure 28: SQNR of the SDM with scaling the coefficients from their quantized values (blue), after scaling for a_1 (red), after scaling for a_1 and c_1 (green)

design were $Q_1 = 19$, $Q_2 = 18$, and $Q_3 = 18$. The Verilog code for this design is available in Appendix A. It is worth noting that in hindsight, the author acknowledges that Q_3 could have been sized smaller by implementing the quantizer to simply be the sign bit of the output of c_3 such that the quantizer threshold was zero, rather than the unsigned value of half the maximum swing of the output of c_3. This Verilog code

was synthesized, then placed and routed using Complimentary Metal Oxide Semiconductor (CMOS) logic gates within TSMC's 7 nm FinFET standard cell library.

3.4.2 Frequency Divider

The architecture used to implement the programmable frequency divider portion of the sigma-delta divider was the "truly modular" MMD described in Chapter 2. It was selected because an 8-stage CMOS design was already placed and routed in the 7 nm process, which could be repurposed for this application to save on design effort. The divisor extension circuit was included for each of the 8 stages, meaning the programmable control word was 9 bits with a step size of 1. Substituting 8 for k stages into (2.15), the programmable divisor range for the divider is from 2 to 511. To cover the unsupported input values, if 0 is programmed, the MMD divides by 2, while a programmed value of 1 would divide by 3. While this range is substantially larger than what was required, since the layout area of this digital circuit is miniscule when compared to the total layout area, the additional area required was not of concern. Unfortunately, this comes at the cost of additional dynamic power consumption, since 3 unused stages will have their respective divide by 2 portions switching at one half, one quarter, and one eighth of the output clock frequency.

In general, there are some potential caveats to using this architecture for a sigma-delta divider instead of the generic MMD architecture, also discussed in Chapter 2, that are worth mentioning. Firstly, the duty cycle becomes increasingly narrow as the programmable divisor is increased and can be less than 25% of the period. Despite the PFD being a rising-edge triggered circuit, narrow pulses raise concern for creating a timing violation if the circuit is not sized accordingly. However, the divide by 2 at the reference input of the PLL will bring the duty cycle closer to 50% (recall that the input frequency is modulated by the SDM). This should not be problematic so long as there is sufficient swing at the output and downstream from the divider up until the PLL input. Another disadvantage is that since the output is taken at Mod_{out} of the first stage, and Mod_{out} is a function of Mod_{in} for a divide by 2 or 3 cell (refer to Figure 4), it will take longer for the frequency to respond to a change in the programmed divisor word. For example, if the LSB is changed, not only does the change in frequency have to ripple down the chain from the clock output of the first block to the clock input of the last active block, the last active block needs to generate the corresponding Mod_{out} that will ripple back up the chain to Mod_{in} of the

first block.

3.4.3 Adder

There are several adder architectures that could have been selected to implement the sigma-delta divider. Since an existing placed and routed design already existed for a 14-bit CMOS Kogge-Stone adder, this design was used to save on design effort. This is a common carry-look-ahead full-adder architecture [14], was not designed by the author, and will not be covered in detail.

3.4.4 Integrated System

A system level block diagram of the integration of the three core blocks discussed in this chapter, namely the SDM, the MMD, and the adder, is available in Figure 29. This diagram specifically represents the first of the two sigma-delta divider variants, where the second implementation is discussed in subsequent paragraphs. Here, Verilog syntax is used to express the allocation of bits, where 0 and 1 represent logic zero and logic one. The divider output is inverted prior to being fed back to the SDM sample clock input; driving rising-edge accumulators. The single-bit SDM output is concatenated with 13'd0 for one adder input, while the 8-bit programmable divisor control word M is concatenated with 6'd0 for the other input. 9 LSBs of the adder output are passed to the MMD programmable control word. The SDM is input with a 16-bit FCW and has a reset input to initialize the accumulators to zero. The nature of this feedback loop allows the frequency divisor of the MMD to be modulated between M and $M + 1$ by the SDM in a coherent manner.

To allow the clocks to be measured and input to the integrated circuit, Current-Mode Logic (CML) input and output drivers were included in the design implementation. Due to these circuit elements interfacing directly with lab instruments, their designs include ElectroStatic Discharge (ESD) protection circuitry. The design of these drivers and ESD protection circuits were not completed by the author and will not be discussed. A clock tree was routed from the input driver to two CML to CMOS converters, each corresponding to one of the two sigma-delta divider variants. This 7.1568 GHz single-ended clock drives the CMOS sigma-delta divider, whose output is then converted back into CML prior to the output driver.

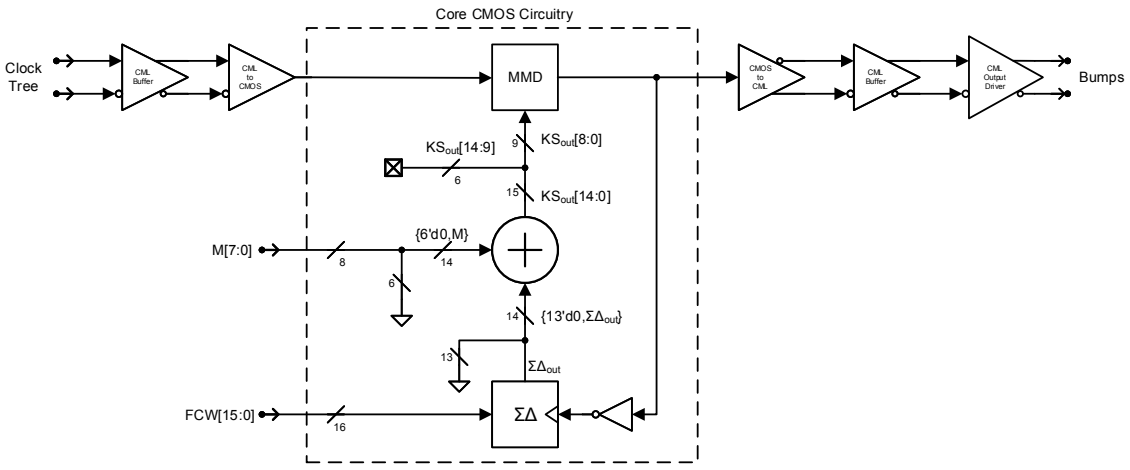

Figure 29: System block diagram of the design implementation of the sigma-delta divider

Feedback Inverter

It was mentioned that an inverter was placed at the output of the MMD prior to clocking the SDM. This inverter is used to delay the divider code word such that the sigma-delta is updated on the falling edge of the output clock. This prevents the divider from changing its programmed value, due to the sigma-delta modulation, while a pulse is being output, mitigating potential timing violations. While the propagation delay from the SDM to the divider input, mostly dominated by the propagation delay of the adder circuit, was likely sufficiently long to hold the value long enough such that Mod_{out} would already go low prior to R switching value, placing an inverter ensures this is the case. Equivalently, the SDM could have been designed to be falling-edge triggered, or the final divide by 2 or 3 stage could have had its output inverter removed. This timing is best observed through simulation results, where waveforms of such nature are available in Chapter 4.

Divisor Step Size

The implementation of the second variant of the sigma-delta divider to drive the reference of the communication clock PLL is shown in Figure 30. The difference between the two variants lies in the bit allocations of the buses input to the SDM and input to the adder. The 16-bit FCW was effectively shifted to the right to divide its value by two. The output of the SDM was shifted to the left to multiply its value by

two. This maintains the equation for $\overline{f_{ref}}$ of (3.14). However, it changes the criteria for valid values of the FCW. Not only will odd values now function as if they were rounded down to the nearest even value, the minimum and maximum stable values for FCW, limited by the MSA, are both increased by a factor of 2 up until the maximum value of the FCW of $2^{16} - 1$. Since the MSA is greater than the specification of half the FSR and M is 45 for this variant as determined in (3.18), it can be said that the tunable fractional divisor range now covers at least from 45.5 to 46. If the FCW was not shifted to the right, the fractional divisor range would have covered from at least 45.5 to 46.5, similar to that depicted in Figure 17 for a divider step size of 2. While the MSA was approximated to be $1/\sqrt{2}$, thus technically supporting the fractional divisor of 45.833 with the first variant of the sigma-delta divider, it was opted not to operate the sigma-delta past the amplitude of half the FSR, corresponding to the dynamic range as shown previously in Figure 26.

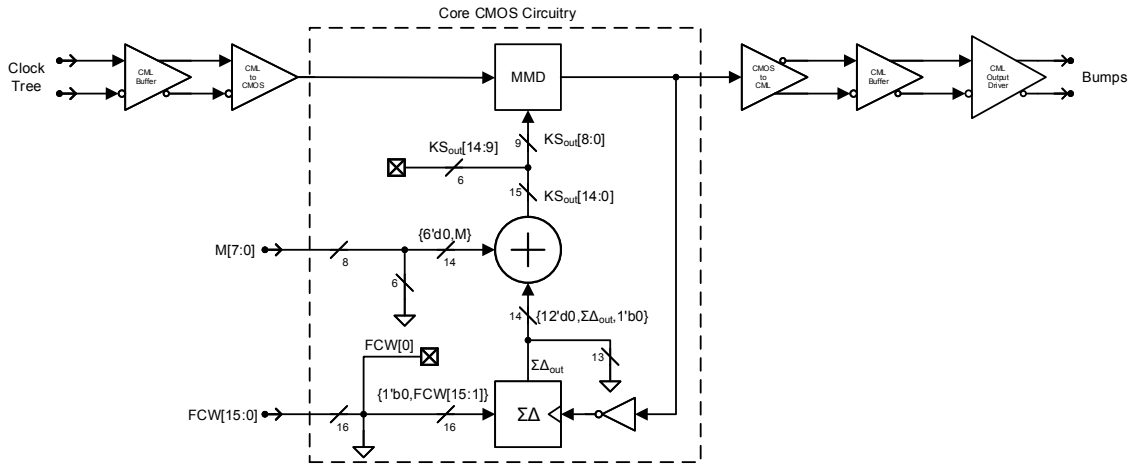

Figure 30: System block diagram of the design implementation of the sigma-delta divider with the divider step size increased by a factor of 2 and FCW decreased by a factor of 2

Layout

While the design and integration of the system was completed by the author, the layout integration of the blocks within the sigma-delta divider system was not. A single image of the layout for the first sigma-delta divider variant is included in Figure 31 for completeness while creating a visual for the layout area used for the design.

The layout of the two variants are nearly identical, and thus only the layout for one was included here. Note that this image is of a single instance of the sigma-delta divider and does not include the entirety of the clock tree nor the input driver. It also does not include the power mesh or decoupling capacitors to not obscure the main circuit elements. The layout dimensions of a sigma-delta divider slice are 220.5 µm by 372 µm, where the area is largely dominated by the CML output driver and output bumps. The core CMOS circuitry dimensions are only approximately 45 µm by 38 µm. Due to the nature of CML and the size when compared to the core CMOS circuitry, the CML output driver was anticipated to dominate the power consumption of the system prior to its simulation in Chapter 4.

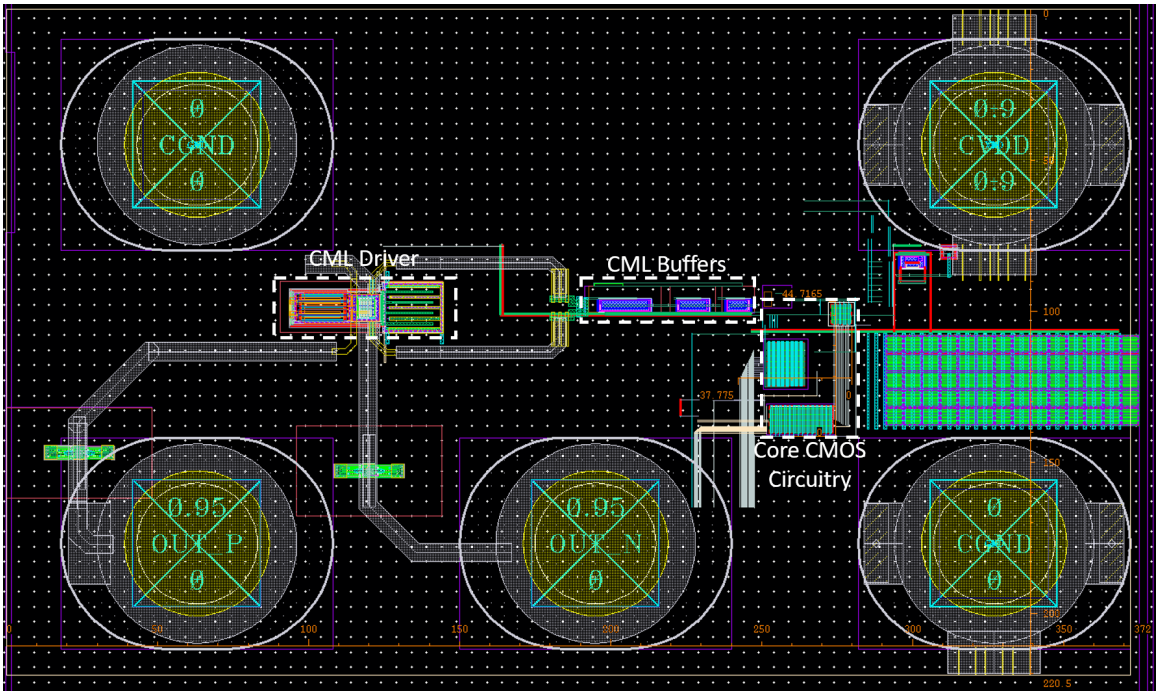

Figure 31: Layout of the sigma-delta divider and CML output circuitry. Units of length measurements are in µm

3.5 Chapter Summary

To understand the concept of using a sigma-delta divider in the reference path and allow reuse of existing theory,the system it was compared to two analogous systems; sigma-delta converters and fractional-N PLLs. Once the concept was demonstrated,

design considerations for the SDM and MMD as well as the interactions between them were introduced. Using these design considerations, a set of design specifications were developed for the blocks in the system in order to meet the system requirements. Finally, the implementation of the SDM was detailed along with the descriptions of the other blocks to explain how these specifications were met. The implemented design was taped in the 7 nm FinFET process and an image of the layout was included, where the simulations and measurements were conducted to verify the design and implementation, as discussed in the following chapter.

Chapter 4

Simulations and Measurements

To verify that the implemented design met the system requirements and behaves as designed, the sigma-delta divider system was simulated and measured both with and without the integer-N PLL. The methodology of these simulations and measurements will be detailed herein and compared with the expected results from the theory and the design discussed in previous chapters.

4.1 Simulations

Prior to sending the design out for fabrication in the 7 nm process, it was crucial to ensure the system behaves as designed. Simulations were also used to verify the design and confirm design decisions yet to be sufficiently validated in previous chapters. The results and methodologies for simulations run for the sigma-delta divider system on its own will be covered first, then two models of the PLL will be included. These will be compared with the design theory, calculation results, and graphs presented in previous chapters to confirm the behaviours are as expected.

4.1.1 Sigma-Delta Divider

The two core blocks, namely, the SDM and the MMD, within the sigma-delta divider system were simulated on their own prior to simulating them as the sigma-delta system. Simulation results for various design aspects of SDM are presented herein, followed by a demonstration that the MMD does indeed operate within the required region. Then, simulations and methodologies for the sigma-delta divider feedback timing, average frequency verification, and average power consumption estimation

are shown.

SDM Resolution, Dynamic Range, and MSA

To ensure the implemented SDM design meets the performance requirements, the Analog Mixed-Signal (AMS) simulator in Virtuoso was used to simulate the Verilog code available in Appendix A. Knowing that the dynamic range should be measurable with an input amplitude of -6 dB with respect to the FSR, a discrete-time sinusoid sampled at a rate of $f_s = 1.6\,\text{GHz}$ was applied. To implement this discrete-time sinusoid, the *ahdlLib* Verilog-A behavioural library available from Cadence was used. The *vco* cell, input with 0 V, was used together with the *adc_16bit_ideal* block to implement the test signal. With this setup, the frequency resolution of the input test signal is limited to 1 Hz, which is problematic for using the value of f_{in} determined previously in (3.28). This limitation was countered by setting the number of points to be a multiple of the OSR, which allows f_{in} to be an integer value. Setting P to be 20480 and modifying bin_{prime} to be 17, a new value for f_{in} was calculated:

$$\begin{aligned} f_{in} &= \frac{bin_{prime} \cdot f_s}{P} \\ f_{in} &= \frac{17 \cdot 1.6\,\text{GHz}}{20480} \\ f_{in} &= 1.328\,125\,\text{MHz} \end{aligned} \tag{4.1}$$

The measured SQNR of 106.0 dB is observable in Figure 32. This corresponds to an ENOB of 17.3 bits, greater than the required 16 bits for this design. The SQNR matches very closely with the result from the SDM modeled in MATLAB in Chapter 3 of 106.9 dB. The small difference can be mostly attributed to the difference in the values of P used, which altered the number of frequency bins within the integration bandwidth and forced f_{in} to change in value.

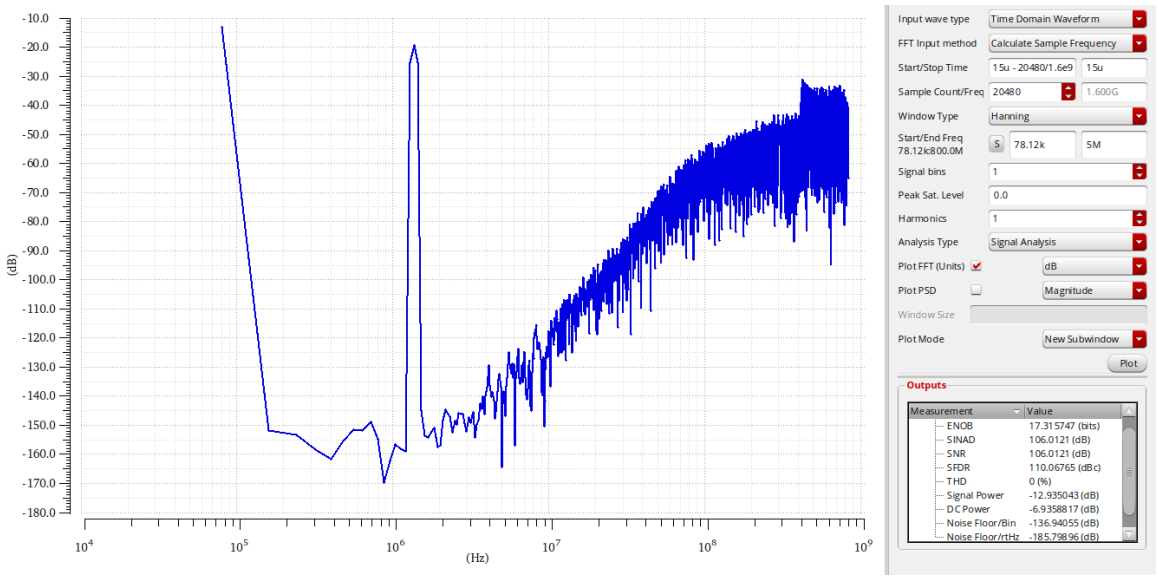

Figure 32: SQNR measurement of the SDM Verilog code with the amplitude of the input sinusoid set to roughly half the FSR

To evaluate that this measured SQNR still corresponds to the dynamic range, the amplitude of the discrete-time sinusoid was swept while measuring the SQNR with the described test bench. The input amplitude incremented from $-60\,\text{dB}$ to $0\,\text{dB}$ in steps of $3\,\text{dB}$; where $0\,\text{dB}$ corresponds to FSR. Shown in Figure 33, the dynamic range corresponds to an amplitude of $-6\,\text{dB}$ and MSA is $-3\,\text{dB}$, agreeing with the SDM model from Chapter 3.

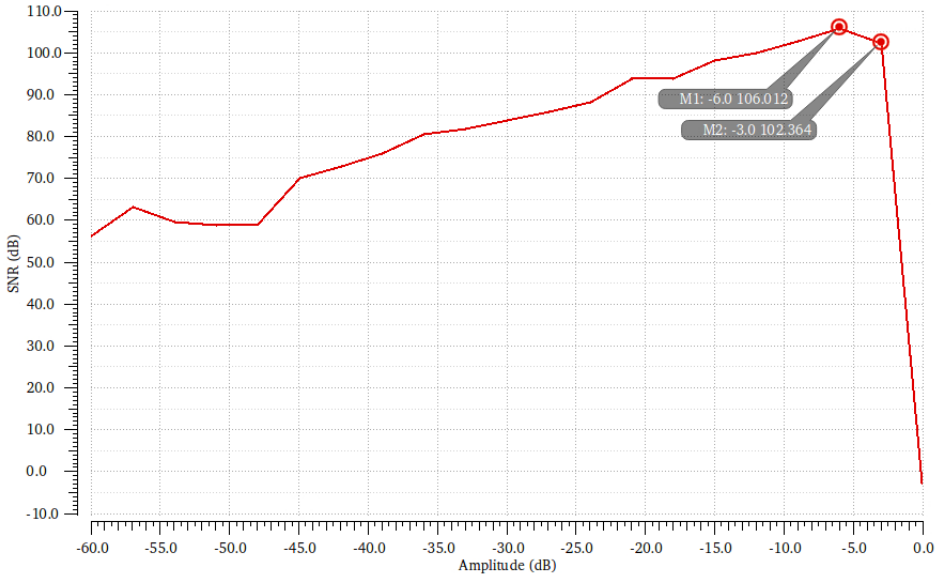

Figure 33: Dynamic range measurement of the SDM Verilog code. Amplitude is presented in dB with respect to the SDM FSR, and was swept from -60 dB to 0 dB in steps of 3 dB

MMD Operation Range

To verify the MMD design was suitable for this application, a testbench was created to test its programmability. Since the MMD divisor control word R should toggle between 20 and 21 for the data PLL reference clock and 45 and 47 for the communication PLL reference clock, R was swept from 2 to 64; a subset of its full programmable range of 2 to 511. Verilog-A models were used again to construct a testbench of accumulators which toggle R between M and $M+1$ for 8 divider output periods prior to incrementing M by 1. To generate R, the Kogge-Stone adder was used. The simulation result of inputing a 50% duty cycle 10 GHz clock into the MMD of the described testbench is visualized in Figure 34, where *mmd_integer_divisor* represents M and *mmd_divisor* is the frequency input to the MMD divided by its output frequency.

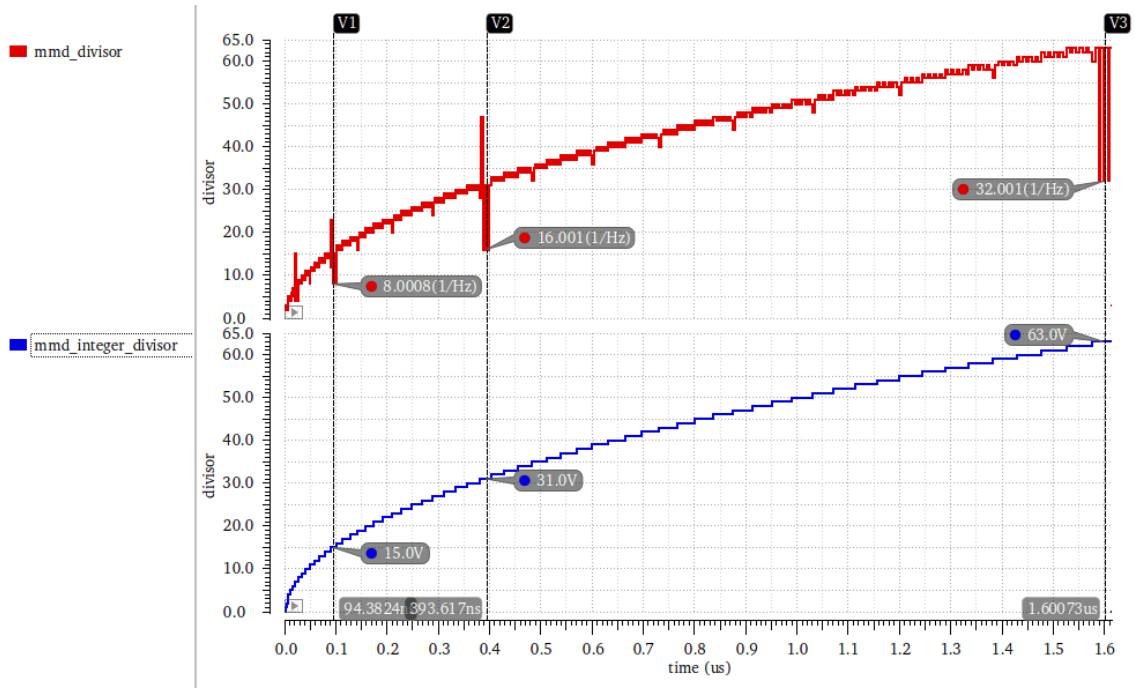

Figure 34: Demonstration that the divisor programmed on the MMD can be modulated in the required values without glitching

It can be observed that M was swept from 2 up to 64. In most cases, the result for *mmd_divisor* tracks the expected result of R. The exceptions to this are when M is incremented and when M is set to $2^k - 1$. It can be seen that in the cycle following M being incremented, *mmd_divisor* does not correspond to the expected value. However, in the subsequent cycles for that value of M, it does match if M is not $2^k - 1$. This was deduced to be because there is a nonzero delay time to generate R along with the MMD requiring a nonzero setup time to switch to the programmed value while remaining coherent. This was nonproblematic in the actual sigma-delta divider loop and emphasizes the importance of the feedback inverter, as discussed in the subsequent section. the MMD output is located in the first divide by 2 or 3 stage which is controlled by the LSB of R, and if incrementing M causes the LSB of R to change from logic 1 to logic 0, there needs to be sufficient time allocated for the clock and modulus signals to propagate down then back up the chain of active divide by 2 or 3 stages. The timing becomes increasing complicated when M is set to $2^k - 1$, since toggling between M and $M + 1$ would involve changing the active number of stages while the clock and modulus signals are propagating up and down the chain of divide by 2 or 3 stages. This can be potentially problematic; observable

in the MMD behaviour of Figure 34 when *mmd_integer_divisor* is $2^k - 1$. However, for this design, the required divisors do not cross the $2^k - 1$ to 2^k boundary when toggled by the SDM, and thus the number of active stages used during operation is static. Therefore, this complication was not of concern, and this MMD architecture was deemed sufficient for this application so long as its programmed divisor could be provided sufficient setup and hold time.

Feedback Timing

To allow the average value of the output period follow the derived expression from (3.2), R must be phase coherent with MMD output period, and the output period must respond correctly to the programmed value of R. This means that the rate and phase of R must be set carefully with respect to the MMD output to not cause a setup or hold violation while remaining coherent. To prove the timing works between the blocks within the loop, the parasitic resistances and capacitances of each core CMOS block were extracted and simulated to produce the waveforms shown in Figure 35. For the waveforms shown, the model parameters for the simulation were set to the SS process corner, nominal supply voltage, and temperature of 125 °C.

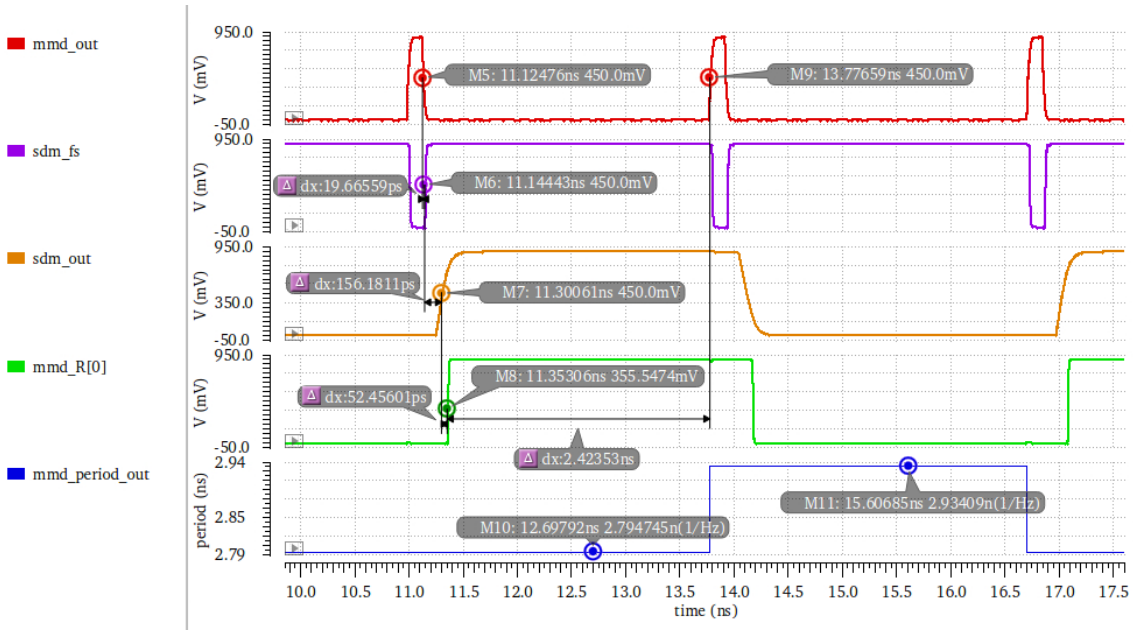

Figure 35: Timing diagram of the subcircuitry within the sigma-delta divider loop

As mentioned previously, an inverted version of the MMD output is used to clock

the SDM. As the divisor of the MMD is modulated, having MMD output period change at the same rate as R allows the two signals to maintain phase coherency. Shown in Figure 35, inverting the MMD output clock *mmd_out* to generate sample clock *sdm_fs* forces the rising-edge triggered SDM to output a new sample at *sdm_out* after pulse output from the MMD has already fallen back low. This forces the LSB of R, *mmd_R[0]*, to not have an effect on *mmd_out* when Mod_{in} of the first divide by 2 or 3 stage is high until the next cycle. This is observable by comparing the output period changing in time, *mmd_period_out*, to *mmd_R[0]*. The output period, measured between consecutive rising edges, does not respond to a change in R until one pulse width prior to the end of the sample. This occurs because the Mod_{out} pulse of the first divide by 2 or 3 stage, corresponding to *mmd_out*, interacts with *mmd_R[0]* at the end of its sample. This interaction causes a ripple effect down and up the chain of divide by 2 or 3 stages and the result is not observed until the end of the next cycle. This provides ample setup time for the Kogge-Stone adder to sum M and *sdm_out* and ensure each resulting bit of R has arrived to their corresponding divide by 2 or 3 stage within the MMD. The propagation delay of the feedback loop itself provides the hold time to ensure *mmd_out* has fully discharged prior to R proceeding to the subsequent sample. The setup and hold times were also evaluated at the FF process corner at 25 °C, with no timing violations observed.

Time-Averaged Frequency

While it is important to simulate the extracted layout, it was quickly realized that verifying that the time-averaged frequency is correct with such accurate models was not feasible. In general, Post-Layout Simulation (PLS)s can be time consuming when simulating nested feedback loops. This is particularly true for a simulation to verify the time-averaged frequency is correct, since the simulation length would need to be greater than 2^{16} MMD output periods plus some time for the SDM accumulators to fill up after their resets are deasserted. For perspective, $2^{16}/156.25\,\text{MHz}$ corresponds to more than 400 µs of simulation time. It would take several weeks if not months to simulate across corners to account for process, temperature, and voltage variations. To circumvent this, it was opted to verify the time-averaged frequency using AMS with Verilog code for each of the core CMOS blocks within the sigma-delta divider. Then, the extracted layout could be simulated over corners for a subset of output periods and contrasted with the AMS results to confirm they match.

Since the core blocks of the sigma-delta divider were CMOS blocks that were placed and routed, their corresponding Verilog codes were available. CML buffers do not swing from rail to rail, thus including them would slow down the simulation since the simulator would need to include real/analog values rather than having clock signals that simply toggle between logic 0 and 1. These blocks were substituted with Verilog assign statements to function as logical buffers since the amplitude level was not important in capturing the core behaviour of this circuit. One of the data clock PLL desired tunable frequencies was 13.9762 GHz, corresponding to an $\overline{f_{ref}}$ of 349.406 211 MHz. To test the baseline model, f_{in} was set to 7.156 808 GHz, M was set to 20, and FCW was set to 31639. To satisfy (3.2), the output period was expected to be 2.9342 ns for 31639 periods and 2.7945 ns for 33897 periods on average over 2^{16} periods. The output period was confirmed to toggle between these two values in a sigma-delta modulated way, as shown in the snapshot of Figure 36. Taking the reciprocal of the average period within a $2^{16}/349.406\,211$ MHz time window, the average period was calculated to be 349.406 345 MHz. This corresponds to an error of 0.385 PPM. The simulated average frequency is within the theoretical PPM/LSB, which was calculated to be 0.745 PPM using (3.6). This simulation was reproduced with transistor level models and yielded similar results.

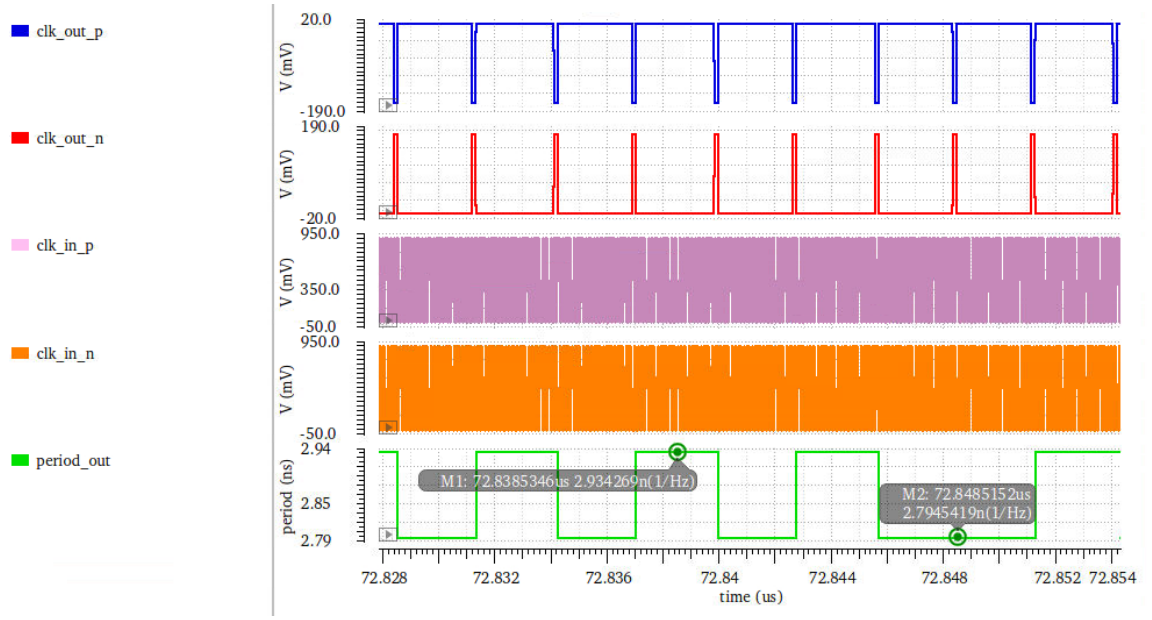

Figure 36: Snapshot of the AMS simulation of the Verilog models of the sigma-delta divider system

The goal of the AMS simulation was to provide a baseline of the circuit behaviour and a method to compare the baseline to the extracted layout. It was reasoned that if the output period was observed to be modulated between the two expected output periods by the SDM, according to the timing demonstrated in the previous section, in both the Verilog and extracted layout systems, the SDM output of both systems could be contrasted to confirm the behaviour matches over time. Despite the output of the SDM being pseudorandom, it has a consistent signature depending on the setting of the FCW and the accumulator initial conditions. Since the layout is just a synthesized, placed, and routed version of the Verilog code, the same signature can be observed in both the layout extraction and Verilog model so long as they provide the same initial conditions.

A testbench was constructed to perform the comparison between the extracted layout and Verilog model of the sigma-delta divider system. The input clock, FCW, and reset signals were common to both models, and an assertion statement was placed into the Verilog code to output to the simulation log if the SDM samples do not match at any point in time. To account for the propagation delays that exist in the extracted layout model that would not exist in the Verilog without having to model them, the rising edge of each model's MMD output was used to sample their corresponding SDM outputs, whereas the extracted layout MMD was also used to trigger the comparison. The extracted layout was confirmed to match the Verilog for more than 461 consecutive samples after the SDM was reset. Applying the same assumption that quantizer error can be approximated as white-Gaussian noise, the samples were assumed to be independent of one another. Since each sample either matches the expected value or does not match and all of the 461 samples match, the bitstream can be said to agree with the expected with 99% confidence and 99% reliability according to a binomial distribution. This process was repeated for all desired frequencies and all sigma-delta divider variants.

Power Consumption

While the power consumption of the sigma-delta divider system was not a key consideration of this work from an academic standpoint, it is worth mentioning it since having high power consumption would make it debatable to use this system instead of an off-the-shelf NCO. The sigma-delta divider system was taped out amongst other circuits on a test chip, and shares a common voltage supply with other circuitry.

Unfortunately, the ability to power down the sigma-delta divider was not included in the design, and the ability to power down everything but the sigma-delta divider was not an option either. The average power consumption was simulated to be around 28 mW. This was determined by obtaining two values for the average current from simulation, one for each of the two possible output periods, and multiplying the result by the voltage supply. The result of these can be time averaged based on the FCW to produce an estimate for the power consumption of the sigma-delta divider for a given input. However, as anticipated, the CML circuitry dominated the power consumption, and the delta in power consumption between the two periods was less than 10 µW. Therefore, regardless of the programmed input, it is sufficient to say either of the sigma-delta divider systems depicted in Figures 29 and 30 consume around 28 mW each.

4.1.2 Sigma-Delta Divider in the Reference Path of the PLL

Two simulations of sigma-delta dividers in the reference path of the data clock PLL are detailed. Since the sigma-delta divider system and its corresponding blocks were expected to operate as designed according to the simulations presented, it was important to confirm the theory that a sigma-delta divider in the reference path of an integer-N PLL behaves as a fractional-N PLL in simulation. Following the presentation of these simulation results, the results of simulating linear models of the phase noise transfer through PLLs in the presence of non-linearities will be detailed to confirm the design decision of using a single-bit quantizer SDM.

Modeling of Fractional-N Locking

Since the design of the PLL which the sigma-delta divider is generating the reference clock for was not designed in the same technology process, a model was used to confirm fractional-N locking behaviour. The structure depicted in Figure 20 was constructed using modeled blocks. To model the VCO, the Verilog-A model aptly named *vco* was used from the *ahdlLib* library. To model the divide by 2 frequency divider blocks, the cell *d_ff* was used from the same library. For the charge pump and loop filter, the cells *vccs*, *res*, *cap*, and *gnd* were used from the *analogLib* library. Finally, the PFD and divide by 40 models were realized using simple custom Verilog-A models. Again, the parameters provided to these blocks were redacted to preserve the confidentiality of the PLL design. The sigma-delta divider schematic was instantiated

to generate the clock input to the model, input with a 7.1568 GHz frequency clock, and M and FCW were programmed to 20 and 31637 respectively. This corresponds to an expected f_{out} of 13.976 254 GHz. This expected value matches within reason to the frequency the VCO settles to shown in Figure 37. Clipping the waveform to the settled region, approximated to be from 30 µs to 60 µs, and taking the average gives the result of 13.976 251 GHz. While normally the ripple of this settled region can be used to approximate the jitter, the phase noise performance of the individual PLL blocks was not provided, and this would only represent the jitter contribution from the SDM projected to output. This analysis was performed by simulating linear models using Simulink instead, available in the subsequent section.

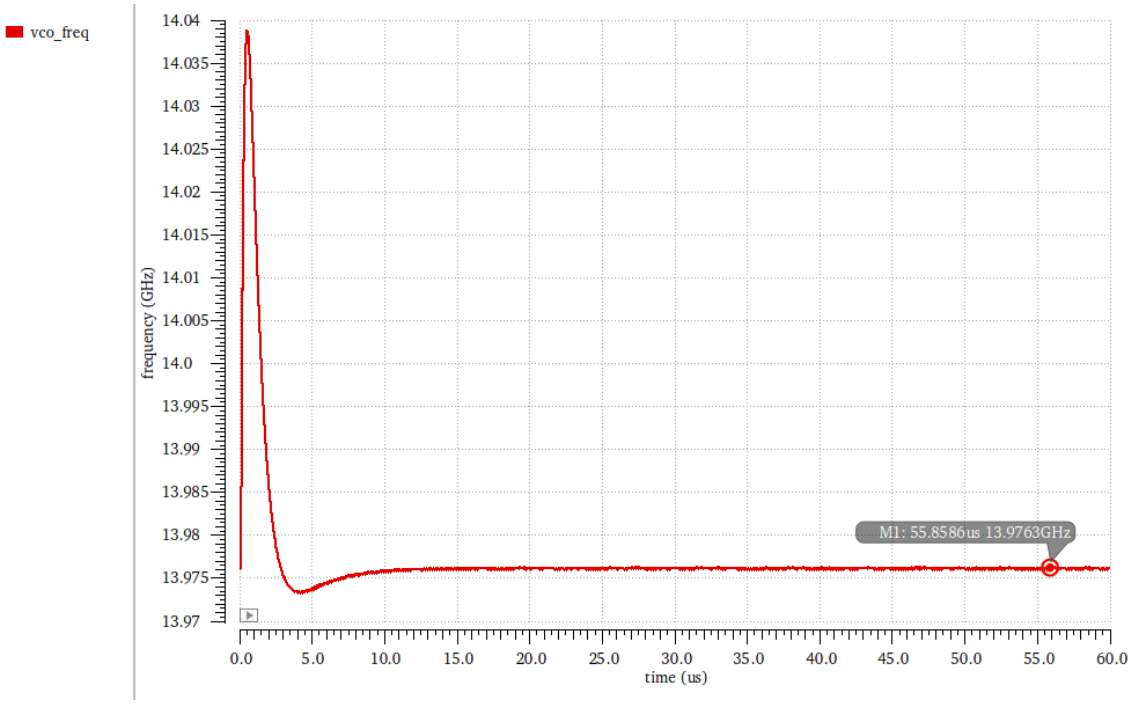

Figure 37: Frequency output of the VCO of a Verilog-A PLL model with the sigma-delta divider generating its reference clock

PFD and Charge Pump Non-Linearities Effect on Phase Noise

To confirm the design assumption of using a single-bit quantizer for the SDM to be more immune to non-linearities of the PFD and charge pump in PLL-based fractional-N synthesis, linear models for both the SDM and integer-N PLL were constructed using Simulink. The PLL model was constructed according to the linear model of the phase transfer of an integer-N PLL as shown in Figure 8 with two exceptions. First,

the phase noise profile of the VCO was not known, and $\Phi_{noise,VCO}(s)$ was effectively set to 0 by removing its corresponding adder. Second, K_{phase} was replaced with the simulated worst-case non-linear transfer characteristic that was provided for the PFD and charge pump system. This characteristic was not included to preserve the confidentiality of the PLL design, however, it resembles the sketch shown in Figure 19 and is typically attributed to the mismatch between "up" and "down" currents within the charge pump. The implemented CIFB structure from Figure 28 was constructed along with a MASH-111 structure with a 3-bit output for comparison, available in [4]. Each of these structures were input with an FCW of 32768 and placed within the linear system depicted in Figure 18 to produce a model of the phase noise contribution from each SDM. Applying the phase noise output for each SDM, $\Phi_{\Sigma\Delta}(s)$, to the reference input to the linear model of the PLL, $\Phi_{noise\_in}(s)$, the simulated phase noise contributions of the CIFB and MASH-111 structures shaped by the described PLL model are shown in Figure 38 in yellow and blue respectively.

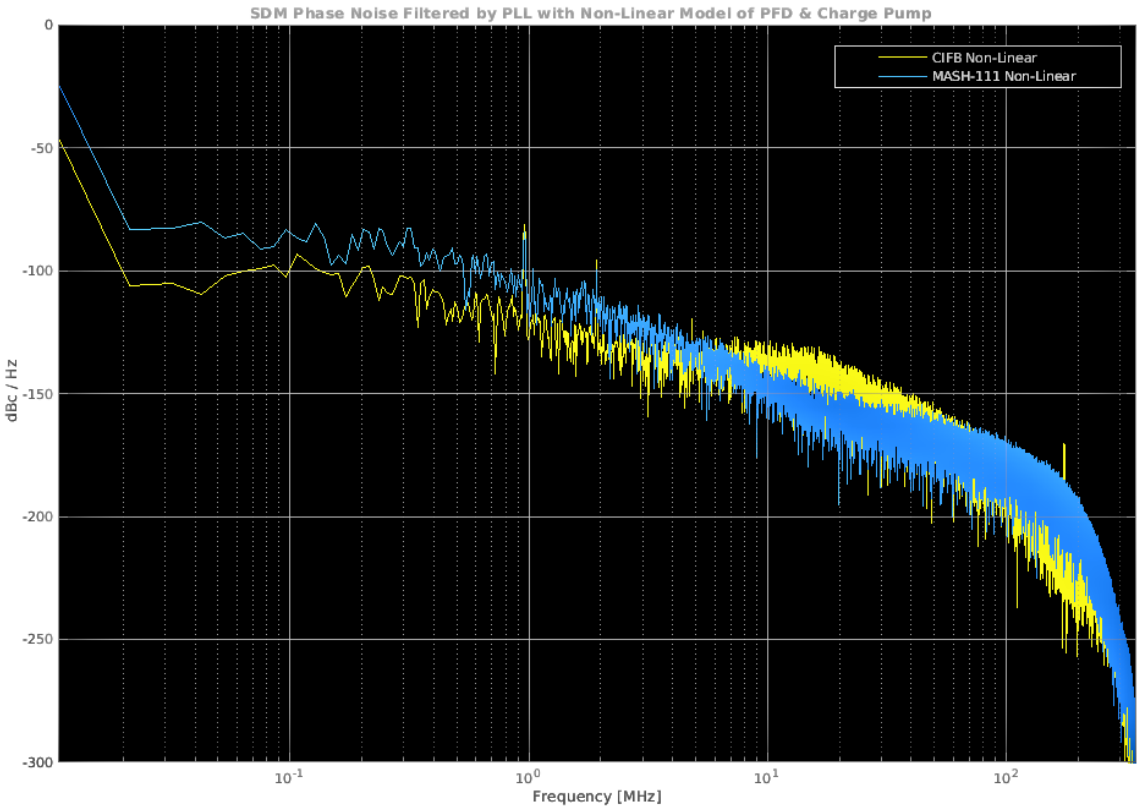

Figure 38: Comparing the phase transfer of a PLL with PFD and charge pump non-linearities input with the phase noise contributions of a sigma-delta divider containing the third-order single-bit quantizer CIFB SDM (yellow) versus a MASH-111 (blue)

When comparing the phase noise profiles shown in Figure 38, it is evident that the in-band phase noise is significantly higher when the MASH-111 is presented with the non-linearity than when the CIFB is. Since the MASH-111 has a 3-bit output and the implemented CIFB has only a single-bit output, the peak-to-peak phase excursion of the MASH-111 is larger and experiences more of the non-linearity. With this worst-case non-linearity characteristic, there was observed to be nearly a 20 dB delta in phase noise at a 100 kHz offset frequency. Replacing K_{phase} with linear transfer characteristic in both PLL models produces the phase noise profiles shown in Figure 39. When comparing the yellow and blue waveforms, again corresponding to the linear phase transfer of the phase noise contributions of the CIFB and MASH-111 respectively, it can be seen that the MASH-111 actually provides better in-band noise performance. This is because the multibit output allows for the NTF to have a higher OBG, and allows for a higher SQNR according to (2.12). Thus, if the PLL

could have been linearized sufficiently, indicating the MASH-111 could have been a superior choice of SDM between the MASH-111 and the third-order single-bit CIFB.

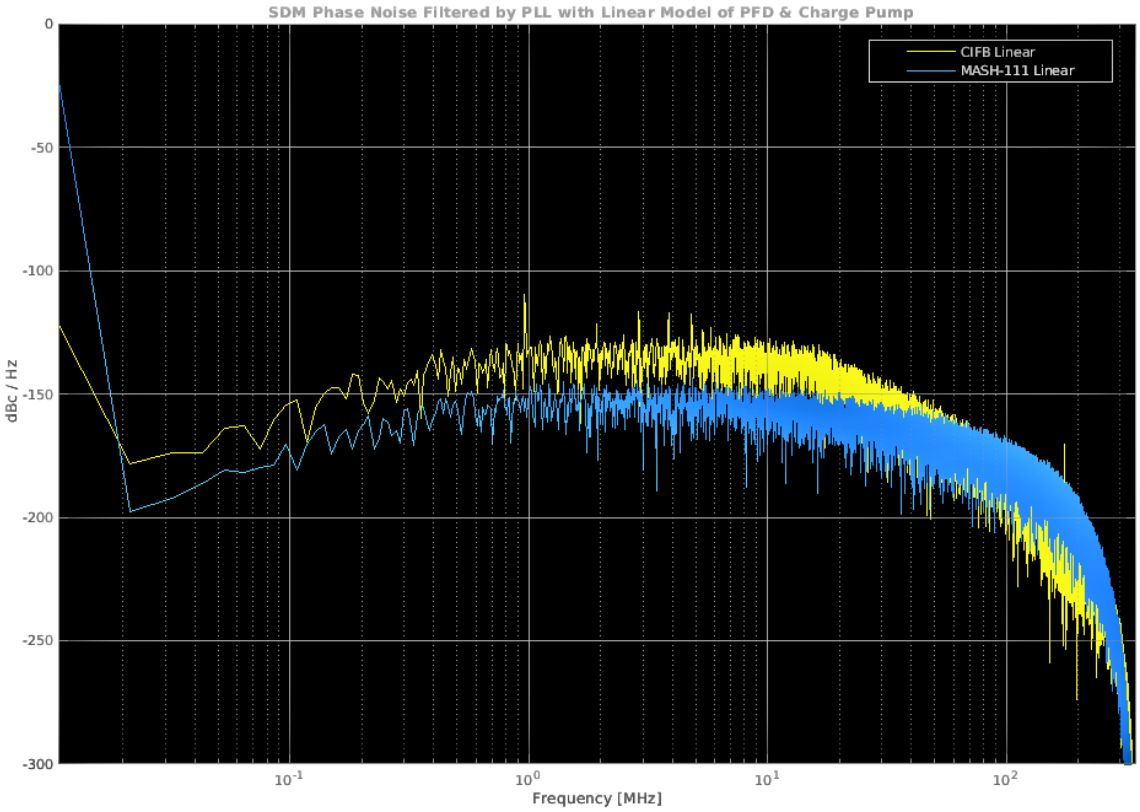

Figure 39: Comparing the phase transfer of a linear PLL input with the phase noise contributions of a sigma-delta divider containing the third-order single-bit quantizer CIFB SDM (yellow) versus a MASH-111 (blue)

4.2 Measurements

Once the chip was fabricated and assembled, measurements were performed to test the sigma-delta divider system. The sigma-delta divider was first tested on its own, and once it was confirmed to be operational, the data clock PLL was introduced. All measurements shown had the 7.1568 GHz clock sourced from an Anritsu signal generator, as the low-jitter PLL intended to provide the input clock was not yet configured to supply it at the time of these measurements.

4.2.1 Sigma-Delta Divider

The spectrum analyzer and phase noise analyzer measurements of the sigma-delta divider will be presented herein. First, the frequency spectrum will be presented to prove the average frequency is correct and confirm operation. Next, the phase noise will be shown and discussed. Finally, the methodology of an experiment to estimate the MSA of SDM and its results will be shown.

Frequency Spectrum

The positive output of the CML driver, corresponding to the sigma-delta divider circuitry for the data clock PLL reference generation, was AC coupled and connected to a spectrum analyzer as shown in Figure 40. M and FCW were programmed to be 20 and 29360, corresponding to an average output frequency of about 350 MHz. When observing the frequency spectrum, a clear tone of the average frequency is observed. This is a similar phenomenon of being able to observe a tone at the frequency of the sinusoid input to the SDM while testing for the SQNR, despite the time-domain output toggling at a much higher rate. In both cases, when observing the frequency spectrum in band around the signal of interest, a tone at the average frequency is visible. While the order of the noise shaping is not immediately obvious from this measurement, since the x-axis is not plotted on log-scale, it is evident that the frequency noise shaping property of the SDM is present. The slope of the noise shaping will be observed in the phase noise measurement of this circuit in the subsequent section.

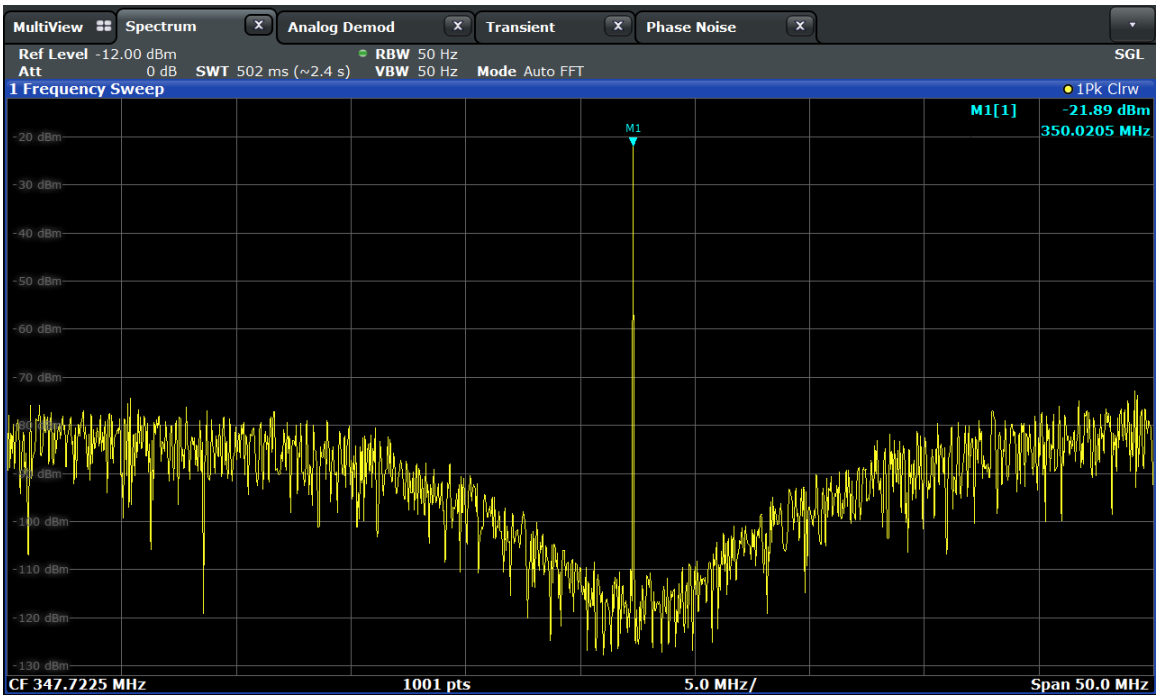

Figure 40: Frequency spectrum of the data clock variant of the sigma-delta divider with M set to 20 and FCW set to 29360

The other variant of the sigma-delta divider for generating the reference frequency for the communication clock PLL was also tested. Unfortunately, due to the restricted access to the lab, Figure 41 is the only measurement the author has that proves that this variant of the circuit works. By programming M to 45 and FCW to 52662, the average frequency was measured to be at the expected value of 156.25 MHz.

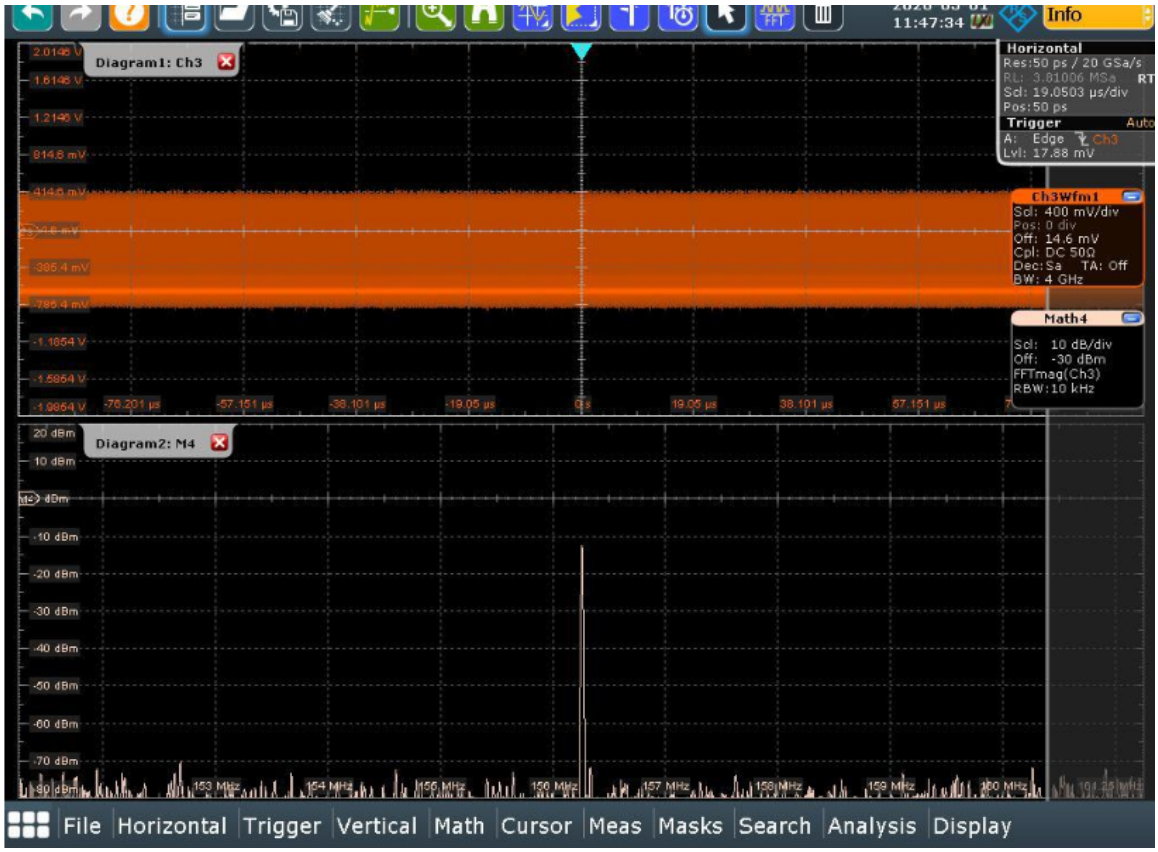

Figure 41: Frequency spectrum of the communcation clock variant of the sigma-delta divider with M set to 45 and FCW set to 52662

Phase Noise

Using the same settings for M and FCW used for taking the measurement depicted in Figure 40, the phase noise was measured as shown in green in Figure 42. For reference, the phase noise profile for the input clock, shown in yellow, was offset by 26 dB to roughly compensate for the divide ratio. The SDM was also held in reset, observable in blue, to show the phase noise profile for a fixed divide by 20.

Since the OSR of the SDM was designed to be 160 and the average frequency output of the sigma-delta divider in this measurement is 350 MHz, the bandwidth of the SDM in this measurement can be calculated to be 1.093 75 MHz by rearranging (2.2). When comparing the phase noise profile of the SDM being active and inactive, plotted in green and blue, respectively, in Figure 42, the phase noise shaping can be observed to begin ramping around 1 MHz, agreeing with the theoretical bandwidth. The shaped phase noise has a slope of 40 dB/decade, aligning with the expected phase

noise shaping for a third-order SDM.

The signal frequency of the phase noise measurement with the SDM active was observed to be 350.000 033 MHz in Figure 42 using the auto frequency feature on the phase noise analyzer. Substituting $f_{in} = 7.1568\,\text{GHz}$, $M = 20$, and $FCW = 29360$ into (3.3) gives exactly this result, to the precision of the instrument. The PPM error of this measured frequency compared to the desired average frequency of 350 MHz is 0.096 PPM, and well within the expected maximum of 0.746 PPM/LSB, calculated using (3.6).

While there are several spurious tones seen in the phase noise profile of the sigma-delta divider, they also appear in the phase noise of the input clock, and nearly all of them appear outside the bandwidth of the PLL. These spurious tones will be further discussed when measuring the phase noise at the output of the PLL.

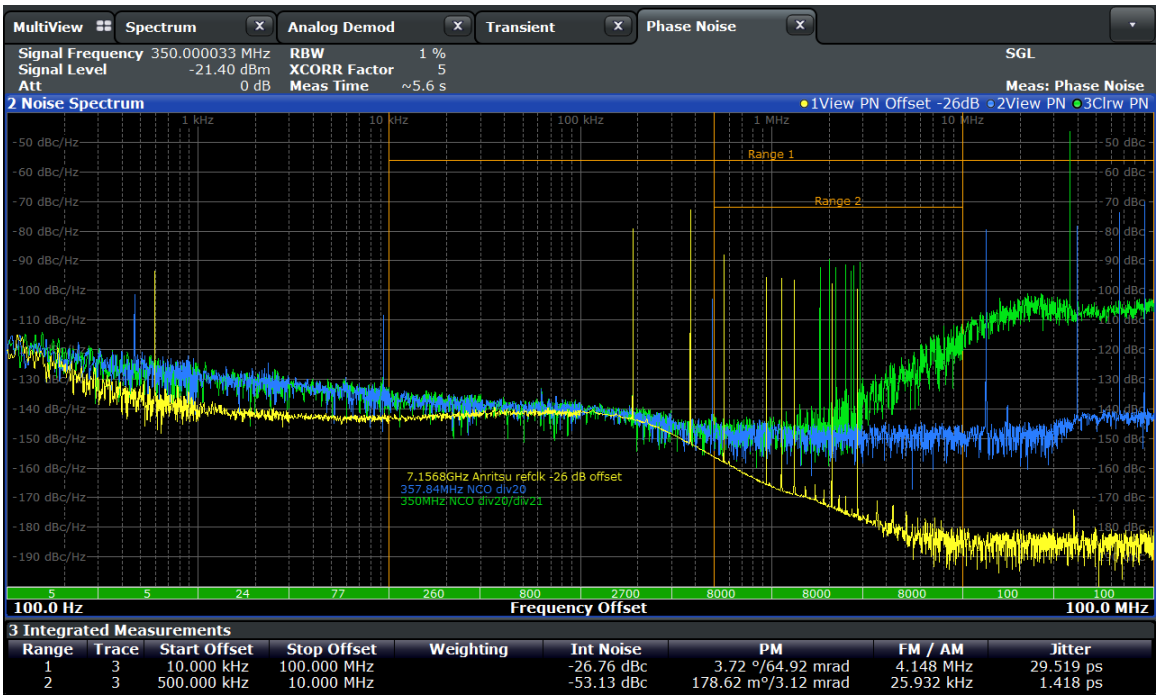

Figure 42: Phase noise of the input clock (yellow), output of the sigma-delta divider (green), and output of the sigma-delta divider with the SDM held in reset (blue)

SDM MSA Approximation

While access to the SDM output itself was not possible, an experiment on the sigma-delta divider was performed to approximate the MSA of the SDM. Setting M to 20 and using the data clock variant of the sigma-delta divider, the FCW was set to 32678

and swept up, then down, in steps of 1. The assumption was that once the frequency shaping was no longer observed in the spectrum, the corresponding FCW had set the SDM input amplitude beyond its MSA. Thus, by performing this experiment, the difference between the upper and lower limits of the FCW where the frequency noise shaping effect was observed could be divided by the FSR to approximate the MSA.

Figure 43 corresponds to FCW set to 55925 while Figure 44 has the FCW set to 55926. Similarly, the FCW is set to 7864 in Figure 45 and 7863 in Figure 46. Taking the difference between the limits of the FCW working properly and dividing the result by 2^{16} yields an approximate MSA measurement of 0.733 of the FSR. This agrees within reason to the approximated theoretical and simulated value of $-3\,\text{dB}$ or $1/\sqrt{2}$ with respect to the FSR. It should be noted that $-3\,\text{dB}$ was known to be an approximation, as this amplitude is slightly less than the amplitude corresponding to the SQNR $3\,\text{dB}$ lower than the dynamic range, observable in Figure 26, which is the true definition of the MSA. However, approximating the MSA to be $-3\,\text{dB}$ was convenient to confirm in simulation, shown in Figure 33. Also, as previously mentioned, it is not recommended to operate the SDM near the edge of its working range. This measurement confirms the working range of the SDM is as designed, and that it has more than $3\,\text{dB}$ of margin on the MSA specification of half the FSR.

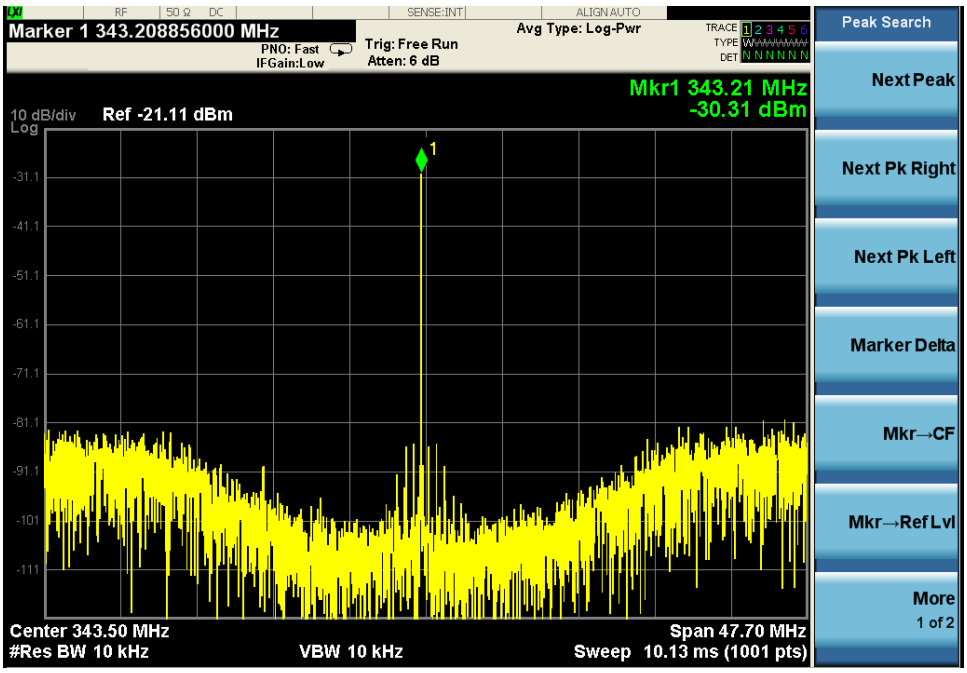

Figure 43: Frequency spectrum of the sigma-delta divider with *FCW* set to its highest operational value of 55925

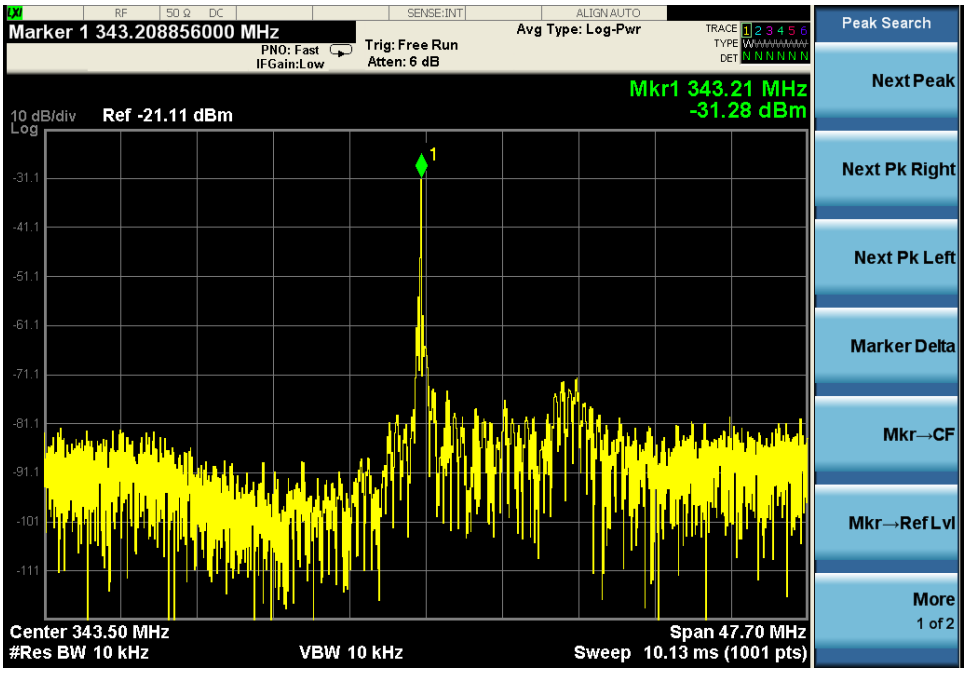

Figure 44: Frequency spectrum of the sigma-delta divider with FCW set 1 LSB higher than its highest operational value

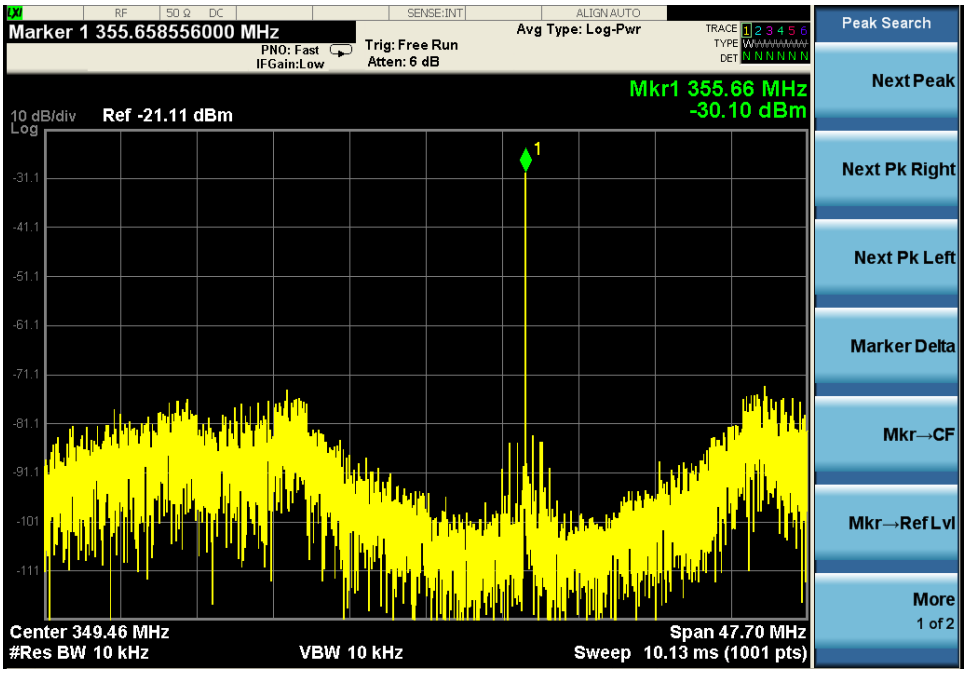

Figure 45: Frequency spectrum of the sigma-delta divider with FCW set to its lowest operational value of 7864

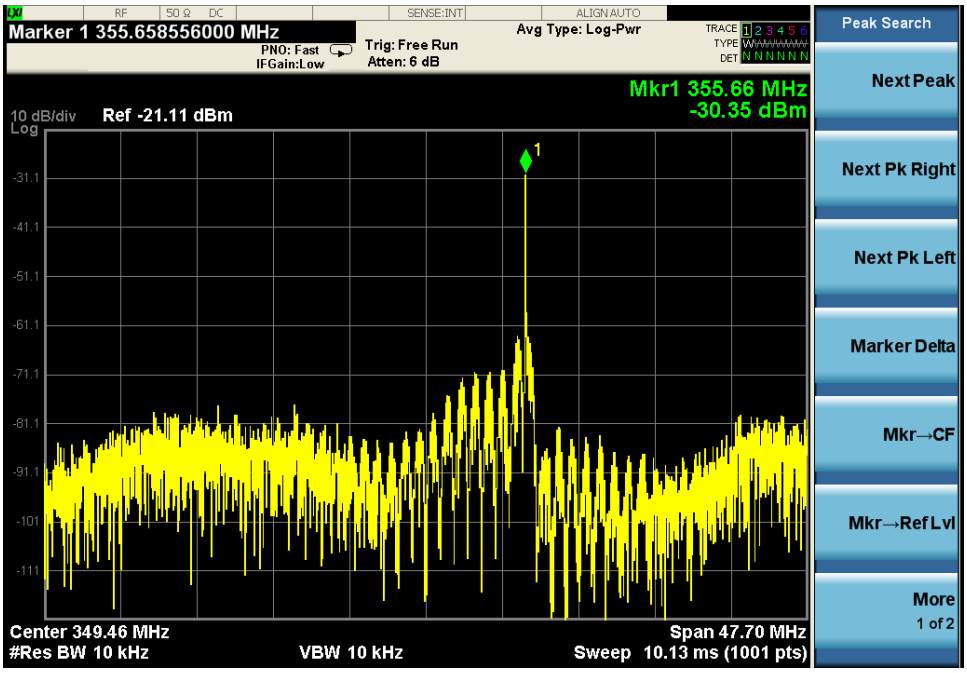

Figure 46: Frequency spectrum of the sigma-delta divider with FCW set 1 LSB lower than its lowest operational value

4.2.2 Sigma-Delta Divider in the Reference Path of the PLL

Measurements demonstrating the fractional-N locking of the sigma-delta divider in the reference path of an integer-N PLL are shown. Following this, the phase noise measurement will be analyzed and discussed, comparing it with theory and system specifications.

Demonstrating Fractional-N Locking

Measurements of the frequency spectrum of the data clock PLL driven by the sigma-delta divider circuit are shown in Figures 47 and 48. In Figure 47, M is set to 20 and FCW is set to 29360. This corresponds to an $\overline{f_{ref}}$ of 350.000 033 MHz, which when multiplied by the PLL gives f_{out} of 14.000 001 34 GHz. The resulting peak search function on the spectrum analyzer shows a measured frequency of 14.000 001 07 GHz. This agrees with the expected value within the resolution of the measurement, where the frequency resolution was reasoned to be around 550 Hz given the 550 kHz span divided by 1001 points. Similarly, Figure 48 had M set to 20 and FCW set to 31639, with an expected $\overline{f_{ref}}$ of 349.405 818 MHz and f_{out} of 13.976 232 73 GHz. The marker

placed using the peak search function indicated a measured f_{out} of 13.976 232 77 GHz, and also agrees with the expected result within the resolution of the measurement. Since these frequencies match the expected values within the resolution of the measurement which is less than 1 PPM and the expected frequencies were designed to be within 1 PPM of their target, the frequency resolution tuning specification was met. Another thing to note was the close-in spur noticed in both of these measurements at the same offset of 210 kHz, which will be discussed further in the subsequent section.

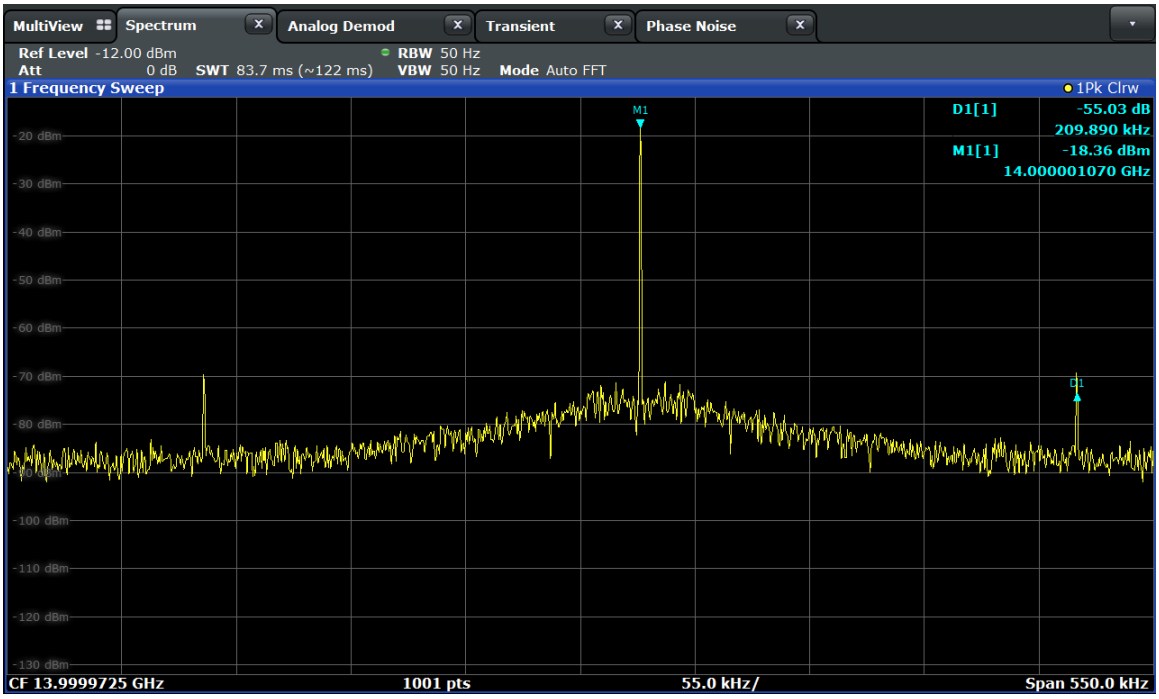

Figure 47: Frequency spectrum of the PLL output with M set to 20 and FCW set to 29360

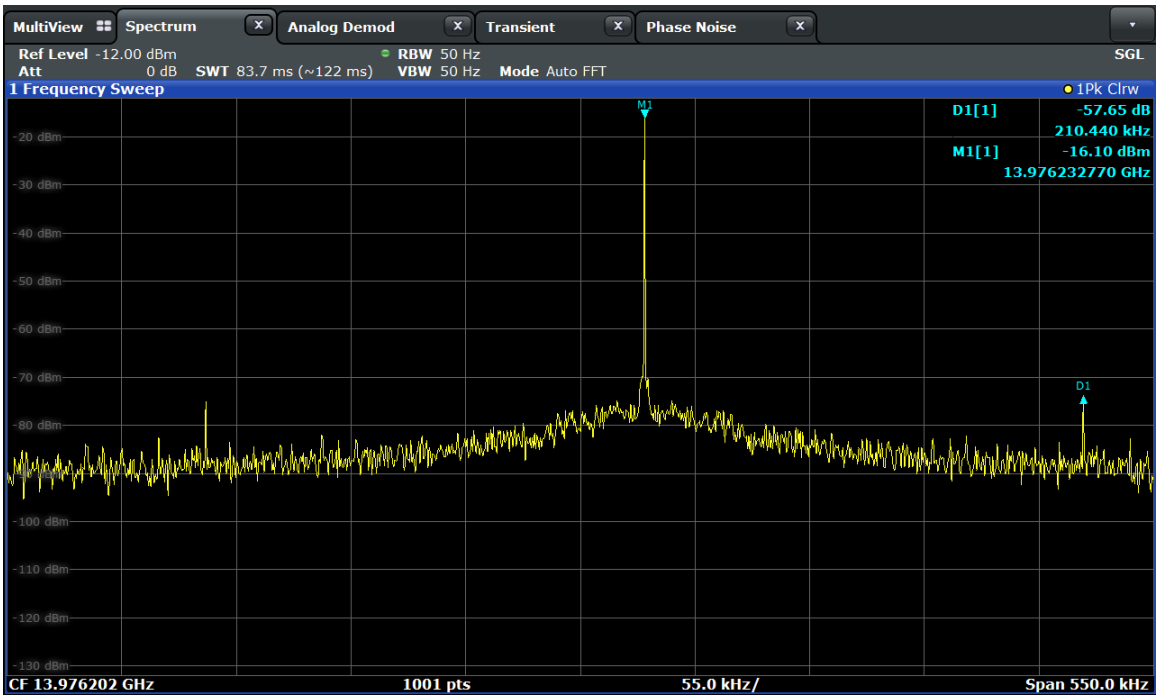

Figure 48: Frequency spectrum of the PLL output with M set to 20 and FCW set to 31639

Phase Noise and Fractional Spurs

Using the same settings to generate the frequency spectrum measurement depicted in Figure 48, the phase noise was also measured; as shown in Figure 49. Over the specified bandwidth of 500 kHz to 10 MHz, $\tau_{jitter,rms}$ was measured to be 487 fs. This was within the design specification of 500 fs. It is also greater than the estimated jitter contribution of the SDM within this integration band of 365.8 fs. Since the integration band is slightly outside the PLL bandwidth, the remaining jitter is speculated to be largely sourced from the VCO since its phase noise contribution has a high-pass phase transfer through the PLL. Over the wider integration range of 10 kHz to 10 MHz, the PLL was measured to have 816 fs of jitter with the spur at 210 kHz, and 813 fs of jitter without it. It is speculated that the source of this tone could be from a switched mode power supply given its frequency location and constant offset from the frequency tone between measurements.

Figure 49: Phase noise profile of the PLL output with M set to 20 and FCW set to 31639

While the 210 kHz tone was the only spur seen in band, two other spurs were observed out-of-band; one at 12 MHz and another at 96 MHz. With the setting used for this measurement, one would expect that a fractional spur would be observed at 84 MHz according to (2.27), however, no tone was observed at this frequency. Since this is out of band, it is uncertain whether it was simply filtered by the PLL until it was indiscernable or if these two visible tones are fractional spurs and follow a different phenomenon since the sigma-delta divider is in the reference path of the PLL. Otherwise, the origin of these two visible spurs is still unknown. Further analysis would be required to confirm whether either of these two tones could be fractional spurs.

4.3 Chapter Summary

The sigma-delta divider system and its two core blocks, the SDM and the MMD, were all simulated to confirm they operate as designed. The average power consumption was estimated. Then the sigma-delta divider was simulated with the inclusion of the integer-N divider to confirm fractional-N operation and the choice to use a single-bit

quantizer SDM design in the prescence of non-linearities. Finally, the sigma-delta divider was measured with and without the integer-N PLL, and its characteristics and performance were discussed and compared to the design theory and requirements. Conclusions and future work to be performed will be discussed in the following chapter.

Chapter 5

Conclusion and Future Work

5.1 Summary

Fractional-N PLL based synthesis provides a method for non-integer multiplication of a reference clock frequency by averaging the modulation of two integer multipliers. A convenient method for allowing an integer-N PLL to achieve fractional-N synthesis by sigma-delta modulating the reference clock frequency was proposed, which does not require modification to the integer-N PLL design. The objective of this work was to demonstrate that a sigma-delta divider could be designed and implemented in TSMC's 7 nm FinFET technology capable of generating the reference clock for a separately packaged integer-N PLL such that fractional-N synthesis could be demonstrated. To be acceptable for one of the design applications, the sigma-delta divider was required to provide less than 1 PPM/LSB of tuning resolution and less than 500 fs of RMS jitter when integrated from 500 kHz to 10 MHz.

Some background theory relevant to understanding the design considerations and performance metrics were presented. The concept of phase coherency was introduced. Some sigma-delta converter theory was presented, including OSR, STF and NTF, noise shaping and its benefits, SQNR, ENOB, MSA, and dynamic range. Two architectures of MMDs were introduced, along with their main sub-structure: the dual-modulus divide by 2 or 3. Finally, some important theory for both integer-N and fractional-N PLLs were covered; including phase transfer functions, loop dynamics and bandwidth, third-order loop filters, RMS jitter, and fractional spurs.

To relate the theory in the background chapter to the concept sigma-delta modulating the reference clock of an integer-N PLL, analogies were made to sigma-delta converters and conventional fractional-N PLLs. This led to the derivation of the

equation for calculating the average frequency output of the PLL when the sigma-delta divider generated its reference clock. Important analyses were conducted for frequency resolution and phase noise. It was ultimately discovered that a higher frequency input clock allows for a higher reference divisor M to be used, which leads to increased frequency resolution and reduced in-band phase noise.

Several design decisions were considered to meet these design specifications. The number of input bits of the SDM was set to 16 such that the sigma-delta divider frequency resolution could be less than 1 PPM/LSB. This allowed an FCW to be calculated that could tune the frequency within 1 PPM of the desired value. Since only the sigma-delta divider was designed here and the bandwidth of the PLL was fixed, the only option was to make design considerations that kept the jitter contribution from the sigma-delta divider as low as possible. A single-bit quantizer SDM was opted for to be more immune to non-linearities within the PLL. A high OSR should be selected for the SDM to allow low in-band quantization noise, but kept lower than OSR of the PLL to ensure the shaped quantization noise began outside the PLL loop bandwidth. The OSR and loop bandwidth of the PLL were approximated to be about 291 and 300 kHz, respectively. The order of the SDM was selected to be equal to the order of the PLL loop filter to allow the shaped quantization noise to be attenuated. The MSA of the SDM was desired to be greater than half the FSR. Given the single-bit quantizer, this would allow the programmable values of FCW within the MSA to cover a continuous range of tunable frequencies if the divider step size was increased from 1 to 2. The divider step size of 1 should be maintained if this additional tunable range is not required, since it also increases the PPM/LSB by a factor of 2. Since the desired reference frequencies for the data clock PLL and communication clock PLL tested required fractional divisors of 20.429 and 45.833, respectively, a divider step size of 2 was used for the latter and 1 for the prior.

The sigma-delta divider system implemented consisting of three main blocks: the SDM, the MMD, and full adder. The implemented SDM architecture selected for this design was a single-bit quantizer CIFB. The OSR of the NTF was 160, within the specified range. The coefficients of the structure were quantized to be easily implemented using binary shift operations and scaled to mitigate analytically determined quantization noise. The final structure had a modeled dynamic range of 106.9 dB, corresponding to an ENOB greater than the required 16 bit resolution, and an MSA of greater than $1/\sqrt{2}$, which is greater than the specified value of 0.5. The implemented

MMD used in this design could be programmed from 2 to 511, easily covering the required range of values for the reference divisor M. The full-adder architecture used was an available Kogge-Stone adder. Additional CML driver and conversion circuitry were included to route the input clock to the MMD and also drive the output bumps of the chip.

To verify the structure would work prior to its fabrication, confirm design considerations, and show performance metrics that were not plausible in measurement, several simulations were conducted. The dynamic range and ENOB of the SDM were simulated to be 106.0 dB and 17.3 bits, greater than the required 16 bits. The MSA was found to be about the expected value of $1/\sqrt{2}$. The MMD was found to glitch when M was set to $2^n - 1$ for a divider step size of 1, which not a required value of M for this design. The timing of the sigma-delta divider loop was confirmed to be free of timing violations with the inclusion of a CMOS inverter in the feedback path. The average frequency was simulated to be correct in Verilog of modeling of the design with 0.385 PPM, was reproduced at the schematic level, then extrapolated to work for the extracted layout. The average power consumption of the implemented sigma-delta divider system was simulated to be about 28 mW. The integer-N PLL was modeled using Verilog-A and simulated with the sigma-delta divider in its reference path, with the VCO frequency measured to be within reason of the expected output frequency. A MASH-111 SDM was modeled to contribute more in-band phase noise through the non-linear transfer of the PFD and charge pump of the PLL than the single-bit CIFB implemented, clarifying the reasoning of selecting a single-bit quantizer.

The sigma-delta divider circuit was fabricated in TSMC's 7 nm process and measured both independently and with the data clock PLL. A phase noise analyzer was used to measure the average frequency output from the sigma-delta divider to be 0.096 PPM from the expected frequency. The spectrum analyzer feature was used for the sole measurement of the sigma-delta divider variant with a divider step size of 2, which was also seen to have the correct average frequency. The phase noise profile of sigma-delta divider was seen to be 40 dB/decade of phase noise shaping starting at approximately at the expected bandwidth, confirming the order and OSR of the SDM was as expected. An experiment to approximate the SDM MSA was conducted, resulting in a value greater than $1/\sqrt{2}$ as expected. The PLL output was measured with the sigma-delta divider in its reference path. The frequency spectra for two different values of FCW were confirmed to match the expected frequencies

within the resolution of the measurement. Using the phase noise profile at one of these frequencies, the RMS jitter $\tau_{jitter,rms}$ was integrated from 500 kHz to 10 MHz, with a result of 487 fs; less than the required value of 500 fs.

5.2 Contributions

The contributions to the art covered in this thesis include the design, analysis, and demonstration of using a sigma-delta divider to modulate the reference clock frequency of an integer-N PLL to effectively implement fractional-N synthesis. To the author's knowledge, this is a novel method of fractional-N synthesis and protection for it has been published in the form of a patent [1].

5.3 Conclusions

Through the use of existing theory and design methodologies, it was possible to implement a sigma-delta divider system capable of modulating the reference clock frequency of an integer-N PLL to achieve fractional-N frequency synthesis. Using the design metrics as validation, the implemented design was found to match the design very closely. This proves the validity of using a sigma-delta divider as a replacement for an NCO specified to generate the reference clock frequency of the PLL. While ultimately this design was considered successful and recommended for use in the design application, it is important to highlight some advice and caveats to benefit future design applications.

It was shown that a conventional PLL-based fractional-N frequency synthesizer operates in a similar way to sigma-delta modulating the reference clock frequency of an integer-N PLL. After completing the design analysis and considering the frequencies used in this design scenario, it was realized that a fractional-N synthesizer could have been a preferable solution in a different design scenario with the same frequencies used. It should be emphasized that this was not an option for this work, since the PLL was an external off-chip design, however, in other design scenarios, one may have to choose whether to place the sigma-delta divider in the reference path or the feedback path. If this was an option and the same frequencies were required/available, a reason to use a conventional fractional-N synthesizer design instead would be because the frequency to be synthesized is less than the frequency of the available input clock.

When transferred through the PLL, the gain of the in-band phase noise contribution from the SDM is approximately the ratio of the feedback divisor of the PLL N to divisor of the sigma-delta divider M. If the sigma-delta divider is in the feedback path of the PLL, this gain is 1. If it is placed in the reference path, the phase transfer gain is N/M and would be greater than 1 for these frequencies. While one could argue that having the sigma-delta divider in the reference path allows the frequency tuning resolution achieved to be independent of N, avoiding a potential design tradeoff, a counter argument is easily made that the number of input bits in the SDM design can be increased to achieve the desired tuning resolution. Thus, when designing an entire fractional-N PLL-based synthesizer, it is recommended to only place the sigma-delta divider in the reference path if the desired frequency is lower than the available input clock frequency; making this concept well suited to low-frequency applications on a chip with other higher frequency functionality.

While designing the PLL and sigma-delta divider independently was demonstrated to be a valid option, designing the two together is recommended whenever possible in future designs. Designing the PFD and charge pump with the SDM would allow the non-linear transfer due to charge pump current mismatch to be properly mitigated in the charge pump or PFD design instead of placing restrictions on the SDM. This could allow a higher number of quantizer bits to used without concern, and therefore the OSR of the SDM could be truly maximized for its given order. Then, the PLL loop bandwidth could be sized to optimize the phase noise while including the SDM, rather than limiting the OSR of the SDM to meet the PLL bandwidth requirements. Finally, designing them together would eliminate the need for the output drivers between the sigma-delta divider and PLL; significantly lowering the overall power consumption.

The layout for the MMD used in this design was already available, lowering the design effort, and the implemented design successfully generated the input clock signal for the PLL. However, the architecture that was used did place restrictions on the design that could have been avoided with a different MMD design. As observed in simulation, the "truly modular" MMD architecture is not suited for having its programmable input quickly toggled across the boundary of $2^k - 1$ and 2^k, where k is the active number of stages for a programmed value of $2^k - 1$. Furthermore, the narrow duty cycle of the output requires the routing to the PLL to support a much higher bandwidth than if the duty cycle was closer to 50%. Two custom generic MMD

designs could have been used instead, which would have less tuning range than the design that was used, but circumvents the possibility of having the number of active stages being modulated by the SDM and lowers the bandwidth requirements of the routing. Future design iterations should consider placing variants of the generic MMD architecture or instances of a custom MMD design that mitigates these issues.

5.4 Future Work

There is additional work in the area of the research topic of this thesis that was not completed. Several of these activities will be introduced here; to be potentially completed by the author or others.

When using SDMs for fractional-N PLL-based synthesis, it is important that the fractional spurs are out of band such that they are sufficiently attenuated. While it was confirmed that the fractional spurs were out of band for a conventional fractional-N PLL design, it was never properly confirmed that the frequencies at which fractional spurs occur follow the same phenomenon if the sigma-delta divider is moved from the feedback path to the reference path. Further research of why fractional spurs occur at the frequencies at which they are observed for conventional fractional-N PLLs and an analysis of whether this applies when the sigma-delta divider is instead in the reference path should be conducted.

The average frequency of the sigma-delta divider variant with a divider step size of 2 was measured to be the expected value using a spectrum analyzer, however, it has not yet been confirmed to tune its corresponding PLL to the desired frequency through measurement. While it is expected that the PLL to lock without issue, this activity should be completed to be certain.

There are several analyses related to the SDMs and PLLs that were not explored in this thesis that could be covered for the application of using a sigma-delta divider in the reference path of the PLL. A comparison could be made for the performance of different SDM architectures within the sigma-delta divider. The presence of limit cycles within the SDM and their effect on the average frequency could be explored. The functionality of placing the sigma-delta divider in the reference path of a digital PLL, or different PLL architecture than the analog charge-pump PLL covered in this work, could be verified. A large percentage of the plethora of analyses and innovations that have been uncovered for conventional fractional-N PLL design over

the years since its inception could be applied to sigma-delta modulating the reference path of an integer-N PLL.

List of References

[1] S. Aouini, M. Mikkelsen, N. Ben-Hamida, M. Parvizi, T. Wen, and C. Plett, "Fractional frequency synthesis by sigma-delta modulating frequency of a reference clock," US2019056029, 2020.

[2] S. Aouini, *Extending Test Signal Generation Using Sigma-Delta Encoding Beyond the Voltage/Amplitude Domain*. PhD thesis, McGill University, 2011.

[3] H. Nyquist, "Certain Topics in Telegraph Transmission Theory," *Transactions of the American Institute of Electrical Engineers*, vol. 47, pp. 617–644, 1928.

[4] J. Rogers, C. Plett, and F. Dai, *Integrated Circuit Design for High-Speed Frequency Synthesis*. Massachusetts: Artech House, 2006.

[5] S. Norsworthy, R. Schreier, and G. Temes, *Delta-Sigma Data Converters: Theory, Design, and Simulation*. New York: IEEE Press, 1996.

[6] J. A. Cherry and W. M. Snelgrove, *Continuous-Time Delta-Sigma Modulators for High-Speed A/D Conversion*. Dordrecht: Kluwer Academic Publisher, 1999.

[7] C. Vaucher, I. Ferencic, M. Locher, S. Sedvallson, U. Voegeli, and Z. Wang, "A Family of Low-Power Truly Modular Programmable Dividers in Standard 0.35-μm CMOS Technology," *Journal of Solid-State Circuits*, vol. 35, pp. 1028–1038, 2000.

[8] C.-L. Ti, Y.-H. Liu, and T.-H. Lin, "A 2.4-GHz Fractional-N PLL with a PFD/CP Linearization and an Improved CP Circuit," in *IEEE International Symposium on Circuits and Systems*, pp. 1728–1731, 2008.

[9] M. H. Perrott, M. D. Trott, and C. G. Sodini, "A Modeling Approach for Σ-Δ Fractional-N Frequency Synthesizers Allowing Straightforward Noise Analysis," *Journal of Solid-State Circuits*, vol. 37, pp. 1028–1038, 2002.

[10] S. Gierkink, "An 800MHz -122dBc/Hz-at-200kHz Clock Multiplier based on a Combination of PLL and Recirculating DLL," in *IEEE International Solid-State Circuits Conference*, pp. 454–455, 2008.

[11] S. Aouini, N. Ben-Hamida, and M. Parvizi, "High order hybrid phase locked loop with digital scheme for jitter suppression," US9787466B2, 2019.

[12] Q. Meng, *Low-voltage Data Converters*. PhD thesis, Oregon State University, 2007.

[13] W. Lee, "A Novel Higher Order Interpolative Modulator Topology for High Resolution Oversampling A/D Converters," Master's thesis, Massachusetts Institute of Technology, 1987.

[14] M. Horowitz, "Lecture 4 - Adders." https://web.stanford.edu/class/archive/ee/ee371/ee371.1066/lectures/lect_04.2up.pdf, 2006.

Appendix A

CIFB SDM Verilog Code

```
1  //Compensate for a1
2
3  `define sizeF 16
4  `define sizeA1 19
5  `define sizeA2 18
6  `define sizeA3 18
7
8  module CIFB_SDM(
9  input               clk, rst,
10 input [`sizeF-1:0]  FCW,
11 output              out,
12     );
13
14 wire signed [`sizeF:0] x1a;
15 wire signed [`sizeA1-3:0] x1c;
16 wire signed [`sizeA1:0] x1;
17
18 wire signed [`sizeF:0] x2a1;
19 wire signed [`sizeF:0] x2a2;
20 wire signed [`sizeA1-3:0] x2a;
21 wire signed [`sizeA2:0] x2;
22
23 wire signed [`sizeF+1:0] x3a1;
24 wire signed [`sizeA2:0] x3a;
25 wire signed [`sizeA3+1:0] x3c1;
26 wire signed [`sizeA3+3:0] x3c2;
27 wire signed [`sizeA3:0] x3;
28 wire signed [`sizeA3+4:0] x3c;
29
30 wire signed [`sizeF:0] F, fb;
31
32 reg signed [`sizeA1:0] x1z;
33 reg signed [`sizeA2:0] x2z;
34 reg signed [`sizeA3:0] x3z;
35
36 wire nor1,nor2;
37
```

```
38 always@(posedge clk or posedge rst)
39     if(rst)
40     begin
41         x1z[`sizeA1:0] <= {(`sizeA1+1){1'b0}};
42         x2z[`sizeA2:0] <= {(`sizeA2+1){1'b0}};
43         x3z[`sizeA3:0] <= {(`sizeA3+1){1'b0}};
44     end
45     else
46     begin
47         x1z[`sizeA1:0] <= x1[`sizeA1:0];
48         x2z[`sizeA2:0] <= x2[`sizeA2:0];
49         x3z[`sizeA3:0] <= x3[`sizeA3:0];
50     end
51
52 assign F = FCW;
53
54 assign x1a = F + fb;
55 assign x1 = x1z + x1a;
56 assign x1c = x1z >>> 3;
57
58 assign x2a1 = fb >>> 1;
59 assign x2a2 = x2a1 >>> 1;//reuse existing shift
60 assign x2a = x2a1 + x2a2 + x1c;
61 assign x2 = x2z + x2a;
62
63 assign x3a1 = fb <<< 1;
64 assign x3a = x3a1 + x2a2 + x2z;
65 assign x3 = x3z + x3a;
66 assign x3c1 = x3z <<< 1;
67 assign x3c2 = x3c1 << 2;
68 assign x3c = x3z + x3c1 + x3c2;
69
70 assign nor1 = ~(x3c[`sizeA3+2] || x3c[`sizeA3+3]);
71 assign nor2 = ~(nor1 || x3c[`sizeA3+4]);
72
73 assign out = nor2;
74 assign fb = {out,{(`sizeF){1'b0}}};
75
76 endmodule
```