FRACTIONAL-N SYNTHESIZER ARCHITECTURES WITH DIGITAL PHASE DETECTION

by

Mark A. Ferriss

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Electrical Engineering)
in The University of Michigan
2008

Doctoral Committee:

Professor Michael Flynn, Chair
Professor Dennis M. Sylvester
Assistant Professor Jerome P. Lynch
Assistant Professor David D. Wentzloff

ACKNOWLEDGENTS

Thanks Everyone!

Firstly, I would like to thank my advisor, Professor Michael Flynn. It has been a long journey to get here, and Professor Flynn provided just the right combination of support, prodding and technical advice to get me through my Ph.D. I would also like to thank the rest of my committee members: Professor Dennis Sylvester, Professor Jerome Lynch, and Professor David Wentzloff. They were always approachable and had useful advice.

I couldn’t have survived this process if it wasn’t for the support of all our group members, past and present who helped with both technical support, and keeping each other sane when working the many late nights in the Michigan winters. In no particular order these include Ivan Bogue, Joshua Jaeyoung Kang, Jongwoo Lee, David Lin, Junyoung Park, Andres Tamez, Chun Chieh Lee, Li Li, Shahrzad Naraghi, Jorge Pernillo, Dan Shi, Jia-Yi Chen, Sunghyun Park, and Fatih Kocer.

I’d also like to thank Intel and the WIMS Engineering Research Center for funding this research project. This work would not have been possible without their generous support.

CONTENTS

LIST OF FIGURES

LIST OF APPENDICES

ABSTRACT

FRACTIONAL-N SYNTHESIZER ARCHITECTURES WITH DIGITAL PHASE DETECTION

by

Mark A. Ferriss

Chair: Michael Flynn

During the last decade there has been unprecedented growth in the use of portable wireless communications devices for applications as diverse as medical implants, industrial inventory control, and consumer electronics. If these communication devices are to be low power, flexible, and reconfigurable, new radio architectures are needed which take advantage of the major strength of state-of-the-art digital manufacturing processes; that is the ability to build large, complex low power signal processing circuits, with extremely fast clocking speeds. However, traditional radio architectures rely on analog techniques which are ill suited for the use in modern highly integrated digital systems.

A critical component of a radio system is the frequency synthesizer, a circuit which can accurately synthesize and modulate high frequency signals. Traditional synthesizers still utilize a significant amount of analog circuitry. In this work, techniques are developed to replace this analog circuitry with digital equivalents. To do this, a digital phase detection scheme for a Fractional-N Phase Lock Loop (FPLL) is presented. The all-digital phase detector can be used as an alternative to a conventional analog-intensive phase detector, charge pump and loop filter blocks.

Another limitation of traditional synthesizers is the difficulty in modulating the frequency of the output signal at speeds larger the FPLL's bandwidth. A new technique is developed for modulating the output frequency of the FPLL at rates significantly faster than the loop bandwidth would typically allow. A digital sampling scheme that enables FSK modulation rates much larger than the loop bandwidth is demonstrated. The new scheme does not compromise on the frequency accuracy of the output signal. The key ideas presented have been proven in a proof of concept design. A prototype 2.2GHz fractional-N synthesizer, incorporating the digital phase detector and sampling scheme is presented as a proof of concept. Although the loop bandwidth is only 142kHz, an FSK modulation rate of 927.5kbs is achieved. The prototype is implemented in 0.13μm CMOS and consumes 14mW from a 1.4V supply.

CHAPTER 1

INTRODUCTION

1.1 Background

The wireless communication industry’s exponential expansion is fuelling a search for the next generation of circuit tools and techniques which will enable faster, lower power and cheaper wireless devices. The starting point of this research is to identify the limiting features of today’s technologies, and develop circuit concepts and architectures which address and overcome these short comings.

Most modern wireless communications systems are in essence frequency multiplexed systems, which consist of the simultaneous transmission of independent signals in separate frequency channels. For example in the GSM 900MHz band the channels are spaced just 200kHz apart. The ability to select a single channel from the myriad of other signals is a crucial part of the design of a modern radio system. (This is in contrast with the first three decades of radio (1885-1916) when spark-gap generators transmitted pulses of wide band energy). A crucial part of channel selection is the task of synthesizing signals with the correct frequency. As communications devices become ubiquitous, there is continuous downward pressure on the size and cost of the devices needed to modulated and demodulate radio signals. This has consequences when considering architectures of the future. Firstly, techniques are needed to ensure that the radio circuitry of tomorrow can coexist on the same substrate as the large DSP microprocessor cores. This is essential if the level of integration is to be maximized, and the costs reduced. If follows that the ability to implement Radio Frequency (RF) functions in standard Complementary Metal-Oxide-Semiconductor (CMOS) processes, and avoid more exotic and expensive manufacturing processes, is critical. It makes sense to implement as much of these systems as possible in the digital domain in order to take advantage of the fast clock speeds of fine line CMOS. In addition tomorrow's circuits must be programmable and reconfigurable in order to support multiple standards.

This thesis focuses on the fractional-N Phase Locked Loop (FPLL). This circuit can fulfill many of the functions in the wireless devices of the future. Firstly, a critical function in all RF systems is the requirement to synthesis high frequency, low noise clocks. The FPLL can meet these needs and can be programmed to synthesize RF clocks with a wide range of frequencies, without gaps in the tuning range, as is the case for integer division PLLs. Much of the signal processing can be done in the digital domain, making it suitable for integration in a standard digital CMOS process. Another benefit of the FPLL is the ability to modulate the output signal using the same circuitry that is used to synthesis the LO signal in the first place.

While much of the signal processing in a classical FPLL is done in the digital domain, there is still a significant amount of analog circuitry. The design of analog circuits in the deep sub-micron (DSM) age is challenging, and can often lead to excessive power dissipation, increased sensitivity to substrate/power supply noise, and sensitivity to process variation which can compromise performance or yield. Many of these problems are evident in fractional-PLL (FPLL) design. Charge pumps require good matching between currents of opposite polarity. Low loop-time-constants are typically required, so the loop filter must be implemented using large area capacitors or expensive off chip components. Furthermore, these blocks do not take advantage of the major strength of DSM processes, which is the ability to build fast, complex, low power digital signal processing circuits. This thesis addresses two limitations of this architecture; the reliance on analog circuitry in deep sub-micron technology, and the trade off between low loop bandwidth for good ΣΔ noise rejection, and high loop bandwidth for fast modulation rates.

The fractional-N frequency synthesizer is a key building block of wireless systems as it can both generate a high frequency signal with a well defined frequency and modulate that signal [1], [2]**.** An introduction to the use of ΣΔ modulator in frequency synthesis can be found in [4]. The popularity of this architecture is derived from its ability to do much of the signal processing required for control of the output frequency in the digital domain. Nevertheless, this architecture still relies on a significant amount of analog circuitry. In the remainder of this Chapter some of the uses of FPLLs are briefly reviewed, and some basic PLL concepts are introduced.

1.2 A brief overview of Radio Frequency transmission schemes

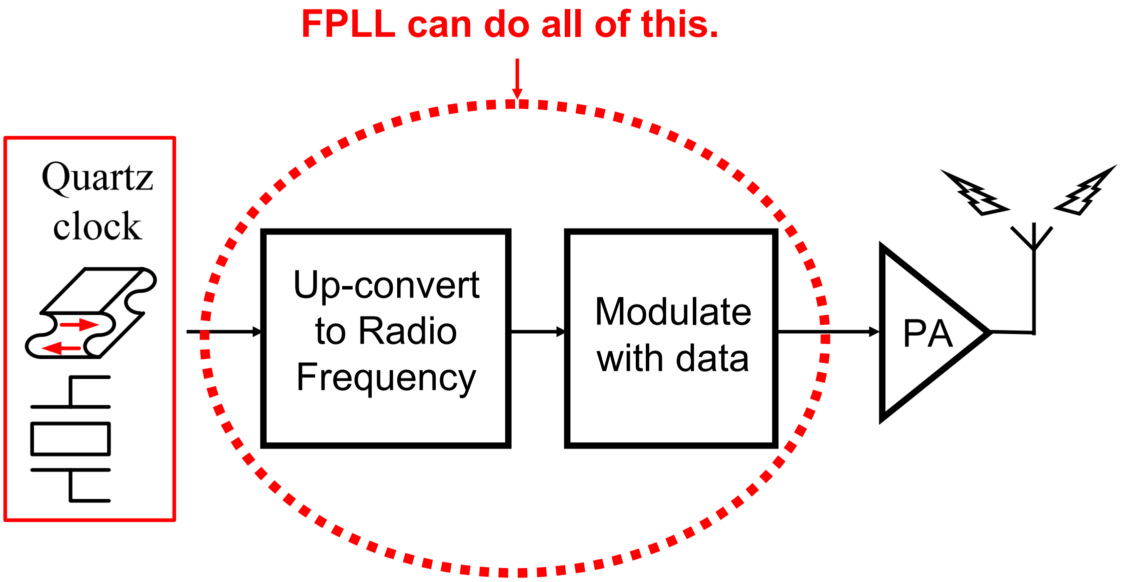

Figure 1.1: An overview of a transmission scheme. A quartz based clock is up-converted to high frequencies and modulated with transmission data before the Power Amplifier.

A conceptual overview of a typical radio transmitter is shown in Figure 1.1. There are two primary steps involved in generating the RF signal. Firstly a carrier signal, which is well defined, stable, and has low noise, must be synthesized. Typically, a low frequency clock source such as a quartz crystal is used as a reference, and a Phase Lock Loop (PLL) up-converts this source to the carrier frequency. The second task is to modulate the RF signal with the information to be transmitted. This task is typically accomplished with the use of a mixer. However, a Fractional-N synthesizer (or FPLL) can do both of these tasks. If an FPLL is described as a black box, then we could say that it has two inputs, and one output. The output is an RF signal of frequency equal to the product of its two inputs; the reference frequency and the digital input division ratio. As an FPLL can synthesize an RF signal of any frequency, it follows that it can also be used as modulator.

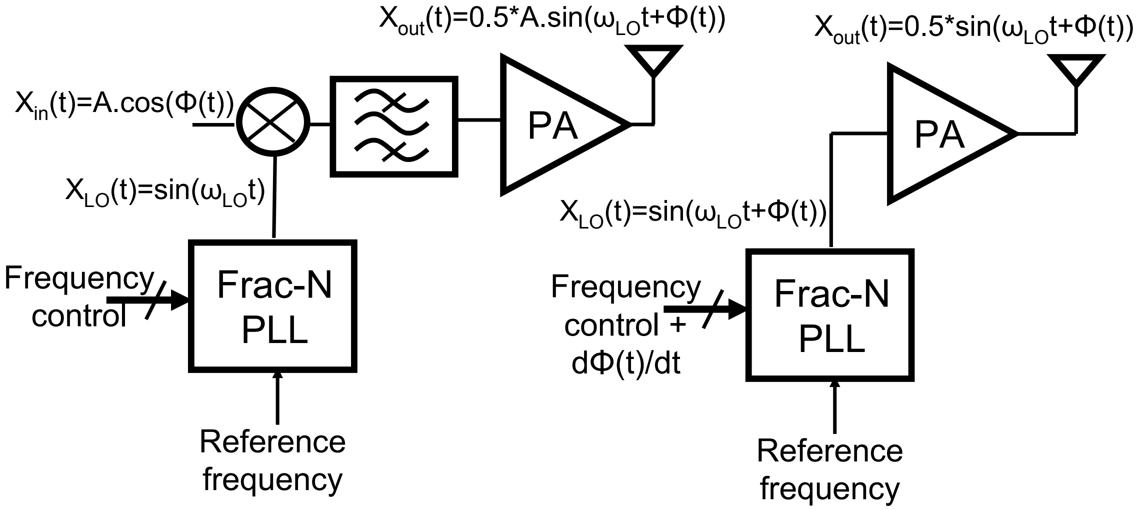
$X_{out}(t)=0.5*A.sin(\omega_{LO}t+\Phi(t))$
$X_{in}(t)=A.cos(\Phi(t))$
PA
$X_{LO}(t)=sin(\omega_{LO}t)$
Frequency control
Frac-N PLL
Reference frequency
$X_{out}(t)=0.5*sin(\omega_{LO}t+\Phi(t))$
PA
$X_{LO}(t)=sin(\omega_{LO}t+\Phi(t))$
Frequency control + $d\Phi(t)/dt$
Frac-N PLL
Reference frequency

Figure 1.2: Possible uses of synthesizers in an RF transmission scheme. a) In a direct conversion transmitter. b) Using just the FPLL as a modulator.

In Figure 1.2(a) an overview of a direct conversion transmitter architecture is shown. A low frequency signal is up-converted to RF by mixing it with a Local Oscillator (LO) signal. In this case a Fraction-N PLL (FPLL) can be used to generate the LO signal. The frequency of the LO signal is given by $\omega_{LO} = \omega_{ref} \cdot N$ where ω_{LO} is the frequency of the LO signal, ω_{ref} is the frequency of the reference and N is the Frequency control word that is fed into the synthesizer. If N is changed then the frequency of the LO changes, hence the synthesizer can be used to select the channel.

If N can be changed to select the output frequency, then if N is slowly modulated then the frequency of the LO signal is slowly modulated. In other words, the original synthesizer can be used to modulate the signal, and no other circuitry is required. Figure 1.2(b) shows a synthesizer being used in the transmission path. No other circuitry apart from the power amplifier is required. So what exactly does an FPLL consist of? To answer this question we first start with an introduction to some of basic PLL concepts.

1.3 Introduction to the PLL

The earliest descriptions of the principles of PLLs date to the 1920's [5], and 1930's [6]. In the 1970's control theory was used to describe PLL's for the use in motor speed control systems [7], and many of the earliest IC implementations of PLLs were used for servo-mechanics [8]-[16]. An introduction to some of the underlying concepts can be found in [17].

Broken down to its core function, a PLL is simply a feedback system acting on phase. The simplest configuration of a PLL is shown in Figure 1.3. The core components include a Voltage Controlled Oscillator (VCO) which produces a signal at its output which has a frequency proportional to the voltage at the VCO's input. In the phase domain this is the equivalent to producing a phase change at the output proportional to the integration of the input. Next a phase detector measures the phase difference between the output of the VCO and a suitable reference clock. Finally a loop filter[1] completes the loop.

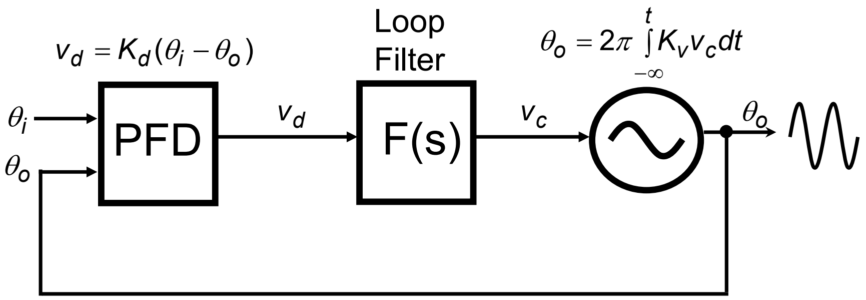
$v_d = K_d(\theta_i - \theta_o)$
Loop Filter
$\theta_o = 2\pi \int_{-\infty}^{t} K_v v_c dt$
θ_i
θ_o
PFD
v_d
F(s)
v_c
θ_o

Figure 1.3: A simple PLL includes a Phase/Frequency detector, a Loop Filter, and a Voltage Controlled Oscillator.

[1] In fact the description "Loop Filter" is somewhat misleading. The objective is not to filter the phase information, but to establish the dynamics of the feedback loop. However, in order to follow convention, the expression "Loop Filter" will be used in this work.

The VCO gain is K_v, and has units of Hz/Volt. (An occasionally used alternative notation is K_o which has units of Radians/Volt.) The phase detector gain is K_d, which has units of Volts/Radian. The loop filter typically includes an integrator and a proportional path, an example for a Type II[2] system is given by Equation (1.1).

$$F(s) = K_1 + \frac{K_2}{s} \tag{1.1}$$

The loop gain is typically defined as the gain around the proportional path of the loop, which is given by Equation (1.2).

$$K = 2\pi \cdot K_v \cdot K_d \cdot K_1 = K_o \cdot K_d \cdot K_1 \tag{1.2}$$

Finally, the system transfer function is given by Equation (1.3).

$$H(s) = \frac{\theta_o(s)}{\theta_i(s)} = \frac{2\pi \cdot K_v \cdot K_d \cdot F(s)}{s + 2\pi \cdot K_v \cdot K_d \cdot F(s)} \tag{1.3}$$

If a divider divides down the output frequency before the phase detector, then the PLL transfer function becomes:

$$H(s) = \frac{\theta_o(s)}{\theta_i(s)} = \frac{2\pi \cdot K_v \cdot K_d \cdot F(s)}{s + 2\pi \cdot K_v \cdot K_d \cdot F(s)/N} \tag{1.4}$$

Notice that at low frequencies (s→0) Equation (1.4) simplifies to H(s)≈ N. Hence when PLLs are used to up-converted a reference signal to a higher frequency, then low frequency phase changes at the input are multiplied by N before reaching the output.

[2] In control theory the Type refers to the number of integrators in the loop. The VCO provides one integrator, so PLLs are always at least Type I, and typically Type II.

1.4 A practical example of a modern PLL

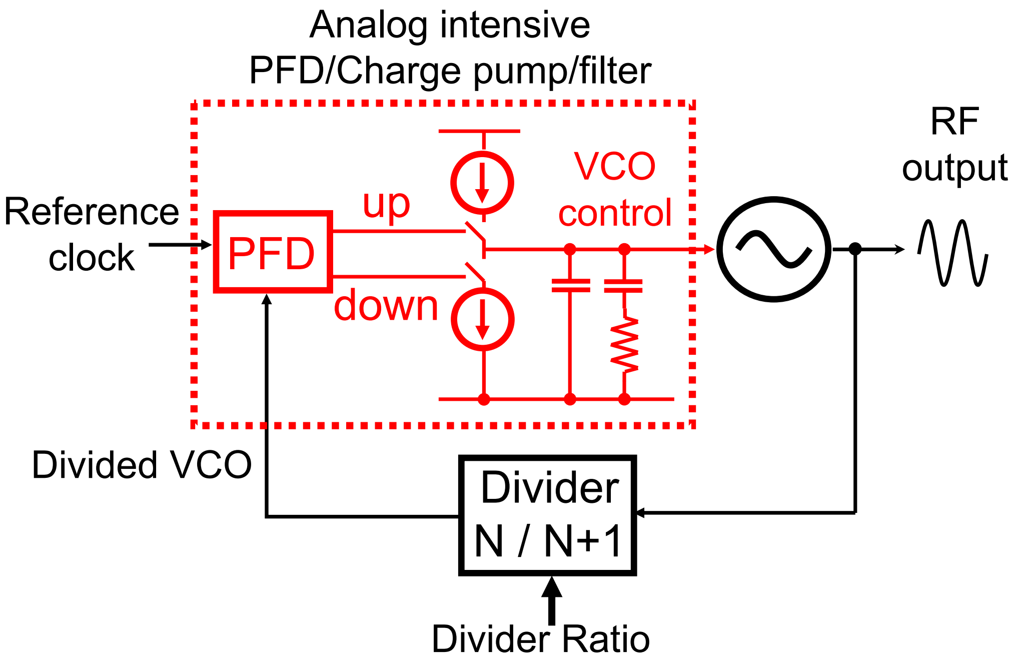

Figure 1.4: A standard PLL. A charge pump is used to convert the output signals form the phase detector to a voltage.

Shown in Figure 1.4 is a typical implementation of an integer-N PLL. On the right of the figure an oscillator generates a high frequency signal. The oscillator output signal is then divided down using a digitally programmable divider. The divider works by counting out phases of the high frequency signal. The counter is digital, and can be made programmable. However the count value, and hence the division ratio is restricted to integer numbers, as a digital counter can only count out whole phases of the output. The divided-down signal is then compared in phase with the reference clock, and the loop is completed with a charge pump and a loop filter. This is a negative feedback loop, the phase and hence the frequency of the divided-down signal is forced to be identical to that of the reference, therefore the frequency of the output signal is given by $\omega_{out} = \omega_{ref} \cdot N$ where N is the division ratio of the divider.

At this point some observations are in order. The frequency of the output can be changed by changing the division ratio, N. However, N is restricted to integer numbers, therefore the frequency-step size at the output is restricted to the frequency of the reference. In other words, if a small frequency step at the output is required then a small reference frequency should be used. However, it is impractical to use references whose frequency is close to the bandwidth of the PLL. This is because a PLL is a sampled system, it can only represent phase information at frequencies less than Fs/2, where Fs is the reference frequency (This should not be mixed up with the frequency of the output signal, which can be order of magnitude higher). If aliasing is to be avoided, a reference frequency should be used which is significantly faster than the loop bandwidth. On the other hand, the PLL bandwidth needs to be sufficiently wide (a typical value is 100kHz) if the low-frequency VCO phase noise is to be suppressed. For these reasons, when using practical values of reference frequency, changing the value of N by an integer usually leads to a frequency step at the output that is too large to be used to select a channel in a conventional RF communication system. A second observation is that the input reference is typically a digital clock, useful phase comparisons can only be made on the edges of the reference clock[3]. Therefore this type of system can be considered to be a sampled system, even though the output is a continuous time signal.

1.5 A first attempt at non-integer based division

If the division ratio is set to a fractional number, then the frequency of the output signal is no longer restricted to integer multiples of the reference frequency. A potential

[3] If the reference clock is a sine wave, and if the divider is not a digital divider (i.e. it also produces sine waves as a mixer would) then information can be extracted from the phase detector at time points other than the clock rising edge.

solution to this problem is to switch the divide ratio between different integer values so as to get an average divide ratio which is a fractional number. One approach is to use the carry out bit of an accumulator to control the divide ratio, so that the average output of the accumulator is equal to the input, similar to a first order ΣΔ [18].

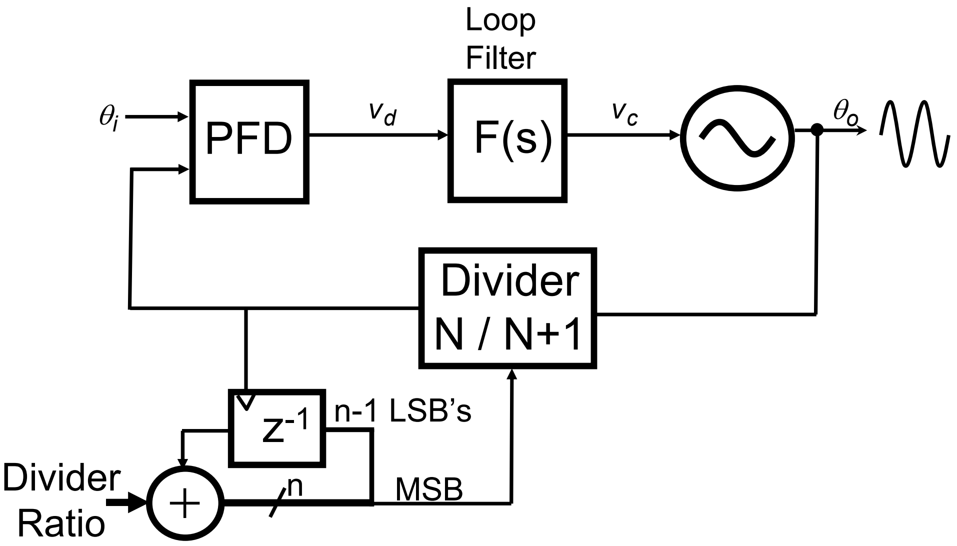

Figure 1.5: A first attempt at a Fractional-N Synthesizer [18]. The MSB from an accumulator controls the divider. The LSBs are recycled so there is no DC divider error.

This method can provide fractional synthesis, but leads to a periodic phase error. For example, to achieve an average divide ratio of 8.9, the accumulator output becomes a repeating pattern of {9,9,9,9,9,9,9,9,9,8}. This pattern is highly periodic, and results in fractional spurs. Bjerede, et al., introduced the idea of cancelling out the periodic phase error with a second path to the output of the phase detector [19]. Differences in the gain of the two paths can limit the effectiveness of this technique, however this idea remains popular to this day, where an alternative path is often used to cancel out ΣΔ noise [20].

Riley, et al., recognized that the accumulator in Figure 1.5 is actually a first order ΣΔ modulator. If the modulator is replaced with a higher order modulator, then the quantization noised can be shaped to high frequencies, and effectively filtered out without effecting the output signal [1],[20]. Miller, et al., independently cane to the same conclusion [22], [23]. Using these methods the frequency of the output signal can be changed indirectly, by change the division ratio within the loop, without adding low frequency noise to the output.

1.6 A brief introduction to the ΣΔ modulator

In this section a brief overview of a ΣΔ modulator is given.[4]

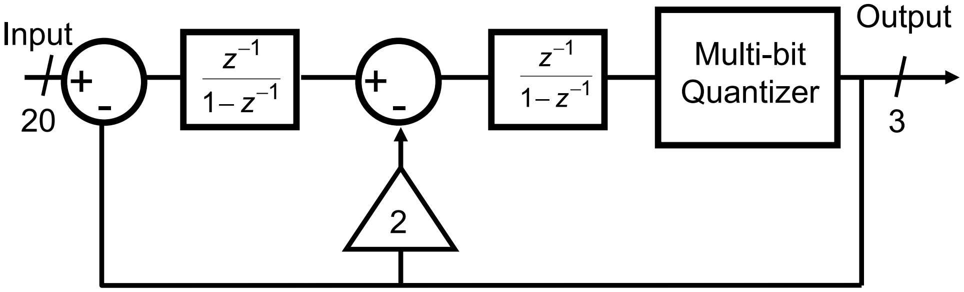

Figure 1.6: A second order digital ΣΔ modulator, which includes two integrators, and a quantizer.

A ΣΔ modulator can switch its output between different quantization levels, while keeping the low-frequency average of the output equal to the input. For example, in Figure 1.6 the architecture of a standard 2nd order ΣΔ is shown. The forward path consists of two integrators and a quantizer. The feedback forces the average output signal to be equal to the average input signal, despite the fact that the output is a low resolution bus (3bits) and the input is a high resolution bus (20bit in this case). The second order nature of the feedback system means that the quantization noise is shaped to high frequencies

[4] For a more in-depth analysis, there are some excellent texts on this subject [46]

according to Equation (1.5), where *n* is the order of the feedback loop, E(f) is the quantization noise and N(f) is the quantizer noise as seen at the output [46].

$$N(f) = E(f)\left|1 - e^{-2\pi jfT}\right|^{n} \tag{1.5}$$

Using a higher order feedback system results in less noise at low frequencies at the expense of more noise at high frequencies (which can be filtered out by the PLL). As an example, Figure 1.7(a) shows the input and output, and Figure 1.7(b) shows the spectrum of these signals if the input of the ΣΔ is a sine wave. Although at first glance the quantized output might not look much like the input, at low frequencies they are the same, as can be seen in the output spectrum. A key point is that the quantization noise is pushed to high frequencies, so if the output is low-pass-filtered then it faithfully represents the low frequency content of the input.

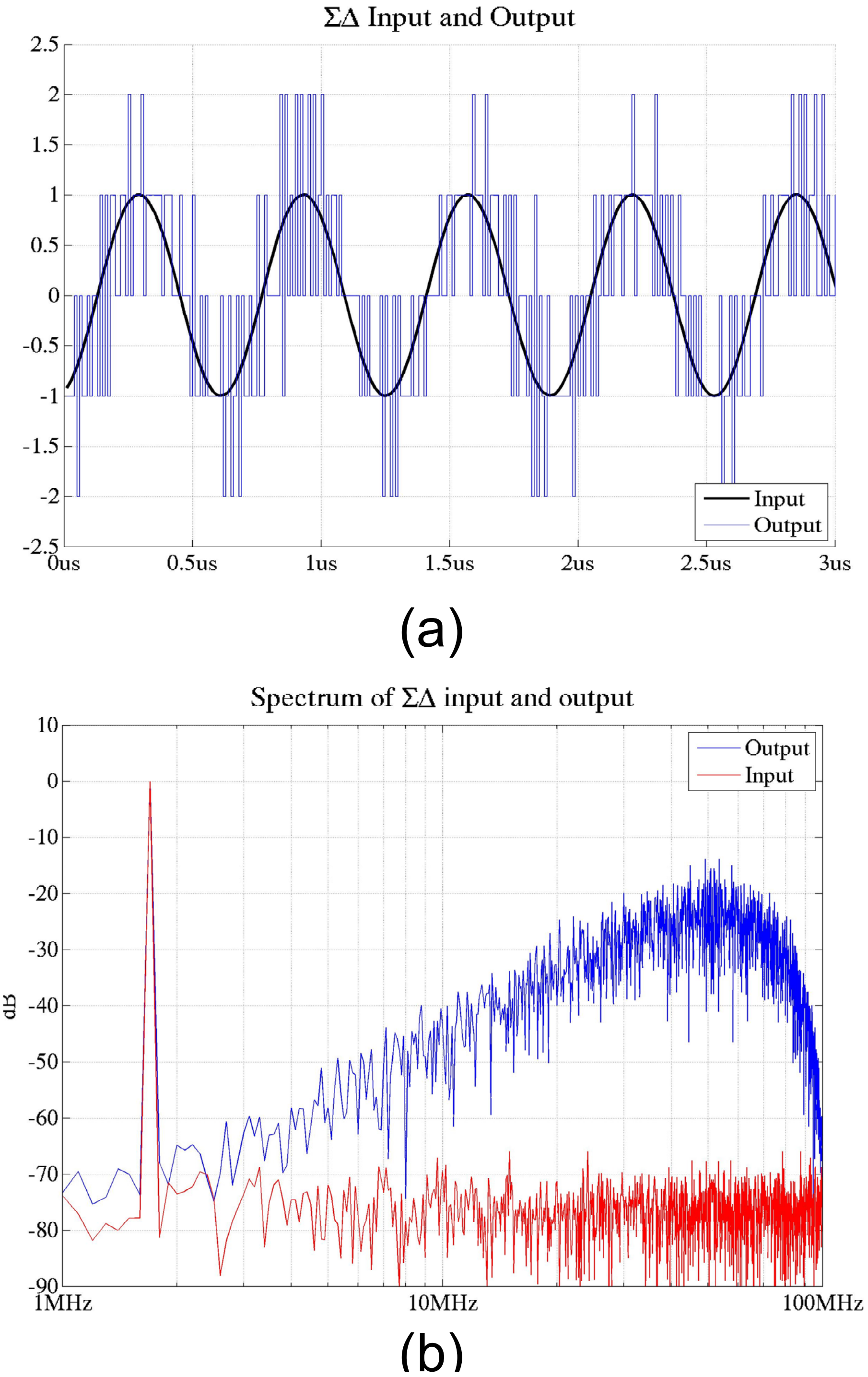

Figure 1.7 Waveforms at input and output of the ΣΔ. (b) The spectrum of the input and output.

1.7 A conventional Fractional-N PLL(FPLL)

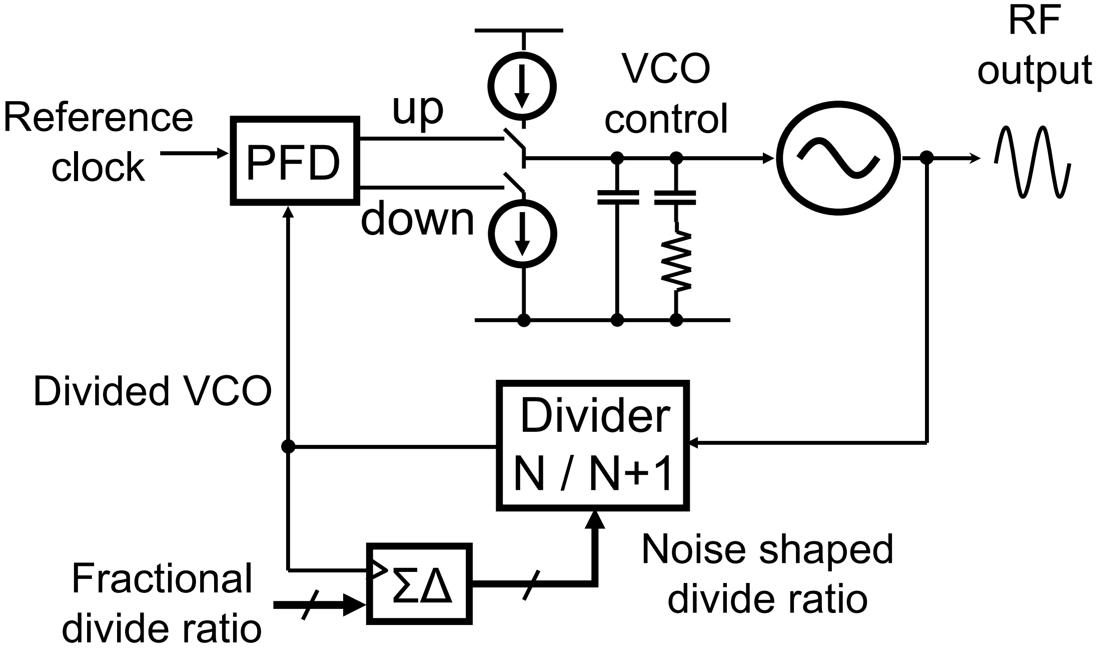

Figure 1.8: A conventional fractional-N synthesizer. In this example a ΣΔ modulator controls the programmable divider.

The ΣΔ can be used in the PLL to control the average divide ratio, as shown in Figure 1.8. The divider itself is restricted to integer numbers, but the ΣΔ modulator allows us to set the average divide ratio to a non-integer or fractional number. The PLL has an overall transfer function which is low pass in nature, therefore the ΣΔ noise, which is shaped to high frequencies, is filtered out.

This architecture derives much of its popularity from the fact that the frequency and phase of the output signal can be controlled and modulated using digital blocks; the programmable divider and the ΣΔ modulator.

1.8 The programmable frequency divider

In this section a brief introduction to the programmable divider is given. The programmable divider, or pre-scalar, divides the RF signal by one of several possible integer divide ratios. Most implementations consists of asynchronous dividers in series where the first divider is programmable [24],[25],[26],[27], or where all of the dividers are programmable [28] which can used to achieve a wider division range. The basic idea is that one of the dividers can "swallow" a pulse, resulting in a higher count, and hence a larger divide ratio. Asynchronous designs are used so that only the first divider block is running at the full RF frequency. An example of a 2/3 divider is shown in Figure 1.9.

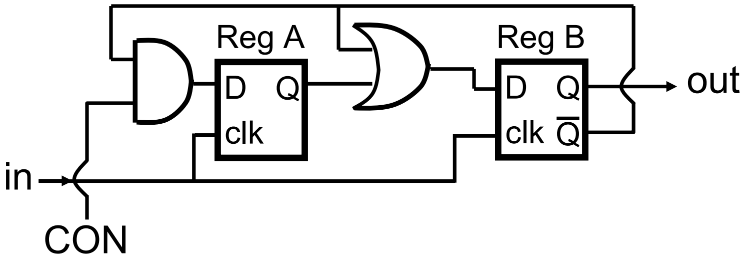

Figure 1.9: A simple 2/3 Divider.

If the CON signal is low then the output of *Reg A* is low, and hence *Reg B* simply acts as a divide-by-2 flip-flop. On the other hand, if CON briefly goes high, then the circuit "swallows" an extra pulse, resulting in a divide by 3. CON can be controlled by the divider so that it goes high only once per output cycle, as shown in Figure 1.10.

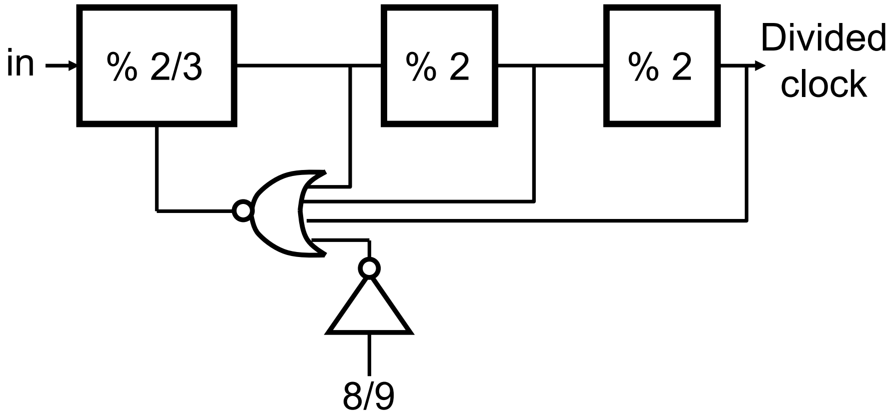

Figure 1.10: A 8/9 bit asynchronous divider. The additional control logic tells the %2/3 to divide by 3 once per output period.

By using 2/3 divider cells for each of the dividers, this idea can be easily extended to implement a divider which can be programmed to divide by values from 8 to 15. The divider used in this work is slightly more complicated, and will be described in Section 5.5.

1.9 Chapter summary

This chapter began with a discussion on the applications and background of PLLs. Next, we introduced some of the key ideas needed to understand the basic FPLL. Although only a brief introduction to these concepts is provided, hopefully it gives enough background so that some of the more novel concepts in later chapters can be understood.

In this research work, techniques are proposed and demonstrated which address two limitations of the FPLL modulator architecture; the reliance on analog circuitry in deep sub-micron technology, and the trade off between low loop bandwidth for good ΣΔ noise rejection, and high loop bandwidth for fast modulation rates. Chapter 2 introduces an all-digital phase detector, which relies on a single flip-flop for phase quantization.

Chapter 3 focuses on the small signal analysis of the new architecture. Chapter 4 introduces a digital dual-modulation scheme that alleviates the tradeoff between loop bandwidth and switching speed. Chapter 5 discusses the physical prototype, and Chapter 6 discusses measurements from silicon. Finally, Chapter 7 presents conclusions.

1.10 Thesis Contributions

This thesis covers several aspects of digital Fractional-N Phase Lock Loops. The broad theme of this thesis is to implement an all digital phase detector, and to take advantage the digital nature of the system to increase the modulation rate. The key contributions are listed below:

- *Digital phase detection*: In a standard FPLL an analog intensive charge pump and loop filter is typically required. In this work a technique has been developed for implementing the phase detector in the digital domain.

- *High Speed Switching*: In an FPLL the speed at which the output frequency can be switched is limited by the bandwidth of the loop. In this work we propose and demonstrate a technique which breaks this trade off for 2-FSK switching schemes.

- *Small signal modeling*: A small signal model has been developed for the quantizer, which allows for the derivation of the loop dynamics and phase noise at the output.

- *Removal of jitter peaking*: Analysis of the design reveals that the proposed architecture avoids jitter peaking.

CHAPTER 2

DIGITAL PHASE DETECTION

This chapter introduces the new phase detection technique, and compares it with prior art. The focus will be on qualitative explanations, with some of the more quantitative derivations presented in Chapter 3.

2.1 Background

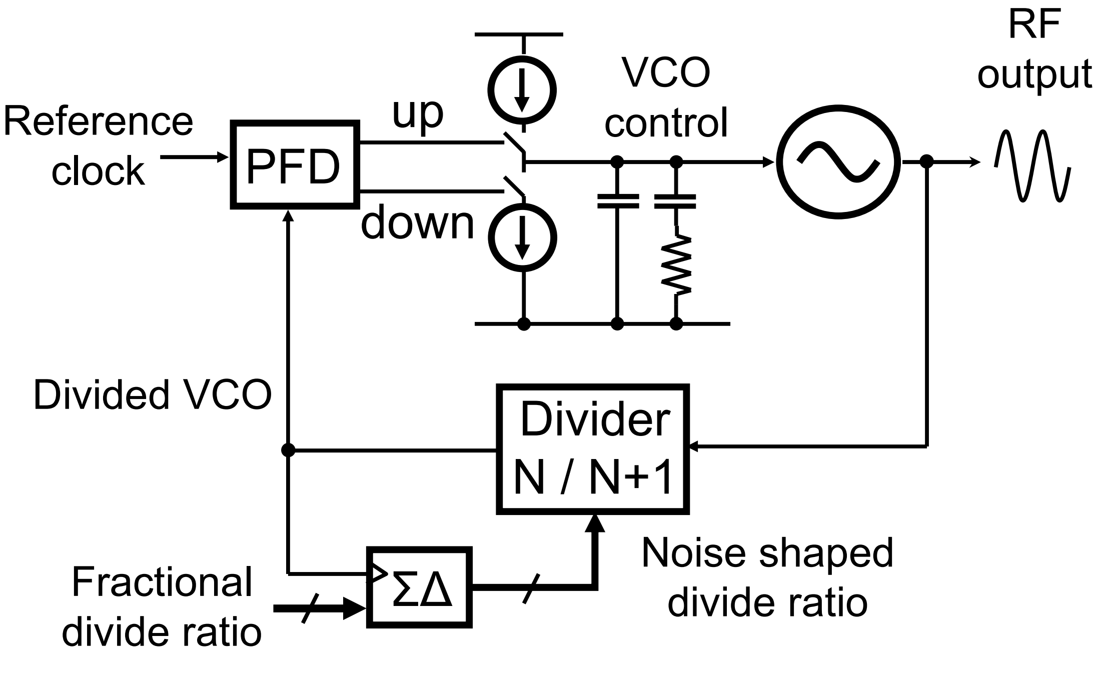

Figure 2.1: A conventional fractional-N synthesizer (Reproduced from Figure 1.8).

We start by taking a fresh look at the FPLL shown in Figure 2.1. An FPLL system can be considered to be a type of digital-to-analog or more specifically a digital-to-frequency converter, with the phase of the input clock acting as the reference, the frequency of the input clock corresponding to the sampling rate, the divider control corresponding to the digital input, and finally, the frequency of the RF signal corresponds to the output. It should come as no surprise that many of the challenges associated with building these systems in deep sub micron (DSM) processes are similar to the challenges associated with ADC/DAC design. In fact a typical FPLL contains a mixture of analog and digital sub-blocks within the loop, with implicit conversion from digital to analog. For example the combination of the $\Delta\Sigma$ and the programmable divider are used to control the analog phase of the divider's output which implies the presence of a DAC. However, much of the loop remains analog.

In Figure 2.1 the information extracted by the phase detector is inherently analog in nature, since the phase information is not synchronized to either the reference clock or the divided down VCO clock, and is not quantized. Although conventional XOR and tri-state phase detectors utilize digital building blocks, a charge pump and filter are still required to extract useful phase-difference information. If a phase-detector with a digital output were available then a digital loop filter could be used in place of the troublesome analog filter.

In this chapter a new method of digital phase detection is introduced. First, some examples of digital phase detection from the literature are discussed. Next, a method of using a single flip-flop as a phase quantizer is introduced. This phase quantizer scheme requires an extra loop, which we call a phase minimization loop.

2.2 A Time to Digital Converter (TDC)

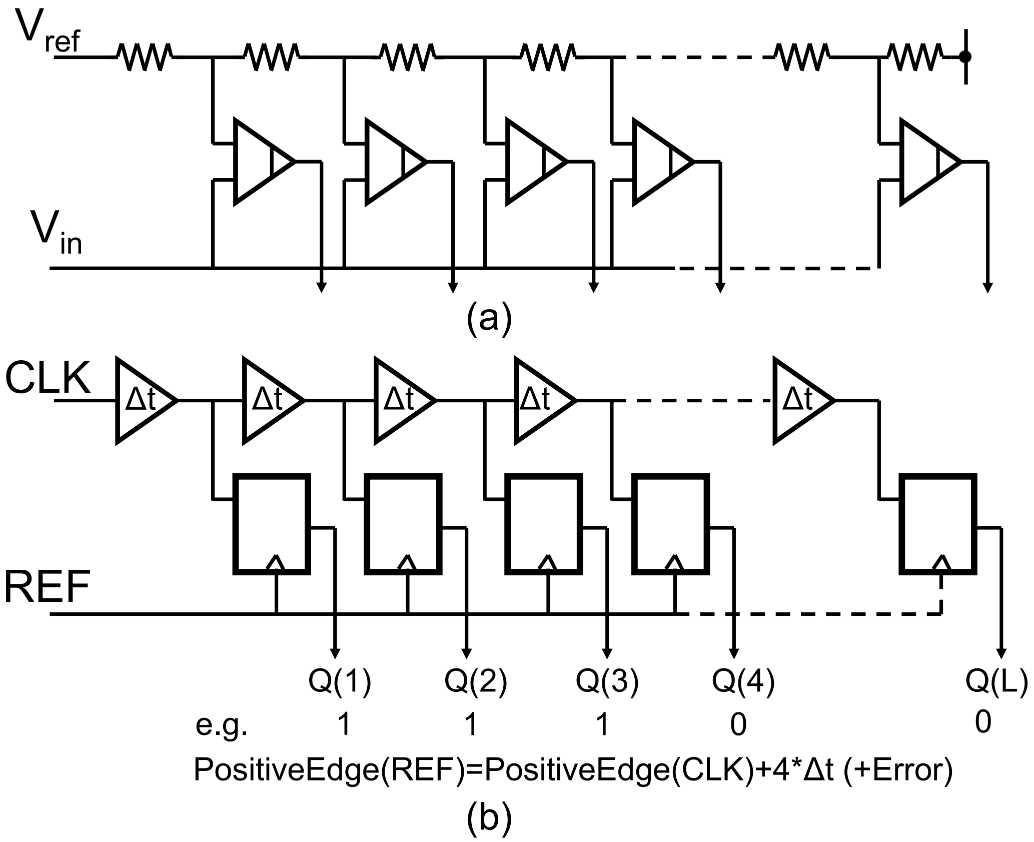

Figure 2.2: (a) A Flash analog-to-digital converter. (ADC) (b) A simplified time-to-digital converter (TDC), which can be considered a flash converter acting on phase.

An example of a digital phase detector is found in [33], where a time-to-digital converter (TDC) uses multiple flip-flops and unit delays (in practice inverters) to quantize the time difference between the edges of the reference clock and feedback clock, as shown in Figure 2.2(b). If each inverter has a unit delay of Δt, then the phase detector has a minimum phase step size of $2\pi\Delta t/T$. This architecture is analogous to a flash ADC, where a resistor ladder divides down a reference voltage, and a set of comparators compares the incoming signal with each of the references as shown in Figure 2.2(a). In a TDC the resistors are replaced with unit delay elements, and the comparators are replaced with flip-flops as shown in Figure 2.2(b). There are some interesting differences between these ADC configurations. In a Flash ADC the two ends of the divider ladder are at well defined voltages, V_{ref} and ground, therefore the largest Integrated Non-Linearites (INL) will occur at the mid point at the ladder. In contrast we cannot say that the end of a TDC is at a well defined phase. The effects of random jitter and inverter mismatches accumulate as the signal progresses down the inverter chain, therefore it makes sense to use as short a chain as is possible.

While TDCs have been used successfully in all-digital PLLs [36], they have some limitations. Firstly, the resolution and linearity are dependant on the speed and matching of the unit delay elements, and hence are inherently process dependant. One of the key motivations for this work is to reduce process dependency, and to increase the accuracy of the phase detector. Secondly the gain of the TDC in [36] is a function of the inverter delays. While these delays could be controlled with a Delay Locked Loop (DLL), this would add significant additional analog processing. As will be shown, a new technique introduced in Section 2.4 addresses these limitations.

Recently other more complex TDC architectures have been proposed [34], which use techniques developed for Multiplying Delay Locked loops (MDLL) [35]. In these systems a Gated Ring Oscillator (GRO) is used in the TDC, which has been shown to shape the quantization noise to high frequencies, like a first order ΣΔ.

2.3 A single bit phase quantizer as a phase detector

If a conventional TDC is analogous to a flash ADC, with the unit delays setting the quantization steps, then the proposed phase detector is analogous to an over-sampling 1bit ADC, with over-sampling and a phase integration loop used to improve the performance of a coarse single-bit phase quantizer. If only a single flip-flop is used as the quantizer, then the problem of mismatch between unit delay elements is eliminated. There are some properties of a PLL which make this approach practical. First the bandwidth of the loop can be made to be a small fraction of the reference sampling rate. Secondly the ΣΔ controlled divider adds dither, which is an important requirement of any over-sampled quantizer.

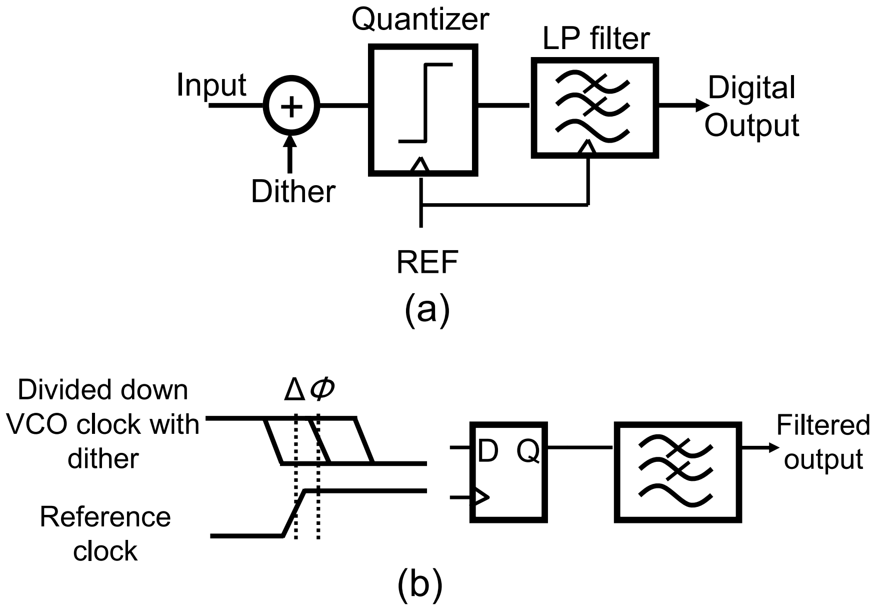

Figure 2.3: (a) Conceptual overview of a 1bit ADC. (b) Using a flip-flop as a phase quantizer. If dither is added to the input, the quantizer's output can be filtered, to reproduce the input.

The new phase detection technique uses a single flip-flop as a phase comparator, while an additional negative feedback loop around the programmable divider keeps the phases of the two clocks aligned to within a single quantization step. On the rising edge of the reference clock, the flip-flop samples the divided-down VCO signal, determining whether the divided clock is ahead or behind the reference clock. In this way, the flip-flop effectively acts as a one-bit phase quantizer. In Figure 2.3(b) a single flip-flop is used to quantize the phase difference, $\Delta\Phi$, between the reference clock and the divided down VCO clock. The quantization noise of the ΣΔ controlled divider is added to the divided down VCO clock and this acts as dither for the phase quantizer. If this dither were absent and for example if $\Delta\Phi$ were positive, then the output of the flip-flop (quantizer) would always be one, irrespective of the magnitude of the phase difference between the divided down clock and reference clock. In the presence of the dither, the output of the flip-flop is sometimes one and sometimes zero. Similar to an over-sampled ADC, the low-pass-filtered output of the flip-flop can be proportional to $\Delta\Phi$, depending on the properties of the dither. An approximately linear relationship is achieved if the phase difference $\Delta\Phi$ is small compared with the added dither. A more detailed discussion of the characteristics of the dither, and its effects on the quantizer gain takes place in Sections 3.2 and 3.3.

While using a single flip-flop as a phase detector is conceptually possible, there are some limitations which would make this scheme difficult to use without some additions to the loop. Firstly, the quantizer gain is a function of the probability density function of the ΣΔ quantizer noise. If this *pdf* is non-uniform then the gain of the phase quantizer is non-linear, particularly for large phase deviations. For this reason a new technique is used to keep the phase difference at the input of the quantizer small. This is called a Phase-Minimization Loop (PML).

2.4 Phase Minimization Loop (PML)

In conjunction to the use of a sampling flip-flop, an additional feedback loop is introduced, as shown in Figure 2.4. The new feedback loop has two purposes. Firstly, the new loop keeps the phase difference between the two clocks small, i.e. minimizing $\Delta\Phi$. For this reason the new loop is called the *phase minimization loop* (PML). Secondly, the

PML desensitizes the overall transfer function to the small signal phase detector gain. (Although the overall PLL keeps the average of $\Delta\Phi$ at zero, the instantaneous value of $\Delta\Phi$ is a function of the bandwidth of the PLL, which cannot be set arbitrarily large.) The bandwidth of the inner loop, however, can be set to be significantly larger than that of the overall PLL, hence can do a better job keeping the two clocks in close phase alignment. This is necessary in order for the phase detector to behave in an approximately linear fashion. If the phase deviation, $\Delta\Phi$, is too large then the quantizer output becomes stuck at 1 or -1, and the dither will not have the desired effect.

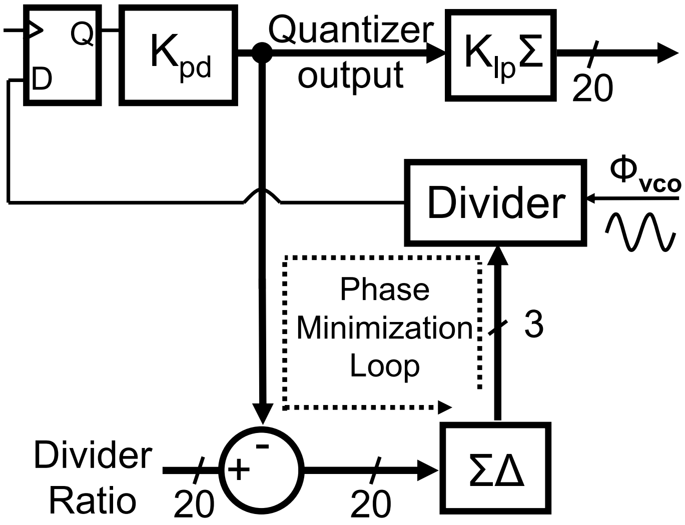

Figure 2.4: New Phase detector configuration with the Phase Minimization Loop (PML). The quantized phase information is fed back to the input of the ΣΔ.

The one-bit output of the flip-flop is passed through a linear scaling block K_{pd}, so that the output of the quantizer is +/- K_{pd}. This quantized information is then fed back to the input of the ΣΔ modulator that controls the programmable divider, forming the phase minimization loop. K_{pd} is used to set the bandwidth of the PML.

The only pseudo-analog component is the decision making flip-flop, everything else is synchronous digital circuitry. The flip-flop should be treated in a similar fashion to a comparator in an ADC, as its set-up and hold times are not necessarily respected, as would be the case in a truly digital synchronous system. Therefore attention must be paid to meta-stability, gain, and other characteristics as for a comparator in an ADC. Some techniques to build a suitable flip-flop are discussed in **[33]**. However, for our design a flip-flop from a standard digital cell library is adequate. This digital approach does not rely on component matching or on any process dependant parameters such as inverter delays.

From a phase perspective, the PML incorporates an integrator in its feedback path. This is because the quantized phase information is fed back to the *frequency* control of the divider. As phase is the integral of frequency, this implies the presence of an integrator in the phase domain. Referring again to Figure 2.4, the presence of this integrator changes nature of the transfer function from the VCO output (Φ_{vco}) to the output of the quantization flip-flop, and also changes the transfer function from the modulator input to the output of the flip-flop. A phase-domain model of the divider is shown in Figure 2.5, the derivation of which is discussed in [37].

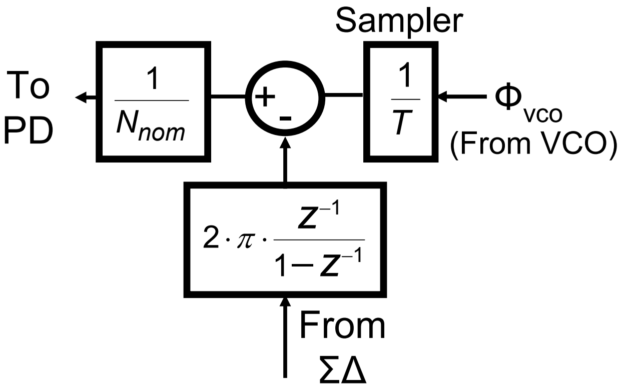

Figure 2.5: The small signal phase domain model of the programmable divider. It includes an integrator in the control path.

The phase transfer function from the VCO output (Φ_{vco} in Figure 2.4 and Figure 2.5) to the output of the flip-flop is now high-pass instead of all-pass, because of the integrator in the feedback path.

Similarly, the transfer function between the input to the modulator ("Divider ratio" in Figure 2.4) and the quantizer output becomes flat instead of low-pass. Without the PML, the transfer function between the input to the modulator and the phase detector output would be low-pass, because the frequency control signal passes through an integration associated with the conversion to phase. However, the integrator is now within the forward path of a feedback loop. As shown in Figure 2.4 a digital integrator placed at the output of the quantizer compensates for the change in transfer function that the PML has caused.

2.5 Viewed from a *Delta Modulator* perspective

The PML is similar to a delta modulator, a popular ADC technique. (Several standard text books deal with delta modulators, such as [38], [39], [40], and [41].) It is worthwhile reviewing the delta modulator architecture, in order to get a better understanding of the PML. In a delta modulator the forward path consists of a quantizer, and the feedback path includes an integrator. (In this work the integration occurs because phase information is fedback to the frequency control of the divider.) A standard delta modulator architecture is shown in Figure 2.6.

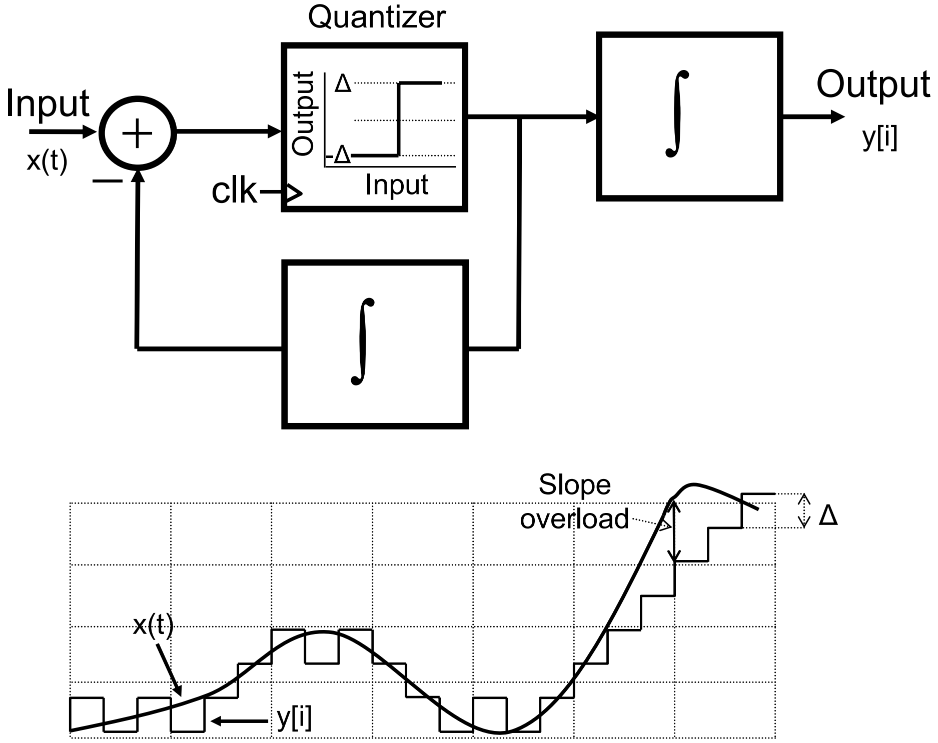

Figure 2.6: A delta modulator is a commonly used precursor to a ΣΔ modulator. There are two types of errors; slope overload and quantization error.

The modulator consists of a feedback system with a 1bit quantizer in the forward path and an integrator in the feedback path. The input signal (x(t) in Figure 2.6) minus the feedback signal is quantized to +/-Δ. The feedback keeps the input to the quantizer smaller than one quantization step. The key advantage to this is that a single bit quantizer with a step size much smaller than the signal swing can be used. If the feedback loop is not included then achieving the same input range would be challenging; either a multi-bit quantizer with the same step size would be needed, or a single bit quantizer with a much bigger step size (resulting in a much greater amount of quantization noise.)

There are two types of noise. Firstly there is regular quantization noise, which can be reduced by using a smaller quantizer step size. Secondly there can be slope overload, which occurs when the input signal changes two quickly with the feedback being unable to catch up. To prevent slope overload a sufficiently large quantization step and clocking rate needs to be used.

The system shown in Figure 2.6 is similar in form to Figure 2.4, when the relationship between the VCO phase Φ_{vco} and the phase detector output is considered. Both of these systems have an integrator in the feedback path. In the PML, the integrator consists of the conversion from the frequency domain to the phase domain i.e. Quantized phase information is fedback to the frequency control of the divider. In addition the feedback path in the PML includes a ΣΔ modulator, which also injects noise into the loop. Therefore the PML is similar to a delta modulator but not identical.

2.6 Comparison between the PML and a frequency discriminator

A similar technique is presented as part of a frequency discriminator[5] in [42], which can be used as an RF demodulator or as part of a Fractional-N PLL [43], [44][6], in which the quantized phase signal is fed directly back to the divider control, and no ΣΔ is used in the feedback path. The lack of the ΣΔ in the feedback path means that the quantization step must be set by the divider step, which results in very large amounts of

[5] In this work *Phase Minimization Loop* is preferred to *Frequency Discriminator*, as the feedback is used to minimize the *phase* difference at the input of the quantizer.

[6] Also [42]- [44] include analog integrators with charge pumps in order to implement higher order noise shaping of the frequency error, a step that has been avoided in this work in order to maximize the digital nature of the system.

quantization noise. In this work, the use of the ΣΔ to control the divider means that the combination of divider and ΣΔ can be treated as one linear block, and hence the quantizer step size can be chosen independently of the divider. The ΣΔ allows the average frequency change at the output of the divider to be smaller than just the divider itself would allow.

2.7 Desensitization to phase detector gain

Consider the feedback system in Figure 2.7, which is intended as a simplified representation of the PML, for explanation purposes only. The digital integrators are replaced with analog integrators and all loop gain terms are lumped together into two constants K_1 and K_2.

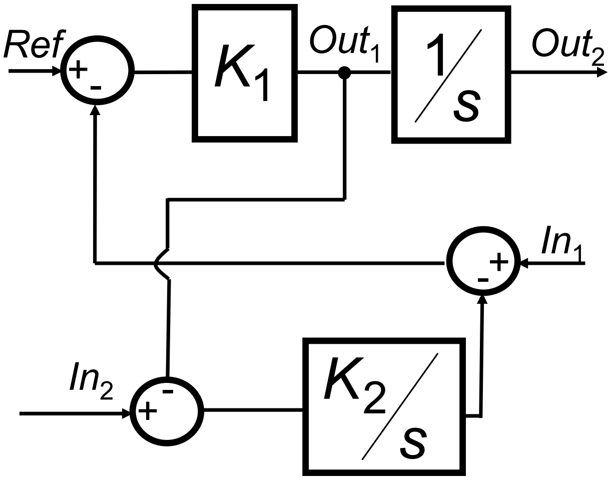

Figure 2.7: A simplified continuous time version of the PML feedback system. K_1 includes the phase quantizer gain.

The phase detector gain is included in the constant K_1 and therefore it is desirable if the closed loop transfer function is not sensitive to this constant. Consider the relationship between the phase signal from the VCO (In_1) and the output of the system (Out_1). If *Ref* and In_2 are set to zero, it is trivial to show

$$Out_1 = K_1 \cdot \left(-In_1 - \frac{K_2}{s} \cdot Out_1 \right) \tag{2.1}$$

This can be rearranged to give Equation (2.2).

$$\frac{Out_1}{In_1} = \frac{-sK_1}{s + K_1 K_2} \tag{2.2}$$

Out_2 is simply Out_1 integrated so the final output is given by Equation (2.3).

$$\frac{Out_2}{In_1} = \frac{-K_1}{s + K_1 K_2} \tag{2.3}$$

Notice that in Equation (2.3) at low frequencies ($s << K_1 K_2$) the gain simplifies to -$1/K_2$. Recall that it is K_1 that contains the phase detector gain, while K_2 is made of constants which are well defined and stable. Therefore the low frequency gain of the overall phase detector is desensitized to the absolute value of the phase detector gain. For this reason variations in the gain of the phase quantizer affect only the bandwidth of the inner loop, not the gain. This is a very useful and important property of the PML, which could well be used in many different digital phase detection schemes. For example, as the TDC's gain is process dependant, the PML loop could be used to desensitize the gain to process variation. This also explains why a phase detector with a poorly defined gain can reliably be used in the system.

Figure 2.8 shows an overview of the new architecture with the new phase detector configuration. The output of the integrator goes to a DAC, which drives the analog control of the VCO, so as to complete the loop. K_{lp} is a digital gain that modifies the bandwidth of the overall loop, while gain block K_{pd} modifies the bandwidth of the inner loop. Notice that almost everything in the loop is now digital apart from the DAC, VCO and output buffer.

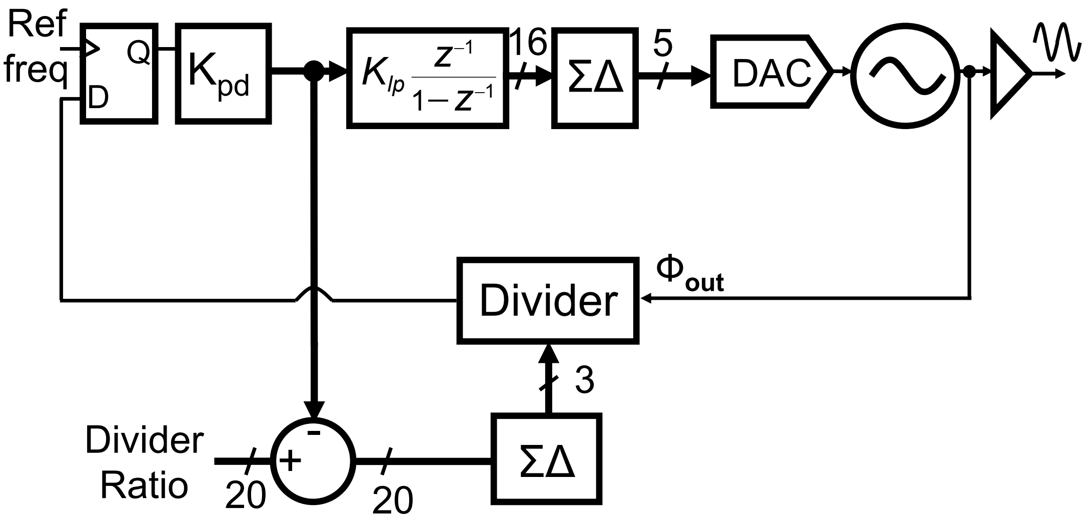

Figure 2.8: The overall architecture of the proposed FPLL, including the new phase detection scheme.

CHAPTER 3

PHASE NOISE AND SMALL SIGNAL MODELING

3.1 Modeling assumptions

A key task in the development of new circuit techniques and architectures is deriving the small signal models for the main components, in order estimate the loop dynamics, and subsequently to estimate the phase noise at the output. Before beginning with the derivation of the small signal model, it is worth discussing some of the assumptions which are typically made in PLL the PLL modeling process, so that the limitations of the model are understood.

The eventual model is a linear model, and non-linear behavior is ignored. For locked or tracking behavior this approximation is reasonable [48], however for large signal behavior such as achieving lock, or for large disturbances, the model is not valid.

An FPLL contains a mixture of continuous time and discrete time blocks. The phase noise at the output is a continuous time signal, therefore a technique known as pseudo-continuous analysis is used, which involves approximating discrete time blocks as continuous time blocks, relying on the fact the PLL has low pass characteristics and only the low frequency alias of the signal need be considered [49].

The spectrum density is defined as the Fourier transform of the autocorrelation function, which exists only if the autocorrelation function is stationary. However the autocorrelation of an oscillator with frequency drift is non-stationary. Therefore when discussing the spectrum density of an open loop oscillator, this should be taken as an engineering approximation of the spectrum, not a strict mathematical formulation.

One of the key modeling tasks is to derive the approximate small signal gain of the phase quantizer introduced in Chapter 2. First, we discuss the small signal gain of a conventional phase detector. Then the model of a tri-state phase detector will be related to that of our quantizer based detector, while making note of all appropriate approximations.

3.2 Phase detector modeling

A standard tri-state phase detector produces a series of pulses which whose widths are proportional to the time difference between the reference clock and the divide down VCO clock. A common modeling approximation [37] is to substitute the time width modulation of the output of the phase detector with fixed width, amplitude modulated signals, as summarized in Figure 3.1.

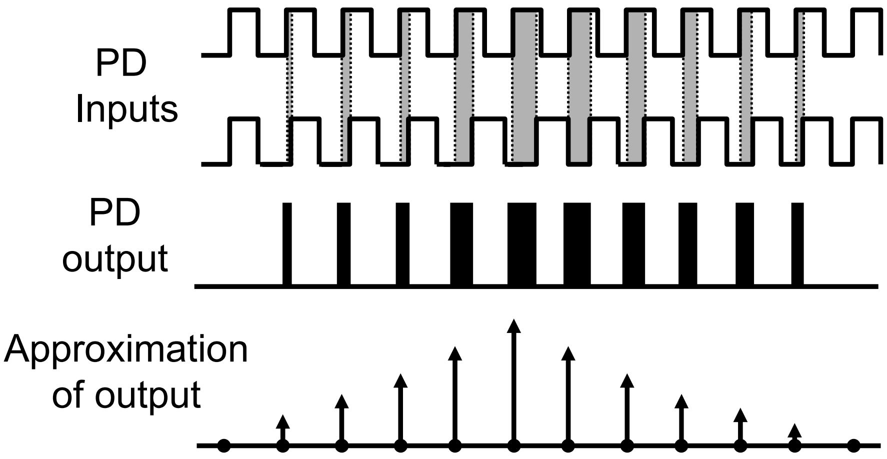

Figure 3.1: Waveforms approximating the signals at the phase detector output.

This approximation allows us to model the phase detector's output as a sequence of impulses whose area is proportional to the phase difference. The approximation has little effect on the loop provided that the loop bandwidth is significantly smaller than the reference frequency as discussed in [29]. A general time domain model for a (tri-state) phase detector is given in Equation (3.1):

$$E(t) \approx \sum_{k=-\infty}^{k=\infty} \Delta t_k \partial(t - kT) \tag{3.1}$$

Where the pulses Δt_k are given by Equation (3.2). Note that the phases of the reference and divided-down VCO signals are treated as sampled signals, as the phase difference information is only extracted on the rising edge of the reference clock.

$$\Delta t_k = \frac{T}{2\pi}\left(\phi_{ref}[k] - \phi_{div}[k]\right) \tag{3.2}$$

From this it can be shown that the small signal gain of the tristate phase detector is given by Equation (3.3) which has units of seconds/radians (typically a charge pump and loop filter are used to convert the phase difference to voltage):

$$K_{pd\_tristate} = \frac{T}{2\pi} \tag{3.3}$$

Our objective is to develop a model for the phase quantizer which is similar in form to (3.3). The gain of the phase detector presented in Chapter 2 needs to be approximated with a linear gain. Even though the output of the phase quantizer is one of two values, if the output is low pass filtered then gain of the quantizer can be approximated, as shown in Figure 3.2.

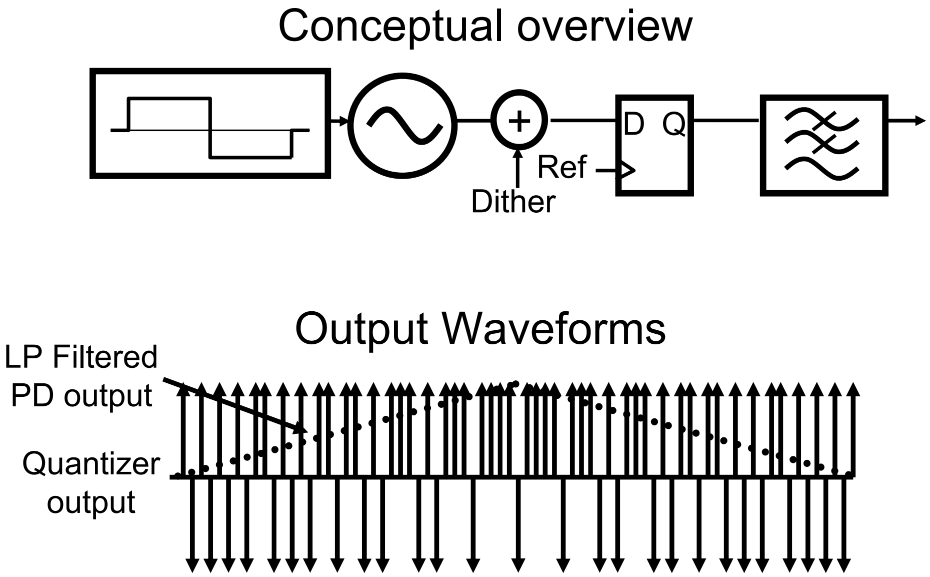

Figure 3.2: In this example VCO's input, and hence the excess phase at the output, increases and then decreases. The output of the phase quantizer is either 1 or -1, but after being low pass filtered it resembles the required phase information.

Calculating the gain involves two steps; calculating the appropriate phase detector small signal gain, K, and calculating the approximate amount of quantization noise. The starting point for this derivation is to look to ADC analysis for estimating quantizer gain and noise. We start by noting that the quantizer rounds the input phase offset to $+K_{pd}$ or $-K_{pd}$ where K_{pd} is a digitally programmable quantization word. The phase-quantizer's low-frequency equivalent gain can be defined as the ratio of the average phase detector output to the input phase difference. Note that this is an over-sampled system, the gain term to be derived is based on an average of the output taken over many samples, and hence it is only valid at low frequencies. The average phase quantizer output can be approximated with Equation (3.4) where p(1), p(-1) refers to the probability that the quantizer produces a 1 or -1respectively[7].

$$Out \cong K_{pd} \cdot (p(1) - p(-1)) = K_{pd} \cdot (2p(1) - 1) \tag{3.4}$$

And the phase detector gain, K, is given by Equation (3.5), where *Out* is the average output from Equation (3.4), and Φ is the phase from the VCO, after it has been divided but the divider :

$$K = \frac{\delta(Out)}{\delta(\phi)} \tag{3.5}$$

The next step is to discuss the chances of the quantizer producing a positive output, p(1) . The quantizer produces a positive output if the combination of the nominal phase difference Φ, and the ΣΔ quantization noise is positive. In other words the probability of the phase detector producing a 1 is given in Equation (3.6) and Equation (3.7), where $n_{\Sigma\Delta}$ is the noise added by the sigma delta modulator and $\text{pdf}_{\Sigma\Delta}$ is its probability density function at the quantizer input:

$$p(1) = p(\phi + n_{\Sigma\Delta} > 0) \tag{3.6}$$

or

$$p(1) = 1 - p(\int_{-\infty}^{-\phi} pdf_{\Sigma\Delta}(\phi)\partial\phi) \tag{3.7}$$

Thus the small signal gain is complicated by the fact that it is a function of the quantization noise added by the ΣΔ modulator. The next step is to approximate the

[7] Of course the flip-flops output is not literally a 1 or -1, its 1 or 0. However, when the flip-flop produces a 0, this is treated as a -1, so the quantizer output is +/-K_{pd}.

probability density function of the noise generated by the ΣΔ. The path from the sigma delta modulator to the quantizer is shown in Figure 3.3:

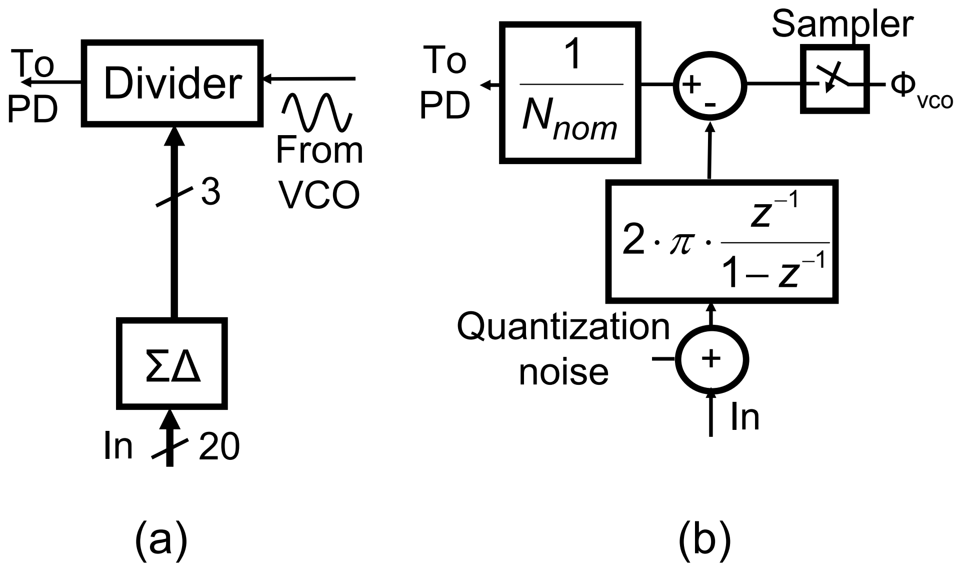

Figure 3.3: (a) The path from the ΣΔ to the quantizer input (b) The phase domain model, which includes an integrator in the divider's control path.

3.3 Estimating the amount of the ΣΔ noise at the quantizer input

Unfortunately a closed form expression for the probability density function of the ΣΔ noise is not available; instead some approximations are made based on simulation experiments. Firstly, a few observations are in order. According to Equation (3.7) the probability of the quantizer producing a one is a function of the *pdf* of the ΣΔ noise. To produce linear behavior, the *pdf* should be continuous and preferably uniform. However the actual output of a ΣΔ is quantized to a small number of fixed levels, therefore for a constant input to the ΣΔ the error (or noise) at the output is restricted to quantized levels. For example if the input to the ΣΔ is 0.73, the output is 1 or 0. In this case the quantization error will always be -0.73 or 0.13. However, what is needed is a noise source with a continuous pdf. Luckily a solution to this problem comes from the realization that the ΣΔ quantization noise passes through an integrator before reaching the phase quantizer, as shown in Figure 3.3(b). It is the pdf of the output of the integrator which is of interest. In order to approximate the pdf of the noise generated by the ΣΔ modulator a simulation experiment, shown in Figure 3.4, is performed.

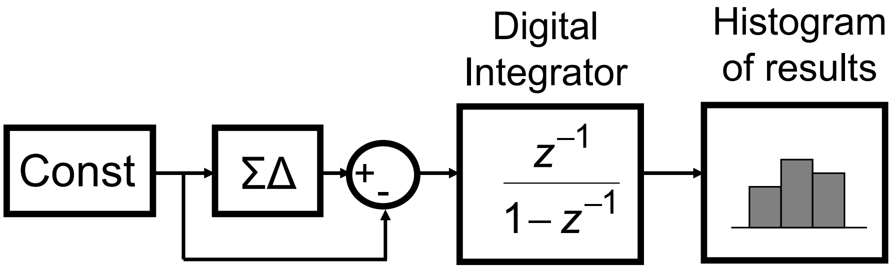

Figure 3.4: Simulation experiment setup. The input is subtracted from the output of a ΣΔ modulator, leaving only the error. This is then integrated, and plotted on a histogram.

In this experiment, the input to a ΣΔ modulator is set to a DC value. The DC value at the input is subtracted from the output, leaving just the error, or quantization noise. This noise is then passed through a digital integrator, which models the effects of the conversion from frequency control to phase control. The output of the integrator is sampled and the results are plotted in histogram form in Figure 3.5:

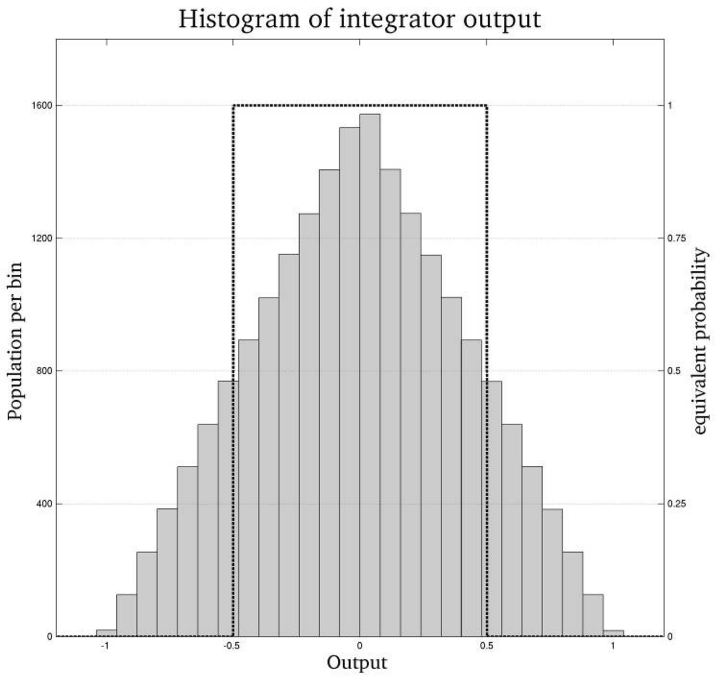

Figure 3.5 Histogram of samples from the integrator output.

On the left y-axis of Figure 3.5 is the population per bin. On the right y-axis is the population per bin expressed as its equivalent *pdf*. As can be seen the shape of the histogram is approximately triangular, not uniform as would be preferred. However the *pdf* can be approximated as a uniform value for small phase deviations, as part of a small signal analysis. (The approximation is shown in Figure 3.5 as the dotted line.) Note that it is the probability when then input is close to zero that is of most interest. Thus the *pdf* of the noise is approximated as having a constant value of 1, according to the right hand y-axis. As is the case in small signal analysis, this is a linear approximation of a non-linear transfer function, and is only valid for small phase deviations. According to Figure 3.3(b) in addition to the digital integration, the noise passes through a gain of 2π and is divided by N_{nom} before reaching the phase quantizer, and hence the *pdf* at the input to the quantizer has an approximate value of $N_{nom}/2\pi$, with a range of +/- π/N_{nom} (for any *pdf* the total area must be 1, so for a uniform *pdf* with a magnitude of *X*, then the range must be from +/- 0.5/*X*.). This can be combined with Equation (3.7) to get Equation (3.8).

$$p(1) = 1 - p(\int_{-\infty}^{-\phi} pdf_{\Sigma\Delta}(\phi)\partial\phi) \approx 1 - p(\int_{-\pi/N_{nom}}^{-\phi} \frac{N_{nom}}{2\pi}\partial\phi) \tag{3.8}$$

This can be simplified to give Equation (3.9).

$$p(1) \approx 0.5 + \phi \cdot \frac{N_{nom}}{2\pi} \tag{3.9}$$

Combining this with Equations (3.4) and (3.5) results in the approximate small signal gain.

$$K = K_{pd}\frac{N_{nom}}{\pi} \tag{3.10}$$

At this point a few comments are appropriate. The expression for small signal gain is calculated using a combination of derivations and simulation results. Because of the approximation in calculating the *pdf* of the noise, this expression is only valid for small phase deviations. However, as discussed in 2.7, the quantizer is nested in a feedback loop which includes an integrator; therefore the overall phase detector gain is desensitized to the quantizer gain. This is advantageous, as the derived gain can only be used as an approximation.

3.4 Transfer function of the other blocks

Now that a model for the gain of the phase detector has been developed, we turn our attention to some of the other blocks. The phase at the output of the VCO can be summarized with Equation (3.11).

$$\phi_{vco}(t) = 2\pi \cdot f_{nom}t + \phi_{out}(t) \tag{3.11}$$

Where f_{nom} is the nominal VCO frequency and Φ_{out} is the excess phase, given in Equation (3.12).

$$\phi_{out}(t) = \int_{0}^{t} 2\pi K_{vco} V_{in}(t)\partial t \tag{3.12}$$

The small signal model for the divider is shown in Figure 3.3. This can be summarized by observing that the phase at the output of the divider is given by Equation (3.13), which is derived in [37]. Here $n[i]$ is the i^{th} sample of the ΣΔ, $\Phi_{div}[k]$ is the phase of the k^{th} output of the divider, and $\Phi_{out}[k]$ is the excess phase of the VCO output.

$$\phi_{div}[k] = \frac{1}{N_{nom}}(-2\pi \sum_{m=1}^{k} n[m-1] + \phi_{out}[k]) \tag{3.13}$$

The small signal models for the divider, the phase detector, the VCO and the divider are put together to give a complete small signal model, as shown in Figure 3.6. The two loops are labeled Loop 1 (which is the PML) and Loop 2. Loop 1 consists of the phase detector, divider and ΣΔ, while Loop 2 consists of the digital integrator, the VCO, and Loop 2 also includes Loop 1.

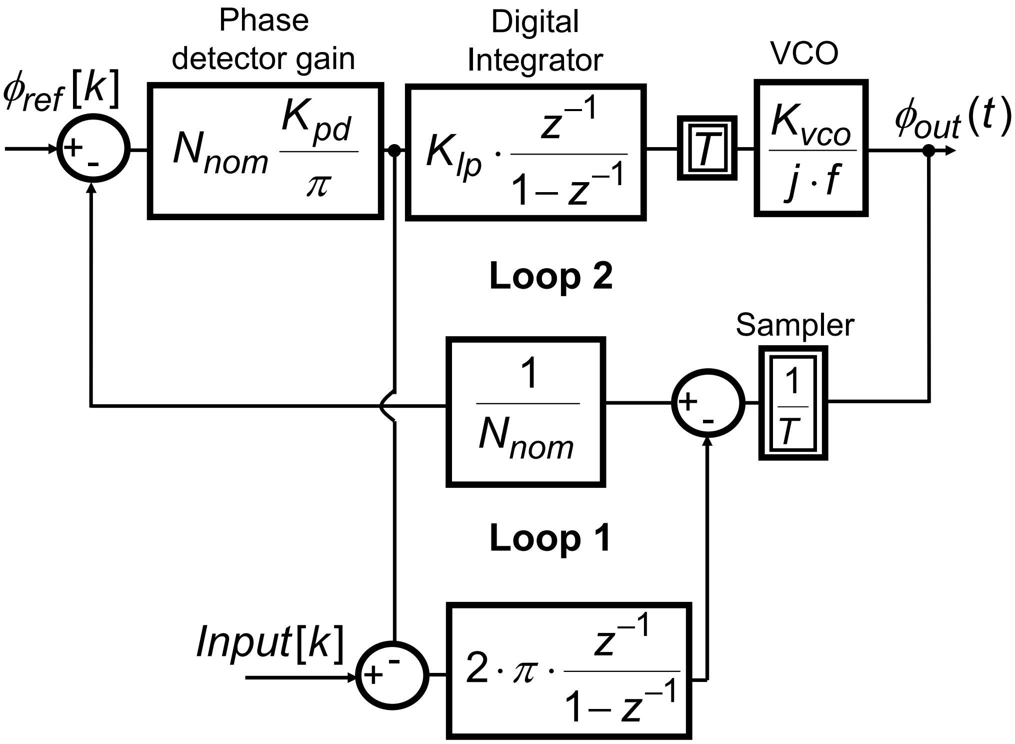

Figure 3.6: An overview of the small signal model for the FPLL.

As can be seen the model contains a mixture of continuous time and discrete time components. For simplicity, the poles at the output of the DAC are not included. These higher order poles are placed well outside the loop bandwidth, and hence have little effect on the loop settling characteristics (The effects of the poles are included in the numerical calculations in Chapter 6). To analyze Figure 3.6 pseudo-continuous analysis can be used, which involves modeling discrete time components as continuous time blocks. The relationship between the Fourier transform of a continuous time signal, *Y(f)*, and its sampled equivalent, *Ŷ(f)*, is given by Equation (3.14).

$$\hat{Y}(f)=\frac{1}{T}\sum_{k=-\infty}^{\infty}Y(f-\frac{k}{T}) \tag{3.14}$$

The discrete time transform consists of the continuous time transform replicated in frequency, and scaled by the sampling frequency. However the bandwidth of the loop must be a small fraction of the sampling rate, hence all replicas of the transform are suppressed, except for the fundamental. Therefore the transform from continuous domain to the sampled domain can be approximated by scaling the signals by 1/T. An analog model needs to be developed for the digital integrators. It has been shown in [37] that a digital integrator can be approximated using Equation (3.15).

$$\frac{z^{-1}}{1-z^{-1}} \approx \frac{1}{j2\pi fT} \quad for \quad f << \frac{1}{T} \tag{3.15}$$

Using this model the loop dynamics can be analyzed.

3.5 The loop transfer functions

In this section the transfer function of each loop will be derived. In Figure 3.6 the two loops are labeled Loop 1 and Loop 2. The open loop ($A_1(f)$) and closed loop ($G_1(f)$) transfer functions for Loop 1 are given by Equation (3.16) and Equation (3.17).

$$A_1(f) = \frac{1}{N_{nom}} \cdot 2\pi \frac{z^{-1}}{1-z^{-1}} \frac{N_{nom} K_{pd}}{\pi} \approx \frac{K_{pd}}{j\pi fT} \tag{3.16}$$

$$G_1(f) = \frac{A_1(f)}{1+A_1(f)} \quad \left(\cong 1 \, for \; f << \frac{K_{pd}}{j\pi fT} \right) \tag{3.17}$$

The unity gain frequency of this loop can be controlled using K_{pd} as shown in Equation (3.18):

$$|A_1(f)| = 1 \,@\, f \approx \frac{K_{pd}}{\pi T} \tag{3.18}$$

The next step is to focus on Loop 2, G$_2$(f). For this, the inner loop is replaced with its transfer function, from the divider input to the quantizer output. This is given by Equation (3.19).

$$\frac{PDout}{Divider\_in} = G_1(f) \Big/ 2\pi \frac{z^{-1}}{1-z^{-1}} \cong G_1(f)jfT \tag{3.19}$$

A block diagram for Loop 2 is shown in Figure 3.7. Loop 1 has been replaced by its equivalent transfer function, from Equation (3.19). As will be explained in Section 3.9, the phase of the reference signal is gained by before being inserted into the loop.

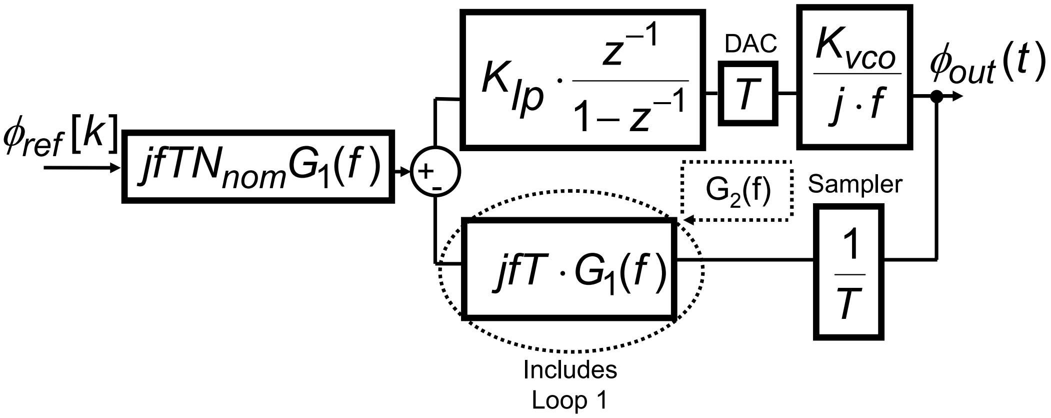

Figure 3.7: Overview of small signal model for outer loop (Loop 2), with Loop 1 replaced by its equivalent transfer function.

Similar to the inner loop the open loop gain is given by Equation (3.20).

$$A_2(f) = \frac{K_{vco}}{jf} \cdot \frac{1}{T} \cdot jfTG_1(f) \cdot K_{lp} \frac{z^{-1}}{1-z^{-1}} \cdot T \tag{3.20}$$

Using Equation (3.15) this can be simplified to the transfer function given by Equation (3.21). This results in the closed loop transfer function given in Equation (3.22).

$$A_2(f) \approx \frac{G_1(f)K_{vco}K_{lp}}{j2\pi f} \tag{3.21}$$

$$G_2(f) = \frac{A_2(f)}{1 + A_2(f)} \tag{3.22}$$

Equations (3.16), (3.17), (3.21) and (3.22) can be combined to get the overall transfer function given by Equation (3.23).

$$G_2(s) = \frac{\frac{2}{T}K_{pd}K_{vco}K_{lp}}{s^2 + s\frac{2K_{pd}}{T} + \frac{2K_{pd}K_{vco}K_{lp}}{T}} \tag{3.23}$$

This can be put in a more traditional form, to give (3.24).

$$G_2(s) = \frac{{\omega_n}^2}{s^2 + s2\zeta\omega_n + {\omega_n}^2} \tag{3.24}$$

where the natural frequency is given by Equation (3.25), and the damping factor is given in Equation (3.26).

$$\omega_n = \sqrt{\frac{2}{T}K_{pd}K_{vco}K_{lp}} \tag{3.25}$$

$$\zeta = \sqrt{\frac{K_{pd}}{2TK_{vco}K_{lp}}} \tag{3.26}$$

Notice that Equation (3.24) contains no zeros, and hence it can be designed to be completely free of gain peaking, as long as ζ>0.707. This is an unusual property for a PLL, and will be discussed further in Section 3.6.

3.6 Gain (and Jitter) peaking

In Section 3.5 the closed loop response of the loop is derived, as given in Equation (3.24). Notice that this equation is all pole, and there are no zeros. This is in contrast to most second order PLLs, which typically have a closed loop response as given by Equation (3.27).

$$G(s) = \frac{2\zeta\omega_n s + \omega_n^{2}}{s^2 + 2\zeta\omega_n s + \omega_n^{2}} \tag{3.27}$$

In a Type II second order PLL the zero is required to stabilize the loop. In addition to stabilization, the zero also causes gain peaking [48].

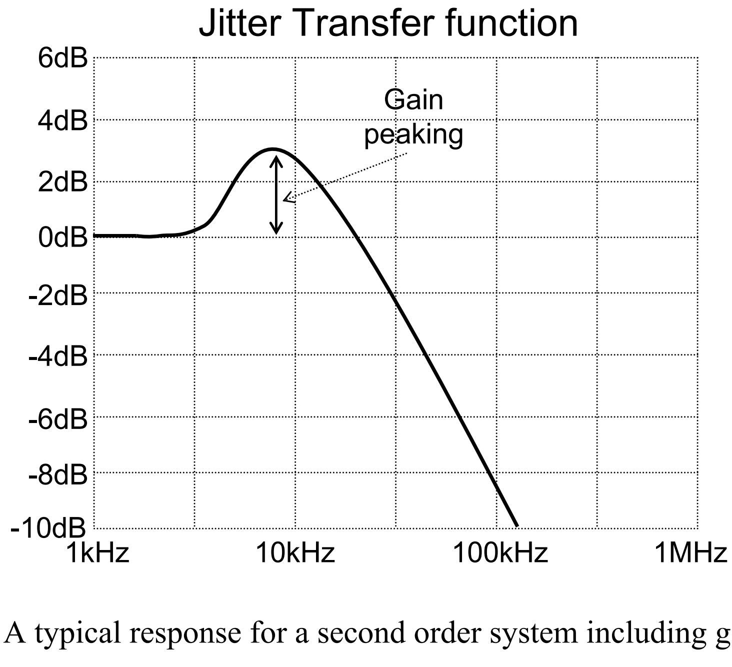

Figure 3.8: A typical response for a second order system including gain peaking.

If gain peaking is present, then noise within a small bandwidth undergoes some amplification, which is undesirable in applications which require a number of PLLs to be placed in series, such as repeaters. (For this reason gain peaking is often referred to as jitter peaking) Gain peaking can be reduced by increasing the damping factor, ζ, but for a second order Type II system with a stabilizing zero the gain peaking can never be fully eliminated. In series repeaters jitter can grow exponentially due to jitter peaking [50]. In [51] and [53] a dual loop architecture is used to remove jitter peaking. While the architecture in [51] is very different to this work, analysis of the two transfer functions reveals the dual loop has the same effect on gain/jitter peaking. This architecture, summarized in Figure 3.9, uses two loops, and is named a Delay- and Phase-Locked Loop (D/PLL). The idea is that each of the loops performs a different function. The Delay Locked Loop (DLL) forces the phase of the fedback signal to be identical to that of the incoming signal, using a Voltage Controlled Phase Shifter (VCPS). As with any DLL, lock can only be achieved if the DLL clock has identical frequency to that of the incoming signal. The second loop is a PLL, which indeed forces the frequency of the feedback signal to be identical to that of the inner loop.

In this type of PLL the incoming signal is a raw data stream, which in effect is a random stream of ones and zeros. Part of the motivation for this type of architecture is to break the link between jitter tolerance, the amount of allowable jitter on the input signal that the PLL can tolerate with out loosing lock, and jitter rejection, the amount of filtering the PLL can do. This type of dual loop architecture breaks this link.

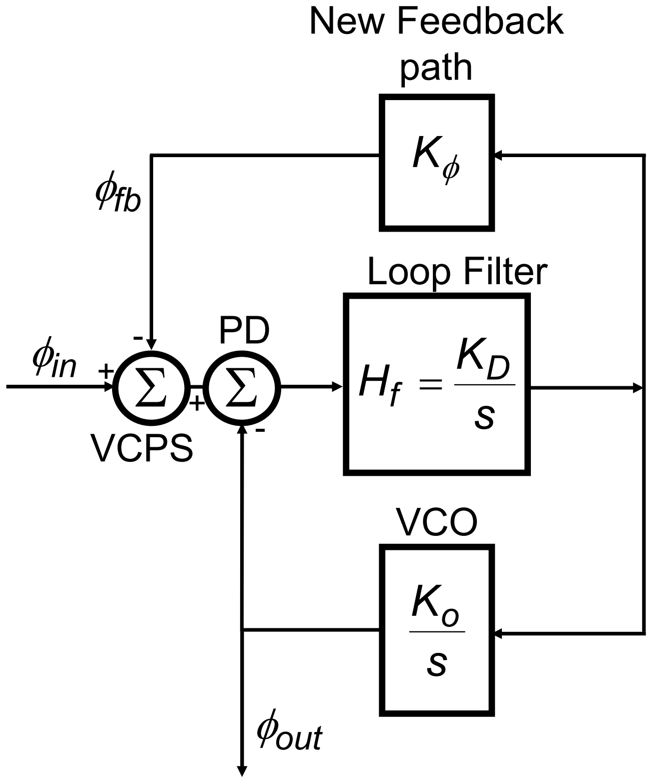

Figure 3.9: Simplified architecture for a D/PLL. There are two feedback paths, one through the VCO, and the other through the Voltage Controlled Phase Shifter (VCPS).

As shown in [51] the phase relationship between input and output is given by Equation (3.28).

$$H(s) = \frac{\phi_{out}}{\phi_{in}} = \frac{K}{s^2 + K\tau s + K} \tag{3.28}$$

Where $K=K_D K_O$ and $\tau=K_\Phi/K_O$ in Figure 3.9. Typically if a feedback system contains two integrators, then there must also be a stabilizing zero. However, if the loop is analyzed in terms of the total feedback signal, then there would be a stabilizing zero, such as in Equation (3.29).

$$\frac{\phi_{out} + \phi_{fb}}{\phi_{in}} = \frac{K + K\tau s}{s^2 + K\tau s + K} \tag{3.29}$$

It is the relationship between input and output in Equation (3.28) which is of importance.

Although this system appears to be very different to Figure 3.6, if Figure 3.6 is redrawn in Figure 3.10 it can be seen that the two systems have the same form, and hence should have the similar jitter transfer functions. In Figure 3.6 there are two different digital integrators. However both integrators have inputs connected to the output of the phase detector, therefore in Figure 3.10 these integrators can be drawn as one. The major difference between Figure 3.10 and Figure 3.9 is the $1/N_{nom}$ in the feedback path of Figure 3.10. This is because the frequency of the input and outputs are identical in [51], there is no divider.

In [51] (and Figure 3.9) the VCO gain K_O has the units Radians per Volts. In this work VCO gain, K_{vco}, is expressed in Hz per Volt, which explains the slightly different form of Figure 3.10. For more details on notation conventions see [48].

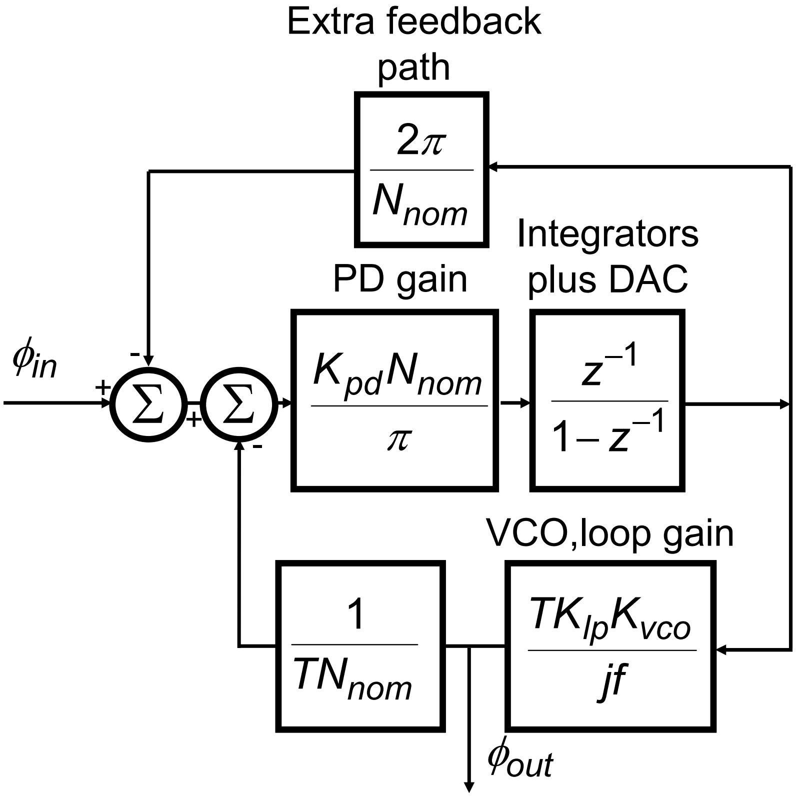

Figure 3.10: Redrawn version of new architecture. The digital integrator represents both the integrator at the output of the flip-flop, and the integrator caused by the divider.

3.7 Additional comments on PLL Type

As discussed in Section 3.5, Loop 2 is a Type I feedback system (it contains a single net integrator), while most PLL's are Type II or higher. The reason that most PLL's must be Type II is because of a subtle DC range issue at the output of the phase detector [52], [48]. For example if the control voltage of a VCO needs to move from 0V to 1V to achieve lock, then in a Type I PLL this must mean the output of the loop filter must be able to reach 1V. The maximum phase difference that can be tolerated at the input of a phase detector is +/-π/2, phase differences greater than this will case cycle slips. Therefore if the output of the loop filter is to reach 1V without cycle slipping, then the product of the loop filter and phase detector gain must be at least 2/π (Volts/radian). On the other hand, the gain also sets the PLL bandwidth. For a Type I PLL, a high gain is required to achieve a reasonably wide tuning range, and a low gain is required to limit the loop bandwidth. For example, if the VCO tuning range is 100MHz/V, the division ratio is 20, and the loop filter/phase detector gain is 2/π, then the loop bandwidth will be at least 100e6/10*(2/π)=3.1847MHz, which is too large for many applications. If a second integrator is used, as in a Type II system, then the net phase difference at the input of the phase detector will be forced to zero, and the bandwidth can be set independently of the phase detector range.

In this work, the phase detector (or phase quantizer) is inside a local feedback loop (Loop 1). The feedback has the effect of differentiating the phase detector's output, to recover the required phase difference information; a digital integrator is placed at the output of Loop 1, as seen in Figure 3.6. The local feedback loop (Loop 1) forces the average phase difference at the input of phase quantizer to zero, irrespective of the value of the output of the digital integrator. For this reason Loop 2 need only be Type I.

3.8 Methodology for calculating output phase noise

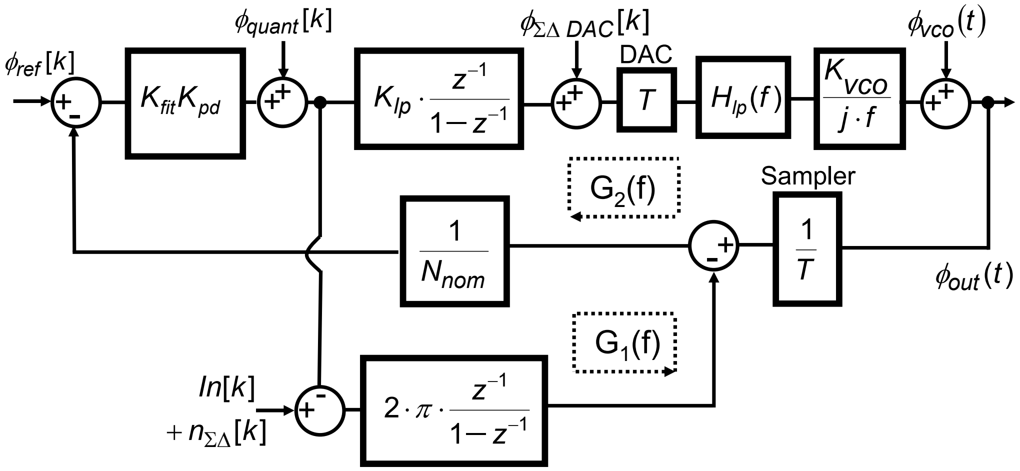

Figure 3.11: Small signal model incorporating all of the noise sources which are considered.

In Figure 3.11 the small signal model is redrawn, with all of the noise sources added. Some of these noise sources are discrete time, and some of them are continuous. In this section a method of calculating the noise at the output will be discussed. [37] introduced a method for calculating the noise at the output based on the loop transfer function, G(f). The loop transfer function can be used as part of a compact formulation for any of the transfer functions, and hence is very useful. For example consider the feedback system shown in Figure 3.12.

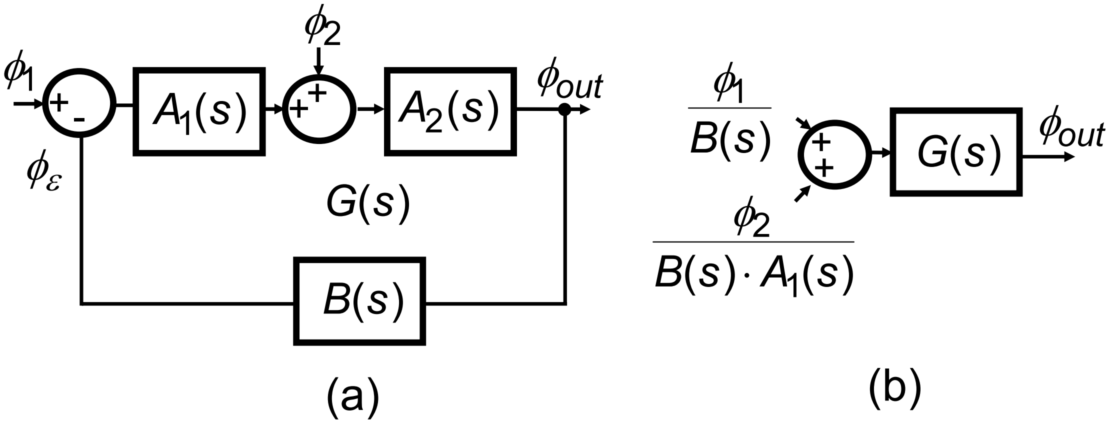

Figure 3.12: A brief review of feedback systems with multiple inputs. (a) System overview with two inputs. (b) Representing each of the transfer functions in terms of G(f).

First, we quickly review how to deal with feedback systems with multiple inputs, similar to [37]. In Figure 3.12 there are two inputs, Φ_1 and Φ_2 which both affect the output. Now suppose that the loop transfer function has already been calculated as G(s). By definition Φ_ε/Φ_1=G(s), and therefore Φ_{out}/Φ_1=G(s)/B(s). It is also desirable to know the transfer function for Φ_2 in terms of G(s). The transfer function for $\Phi_1$is used to help derive the transfer function for Φ_2. We start by determining what signal could be added to Φ_1 to have the same effect on the output as Φ_2 has. The answer is Φ_2 divided by the transfer function from Φ_1 to Φ_2. If follows that Φ_{out}/Φ_2= G(s)/(B_1(s)A_1(s)). In this way the transfer function for a signal added to the loop at any point can be easily calculated in terms of G(s). In this example the transfer functions are straight forward to calculate, however this method can simplify the derivation for more complex loops with many inputs.

Next consider the more complex scenario of using two loops, labeled Loop 1 and Loop 2 in Figure 3.13(a). Again there are two inputs Φ_1 and Φ_2. How should the transfer function from an insertion point in Loop 1 to the output be calculated? The derivation takes place in two steps. Firstly we calculate the transfer function from an insertion point in Loop 1 to an insertion point in Loop 2. Secondly we calculate the transfer function to get from the insertion point in Loop 2 to the output. As can be seen in Figure 3.13(a), Φ_1 is added to the Loop 1. The transfer function between Φ_1 and Φ_2 can be calculated, so that both signals are inserted into the loop at the same point in Loop 2 as seen in Figure 3.13(b). Next the transfer function from Loop 2 to the output can be calculated in terms of $G_2(s)$ as in Figure 4.3(c).

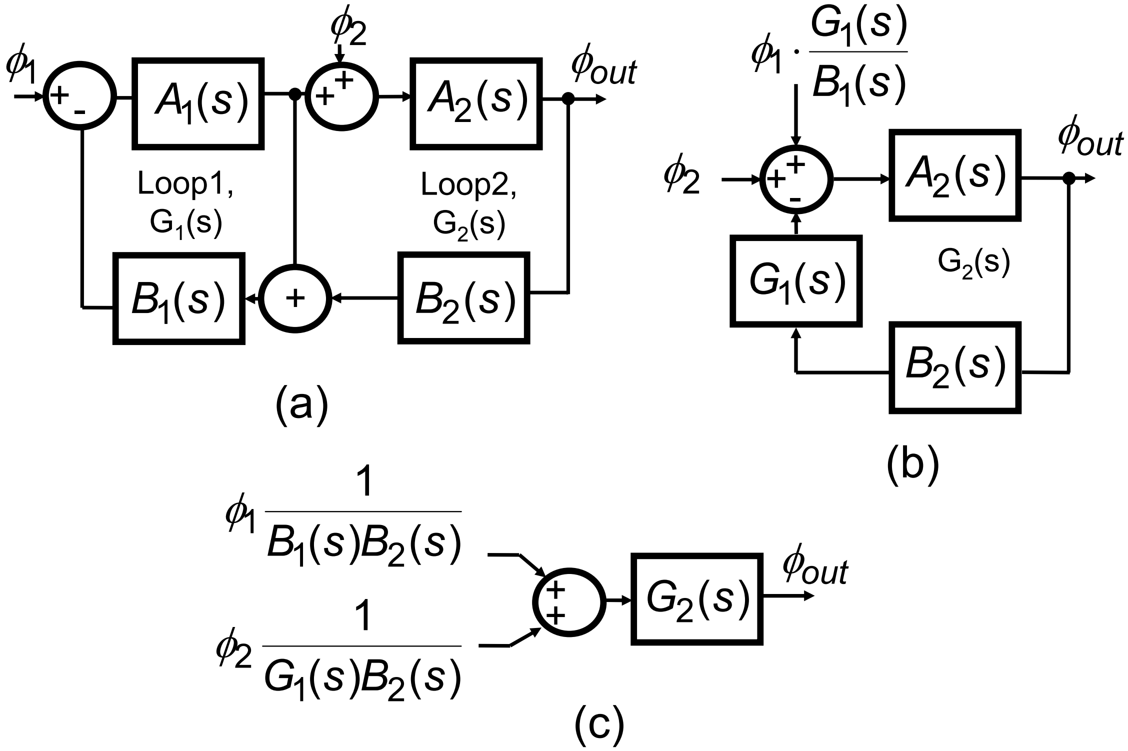

Figure 3.13: Calculating the output phase noise for a dual loop (a) System overview (b) Converted into a single loop (c) overall transfer function, simplified.

To summarize, to calculate the noise at the output caused by a noise signal inserted into the inner loop, two steps are required. The first step is to calculate the transfer function from the insertion point to the output of the first loop. The second step is to calculate the transfer function from this point to the output. This method is used to calculate the phase noise at the output due to each of the noise sources. The transfer functions G_1(s) and G_2(s) have been are calculated in Section 3.5.

3.9 Reference transfer function

In this section the loop gain equations are used to derive the gain between the reference input (Φ_{ref} in Figure 3.11) and the output phase. This derivation takes place in two parts. First the transfer function (TF1) between the reference input and the output of the inner loop is calculated, followed by the transfer function between the output of the inner loop to the output of the outer loop (TF2). This can be summarized in Equation (3.30).

$$\phi_{out(ref)}(f) = TF1 \cdot TF2 \cdot \phi_{ref}[k] \tag{3.30}$$

$\Phi_{out(ref)}(f)$ is the phase change at the output due to $\Phi_{ref}[k]$ (This is a discrete time source which can be approximated as a continuous time source with a value of T $\Phi_{ref}[k]$). It can be shown from Figure 3.11 that TF1 is given by Equation (3.31), and TF2 is given by Equation (3.32), where FB_{TF1} refers to the elements in the feedback path from the output of Loop 1 to the references, and FB_{TF2} refers to the feedback path from the output of Loop 2 to the output of Loop 1.

$$TF1 = G_1(f)\frac{1}{FB_{TF1}} = G_1(f)N_{nom}jfT \tag{3.31}$$

$$TF2 = \frac{G_2(f)}{FB_{TF2}} = G_2(f)\frac{T}{jFT \cdot G_1(f)} = \frac{1}{jf}\frac{G_2(f)}{G_1(f)} \tag{3.32}$$

Equations (3.30), (3.31) and (3.32) can be combined to get

$$\frac{\phi_{out(ref)}(f)}{\phi_{ref}[k]} \cong G_2(f)N_{nom}T \tag{3.33}$$

Or if TΦ_{ref}[k] is expressed as a continuous time source, Φ_{ref}(f).

$$\frac{\phi_{out(ref)}(f)}{\phi_{ref}(f)} \cong G_2(f)N_{nom} \tag{3.34}$$

As would be expected, the output phase is equal to the input phase multiplied by the nominal division ratio, and filtered by the outer loop transfer function G_2(f).

3.10 Divider ΣΔ quantization noise

The same methodology can be used to calculate the relationship between the divider's ΣΔ and the output. Although the methodology is the same as is used in Section 3.9, in this case TF1 and TF2 have different values.

$$\phi_{out(\Sigma\Delta)}(f) = TF1 \cdot TF2 \cdot n_{\Sigma\Delta}[k] \tag{3.35}$$

$\Phi_{out(\Sigma\Delta)}$(f) is the phase change at the output due to the ΣΔ noise, $n_{\Sigma\Delta}[k]$. The two new transfer functions are given in Equations (3.36) and (3.37).

$$TF1 = G_1(f) \tag{3.36}$$

$$TF2 = G_2(f)\frac{T}{jf \cdot T \cdot G_1(f)} = \frac{1}{jf}\frac{G_2(f)}{G_1(f)} \tag{3.37}$$

These can be put together get Equation (3.38).

$$\frac{\phi_{out(\Sigma\Delta)}(f)}{T \cdot n_{\Sigma\Delta}[k]} = \frac{G_2(f)}{jfT} \tag{3.38}$$

This shows that the ΣΔ noise is integrated, and filtered by the loop G$_2$(f). This result is typical for FPLLs. Because of this integration the order of the ΣΔ is typically chosen to be at least one more than the order of the FPLL, as the integration will subtract one from the order of the ΣΔ noise shape.

3.11 Phase detector flip-flop quantization noise

Once again, the same methodology is used. First calculate transfer function from the output of phase detector to the output of the first loop, and from there calculate the transfer function to the output. The noise at the output due to the phase detector is given by Equation (3.39).

$$\phi_{out(quant)}(f) = TF1 \cdot TF2 \cdot \phi_{quant}[k] \tag{3.39}$$

Φ$_{out(quant)}$(f) is the phase change at the output due to *Φ*$_{quant}$[*k*]. Similar to before TF1 and TF2 are given by Equations (3.40) and (3.41).

$$TF1 = 1 - G_1(f) \tag{3.40}$$

Reducing the bandwidth of the loop is an effective method of reducing phase quantizer noise.

$$TF2 = G_2(f)\frac{1}{FB_{TF2}} = \frac{1}{jf}\frac{G_2(f)}{G_1(f)} \tag{3.41}$$

Equation (3.39), (3.40) and (3.41) can be combined to show that the noise is filtered by the loop G$_2$(f). 7

$$\frac{\phi_{out(quant)}(f)}{T \cdot \phi_{quant}[k]} = \frac{\pi \cdot G_2(f)}{K_{pd}} \tag{3.42}$$

Reducing the bandwidth of the loop is an effective method of reducing phase quantizer noise. It also appears at first glance that increasing K_{pd}, will decrease the noise. However, changing K_{pd} affects both $G_2(f)$ and the amount of quantization noise that is produced. Also, notice that this transfer function is the same as for the reference input, but scaled by the gain of the phase quantizer as would be expected.

3.12 VCO phase noise

The VCO phase noise as seen at the output is given in Equation (3.43). At low frequencies the VCO noise will be suppressed by the PLL. At higher frequencies, well outside the loop bandwidth, the VCO noise will be unfiltered.

$$\phi_{out(VCO)}(f) = \phi_{VCO}(t) \cdot \left(1 - G_2(f)\right) \tag{3.43}$$

3.13 DAC ΣΔ noise

Similarly it can be shown that the transfer function for the DAC quantization noise is given by Equation (3.44).

$$\phi_{out(DAC\_\Sigma\Delta)}(f) = \frac{2\pi G_2(f)}{K_{lp} G_1(f)} \cdot \phi_{DAC\_\Sigma\Delta}(f) \tag{3.44}$$

3.14 Summing the noise sources to get the total output phase noise

The next step is to add these noise sources together. Some of the noise sources in the FPLL are continuous time sources and others are discrete time. The output noise caused by a continuous time noise source can be calculated using Equation (3.45), where $S_{in}(f)$ is the input noise nose power, and $H(f)$ is its transfer function from the input to the output.

$$S_{out}(f) = \left|H(f)\right|^2 S_{in}(f) \tag{3.45}$$

The output noise caused by a discrete time noise source can be calculated using Equation (3.46).

$$S_{out}(f) = \frac{1}{T}\left|H(f)\right|^2 S_{in}(e^{j2\pi fT}) \tag{3.46}$$

Next, the values of the actual noise sources are considered. The VCO noise is measured from the silicon prototype, with the PLL turned off. The divider's ΣΔ noise is given by Equation (3.47), which is the shape for quantization noise [46], where *m* is the order of the ΣΔ (*m*=0 if there is no noise shaping).

$$S_{in\_\Sigma\Delta}(f) = \left|\left(1 - e^{-2\pi fT}\right)^m\right| \cdot \Delta^2 / 12 \tag{3.47}$$

The noise at the output due to the divider's ΣΔ this noise source is given by Equation (3.48).

$$S_{out\_\Sigma\Delta}(f) = \frac{1}{T}\left|\frac{G_2(f)}{f}\right|^2 S_{in\_\Sigma\Delta}(f) \tag{3.48}$$

The same calculation can be done for the phase detector noise, with *m*=0, and $\Delta=2K_{pd}$. The noise from the DAC is similar to Equation (3.47), except the step size is equal to $\Delta=1/2^{\text{bits}}$, where "bits" refers to the resolution of the DAC, and *m*=1 as this ΣΔ is a first order system. The noise source powers can be summed to generate the total output noise power.

$$S_{out}(f) = S_{out\_\Sigma\Delta}(f) + S_{out\_pd}(f) + S_{out\_vco}(f) + S_{out\_dac}(f) \tag{3.49}$$

The calculated phase noise at the output is compared with the phase noise measured from a prototype in Chapter 6.

CHAPTER 4

HIGH SPEED FREQUENCY MODULATION

4.1 Introduction

A limitation of the Fractional-N PLL architecture is the difficulty of achieving high-speed modulation of the RF output frequency/phase without widening the loop bandwidth. The ΣΔ modulator, used to control the divider, injects high-pass-shaped quantization noise into the loop, but the PLL itself has a low bandwidth, which naturally rejects the high frequency portion of the ΣΔ noise. Similarly, the PLL's low bandwidth filters high frequency jitter on the reference clock. There are several other noise sources which can potentially benefit from a low bandwidth PLL, such as phase detector noise, divider noise, etc. If rejecting all of theses noise sources is critical, then a low bandwidth PLL is required. On the other hand, the average divide ratio should be modulated at a rate less that PLL bandwidth, otherwise the modulation signal is filtered by the PLL's low pass characteristic. If the PLL bandwidth is widened to accommodate faster modulation, any reference jitter or ΣΔ quantization noise undergoes less filtering, and this can degrade the phase noise of the output signal.

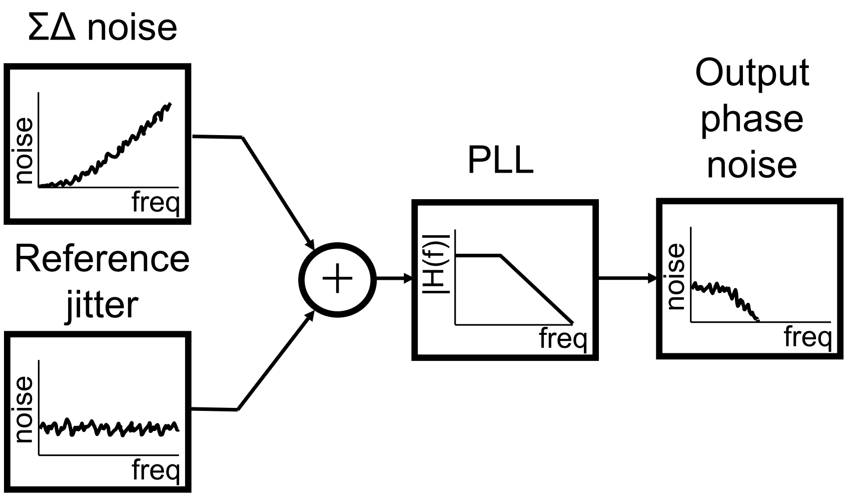

Figure 4.1: A PLL is often used to filter noise sources. In particular, high frequency noise sources such as ΣΔ noise can be effectively filtered by the PLL's low loop bandwidth.

4.2 Comparison with other techniques

Several techniques have been proposed to overcome these shortcomings; however all of these have limitations. In pre-emphasis, the frequency control signal is passed through a transfer function that is the inverse of the PLL's, in order to compensate for the effects of the PLL filtering. This technique can compensate for the limited loop bandwidth [29],[30],[31] but requires knowledge of the loop's frequency response. The pre-emphasis filter is implemented as a digital filter which represents the inverse of the analog filtering characteristics of the PLL. If the loop characteristics are not exactly known, then the information to be transmitted undergoes distortion. Typically PLLs contain a significant number of blocks that can vary significantly due to process, voltage and temperature variations (PVT) and therefore it is difficult to predict a priori exactly what the loop characteristics will be.

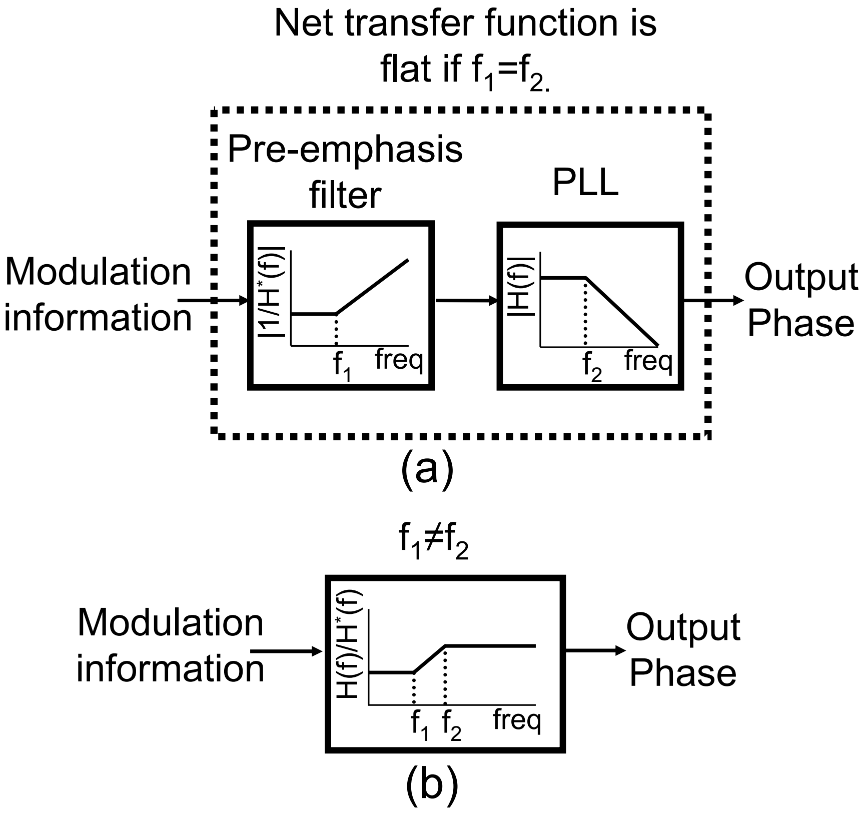

Figure 4.2: (a) Overview of pre-emphasis, where a digital filter is used to estimated the approximate inverse of the PLL characteristics. (b) Exact knowledge of the loop is needed, or the filters will mismatch.

For example in Figure 4.2(a) an overview of the concept is shown. In this case the PLL is modeled as having a single pole. In practice, PLLs can have several poles and zeroes, which significantly complicates the cancellation scheme. As long as f_1 is equal to f_2 then there is no distortion of the modulation data. In Figure 4.2(b) the net transfer function is shown, this time with f_1 mismatching with f_2. As can be seen this results in a slight magnification of the signal at higher frequency.

Another method is two-point modulation, where in addition to modifying the divide ratio, a modulation signal is added directly at the input of the VCO [32]. The concept overview is shown in Figure 4.3. If the frequency control word is changed then this will eventually lead to a change in VCO control voltage. If the gain of the VCO is exactly known, then in addition to introducing a signal into the divide ratio control, a signal is added directly at the VCO input which will cause no net disturbance in the loop. This can be explained as follows. The signal added to the input VCO causes an increase in the frequency of the output signal. The same signal is added to the input of the divider, which causes the divider to divide by a larger number. If the gain of the two paths is matched, then the frequency and phase of the output of the divider does not change, the increase in the frequency of the VCO output is exactly matched by the increase in the division ratio.

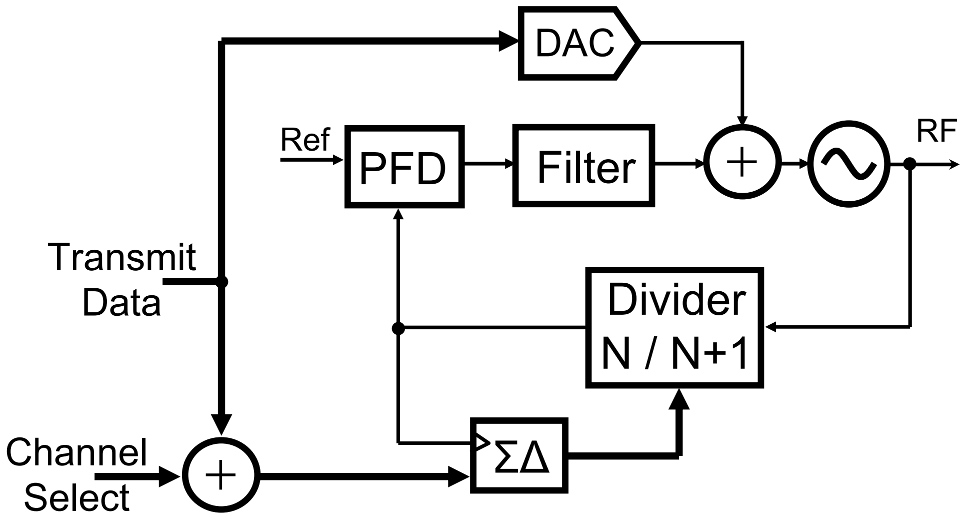

Figure 4.3: Overview of two point modulation. The transmit data is added both to the input to the ΣΔ and to the input to the VCO.

However, this system suffers from similar problems to pre-emphasis. Unless the value of the injected signal exactly corresponds to the VCO gain, then the signal will undergo distortion. Least mean square (LMS) based calibration techniques show much promise, but these can also have very slow associated time constants [45].

4.3 The proposed scheme

As described above, the requirement for a wide fractional-N PLL loop-bandwidth for high-speed modulation, conflicts with the low loop bandwidth favored for suppression of ΣΔ noise, phase detector noise, and reference jitter. In this section, a new technique which breaks the link between loop-bandwidth and the modulation rate of a frequency-shift-keying scheme is presented. In a standard FPLL, the output frequency is modified by changing the division ratio control, eventually leading to a change in the VCO control voltage. In the proposed scheme, knowledge of the exact loop dynamics is not required. Instead, using some simple signal processing, the required change at the input of VCO is learned from previous changes in VCO control in response to changes in the modulator input. In summary, this is a form of two point modulation with a simple calibration scheme.

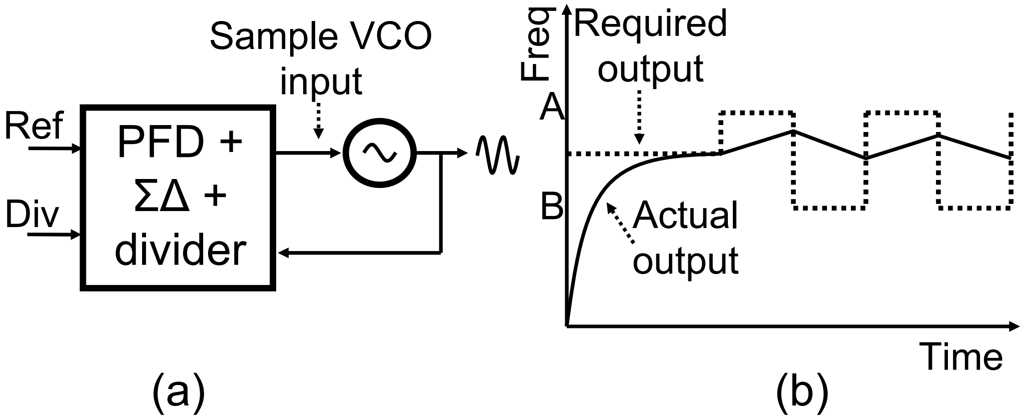

Figure 4.4 (a): Simplified PLL. Input to VCO is analogous to the output frequency. (b): Plot of required output versus actual output.

Consider the case where the loop is to switch between two different frequencies, A and B, such as for 2-FSK, and assume that the loop frequency is initially settled at the average of A and B. If the bandwidth of the loop is not large enough in a standard FPLL then the frequency never settles correctly to A or B (Figure 4.4(b)). In Figure 4.4(a) a simplified version of the PLL is shown. The input to the VCO can be considered to be analogous to the frequency of the output signal, and if the output frequency is to switch instantaneously then the input to VCO must also switch instantaneously. For example if the output frequency is to change by a frequency Δf, then the corresponding voltage change at the input of the VCO is given in equation (4.1).

$$\Delta V = \frac{\Delta f}{K_{vco}} \tag{4.1}$$

Therefore, if the correct step is to be added to the input of the VCO then the gain of the VCO must be known. In the proposed scheme, at the end of each bit period the digital value that determines the VCO input is sampled. When switching between the two desired frequencies, A and B, the most recently sampled values are subtracted from each other.

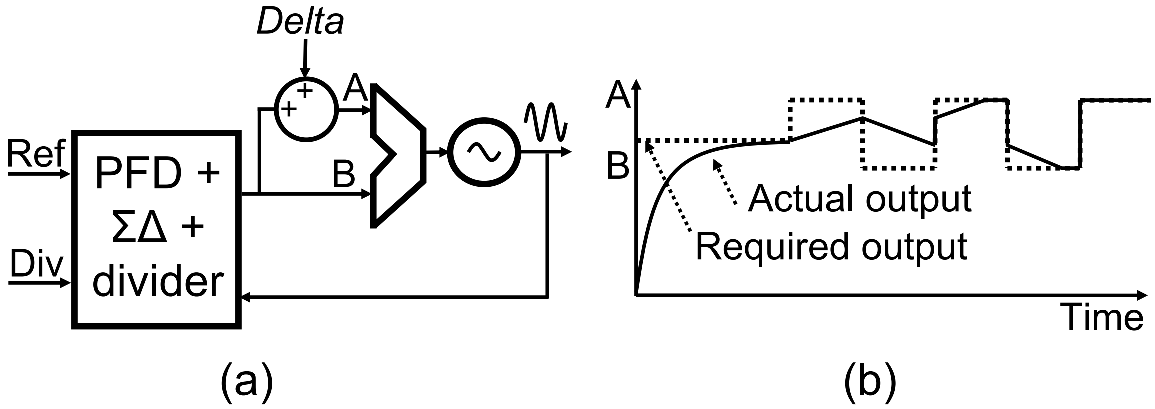

Figure 4.5 (a): PLL with Delta added. (b): Plot of output frequency.

The result of this subtraction ("*Delta*" in Figure 4.5(a)) is added to the VCO control for frequency A, to give an initial digital VCO control value for frequency B.

To analyze the settling behavior consider the diagram in Figure 4.6. We start by breaking the frequency switching segments into pairs, the first segment in the pair is when the PLL attempts to switch to frequency A, and in the second segment the PLL attempts to switch to frequency B. The value of *Delta* is updated twice in each cycle, on the transition from A to B, and again on the transition from B to A. Therefore there are two values of delta in each cycle, $Delta_a$ and $Delta_{\mathrm{b}}$. The two values within each cycle are given by Equations (4.2) and (4.3).

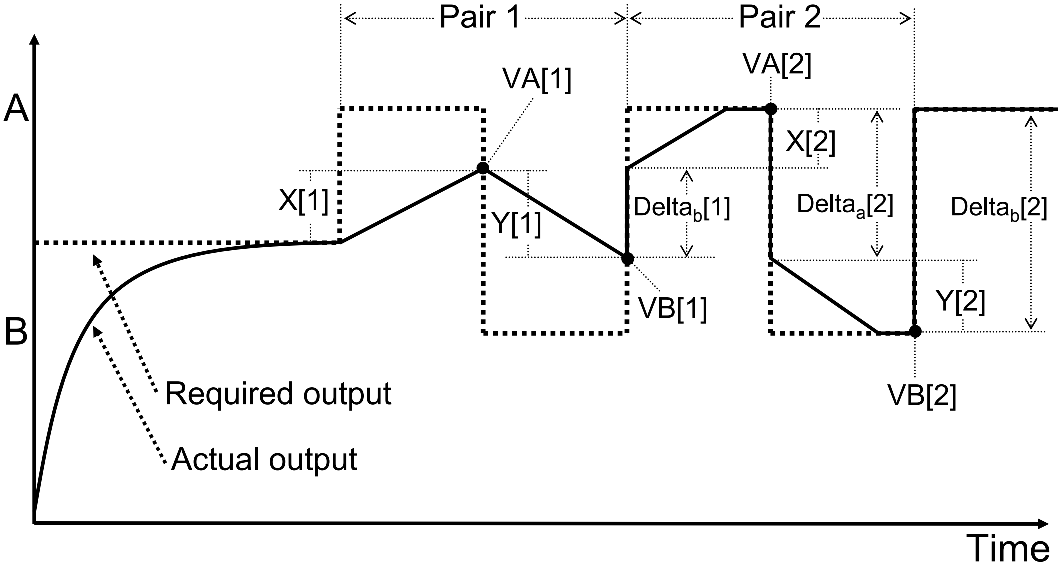

Figure 4.6: Detailed timing diagram of the settling characteristics of the loop.

$$Delta_a[i] = VA[i] - VB[i-1] \tag{4.2}$$

$$Delta_b[i] = VA[i] - VB[i] \tag{4.3}$$

Where *VA*[*i*] corresponds to the VCO input when the transmitter attempts to switch from frequency A to B for the i^{th} time index (i.e. at the end of a bit period for frequency A), and similarly *VB*[*i*] corresponds to the VCO input when the transmitter attempts to switch from frequency A to B for the i^{th} time index. Also, *X*[*i*] refers to the amount that PLL has change the VCO input during the i^{th} time that A is the required output frequency. In other words it is simply the results of the normal settling behavior of the loop, as shown on Figure 4.6. Similarly, *Y*[*i*] is the amount that the VCO input has changed when B is the required output frequency. The relationship between *VA*, *VB*, $Delta_a$ and $Delta_b$ is given in Equations (4.4) and (4.5).

$$VB[i] = VA[i] - Delta_a[i] - Y[i] \tag{4.4}$$

$$VA[i] = VB[i-1] + Delta_b[i-1] + X[i] \tag{4.5}$$

We can combine Equations (4.2), (4.3), (4.4), and (4.5) to get Equations (4.6) and (4.7).

$$VB[i] = VB[i-1] - Y[i] \tag{4.6}$$

$$VA[i] = VA[i-1] + X[i] \tag{4.7}$$

As shown in Equation (4.7) , the current value of *VA* is equal to the previous value of *VA* plus *X*, as explained above, which is the amount the loop has changed the VCO input as the loop attempts to settle to frequency A. In other words *X*[*i*] is the loop's attempt to get from *VA*[*i*-1] to the value of the VCO input for frequency A. While the loop is not able to settle fully in one cycle, as long as *X*[*i*] has the correct polarity (i.e. the loop has moved in the right direction) then *VA*[*i*] will be closer to the correct value than *VA*[*i*-1] was. In practice it only takes a couple of cycles before the loop fully converges. Notice that *Delta* is only updated on the edges of data transition, therefore in the absence of data transitions, the new switching scheme has no effect on the loop. Furthermore the switching scheme does not effect the loop dynamics, and does not add extra poles or zeros, as LMS based schemes can.

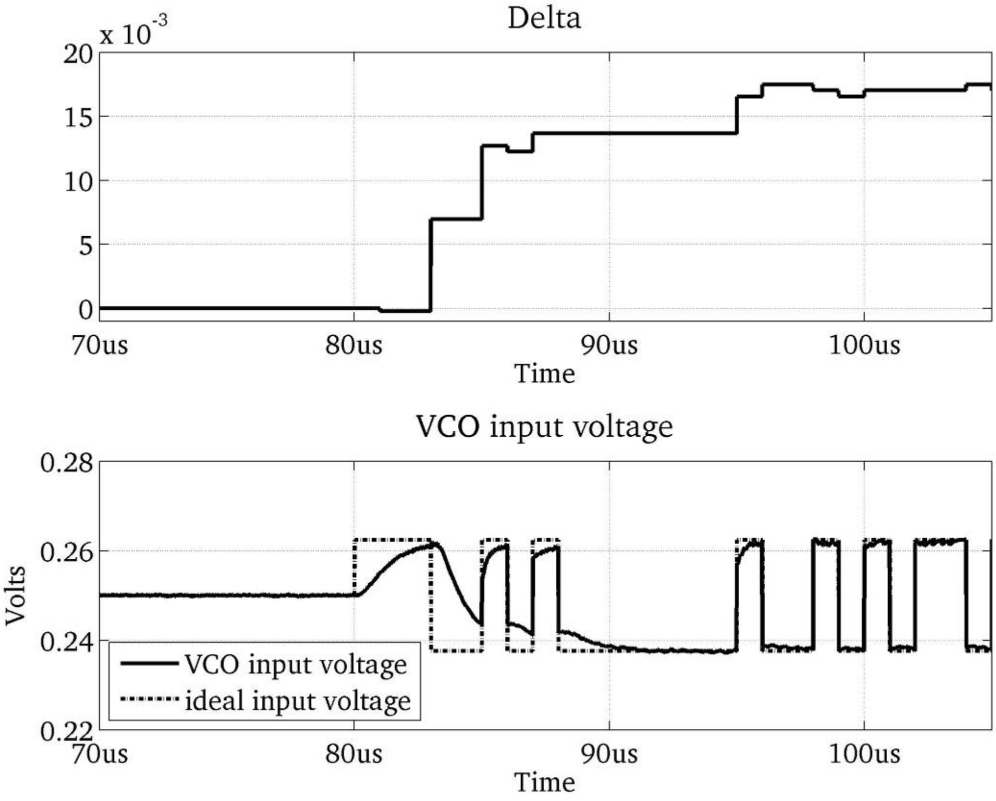

Figure 4.7: Behavioral simulation results. The top plot shows delta, which starts at zero and quickly grows to the required size. The bottom plot shows the ideal and the actual VCO input waveforms.

In Figure 4.7 some simulation waveforms are shown. The top plot shows *Delta* as it starts from zero and quickly grows. In the bottom plot the ideal and actual input voltage to the VCO is shown in the bottom plot. As can be seen, similar to Figure 4.6, the input voltage initially looks like a filtered version of the required signal but quickly converges to the correct value. In this case it only takes around 8 bit periods for *Delta* to converge to the correct value.

4.4 Implementation details

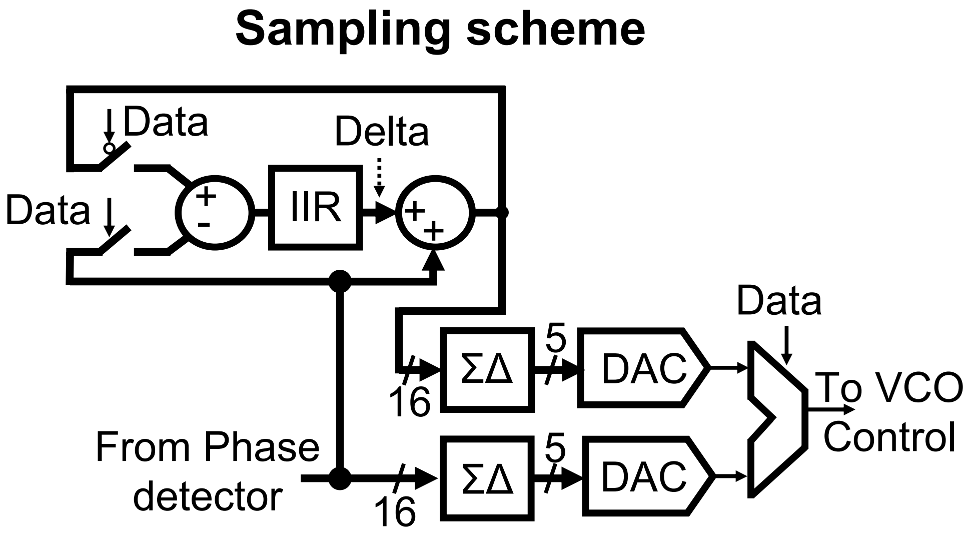

Figure 4.8: Details of the sampling scheme.

A more detailed block diagram of the sampling scheme is shown in Figure 4.8. The control path for the VCO is split into two paths, one path for each of the two frequencies, A and B. Each path contains a ΣΔ DAC, the output of which goes to an analog multiplexer. The two DAC control signals are sampled when the transmission data transitions from zero to one and from one to zero, which corresponds to the switching instants from frequency A to B and from frequency B to A respectively. The difference, *Delta*, is then added to the signal from the phase detector.

Multiplexing between the two DACs is done in the analog domain after the reconstruction filters. The filters contain two higher order RC poles to filter the DAC ΣΔ noise. As the input to the VCO is required to change instantaneously, the multiplexer is placed after these poles. The poles at the output of the DACs are placed well outside the bandwidth of the PLL so that these do not have much effect on the settling characteristics of the loop. The inputs to the sampling system shown in Figure 4.8 include a 16 bit unsigned word from the phase detector, and the raw data stream. The output is the analog control voltage of the VCO. Each first order ΣΔ modulator converts the 16bit bus into a 5bit bus, which controls a 5bit string DACs.

The complete architecture including the digital phase detector and the new sampling scheme is shown in Figure 4.9. The frequency switching method allows fast frequency modulation within a low loop bandwidth, as the frequency switching is not limited by the bandwidth of the loop, provided that the loop needs only to switch between a small number of discrete frequencies. This also demonstrates the usefulness of having all of the relevant signals in the digital domain. An analog equivalent of the above scheme is possible in principle, however implementing the samplers, adders and subtractors as analog circuits would introduce debilitating additional complexity. Almost all of the additional circuitry can be included as part of the synthesized logic, the only additional analog circuitry required is the extra DAC and multiplexer.

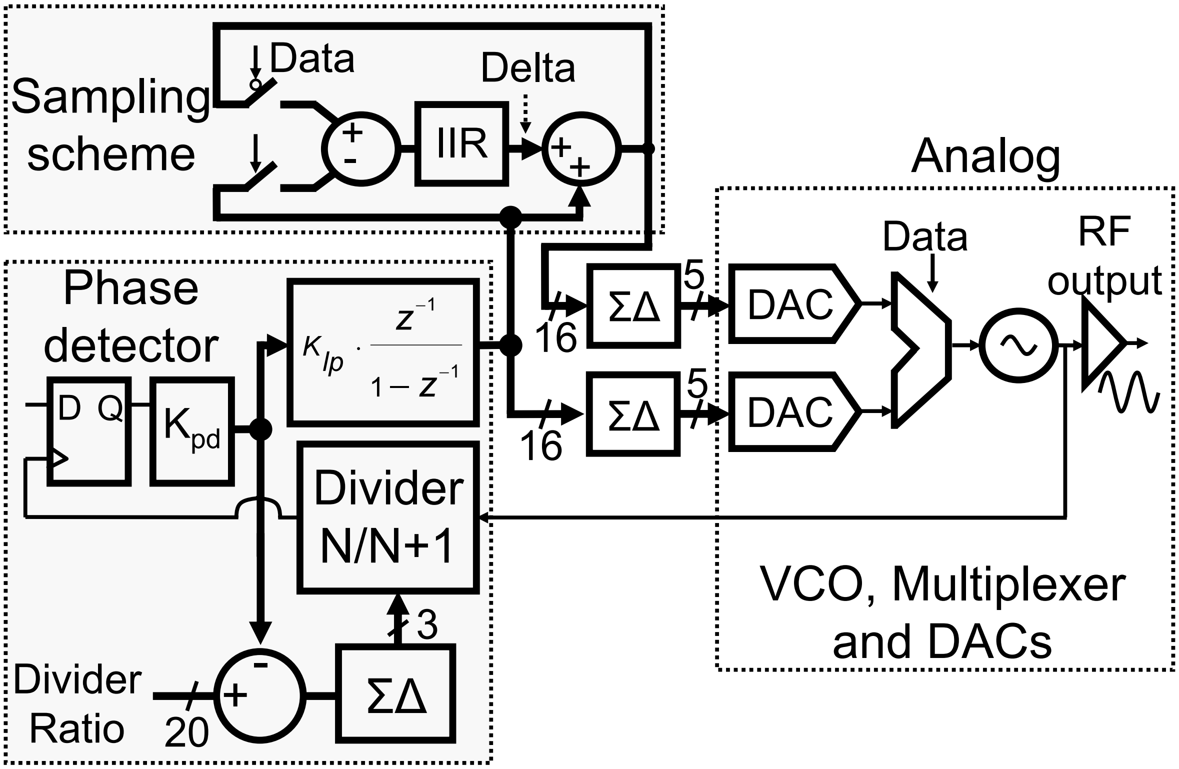

Figure 4.9: The complete architecture including the digital phase detector and the proposed sampling scheme.

CHAPTER 5

DESCRIPTION OF PROTOTYPE

Figure 4.9 shows the proposed architecture, this Chapter discusses the physical implementation of this architecture. While much of the system in Figure 4.9 is digital, and can be implemented using synthesized verilog, there are several analog components such as the VCO, DAC, output buffer, etc which will be described in this Chapter. The design of the programmable divider will also be described. This is an asynchronous divider; it is not part of the synchronous logic. A commercially available 0.13µm mixed-mode CMOS process was used to implement these blocks. A notable aspect of this design is that it is fully integrated, no external components are used.

5.1 Overview

A reference clock of 185.5MHz is used, with a nominal output frequency 2.24GHz, which corresponds to a nominal division ratio of N_{nom}=12.075. K_{pd} is set to 0.01, and the loop gain (K_{lp}) is set at 0.025, resulting in a loop bandwidth of 142 kHz. The VCO analog gain is 25MHz/V, and an additional 500MHz tuning range is achieved using digital switches which add or remove VCO capacitance. The analog tuning range is deliberately made small in order to prevent DAC quantization noise from excessively contributing to the output phase noise. A relatively high reference frequency is required, so that the phase quantizer is adequately over-sampled. Each doubling of over-sampling rate leads to a 3dB reduction in in-band quantization noise.

5.2 A ΣΔ digital to analog Converter

In order to get from the digital to the analog domain, a first order ΣΔ modulator controls a simple 5 bit string DAC. If a higher frequency reference clock or a DAC quantizer with more bits is used, then the VCO analog tuning range can be increased without degrading output phase noise.[8] A block diagram of the ΣΔ DAC is shown in Figure 5.1. The first block is a first order ΣΔ which converts a 16 bit digital word in into a 5 bit word. Next the 5 bit word is decoded to select one of 32 DAC control lines.

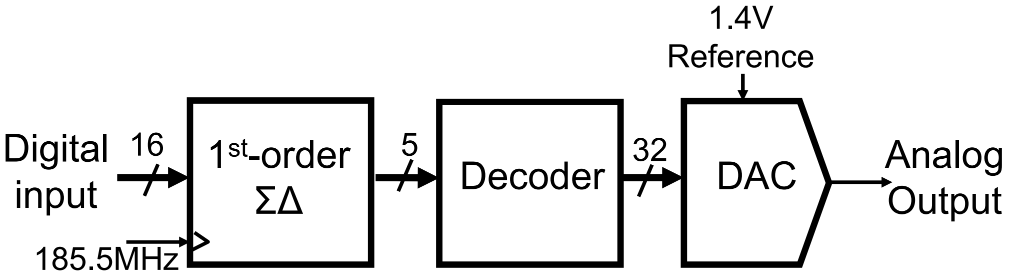

Figure 5.1: Overview of DAC scheme. Includes a ΣΔ modulator, a decoder and a resistor string DAC.

The first order ΣΔ DAC uses a single feedback loop in order to increase the resolution within a limited bandwidth for a quantizer. The architecture of the ΣΔ modulator is shown in Figure 5.1. As can be seen the output is subtracted from the input and subsequently intergraded before being quantized.

[8] An alterative approach is to implement a fully digitally controlled oscillator, such as presented in [55]. The ΣΔ DAC was used in our design for implementation simplicity.

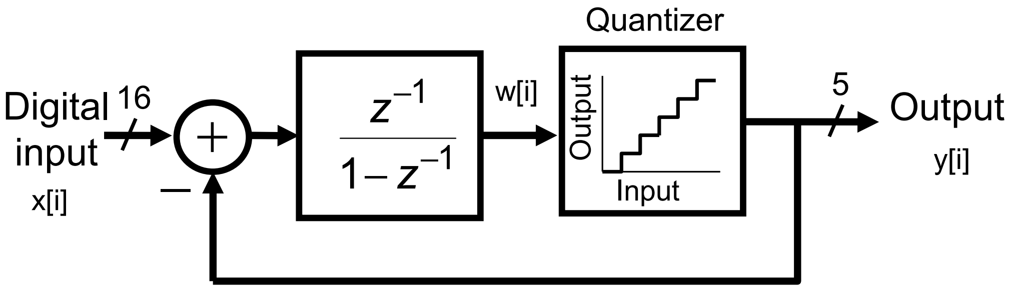

Figure 5.2: Overview of a first order ΣΔ.

Techniques needed to analyze a first order ΣΔ can be found in [46]. To analyze this circuit, we consider all the blocks to be linear, the quantizer simply adds a quantization error at the i^{th} sampling time of $e[i]$. The input ($x[i]$), the output ($y[i]$) and the integrator output (w[i]) need to be related to the error. The output of the integrator is simply the input minus the previous output, plus the previous integrator value.

$$w[i] = x[i-1] - y[i-1] + w[i-1] \tag{5.1}$$

Also, the output (y[i]) is simply the output of the integrator (w[i]) plus the quantization error (e[i]), as shown in Equation (5.2).

$$y[i] = w[i] + e[i] \tag{5.2}$$

Equations (5.1) and (5.2) can be combined to give Equation (5.3).

$$w[i] = x[i-1] - e[i-1] \tag{5.3}$$

Or $w[i]$ can be removed from Equation (5.3), using (5.2) to get Equation (5.4).

$$y[i] = x[i-1] + (e[i] - e[i-1]) \tag{5.4}$$

This can also be expressed in the z domain:

$$Y = X \cdot z^{-1} + E(1 - z^{-1}) \tag{5.5}$$

In other words the quantization error, *E*, is differentiated, while the input signal undergoes a time delay, with no frequency shaping. In order to evaluate the equivalent continuous time response, z is set to z=$e^{j2\pi fT}$. From this the quantization noise at the output can be given by Equation (5.6).

$$N(f) = E(f)\left|1 - e^{-2\pi jfT}\right| = e_{rms}\sqrt{2T}\,\sin(\pi fT) \tag{5.6}$$

From this we can see that the quantization noise is shaped to high frequency by the sin function. At frequencies much smaller than the sampling frequencies the noise falls to zero (sin(0)=0). Analog filters at the output of the DAC effectively filter out the high frequency quantization noise.

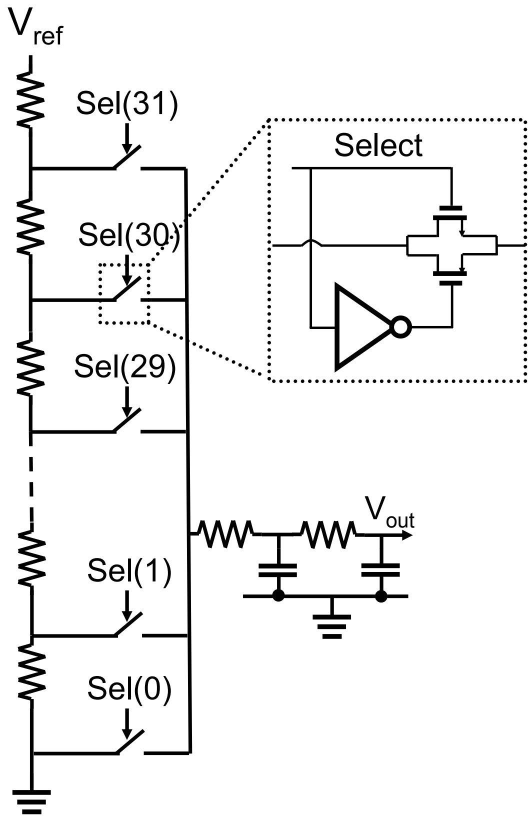

Figure 5.3: An overview of the resistor string DAC with a reconstruction filter at the output.

The analog part of the DAC consists of 32 100Ω N+ diffusion resistors, creating 32 different potential connection points. The string DAC is shown in 5.3. A resistor ladder divides the reference latter into 32 different levels. Then a set of CMOS switches selects one of the 32 levels. The output of the switches goes to an RC filter, which eventually goes to the output multiplexer. The size of the resistors is carefully chosen so that the DAC is able to settle to its final value even with a 200MHz clock. Simulation experiments with the capacitance extracted from the layout were used to confirm this.

5.3 The Voltage Controlled Oscillator (VCO)

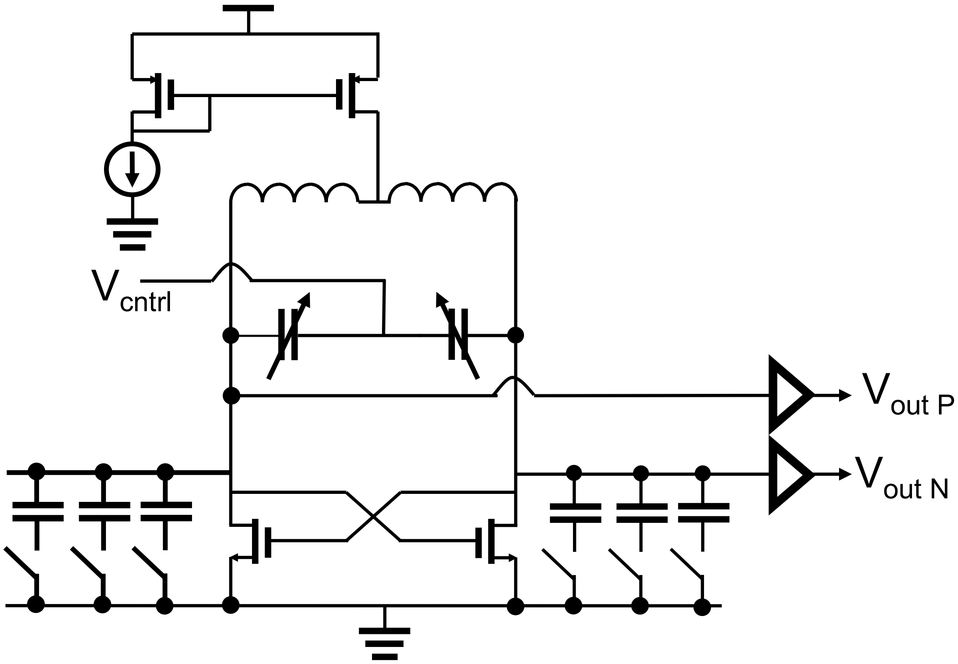

Figure 5.4: VCO overview.

An LC tank based VCO, shown in Figure 5.4, is used in this design. A PMOS current source drives current through a center tapped inductor, and an NMOS cross coupled pair provides the negative resistance. The varactors are implemented as NMOS varactors (NMOS in an n-well). The N+ source and drains for these devices are shorted together, and form one of the terminals, and the other terminal is the gate. The switching capacitors consist of a combination of NMOS varactors and metal-insulator-metal (MIM) capacitors. The analog tuning range for the VCO is approximately 25MHz. A current input mirrors a 50uA reference current into a 2mA current in the VCO core. The symmetrical inductor uses the thick top 2 metals of the CMOS process to give an estimated inductance of 6nH.

The measured phase noise for the oscillator with the rest of the PLL turned off is shown in Figure 5.5. The reason for the spur close to 10MHz is unclear, however we know that the spectrum analyzer uses an internal 10MHz clock. At low frequencies the phase noise of the oscillator appears to level out. This is thought to be due to the dynamic range of the spectrum analyzer (Agilent 4405B spectrum analyzer) measurement equipment, which was found to be the cause in a previous work [47]. In reality as frequency reduces, the noise power per Hertz should increases at a faster rate, it should not level out.

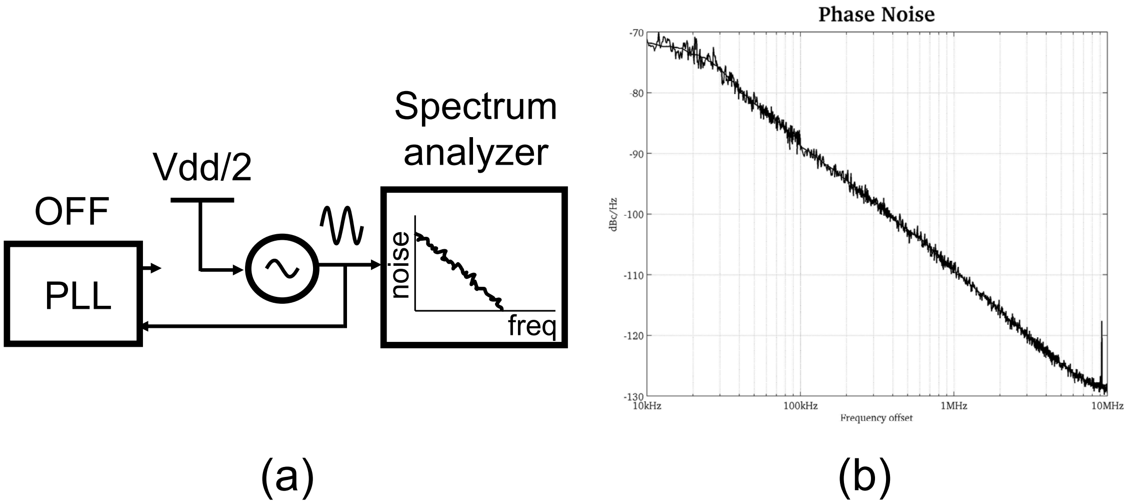

Figure 5.5: Measure VCO Phase noise. (a) The VCO is configured in open loop. i.e. with the rest of the PLL turned off. (b) The measured noise of the VCO by its self.

5.4 The design of a low headroom Output Buffer

The design of even simple output buffers in low headroom processes is not trivial. If a source follower is used, then enough headroom would be required for both the signal swing and the V_t of the MOS devices. In this work, a simple output buffer is proposed, which requires little headroom, and sets the DC value of the output node automatically, without requiring complex common mode feedback circuitry. The buffer is shown in Figure 5.6. At DC, the buffer acts a current source driving current into a diode connected load. In this way the DC point of the output is approximately V_{dc}=V_{dsat} + V_t, which is close to Vdd/2 for this process. On the other hand at high frequencies, the buffer acts as a push-pull amplifier. The DC values at the input and output of the buffer are not important, as both input and output are AC coupled.

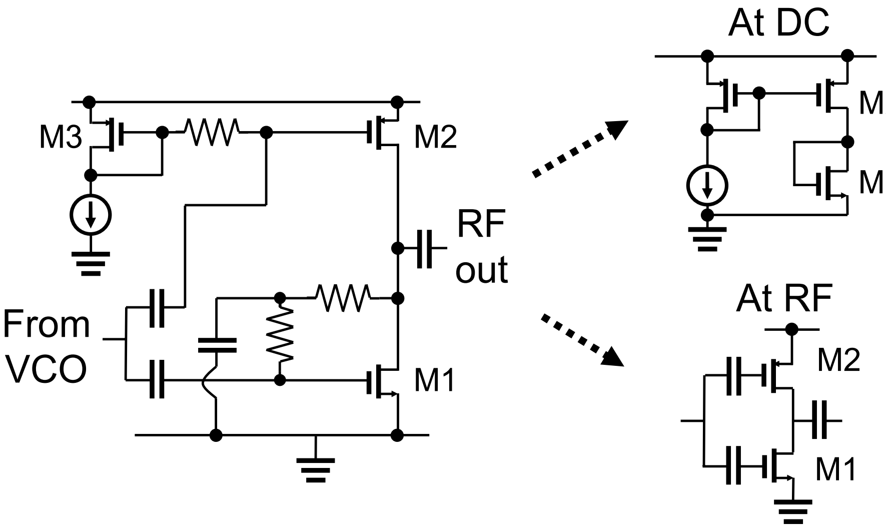

Figure 5.6: A push-pull Output buffer which automatically sets its output bias point without additional circuitry.

When designing such circuits, special care must be taken to avoid excessive resistive loading of the VCO. The resistors shown in Figure 5.6 are separated from the VCO by decoupling capacitors. If these resistors are too small, then they can significantly reduce the effective quality factor of the VCO tank, and hence contribute to phase noise. In addition the buffer contains a feedback loop. The stability of this loop was carefully verified in simulation.

5.5 A Pseudo-Random Bit Sequence (PRBS) generator

The prototype also includes a Pseudo-Random Bit sequence (PRBS) Generator. Using a test mode this random data can be selected as the transmission data, reducing the amount of external test equipment required. The PRBS generator is a digital block, and is embedded in the synthesized logic. A 16 bit Linear Feedback Shift Register (LFSR) in a Galois form (also known as a *multiple-return shift register generator* (MRSRG)) is used to generate the random numbers. The PRBS generator has a sequence length of 2^{16}-1. An introduction to this subject can be found in [56]. In this work the taps are at registers 16, 15, 13, and 4.

5.6 A high speed asynchronous divider

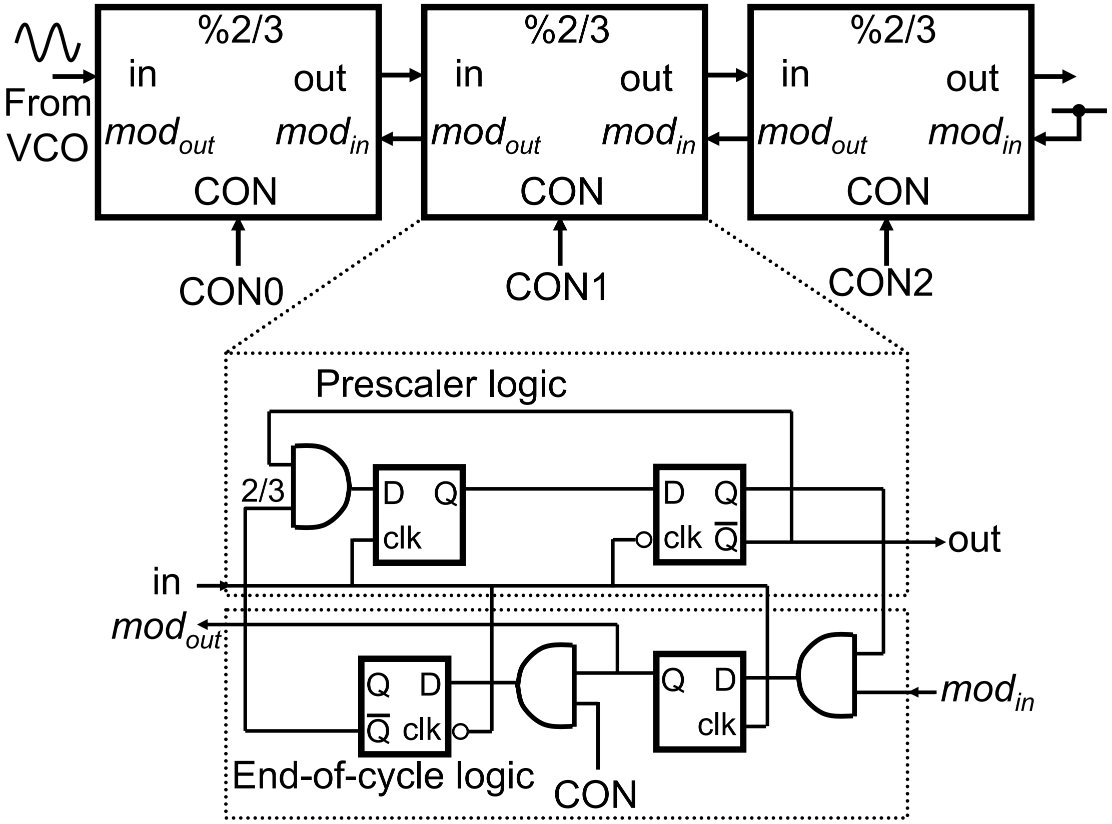

Figure 5.7: Programmable Divider; Architecture and logic.

The programmable divider is based on the modular architecture presented in [57], where each of the 2/3 divider cells is identical. The divider division ratio can be set from 8 to 15. The divider is shown in Figure 5.7. Each modular divider block consists of two parts; the *prescaler logic*, and the *end-of-cycle logic*. The *prescaler logic* divides the input signal by either 2 or 3 depending on a signal which comes from the *end-of-cycle logic*. (Section 1.9 contains a description of a standard prescalar.) The *end-of-cycle logic*, determines whether to divide by 2 or 3, and outputs the mod_{out} signal for the next divider block in the chain. The mod_{out} signal goes high only once per output cycle. On the rising edge of mod_{out} the value of the input *CON* is checked, and if it is high then the prescaler swallows one extra period of the input signal, i.e. it briefly divides by 3 instead of 2. Regardless of the state of the input, the end-of-cycle logic reclocks the mod_{out}, and outputs it to the preceding cell in the chain signal.

The operation of the divider can be explained as follows. Each divider cell produces a mod_{out} signal which is high for half the time that the divider cell's mod_{in} is high. The mod_{in} signal for the last divider in the chain is tied to Vdd, and so it produces a mod_{out} signal which is high half the time. The second to last divider in the chain produces a mod_{out} signal which is high half the time that its mod_{in} signal is high, which means that its mod_{out} signal is high one quarter of the time. When a divider's mod_{in} input is high, then it divides by 2 or 3 based on the CON input. When the divider's $\mathrm{mod}_{\mathrm{in}}$ signal is low then it ignores the CON input and divides by 2. If all of the control inputs are 0, then each of the divider blocks will divide by 2, leaving a net division rate of 8. If CON0 is high, then the left hand divider will divide by 3 only when its $\mathrm{mod}_{\mathrm{in}}$ signal is high, which happens only once per total output period. Therefore only one extra pulse will be swallowed per output cycle, and net division ration will be 9. Similarly, if only CON1 is high, then one extra pulse of the input (which has already been divided in two) is swallowed, resulting in a divide ratio of 10. Similarly, CON2 can add an extra 4 to the net division ratio. The final division ratio is given in Equation (5.7).

$$Division\,Ratio = 8 + CON0 + 2 \cdot CON1 + 4 \cdot CON2 \tag{5.7}$$

5.7 The Analog Multiplexer

The analog multiplexer is placed at the output of DACs, and before the VCO input. This is a straightforward analog switch. It is critical to ensure that both switches in the multiplexer are not on simultaneously, as this would short the outputs of the DAC RC filters. For this reason a break-before-make circuit is added to the block, as seen in Figure 5.8. Two NAND gates with delay in their forward path are used in a cross-coupled configuration to guarantee that both switches are never on simultaneously. The make-before-break includes an RC filter. Extra inverters could have been used to add delay, however it was found in simulation that the RC process variation is less than inverter process variation. For this reason an RC filter is used as a delay element.

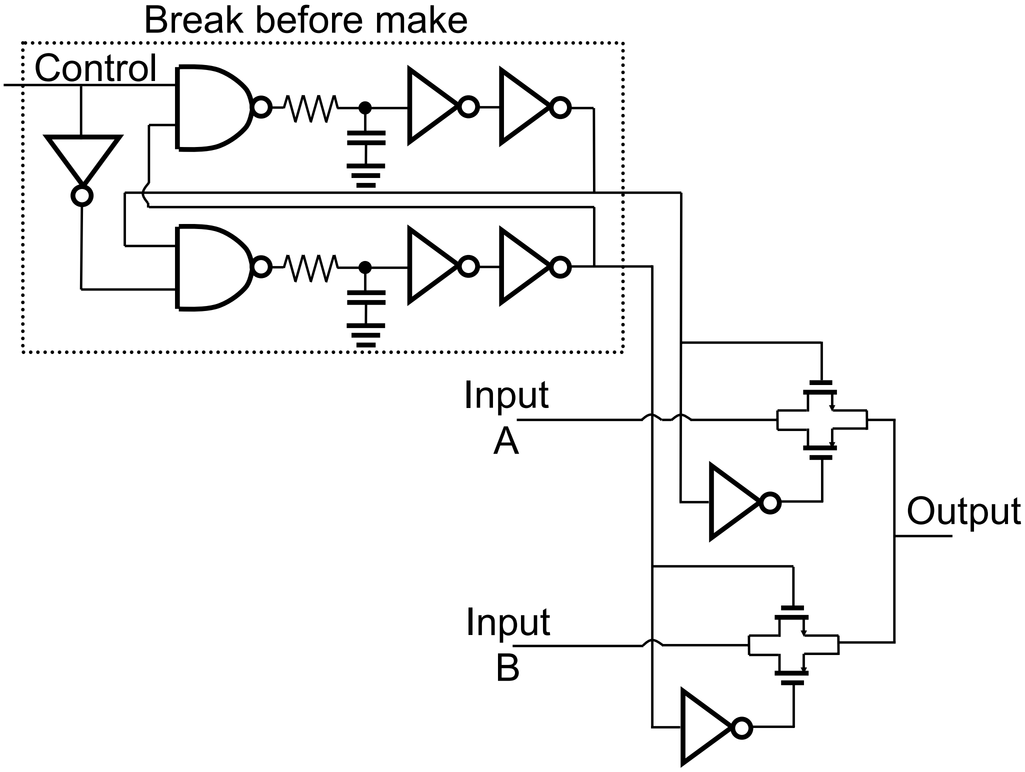

Figure 5.8: Overview of the analog multiplexer

5.8 Prototype layout

The prototype transmitter is implemented on a 0.13µm mixed-mode CMOS, the layout is shown in Figure 5.9. The active design area is 0.7mm^2, with a total area of 2mm^2 including pads. It is worth noting that even though many of the conventional analog components have been removed, the area is still dominated by the remaining analog blocks. The synthesized logic takes up around 0.075mm^2, which is a small fraction of the overall area, despite dominating the architecture shown in Figure 5.9. This demonstrates the area savings that can be achieved by going from an analog dominated architecture to a digital dominated architecture. The inductor in the VCO and the output buffer consume large amounts of area, which is typical for a PLL.

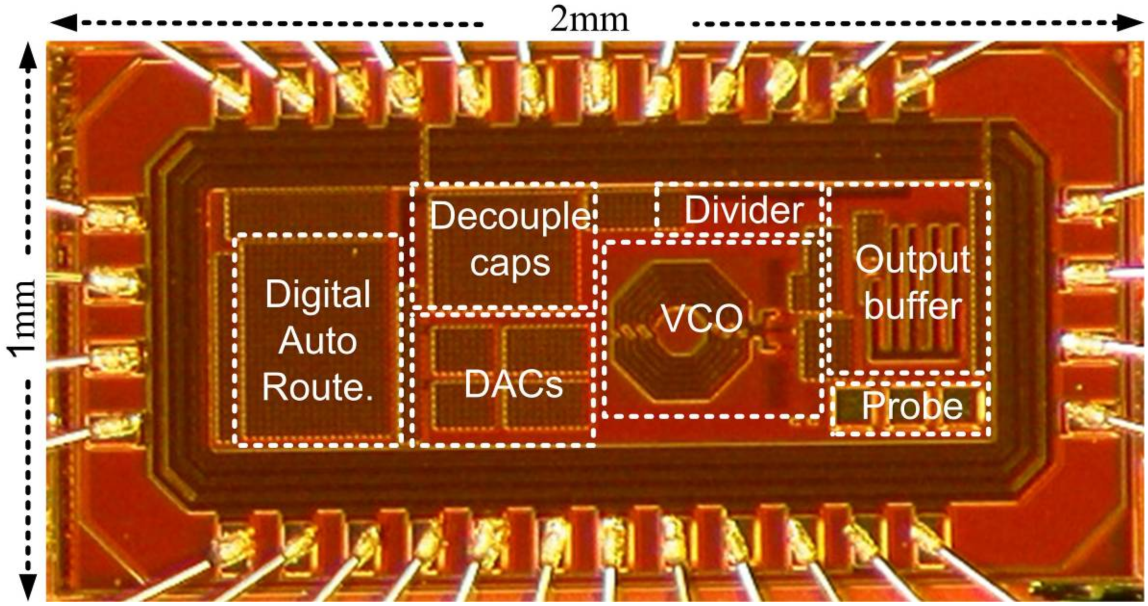

Figure 5.9: Micrograph of Prototype

CHAPTER 6

MEASUREMENT RESULTS

A 0.13µm CMOS process was used to fabricate 40 prototypes and custom printed circuit boards (PCBs) were designed in order to test the parts. Typical output spectrums measured from silicon are shown in Figure 6.1, for a pure synthesized tone and for random data with an FSK data modulation rate of 927.5kbits/s.

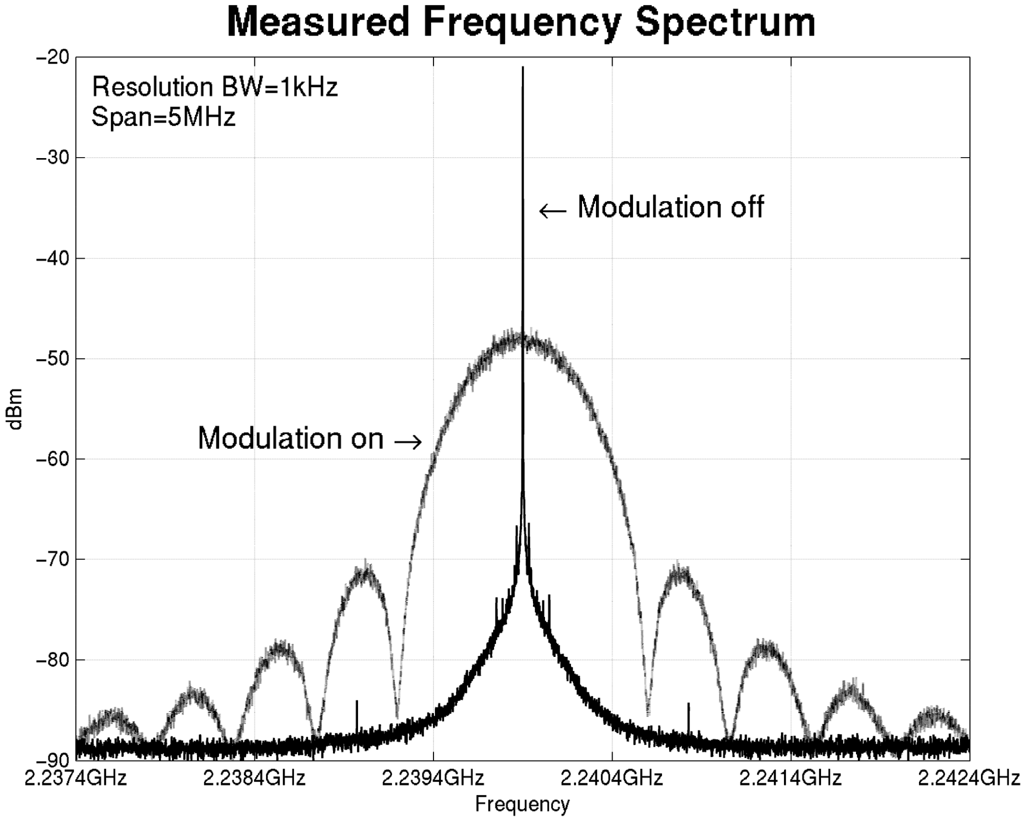

Figure 6.1: The output spectrums, when modulation is turned on and off.

If the synthesizer is functioning correctly, then we expect the output frequency to always be equal to the reference frequency times the division ratio. For example, when using a reference frequency of 185.5MHz, and the division ratio is 12.075 (as is the case for Figure 6.1), then an output frequency of 2.2399GHz is produced. There are 16 available division ratios programmed on the chip, and it was found that the output frequency is always correct (provided the output frequency is within the VCO's tuning range), confirming basic functionality. The modulation scheme used in this work is 2-FSK; the frequency switches between one of two difference frequencies. Simple modulation schemes similar to this are used in low power applications such as IEEE 802.15.4[9]. The modulation spectrum in Figure 6.1 has large side lobes, as would be expected for an IEEE 802.15.4 modulation scheme, an example of such a spectrum can be found in [58].

The prototype consumes 14mW from a 1.4V supply[10]. A 185.5MHz reference clock is used, which allows the phase detector to be significantly over sampled. The VCO was originally intended to operate in the 2.4GHz ISM band. However, due to process variation, the maximum VCO frequency is 2.24GHz, even considering the digital tuning range.

6.2 Phase noise measurements

In this section some of the phase noise measurements are discussed. In the prototype the value of the loop constants are fixed. The various noise sources, and the

[9] IEEE802.15.4 uses an OQPSK modulation scheme, with half sine pulse shaping on the I and Q channels. This form of modulation is spectrally identical to 2-FSK [59].

[10] The recommended nominal supply voltage for this process was 1.2V, all measurements were done with a supply of 1.4V. This is due to the performance of the high speed programmable divider.-

total calculated noise, and measured noise are plotted in Figure 6.2. For low and very high frequencies the phase noise is dominated by the VCO noise, while for frequencies close to the loop bandwidth the phase detector quantization noise (PD in Figure 6.2) is significant. The poor VCO phase noise performance is due in part to the low quality factor of available integrated inductors. The divider ΣΔ noise does not make a significant contribution to the output phase noise. This is because the reference clock frequency is large in comparison to the loop bandwidth, and hence most of the ΣΔ quantization noise is shaped to outside of the PLL loop bandwidth.

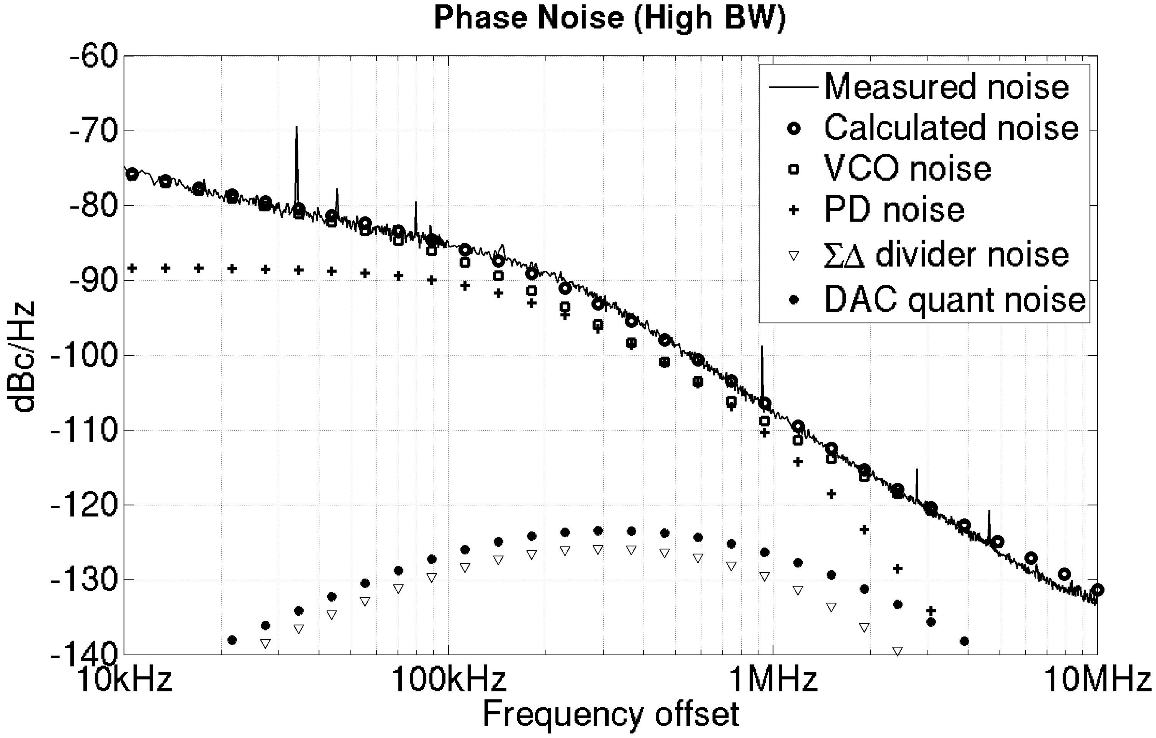

Figure 6.2: Plot of measured and calculated noise sources. The noise is dominated at high frequencies by the VCO, at mid frequency ranges the phase quantizer noise is also significant.

6.3 Frequency switching scheme

Verifying the modulation techniques in silicon presents a challenge, as the only available signal is the RF output signal. (In contrast in behavioral simulations the input to the VCO can be monitored.) A National Instruments (NI) RF Analyzer (NI1042) was used to evaluate the RF signal. The transmission data is generated using an on-chip PRBS generator. The output of the chip is connected to the NI equipment, which based on the expected average output frequency, measures phase changes at the output of the PLL. An overview of the test set up is shown in Figure 6.3. Different data patterns produce different phase changes at the output of the FPLL. The phase changes for different data patterns are overlaid, forming a phase trellis diagram.

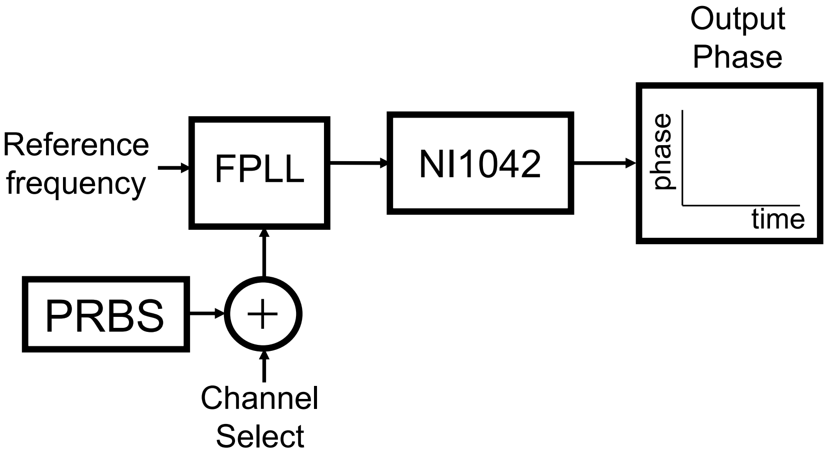

Figure 6.3: Test setup for measuring the phase Trellis diagram at the output.

As an example, consider the case of when the frequency switches between two different frequencies so that the resulting phase changes are shown in Figure 6.4. If we apply two different data patterns we get two different phase changes.

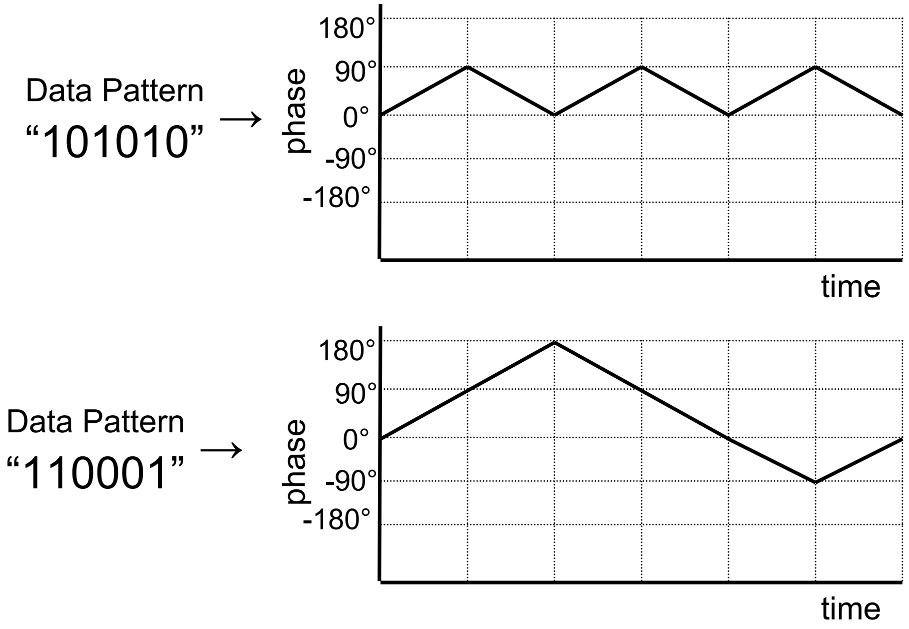

Figure 6.4: Expected phase changes for different data patterns.

If we overlay these patterns on top of each other we get the results similar to those shown in Figure 6.5.

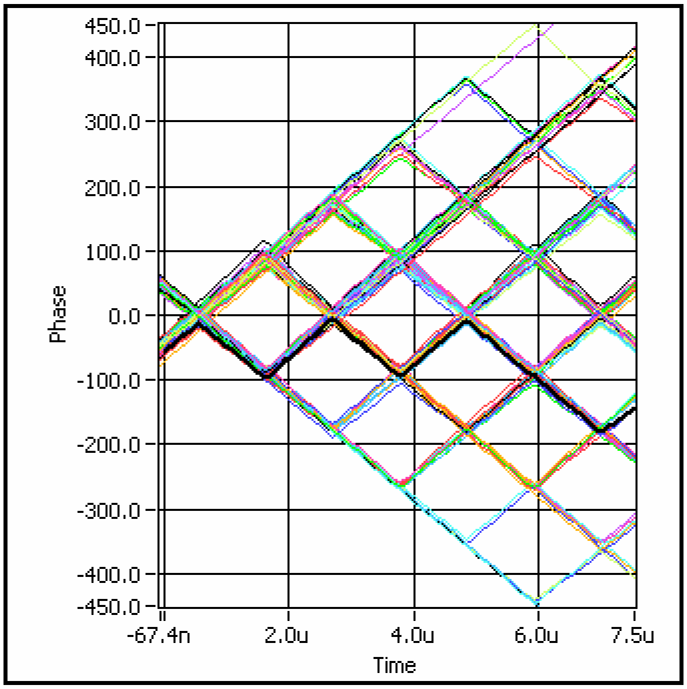

Figure 6.5: Trellis diagram of the output phase, as measured from silicon.

Figure 6.5 shows a trellis diagram of the results measured from silicon. As can be seen, the phase changes direction instantaneously, which is the equivalent to the frequency changing instantaneously. Even with a loop bandwidth of 142kHz, a data rate of 927.5kb/s is still possible. In most FPLLs the data rate is restricted to a fraction of the loop bandwidth. When the experiment is repeated with the switching scheme turned off, the output phase is garbled, as the FPLL cannot switch frequencies fast enough.

6.4 Comparisons with other works

Performing like for like comparisons with other designs is problematic for a number of reasons. Some publications use external oscillators resulting in superior phase noise and lower power dissipations. Other works use external loop filters for loop stabilization. However there are some trends that are worth pointing out. Firstly, designs which are fully integrated, and include the loop filter on chip consume more area and power than this work as shown in Table 6.1. Furthermore, and examination of the layout in Figure 5.9 reveals that although the architecture is made of mostly digital components, the area is still dominated by the analog components such as the VCO, DAC, etc. This demonstrates the value of implementing the architecture the digital domain. For further power/area reductions the focus should be on reducing the use of analog circuitry even further.

	[60]	[61]	[62]	This Work ([65])
Bandwidth	460kHz	700kHz	1MHz	142kHz
Output Frequency	2.4GHz	2.1GHz	3.6GHz	2.2GHz
Phase noise @10MHz	-133dBc/Hz	-135dBc/Hz	-151dBc/Hz	-133dBc/Hz

Includes Oscillator/Filter	Yes/No	Yes/Yes	No/No	Yes/Yes
Area	**6.7mm^2**	**3.4mm^2**	**7.3mm^2**	**2mm^2**
Core Power	**61mW**	**28mW**	**110mW**	**14mW**

Table 6.1: Performance comparison for different synthesizers.

CHAPTER 7

CONCLUSION

The primary contribution of this work is the development and demonstration of new techniques for implementing digital dominated Fractional-N synthesizers. In conventional synthesizers analog intensive charge pumps and capacitor based loop filters are used to extract the useful information from the phase detector and to stabilize the loop. When using the digital phase detector presented in this work, there are no analog components in the phase detector, and no analog loop filter is required. This saves area and power. In addition it results in an architecture which is more suitable for a fine line CMOS manufacturing process, which is an important consideration as transistor sizes continue their downward scaling. As well as proving the phase detector in prototype form, a small signal model for the architecture was developed.

In addition a new method for modulating the output frequency of the FPLL has been developed. This technique can allow the frequency of the output to be modulated at a rate significantly faster than the loop bandwidth would otherwise allow. The key ideas have been proven in a prototype design.

7.1 Dissertation Summary

The key ideas for a digital based phase quantizer are introduced in Chapter 2. Small signal models for the components including phase noise calculations are done in Chapter 3. Chapter 4 introduces a new method for high speed modulation for FSK type schemes. Chapter 5 discusses the implementation details for the prototype, and Chapter 7 discusses the measurements from silicon.

7.2 Key Contributions

This work has made several important contributions to the field of RF synthesis. Firstly, an FPLL has been demonstrated which uses significantly less analog circuitry than a conventional FPLL. It does this by using a phase detection scheme which relies on over-sampling and phase quantization, as described in Section 2.1. Next, a method for modulating the FPLL is developed which overcomes the traditional trade-off between modulation speed and overall loop bandwidth in FSK modulation schemes, as described in Chapter 5. All of these ideas are implemented on a prototype IC incorporating digital, analog and RF signal processing.

As transistor gate length continues to scale down, design of analog circuitry will become even more challenging, while the area and power costs of digital circuits continues to decrease. Techniques such as the ones presented here will become crucial in order to move from analog dominated circuitry to digital circuitry. As these types of systems are to be integrated on substrates with large DSP circuits, it is important to minimize amount of analog circuitry.

7.3 Ideas for future work

As with any research project, each new development opens up many avenues of potential discovery, and this work is no different. There are some fascinating potential follow-on projects from this work. Here are just a few of them.

The Phase detector can also be described as a frequency discriminator, as discussed in 2.6. This opens up the possibility of using the phase detector as an RF receiver for constant envelope communications. Although the frequency discriminator can derive no information about the amplitude of the incoming signal, it could be used to digitize an IF signal and extract frequency or phase information, similar to [42].

In Chapter 2, the single bit phase quantizer with the PML is described as a competing technology to a TDC, however these may in fact be complementary ideas, as the PML could be used to control the closed loop gain of a TDC. It would be fascinating to implement both techniques in the same PLL, the concept is shown in Figure 7.1.

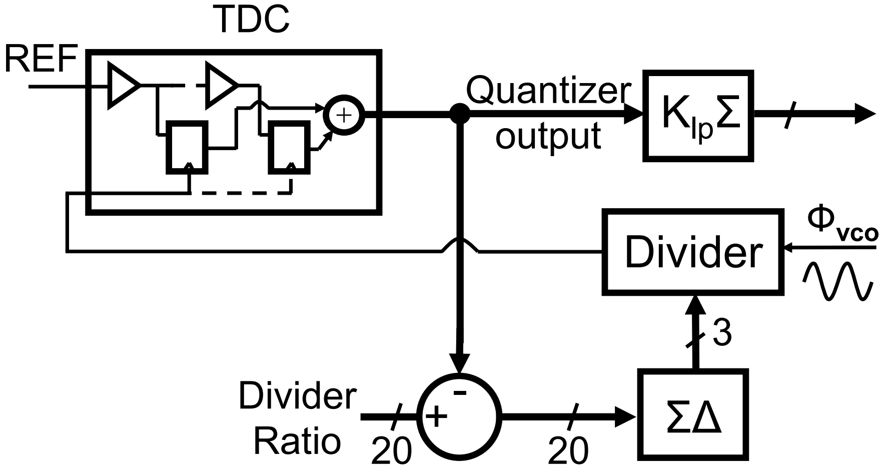

Figure 7.1: Using a TDC with the PML.

This research would dove-tail with the development of a digitally controlled Oscillator. In this way the architecture could be made even more digital in nature, further reducing the process dependency.

Other potential uses for this project include using it as a clock repeater. If a ring is used oscillator instead of a LC tank oscillator, then the design could be made fully synthesizable, and consequently suitable for integration with clock repeaters. The architecture's potential to be free of jitter peaking make it attractive for this application.

APPENDICES

APPENDIX A

DIRECT PHASE MODULATION

This section discusses a technique which has been proven in simulation, but not verified in silicon. The development of an all digital phase detector opens up the possibility of using synthesizers in different ways to previously used. In a conventional FPLL the only digital way to change the output phase is to change the division ratio. However the conversion to an all digital architecture allows for a new method to be used, that is to add modulation data directly to the output of the phase detector.

A.1 Phase control with a digital PLL

The techniques introduced in Chapter 4 take advantage of the digital nature of the architecture to achieve very fast frequency switching for FSK type modulation schemes. However, it would be desirable to support a wider variety of phase modulation schemes. In addition, it would be desirable to utilize some of the unique benefits of the new digital architecture to demonstrate other methods of modulating the frequency and phase of the output signal. A good starting point is to go back to a standard PLL and consider other ways in which to modulate the output signal. Consider the diagram in Figure A.1. As mentioned above, the standard way to change phase/frequency at the output is to change the control word. However, there are two other possible ways to change the phase of the output signal. One possible method is to introduce a time delay Δt into the reference path. This changes the phase of the output signal by $\Delta\Phi_{out}$, which is expanded in Equation

(A.1). The phase change at the output can either be expressed in terms of the time delay, Δt, or the equivalent phase delay, $\Delta\Phi_{in}$.

$$\Delta\phi_{out} = 2\pi \cdot \Delta t \cdot f_{out} = \Delta\phi_{in} \cdot N_{nom} \tag{A.1}$$

As a practical matter, synthesis of exact time delays involves complex analog circuitry and hence this technique should be avoided. However, there is a second approach which is equivalent to the time delay approach, but is easier to implement.

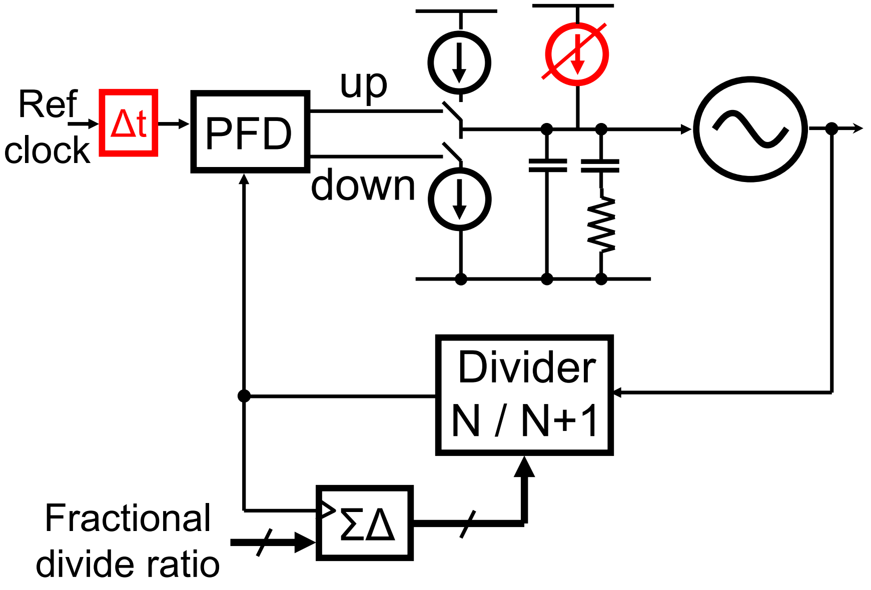

Figure A.1: Alternative phase variation techniques.

If the gain of the phase detector is exactly known, then it is possible to add a current directly to the output of the charge pump. Since it takes a phase difference at the input to produce a net current from the charge pump, adding a DC current to the loop filter is the equivalent of adding a time delay to the reference. In other words, to produce a phase change of $\Delta\Phi_{out}$, a current of I_{in} (given in Equation (A.2)) is injected. K is the phase detector gain, and N_{nom} is the nominal divide ratio.

$$\Delta\phi_{out} = I_{in} \cdot K \cdot N_{nom} \qquad \text{(A.2)}$$

This technique is difficult to implement in a conventional PLL because the current needs to exactly match the gain of the charge pump, if an accurate phase change at the output of the phase detector is to be produced. There are two separate problems here, synthesizing the currents, and evaluating the phase detector gain. In the proposed digital dominated PLL a digital signal can be injected into the output of the phase detector, which solves the first problem. Also, in section 2.7 we discussed how the new feedback loop desensitizes the overall transfer function of the phase detector to the PD gain. Therefore with a digital phase detector, the gain of the phase detector is known, and this technique can be used to modulate the PLL. Notice that in this modulation scheme, the injected phase information passes through just the DAC and VCO before reaching the output.

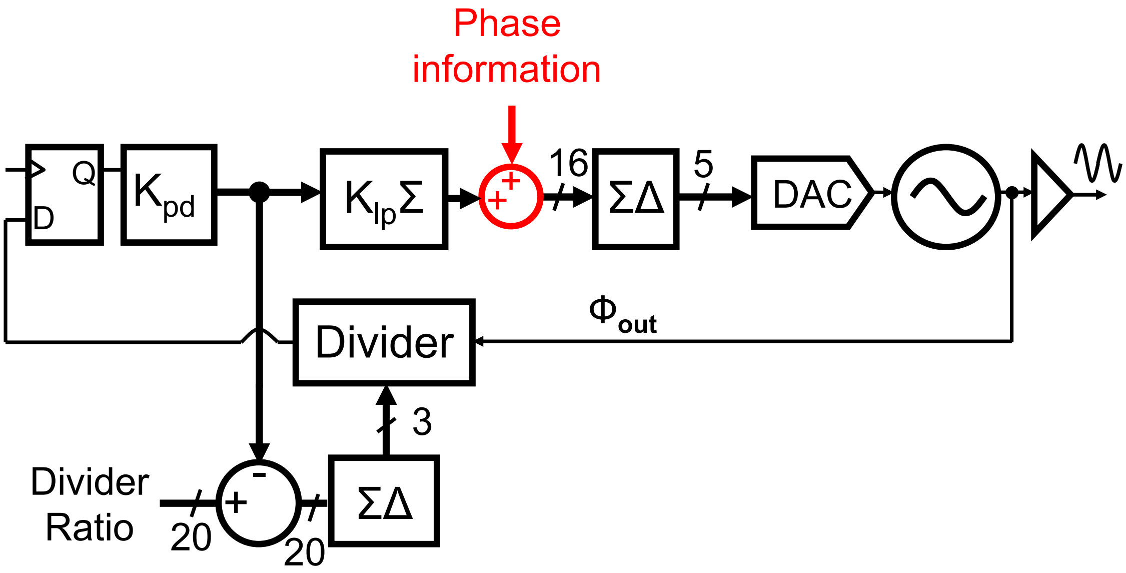

Figure A.2: Digital Phase control technique

A.2 Pre-emphasis

The final proposal is to use pre-emphasis to improve the modulation speed of any phase modulation technique. Pre-emphasis has been used in the division ratio control path to speed up the loop settling time [29]. However this technique could also be used in the new phase control path. The dynamics of the loop can be calculated, and the phase information is passed through a transfer function which is the inverse of the transfer function of the overall PLL, and in this way the pre-emphasis filter cancels out the filtering effects of the PLL itself. Pre-emphasis allows faster modulation of the output frequency.

The advantage of the scheme mentioned above is that it can be used for any phase modulation scheme, not just for FSK. For example, in a simulation experiment QPSK data is injected into the loop. The bandwidth of the loop is only 140kHz, however the data inserted had a data rate of 500kbit/s. A scatter plot of the resulting output phase is shown in Figure A.3. As can be seen, without pre-emphasis the loop distorts the constellation diagram, because the bandwidth of the loop is not wide enough. On the other hand, when pre-emphasis is used, then the constellation diagram is significantly cleaner.

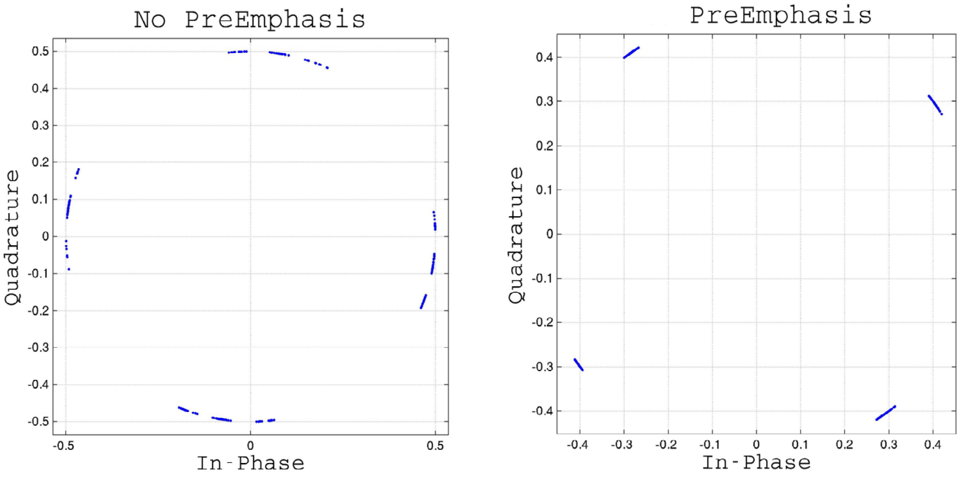

Figure A.3 (a) Direct phase modulation, (b) with pre-emphasis

APPENDIX B

SIMULATION TECHNIQUES

This section will discuss some of the practical problems involved in the simulated verification of PLLs. One of the challenges in the design of synthesizers involves developing successful simulation strategies. The simulation of any complex system can be problematic, but the design of fractional synthesizers provides a unique set of problems that takes quite an effort to overcome.

B.1 An overview of simulation limitations

The primary problems stems from the fact that the ratio of the frequency of the output signal is several orders of magnitude greater than the bandwidth of the system. For example, consider the case when the output frequency is 2GHz, which corresponds to a period of 500ps. If the simulator is to accurately measure the phase of the output signal, then the time step of the simulator must be set to a small fraction of this, typically less than 1% of the signal period, which in this case would be 5ps. On the other hand, the bandwidth of the PLL is typically of the order of 100s of kHz. To perform useful transient simulations the PLL must be simulated for many loop time constants. If the loop has a time constant of 10μs, and we need to simulate the loop for 10 time constants (which is close to the minimum time that useful information can be extracted from the simulation) with a 5ps time step then 2e7 (!) simulation steps are needed. If the simulation contains many analog components, then solving for each time step can involve significant calculations. As a practical matter simulating the entire analog section at the transistor level is found to be close to impossible, just to get the loop to settle several weeks of simulation time are required. In addition, with that many simulation steps, the computer can quickly run out of memory, and the simulation results files eventually reaching the size of 10s to 100s of GBs.

Using a combination of pre-existing tools, behavioral modeling and simulation tricks many of the most challenging aspects of these problems can be dealt with. The initial steps of the architectural investigation were done with CPPsim [58], a free behavioral simulator developed specifically to deal with PLL synthesizers. AHDL models were employed to help which speeding up transient simulations, and SpectreVerilog and AMS were used for mixed-signal simulation. Even with fully behavioral simulations (no transistors) some additional tricks were required to help get the simulations to be sufficiently timely.

B.2 Avoiding large simulation files

In behavioral simulations, even if the simulator only keeps the output nodes, the simulation output files can quickly grow to be too large, causing the simulator to crash, or producing output files that are too large to deal with using the standard plotting tools. Typically we are interested in the phase of the RF output signal, not the signal itself. In a typical transient simulation the simulator stores the high frequency output, and then we do post processing to extract the phase information. Unfortunately the simulation file sizes grow very quickly, hence this can be impractical. The simulator can strobe the output signal, that is it does not store every data point, but only stores a data point periodically at a much slower rate than otherwise would be required. However, this under-sampling results in the loss of phase information from the RF signal.

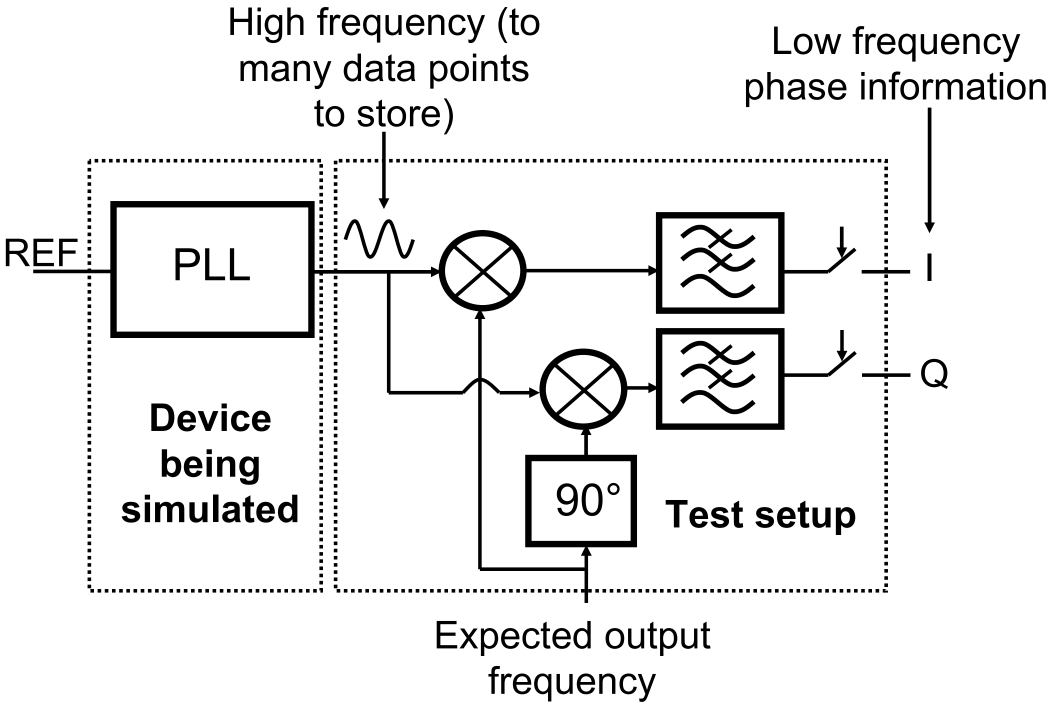
High frequency (to many data points to store)
Low frequency phase information
REF
PLL
I
Q
Device being simulated
90°
Test setup
Expected output frequency

Figure B.1: Simulation technique used to avoid keeping too many data points.

If the high frequency signal is down converted by mixing it with the expected PLL frequency, then the I and Q information can be extracted. These signals can be low passed filtered, and strobed at a rate much slower then the frequency of the output of the PLL (but still significantly faster than the bandwidth of the loop), resulting in much smaller simulation files. The key point is that the signal processing associated with extracting the phase information from the simulation is done within the simulation itself. Behavioral models for the mixers and filters are included in the simulation, as shown in Figure B.1, and only the processed signals are stored, the output of the PLL itself is not stored.

B.3 Modeling the VCO and divider

As part of a typical design phase, the PLL is iteratively simulated as part of a debugging process. The control voltage of the PLL can be monitored to check for settling behavior, there is no need to keep the RF output signal. However, because of the high frequency signals in the system, the simulator still needs to use a small time step. If simulation time is very long then this seriously undermines our ability to do architectural investigations and design iterations. However, there is a way to simulate most of the circuit without generating the high frequency signal. Consider the divider and VCO shown in Figure B.2(a).

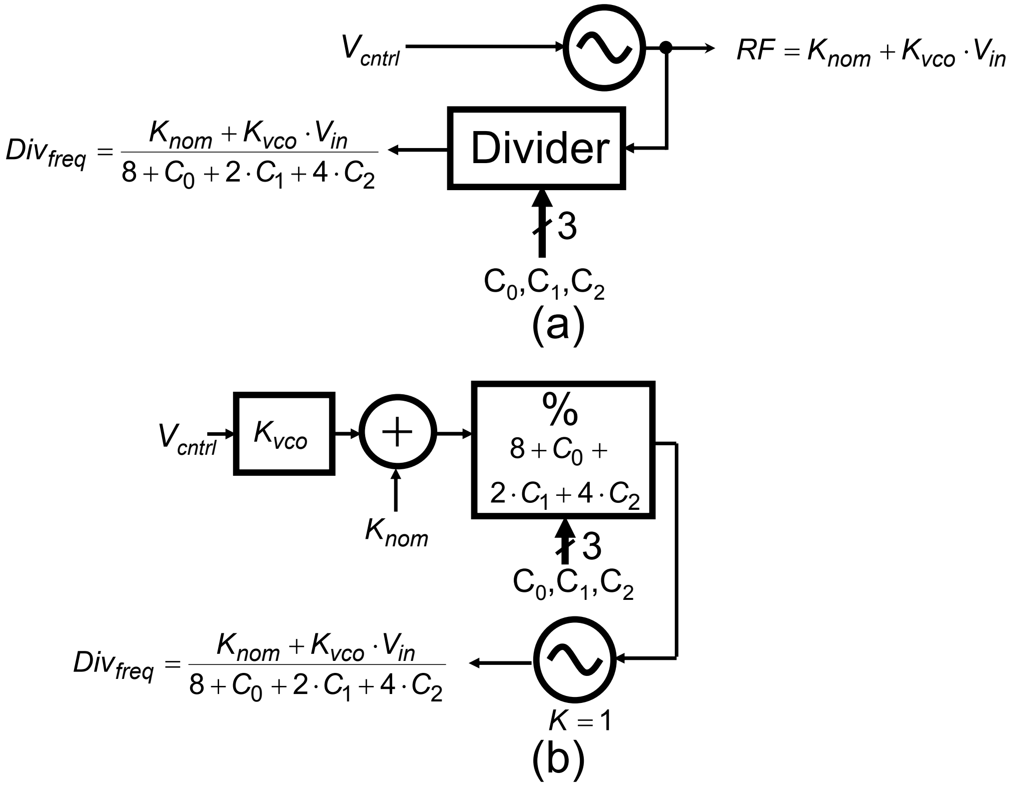
V_{cntrl}
$RF = K_{nom} + K_{vco} \cdot V_{in}$
$Div_{freq} = \frac{K_{nom} + K_{vco} \cdot V_{in}}{8 + C_0 + 2 \cdot C_1 + 4 \cdot C_2}$
Divider
3
C_0, C_1, C_2
(a)
V_{cntrl}
K_{vco}
+
K_{nom}
%
$8 + C_0 + 2 \cdot C_1 + 4 \cdot C_2$
3
C_0, C_1, C_2
$Div_{freq} = \frac{K_{nom} + K_{vco} \cdot V_{in}}{8 + C_0 + 2 \cdot C_1 + 4 \cdot C_2}$
$K = 1$
(b)

Figure B.2: The VCO and divider. (a) The actual configuration. (b) The behavioral model.

In Figure B.2(a) the VCO and divider is shown, which outputs a RF signal of frequency given by (B.1) and a divided down frequency given by (B.2). K_{nom} is the nominal frequency of the VCO, *Kvco* is its gain, and C_0, C_1, C_2 are the divider controls.

$$RF_{freq} = K_{nom} + K_{vco} \cdot V_{in} \tag{B.1}$$

$$DIV_{freq} = \frac{K_{nom} + K_{vco} \cdot V_{in}}{8 + C_0 + 2 \cdot C_1 + 4 \cdot C_2} \tag{B.2}$$

Notice that the RF output signal is the high frequency signal that causes the simulation to slow down, while it is the divided down signal that is required to complete the loop. If the divided down signal can be produced without producing the high frequency signal then the simulation can be significantly speeded up. This can be achieved by moving the divider from the output of the VCO to the input, as shown in Figure B.2(b). Instead of dividing the output signal down in frequency, the input voltage is divided down in voltage. The control voltage is first multiplied by the gain of VCO, and the nominal frequency is added to this to produce a voltage which is proportional to the frequency of the output signal in Figure B.2(b). This signal is then divided by a behavioral programmable divider, to produce a voltage which is proportional to what the divided down frequency should be. This signal is then given to a VCO with a gain of 1 to produce the divided down signal. Notice that nowhere in this signal chain have we produced a signal with a frequency as high as the original output signal. Hence using this behavioral model for the VCO and divider the simulations speed can be significantly reduced. A similar technique can be found in [64].

BIBLIOGRAPHY

[1] T. A. Riley, M. A. Copeland, and T. A. Kwasniewski, “Delta-sigma modulation in fractional-N frequency synthesis,” *IEEE J. Solid-State Circuits*, vol. 28, no. 5, pp. 553-559, May 1993.

[2] E. Gotz, H. Krobel, G. Marzinger, B. Memmler, C. Munker, B. Neurauter, D. Romer, J. Rubach,W. Schelmbauer,M. Scholz, M. Simon, U. Steinacker, and C. Stoger, “A quad-band low power single chip direct conversion CMOS transceiver with ΣΔ-modulation loop for GSM,” in *Proc. Eur. Solid-State Circuits Conf.*, pp. 217–220, 2003.

[3] T. A. Riley and M. A. Copeland, “A simplified continuous phase modulator technique,” *IEEE Transactions on Circuits and Systems — II: Analog and Digital Signal Processing*, vol. 41, no. 5, pp. 321–328, May 1994.

[4] I. Galton, “Delta-Sigma Data Conversion in Wireless Transceivers,” *IEEE Transactions on Microwave Theory and Techniques*, vol. 50, no. 1, pp. 302-315, Jan 2002.

[5] E. V. Appleton, “Automatic synchronization of triode oscillators,” in *Proc. Cambridge Phil. Soc.,* vol. 21, pt. 111, p. 231, 1922-1923.

[6] 121 H. de Bellescize, “La reception synchrone,” *Onde Electr.,* vol. 11, pp. 230-240, June 1932.

[7] G. T. Volpe, “A phase-locked loop control system for a synchronous motor,” *IEEE Trans. Automat. Contr.,* vol. AC-15, pp. 88-95, Feb. 1970

[8] B. K. Bose and K. J. Jentzen, “Digital speed control of a dc motor with PLL regulation,” *IEEE Ind. Electron. Contr. Instrum.,* vol. IECI-25, pp. 10-13, Feb. 1978.

[9] K. Eapen, K. Venkatesan, and S. C. Gupta, “Steady state and stability analysis of an analog-type phase-locked loop dc motor control system,” *IEEE Trans. Ind. Electron. Contr. Instrum.,* vol. IECI-27, pp. 87-91, 1980.

[10] F. Harashima, H. Naitoh, M. Koyama, and S. Kondo, “Performance improvement in microprocessor-based digital PLL speed control system,” *IEEE Trans. Ind. Electron. Contr. Instrum.,* vol. IECI-28, pp. 5661, Feb. 1981.

[11] K. Eapen and K. Venkatesan, “Phase-locked loop dc motor drive with improved transient performance,” *IEEE Trans. Ind. EZectron. Contr. Instrum.,* vol. IECI-28, pp. 347-352, Nov. 1981.

[12] D. F. Geiger, Phase Lock Loops for DC Motor Speed Control. New York Wiley, 1981.

[13] N. Margaris and V. Petridis, “A phase-locked regulator system study of a separately excited dc motor with triangular phase comparator,” *Int. J. Electron.,* vol. 52, no. 3, pp. 241-261, Mar. 1982.

[14] -, “A phase-locked regulator system study of a separately excited dc motor with phase-frequency comparator,” *Int. J. Electron.,* vol. 52, no. 2, pp. 141-156, Feb. 1982.

[15] N. Margaris, V. Petridis, and D. Efthymiatos, “Phase-locked loop control of a nonlinear dc motor,” *IEEE Trans. Ind. Electron.,* vol. IE-29, pp. 91-93, Feb. 1982.

[16] N. Margaris and V. Petridis, “PLL speed regulation of fractional horsepower series and universal motors,” *IEEE Trans. Ind. Electron.,* vol. IE-31, pp. 277-281, Aug. 1984.

[17] G. Hsieh, and J. C. Hung, “Phase-Locked Loop Techniques-A Survey,” *IEEE Transactions on Industrial Electronics*, vol. 43, no. 6, December 1996.

[18] C. A. Kingsford-Smith, Patent No. 3,928,813 (Washington, D.C.: U.S. Patent Office, December 23, 1975)

[19] B. E. Bjerede and G. D. Fisher, “A New Phase Accumulator Approach to Frequency Synthesis,” in *Proc. IEEE NAECON 76*, p. 928, May 1976.

[20] S. Pamarti, L. Jansson, and I. Galton, “A wideband 2.4-GHz delta-sigma fractional-NPLL with 1-Mb/s in-loop modulation,” *IEEE Journal of Solid State Circuits*, vol. 39, no. 1, pp. 49-62, Jan 2004.

[21] T. A. Riley and M. A. Copeland, “A simplified continuous phase modulator technique,” *IEEE Transactions on Circuits and Systems — II: Analog and Digital Signal Processing*, vol. 41, no. 5, pp. 321–328, May 1994.

[22] B. Miller and B. Conley, “A Multiple Modulator Fractional Divider,” in *Proc. 44th Annual Symp. on Freq. Control,* pp. 559–567, May 1990.

[23] B. Miller and B. Conley, “A Multiple Modulator Fractional Divider,” *IEEETrans. Instrumentation and Meas*., vol. 40, no. 3, pp. 578–583, June 1991.

[24] T. Aytur and B. Razavi, “A 2 Ghz, 6 mW BiCMOS frequency synthesizer,” in*Proceedings of IEEE International Solid-State Circuits Conference*, pp. 264–5,1995.

[25] B. Chang, J. Park, and W. Kim, “A 1.2 GHz CMOS dual-modulus prescaler using new dynamic D- type flip-flops,” *IEEE Journal of Solid State Circuits*, vol. 31, no. 5, pp. 749–52, May 1996.

[26] N. Foroudi and T. A. Kwasniewski, “CMOS High-Speed Dual-Modulus Frequency Divider for RF Frequency Synthesis,” *IEEE Journal of Solid State Circuits*, vol. 30, no. 2, pp. 93–100, Feb. 1995.

[27] J. Craninckx and M. S. Steyaert, “A 1.75-GHz/3-V Dual-Modulus Divide-by-128/129 Prescaler in 0.7-um CMOS,” *IEEE Journal of Solid State Circuits*, vol. 31, no. 7, pp. 890–897, July 1996.

[28] T. Kamoto, N. Adachi, and K. Yamashita, “High-speed multi-modulus prescaler IC,” in *Fourth IEEE International Conference on Universal Personal Communications*, pp. 991, 325–8, 1995.

[29] M.H. Perrott, “Techniques for High Data Rate Modulation and Low Power Operation of Fractional-N Frequency Synthesizers”, Ph.D. dissertation, MIT, Cambridge, MA, 1997.

[30] M. H. Perrott, T. L. Tewksbury, and C. G. Sodini, “A 27mW Fractional-N Synthesizer/Modulator IC,” *ISSCC Dig. Tech. Papers*, 1997, pp. 386–387.

[31] M. H. Perrott, T. L. Tewksbury, and C. G. Sodini, “A 27-mW CMOS fractional-N synthesizer using digital compensation for 2.5-Mb/s GFSK modulation,” *IEEE Journal of Solid State Circuits*, vol. 32, no. 12, pp. 2048–2060, Dec. 1997.

[32] R. A. Meyers and P. H. Waters, “Synthesizer review for pan-European digital cellular radio,” in *Proc. IEEE VLSI Implementations for Second Generation Digital Cordless and Mobile Telecommunication Systems Colloq.*, pp. 8/1-8/8, 1990.

[33] R.B. Staszewski, S. Vemulapalli, P. Vallur, J. Wallberg, P.T. Balsara, “1.3 V 20 ps time-to-digital converter for frequency synthesis in 90-nm CMOS”; *IEEE Transactions on Circuits and Systems II*, vol. 53, no. 3, Pages 220 – 224, March 2006.

[34] C.M. Ming, M. Z. Straayer, M.H. Perrott “A Low-Noise, Wide-BW 3.6GHz Digital ΔΣ Fractional-N Frequency Synthesizer with a Noise-Shaping Time-to-Digital Converter and Quantization Noise Cancellation”, *ISSCC Dig. Tech. Papers*, pp. 340–341, 2008.

[35] B. M. Helal, M. Z. Straayer, G. Y. Wei and M. H. Perrott, “A Low Jitter 1.6 GHz Multiplying DLL Utilizing a Scrambling Time-to-Digital Converter and Digital Correlation”, *VLSI Symp. Dig. Tech. Papers*, pp. 302-303, Feb., 2007.

[36] R.B. Staszewski, et al., “All-Digital PLL and Transmitter for Mobile Phones,” *IEEE J. Solid-State Circuits*, vol. 40, pp. 2469-2482, Dec. 2005.

[37] M.H. Perrott, M.D. Trott, “A Modeling Approach for Σ-Δ Fractional-N Frequency Synthesizers Allowing Straight forward Noise Analysis,” *IEEE J. Solid-State Circuits*, vol. 37, pp. 1028-1038, Aug. 2002.

[38] M. Schwartz, *Information Transmission, Modulation, and Noise*, 4th edition, McGraw Hill, 1990.

[39] S. Haykin, *Communication Systems*, 4th edition, J. Wiley & Sons, 2001.

[40] L. W. Couch, II, *Digital and Analog Communication Systems*, 6th edition, Prentice Hall, 2001.

[41] J. G. Proakis, D. G. Manolakis, *Digital Signal Processing*, 3rd edition, Prentice Hall, 2001.

[42] R. Beards and M. Copeland, “An oversampling delta-sigma frequency discriminator,” *IEEE Transactions on Circuits and Systems II*, vol. 41, no. 1, pp. 26 – 32, January 1994.

[43] W. Bax, T. Riley, C. Plett, and M. Copeland “A ΣΔ Frequency Discriminator Based Synthesizer,” *International symposium on Circuits and Systems*, 1995.

[44] W Bax and M Copeland, “A GMSK Modulator Using ΔΣ Frequency Discriminator-Based Synthesizer,” *IEEE J. Solid-State Circuits*, vol. 36, no. 8, pp. 1218-1227, Aug. 2001.

[45] M. Gupta and B. Song, “A 1.8-GHz Spur-Cancelled Fractional-N Frequency Synthesizer With LMS-Based DAC Gain Calibration,” *IEEE J. Solid-State Circuits*, vol. 41, no. 12, pp. 2842-2851, Dec. 2006.

[46] S.R. Norsworthy, R. Schreier, G.C. Temes, *Delta-Sigma Data Converters,* New Jersey, IEEE Press, 1997.

[47] F. Kocer, “A new Transponder architecture for long range telemetry applications,” Ph.D. dissertation, University of Michigan, Ann Arbor, MI, 2005.

[48] F. M. Garnder, *Phaselock Techniques,* 3rd edition, New Jersey, Wiley, 2005.

[49] J. A. Crawford, *Frequency synthesizer Handbook ,*Norwood, MA:Artech,1994.

[50] L. DeVito, et al., "A 52MHz and 155MHz clock recovery PLL," *ISSCC Dig. Tech. Papers*, pp. 142-143, Feb. 1991.

[51] T.H. Lee, J.F. Bulzacchelli, "A 155-MHz Clock Recovery Delay- and Phase-Locked loop", IEEE J. Solid-State Circuits, Volume 27, Issue 7, December 1992, Pages: 1736 – 1746.

[52] M.P. Perrott "PLL design using the PLL design assistant", Available at: http://www-mtl.mit.edu/researchgroups/perrottgroup/cgibin/load.cgi?/~perrott/Distribution/PllDesign/pll_manual.pdf

[53] T.H. Lee, J.F. Bulzacchelli, "A 155-MHz Clock Recovery Delay- and Phase-Locked loop", *ISSCC Dig. Tech. Papers*, pp. 160-161, Feb. 1992.

[54] K.C. Peng, C.H. Huang, C.N. Pan, T.S. Horng, "High-Performance Bluetooth transmitters using two-point delta-sigma modulation"; *Electronic Letters*, vol. 40, Number 9, April 2004.

[55] C.M. Hung, R.B. Staszewski, N. Barton, M.C. Lee, D. Leipold, "A Digitally Controlled Oscillator System for SAW-Less Transmitters in Cellular Handsets"; *IEEE J. Solid-State Circuits*, vol. 41, no. 5, pp. 1160-1170, May 2006.

[56] http://www.newwaveinstruments.com/resources/articles/m_sequence_linear_feedback_shift_register_lfsr.htm

[57] C.S. Vaucher, I Ferencic, M Locher, S. Sedvallson, U Voegeli, Z Wang, "A family of low-power truly modular programmable dividers in standard 0.35-μm CMOS technology", *IEEE J. Solid-State Circuits*, vol. 35, no. 7, pp. 1039–1045, July 2000.

[58] W. Kluge, F. Poegel, H. Roller, M. Lange, T. Ferchland, L. Dathe, D. Eggert, "A Fully Integrated 2.4-GHz IEEE 802.15.4-Compliant Transceiver for ZigBee™ Applications", *IEEE J. Solid-State Circuits*, vol. 41, no. 12, pp. 2767-2775, Dec 2006.

[59] J. Notor, A. Caviglia, G. Levy, "CMOS RFIC architectures for IEEE 802.15.4 networks," Available at http://www.cadence.com/whitepapers/, 2003.

[60] S. Pamarti, L. Jansson, and I. Galton, “A wideband 2.4 GHz delta-sigma fractional-N PLL with 1 Mb/s in-loop modulation,” *IEEE J. Solid-State Circuits*, vol. 39, no. 1, pp. 49–62, Jan. 2004.

[61] E.Temporiti, G. Albasini, I. Bietti, R. Castello, and M.Colombo, “A700 kHz bandwidth fractional synthesizer with spurs compensation and linearization technique for WCDMA applications,” *IEEE J. Solid-State Circuits*, vol. 39, no. 9, pp. 1446–1454, Sep. 2004.

[62] S. E. Meninger, M. H. Perrott, “A 1-MHz Bandwidth 3.6-GHz 0.18-um CMOS Fractional-N Synthesizer Utilizing a Hybrid PFD/DAC Structure for Reduced Broadband Phase Noise”, *IEEE J. Solid-State Circuits*, vol. 41, no. 4, pp. 996–980, Apr. 2006.

[63] M. H. Perrott, “CppSim Behavioral Simulator Package”, Available at: http://www-mtl.mit.edu/researchgroups/perrottgroup/tools.html

[64] M.H. Perrott, "Fast and Accurate Behavioral Simulation of Fractional-N Synthesizers and other PLL/DLL Circuits", *Design Automation Conference (DAC)*, pp. 498-503, Jun. 2002.

[65] M. Ferriss and M. P. Flynn, “A 14mW Fractional-N PLL modulator with an enhanced digital phase detector and frequency switching scheme” *ISSCC Dig. Tech. Papers*, pp. 353–353, 2007.