DLL-based Fractional-N Frequency Synthesizers

by

Farhad Zarkeshvari, B.Sc., M.Sc., M. Eng.

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

Doctor of Philosophy in Electrical Engineering

Ottawa-Carleton Institute For Electrical and Computer Engineering
Department of Electronics
Faculty of Engineering
Carleton University
Ottawa, Ontario
December 2011

Library and Archives Canada

Published Heritage Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et Archives Canada

Direction du Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 978-0-494-87754-8

Our file Notre référence
ISBN: 978-0-494-87754-8

Abstract

Fractional-N phase locked loops (PLLs), widely used for clock/frequency generation in communication/digital systems, offer a frequency resolution tighter than their reference frequency. This work introduces an alternative delay locked loop (DLL) based fractional-N frequency/clock synthesizer and proposes a few architectures for it. A DLL-based synthesizer offers less close-in phase noise as it does not have the jitter accumulation problem of its PLL-based counterpart. Instead, a DLL-based synthesizer poses out-of-band spurious tones in its output frequency spectrum, a reference frequency away from main (carrier) frequency. The level of the spurious tones can be reduced by good design practice. A DLL-based synthesizer is most suitable for compact integration in complementary metal oxide semiconductor (CMOS) technology and can be competitive in terms of power consumption and controllability. The DLL-based fractional-N synthesizer is generally competitive with its PLL-based counterpart but it looks particularly appealing for applications sensitive to close-in phase noise and/or when a high quality LC oscillator (resonance oscillator using inductor and capacitor) is not affordable.

This work also discusses implementation issues of a DLL-based fractional-N frequency synthesizer, explores different topologies and related circuit techniques for all the building blocks and describes associated trade-offs. At the circuit level, this research proposes two novel period synthesis techniques and a unique combination of a charge pump and a voltage to current converter for the delay locked loop.

A practical implementation targeting high-speed serial links applications in deep sub-micron technology is used as a test vehicle for proof of concept and to showcase the abilities and benefits of these architectures. Among the proposed architectures, the single loop one, with the best performance complexity trade-off and the ability of generating low spur levels for fractional multiplication factors close to an integer value is chosen for implementation. This proposed architecture equipped with a $\Delta\Sigma$ modulator, phase interpolator based period synthesis, a differential dual compensated charge pump and a voltage to current converter to generate a fractional delay and achieves acceptable performance. The implemented synthesizer provides output frequencies between 4 GHz and 5GHz with total jitter less than 15ps when consuming about 11.5mW of power at a nominal supply voltage of 1V.

Acknowledgement

I would like to thank my thesis supervisor, Professor Tadeusz Kwasniewski, for his great contribution to this research. Professor Kwasniewski was a real mentor and a constant source of courage and support in this long journey. I also want to thank other faculty members of Department of Electronics, especially Calvin Plett and Ralf Mason, who taught me the art of analog/mixed-signal/RF design, and John Knight for his valuable discussions on digital part of the research.

I would like to acknowledge my former classmates, faculty and staff of department of electronics, whose valuable discussion, valuable hints and encouragements helped me during this research. Especially I am obliged to Peter Noel, Gord Alan, Reza Yousefi, Michel Nakhla, Langis Roy, Irfan Ahmed.

At the end I would like to especially thank my wife, Nazli Javidan, and my son Sam Alan, who put up with long hours, late nights and weekends research; and my parents, Manouchehr and Nahid, for their constant encouragements.

Table of Content

List of Figures

List of Tables

List of abbreviations and symbols:

AC	alternating current
ADC	analog to digital converter
AFC	automatic frequency control
ALPHA	a measure for supply sensitivity define in equation 5.1
AMPS	a family of wireless standards for cellular phone applications
CCBC	current control buffer cell
CCDL	current control delay line
CDR	clock and data recovery
CLK	system clock
CMOS	complementary metal oxide semiconductor
CP	charge pump
DAC	digital to analog converter
DAS	direct analog synthesis
DC	direct current
DDR4	fourth generation of double data rate memory connection standard
DDS	direct digital synthesis
DHDL	digitally-controlled half replica delay line
DLL	delay locked loop
DNL	differential non-linearity
EMI	electro magnetic interference
f_{ref}	reference clock frequency
GDDR5	fifth generation of double data rate memory standard for graphic applications
GSM	global system for mobile communication

IF	intermediate frequency
INL	integrated non-linearity
LF	loop filter
LPF	low pass filter
LSB	least significant bit
LUT	look up table
MASH	multi –stage noise shaping
MDLL	multiplying delay locked loop
MISO	multiple input single output
MOSFET	metal oxide field effect transistor
NMOS	N channel MOSFET
PCIe	peripheral component interconnect express
PD	phase detector
PFD	phase frequency detector
PI	phase interpolator
PLL	phase locked loop
PMOS	P channel MOSFET
PS	period synthesis
PSS	power supply sensitivity
PVT	process voltage temperature
Q	fractional multiplication factor
QAM	quadrature amplitude modulation
RC	resistive capacitive
RF	radio frequency
TETRA	terrestrial trunked radio; a wireless standard for walkie-talkie
T_{ref}	reference clock period
t_d	delay of one delay cell

t_e	in lock error result in fractional-N DLL
UI	unit interval
VCC	common collector voltage; refer to power supply in the circuit
VCDL	voltage control delay line
VCO	voltage control oscillator
VIC	V (voltage) to I (current) converter
VL	variable length
α	fractional part of frequency multiplication factor
β	fractional part of the ratio of reference period and delay cell delay in period synthesis

Introduction

Any synchronous system requires a clock (rectangular wave) or frequency (single sinusoidal tone). As technology advances, the clock rate of digital systems and the data rate for communication links increase which consequently reduce tolerable timing uncertainty in the system. This increases the need for a solid frequency synthesizer to generate the desired clock with the required quality.

Timing uncertainty as a measure of the clock/frequency quality is expressed by phase noise in the frequency domain or jitter in the time domain. In addition to output clock/frequency quality, other factors like channel spacing, power consumption, flexibility and controllability, integration capability, silicon area consumption, and portability between different fabrication technologies impact the architecture and design of frequency/clock synthesizers.

Phase locked loop (PLL) based synthesizers are widely used for clock or frequency generation. Fractional-N PLL relaxes the stringent relation between input reference frequency and output frequency and lets the synthesizer have higher loop bandwidth while satisfying the target channel spacing. Voltage controlled oscillator is a major source of the phase noise in a PLL. Although high loop bandwidth of a fractional-N PLL reduces the VCO noise contribution, the accumulative nature of phase noise in PLL sets a practical limit on possible close-in phase noise reduction. DLL, on the other hand, does not accumulate jitter and thus offers very low close-in phase noise. However, the frequency multiplication factor for DLL based synthesizers is inherently integer and

small. This prevents using DLL for frequency synthesis in spite of its excellent phase noise performance.

This research proposes the techniques to overcome the obstacles of making fractional DLL and introduces several novel architectures for DLL-based fractional-N frequency/clock synthesis. These architectures are alternatives to PLL-based synthesizers and offer competitive and in some aspects superior performance. A DLL-based fractional-N synthesizer shows less jitter/phase noise compared to a ring oscillator PLL based synthesizer. It also has wider frequency range, smaller silicon footprint and area consumption, better portability to a different fabrication technology, and competitive jitter/phase noise performance compared to LC-VCO PLL based counterparts.

This research investigates the circuit techniques for the DLL-based fractional-N synthesizer and explores and compares different topologies for all of its building blocks. The proposed single loop DLL-based fractional-N architecture is selected for implementation to showcase the advantages and challenges of a practical design. The implemented synthesizer is equipped with a novel period synthesis technique, a fully differential and compensated charge pump and a voltage to current converter, and also a variety of analog loops and digital control knobs to compensate for global and local process variations. The implemented single loop DLL-based fractional-N synthesizer performs with less than 15ps total jitter at 4GHz to 5GHz range in nominal voltage and hot temperature in 45nm CMOS technology.

Thesis manuscript contains seven chapters and starts with a summary of thesis objective, motivation behind the research, expectations and possible target applications. It follows with an extensive background and literature review in the next two chapters.

Chapter two presents the fundamental concept of frequency synthesis and describes its related architectural and circuit level techniques. Topics of phase locked loop (PLL), delay locked loop (DLL), phase noise and jitter and their corresponding equations are covered in this chapter.

Chapter 3 focuses on the PLL-based fractional-N frequency synthesizers and covers the historic evolution of these synthesizers to the state of the art $\Delta\Sigma$ fractional PLL synthesizers. $\Delta\Sigma$ modulation in the context of the fractional-N frequency synthesis is presented, different types of $\Delta\Sigma$-modulators are explored and their impacts on the synthesizer performance are explained. Other related techniques including period synthesis, multiplying DLLs and multiphase synthesizers are covered in this chapter.

Chapter 4 introduces the concept of the DLL-based fractional-N frequency synthesis and explains the challenges and the bottlenecks. Inherent in-lock-error in fractional-N DLLs is investigated and several compensation techniques are presented. Proposed architectures for DLL-based fractional-N frequency synthesizer are presented in this chapter. A comparison of main four proposed architectures and a summary of their advantages and disadvantages are given in the end of Chapter 4.

Chapter 5 presents the circuit design consideration for all of the synthesizer building blocks. Different topologies for every sub-block are explored, compared and the best one for the target architecture and application is chosen. Circuit implementation of single

DLL-based fractional-N frequency synthesizer is explained. Interaction between the blocks and timing/switching issues are discussed in detail.

Chapter 6 presents the performance results for the implemented single loop DLL-based fractional-N frequency synthesizer. All the major building blocks are characterized and related issues are discussed.

Thesis conclusion is presented in Chapter7.

Chapter 1 Motivation

1.1 Motivation

Frequency/Clock synthesizer is an important part of any synchronous communication system. From RF transceivers to long haul wire lines, optical lines and high speed interconnects all require a frequency/clock with certain specifications. Depending on the application, frequency synthesizer need to cover range of frequency with specific channel spacing and meet phase noise or jitter requirement. Power consumption, silicon area and portability to different CMOS technology are other important factors for an integrated frequency synthesizer.

A DLL-based frequency synthesizer can be fully integrated. It offers superior phase-noise performance compared to other integrated counterparts [38]. However, it has extremely poor frequency selectivity due to the following reasons. First, to maintain its superior phase-noise performance, a DLL-based frequency synthesizer tends to have a short delay line and a small multiplication factor. Consequently a high reference frequency has to be used to produce the desired output frequency. Second, the edge-combiner structure presents a limitation leading to the use of an integer multiplication factor. The integer multiplication factor together with the high reference frequency results in a very low frequency selectivity. In applications where selectivity is not an issue and low phase noise is required, a DLL-based synthesizer can be an excellent choice for the local oscillator. In practice, DLL-based synthesizers have been used as low phase-noise single-

frequency synthesizers in a double conversion wide-band receiver, where the whole RF band down converts to an intermediate frequency (IF) and channel selection is performed by a second oscillator in the IF band [38, 39].

However, in the most commonly used transceiver architectures (super-heterodyne, direct-conversion, low-IF architectures, high speed I/Os) channel selection is performed at high frequency (RF) band by changing the local oscillator frequency. In this thesis, novel fractional-N DLL-based synthesizer architectures are proposed which provide the required frequency selectivity for popular kinds of transceivers. The proposed architectures benefit from the low close-in phase-noise of a highly integrated DLL-based synthesizer, and the small channel spacing of fractional-N architectures.

1.2 Target Applications

High speed I/O in today technology covers a wide range of applications from chip to chip communication (like front side bus or memory connection in a mother board) to serial link back plans in a large system. Transceivers not only need to comply to latest protocol but in many cases need to be backward compatible to oldest versions which work in a different data rate and thus need a different clock. For example most of the PCIe links need to be compatible with PCIe Gen1,2, and 3 which have data rate of 2.5, 5 and 8 Gb/s, respectively. The new trend is to design the links to work under multiple different protocols which not only have different data rates, but their required frequencies may not be an integer multiply of an available reference clock. In many links it is desired to reduce bit rate to save power, slightly change bit rate to get away from notch of a specific bad channel or cope with a weak receiver.

A DLL-based fractional-N synthesizer is a good candidate for all these cases for the following reasons. First, it can provide high quality clock for both data transmission and clock/data recovery (CDR) circuit. Second, using a fractional-N DLL-based synthesizer offers the advantage and flexibility to work under several protocols/standards with various data rates and different clock frequencies. Third, inductor free structure provides less area consumption and better silicon foot print which helps to have more efficient full chip floor planning and consequently have more saving on silicon area. Fourth, wider frequency range helps to cover desire frequency range with components of only one synthesizer set (doesn't need multiple LC-VCOs to cover the desired range of frequency).

DLL-based Fractional-N architecture also can be a good candidate for wireless applications where a very low close-in phase noise is required or an LC-VCO is not affordable. It is also an excellent candidate when a high frequency clean reference is available.

1.3 Objectives

The objectives of this thesis are:

1. Investigate the possibility of having DLL-based fractional-N frequency synthesizer
2. Develop techniques to remove architectural obstacles of making a fractional-N DLL
3. Develop DLL-based fractional-N synthesizer architecture
4. Proof of concept implementation of DLL-based fractional-N frequency synthesizer

5. Verify the advantages and disadvantages of the implemented architecture

1.4 Contributions

The contributions of this research are outlined as follows:

1. Discovered the obstacles and bottle necks of a practical fractional-N DLL
2. Introduced DLL-based fractional-N frequency synthesizer and presented its related techniques and equations
3. Proposed two novel period synthesis architectures
4. Proposed four novel DLL-based fractional-N frequency synthesizer architectures with different complexity-performance trade-offs
5. Patented DLL-based fractional-N frequency synthesizer
6. Addressed the architectural and circuit level issues related to proposed architectures
7. Implemented fully integrated DLL-based fractional-N synthesizer in deep sub-micron CMOS technology

Chapter 2 Frequency Synthesis; Concept and Techniques

2.1 Introduction

In today's technology, synchronous schemes are used in the most of the practical communication and digital systems. A local frequency is required in both transmitter and receiver paths of a radio communication scheme to up-convert and down-convert information signals, respectively. Digital data is synchronized by a system clock and retimed in a baseband communication scheme. The performance of all these synchronous schemes directly depends on the performance of local frequency or clock. Generating a pure frequency/clock with low phase noise/jitter, low spur, and good stability is an important part of designing a high-speed high-performance communication/digital system. There are many frequency synthesis techniques to generate such a waveform. Each has different complexity, power consumption, cost, integrability, tuning range, and maximum realizable frequency.

In many applications, the target frequency may change either continuously or occasionally. For example, changing the communication channel in wireless systems or changing the communication scheme in a multi-standard communication device. In addition, some schemes like frequency hopping require continuous change of synthesized frequency. Fast frequency switching is an important requirement in these applications.

Frequency synthesis methods are mainly categorized into two classes, direct synthesis and indirect synthesis. Both classes of techniques are discussed in this chapter. Indirect synthesis techniques are more popular due to their complexity-performance trade-off and flexibility.

2.2 Direct Frequency Synthesis

In this class of Frequency Synthesis, desired frequency is generated directly from a source (i.e. stable oscillator) and there is no feedback to control the output. Two architectures for direct frequency synthesis are Direct Digital Synthesis (DDS) and Direct Analog Synthesis (DAS).

2.2.1 Direct Digital Synthesis (DDS)

The block diagram of a standard DDS is shown in Figure 2.2.1. The architecture usually consists of four main components: adder/accumulator, phase to amplitude converter, Digital to Analog Converter (DAC) and Low-Pass Filter (LPF). In each clock period, a predetermined digital value N is added to the value of the accumulator. The accumulator value indicates the phase of the output waveform. A Look-Up-Table (LUT) converts this phase to the digital value of the desired waveform amplitude, which may be converted to actual amplitude voltage by a DAC. The DAC output is connected to a LPF to form the final output waveform.

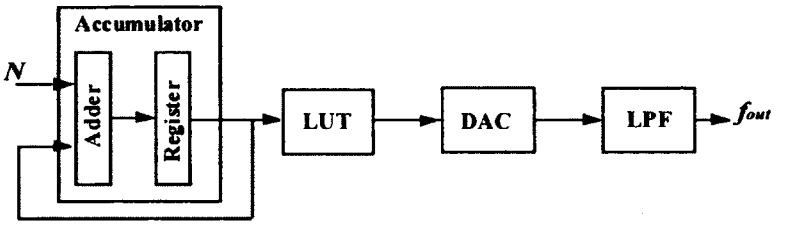


Figure 2.2.1: Diagram of a standard DDS

The frequency of the output waveform is

$$f_{out}=(N/2^M).f_{clk} \quad (2.1)$$

where M is the number of the bits of the accumulator, f_{clk} is the clock frequency and N is the number that is added to the accumulator value in each clock cycle. The output frequency can be changed easily and quickly by varying N. The output frequency is always lower than the clock frequency. The typical maximum output frequency is required to be $f_{clk}/2$ to meet the Nyquist criteria. The LUT maps the instantaneous phase to discrete samples with quantized amplitude of desired waveform. The resolution of DDS can be increased by having a larger LUT. Increasing the size of accumulator reduces the overflow frequency but results in a slower addition due to carry and also larger area and power consumption. The output of LUT has to have the same number of bits as DAC input. In fact, the DAC is the least ideal component of DDS and the main performance bottleneck. Transient quantization error and high frequency spurs result from DAC differential and integral non-linearity. The slew rate also creates harmonic distortion. The advantage of DDS is its capability to switch quickly from one frequency

to another. DDS can also be used as a fractional divider. However, DDS suffers from high-frequency spurs and phase noise and its huge power consumption.

2.2.2 Direct Analog Synthesis (DAS)

In the direct analog synthesis (DAS) approach, the target frequency is produced by adding, subtracting and dividing the reference frequency using analog components. The mixer and the divider are the two main components in DAS. The mixer can be used for frequency addition and subtraction. An example of this approach is shown in Figure 2.3.1. The target frequency can be a fractional multiplication of the reference frequency. Switching to a new target frequency can be performed rapidly using this technique.

DAS is not suitable for integration. It requires a large number of components and the resulting circuit is bulky and power consuming. The non-linear elements in the circuit also produce spurious sidebands. Thus, DAS is not considered to be a good solution for monolithic implementations.

2.3 Indirect Frequency Synthesis

In this kind of frequency synthesis, the generation of desired frequency is achieved through interaction of two or more waveform sources and a feedback loop to control the

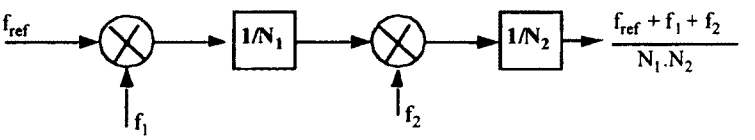


Figure 2.3.1: A Sample DAS approach

output frequency. Indirect frequency synthesis is usually based on phase or delay locking techniques.

2.3.1 Phase Locked Loops

Phase-locking is a well known technique that has been used for more than half a century in many different applications including frequency synthesis, phase modulation/demodulation, clock and carrier recovery, and tracking filters. Phase locked loops (PLLs) demonstrate excellent performance in high speed applications and ease of integrated implementation. As shown in Figure 2.3.2, PLLs use a negative feedback loop to force the phase of output signal to follow the phase of input signal. Phase detector (PD) generates an error signal based on the difference between input and output signals. This error signal is integrated in a loop filter and applied to a voltage controlled oscillator (VCO) to force the phase of its output signal to lock with the phase of the input signal. A measure of the PLL's performance is the phase error between the input signal and the output signal in the lock position. A PLL is also characterized by other main factors including lock-in range, lock-in time, pull-in range, pull-out range, hold-in range, etc. Lock-in range is the maximum frequency difference for which neither discontinuity nor cycle skipping is expected. Lock-in time is the time it takes that the PLL locks to an acceptable value of phase error.

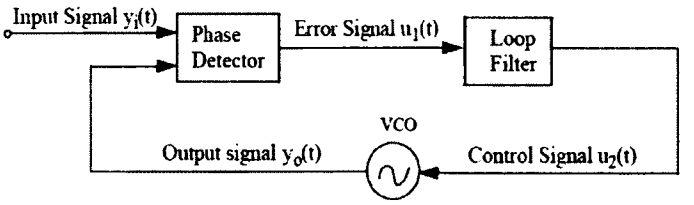


Figure 2.3.2: Basic diagram of a PLL

Pull-in range is the maximum frequency difference for which phase lock can always be acquired. Pull-out range is the largest frequency step below which the loop does not skip cycle and remain locked. In addition, loop stability and output phase noise are important performance factors in frequency synthesis context.

2.3.2 PLL-based Frequency Synthesis

As DDS fails to operate at high frequencies and DAS is too bulky for integration, the indirect frequency synthesis using phase locking technique becomes a good alternative for high speed monolithic applications. The block diagram of PLL-based frequency synthesizer is shown in Figure 2.3.3. The PLL is modified by locating a divider in the feedback loop to achieve the frequency multiplication function.

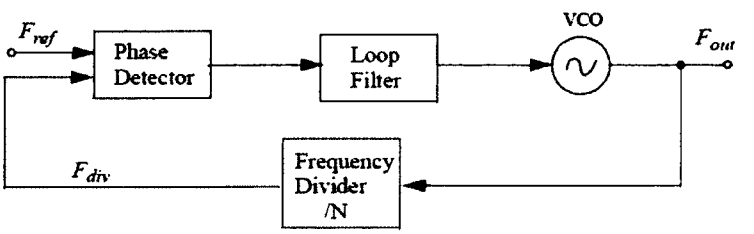


Figure 2.3.3: PLL-based frequency synthesis

The VCO output frequency is divided by the divider modulus, N, and then compared to the reference frequency by PD. The error signal generated by PD, is filtered by a low pass filter (LPF). Then, the filtered error signal tunes the VCO. In lock, the VCO frequency will be N times the reference frequency. Note that N is an integer number and, thus, this architecture is called integer-N frequency synthesizer. The loop filter (LF) has a significant effect on the PLL and the synthesizer operation. The main task of the LF is to suppress the undesired high frequency components of the PD-CP outputs. The LF has a great impact on loop stability, output noise and acquisition time. In general, indirect synthesizers have slower frequency switching than direct synthesizers because of the loop filter.

The high frequency parts of PLL-based synthesizer are the VCO and the first stage of the divider (prescaler). They are also the main power consuming blocks, especially in a CMOS implementation. The noise produced by the loop components can be reduced using an accurate PD, a high quality VCO, and a high speed divider. However, the loop inherently acts like a low-pass filter to noise at the input of PLL (from the reference) and also like a high-pass filter for internal noise of the VCO. As the major source of noise in a VCO is the low frequency noise, the loop filter inherently rejects a considerable part of the VCO noise. The design of a loop filter is a compromise among switching time, frequency settling, the VCO output noise, etc.

2.3.3 Delay Locked Loops

A block diagram of a delay locked loop (DLL) is shown in Figure 2.3.4 It consists of four major blocks: a voltage controlled delay line (VCDL), a PD, a charge pump (CP), and a LF. DLL uses a negative feedback loop to force the overall delay of the VCDL to follow the period of the input signal. The PD generates an error signal based on the timing gap between the rising (falling) edges of the input and the output signals of the VCDL. This error signal is integrated in a loop filter and applied to the VCDL to lock its delay to an integer multiple of the input signal period (usually one period). The achieved resolution of the DLL is limited by the minimum gate delay available in a given technology. DLLs are used for clock alignment, phase/delay adjustments, time interpolation (interval measurement), clock/data recovery and clock/frequency synthesis.

DLLs can be categorized into two main groups, digital and analog. A digital DLL consists of a digital delay line and other digital elements which makes it portable across multiple processes. Digital DLLs may use a digitally controlled delay cell. The digitally controlled delay cell can be implemented by a standard buffer with a digitally controlled capacitor at its output [1]. This digitally controlled capacitor is made of a parallel connection of a large number of small parallel capacitors (usually smallest possible in a given technology) series with a pass switch [1,2]. The capacitance value is determined by the number of ON switches; usually by applying a thermometer code as the digital control signal. In [1] the difference between the parasitic capacitance of a pass switch in ON and OFF condition is used to increment or decrement the digitally controlled capacitor. In addition fine tuning is achieved by half switching the pass switch; i.e., charging the gates of the pass switch transistors to an intermediate voltage [1].

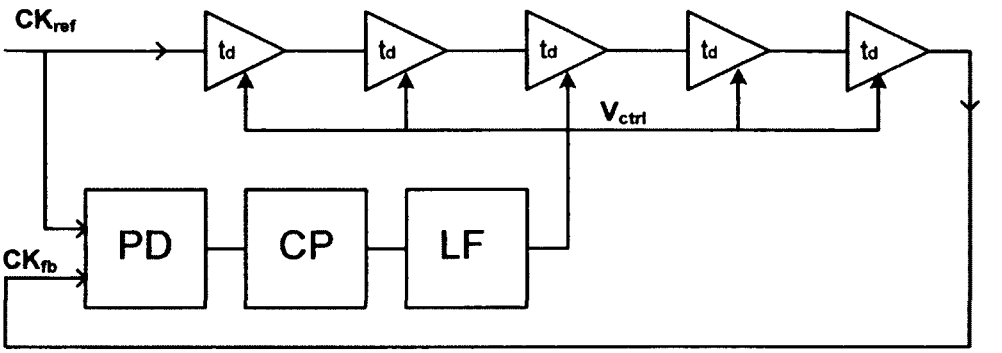


Figure 2.3.4: A typical Delay Locked Loop (DLL)

In another approach fine tuning is performed by switching one of the capacitors with an over sampling rate (much more than the reference clock) using a $\Delta\sum$ modulator [2]. Fast timing recovery is possible after placing the digital DLL in a low power mode because the phase information can be stored as a digital state. This topology requires a lower supply voltage to operate. Major drawbacks of the digital DLL are quantized delay time and large jitter. The phase blending technique [3] is proposed to decrease the quantization effect and improves the phase resolution. However, the inherent problem of the digital DLL is not solved entirely.

The analog DLL typically has a better jitter performance, less area, and lower power consumption. The differential delay cell and its good power supply rejection ratio can result in even less jitter. Dual loop DLL [4] is proposed to minimize the problems of the digital DLL. In dual loop architecture, a fine delay line controlled by an analog signal is attached to a digital DLL in the subsequent stage. In [4], a phase interpolator is cascaded with a digital DLL to increase the phase capture range. These dual loop architectures achieve both wide frequency range and relatively lower jitter performance at the expense of a significant increase in area and power consumption.

In comparison to PLLs, the DLL has unconditional stability and faster locking time. VCDLs introduce much less jitter than VCOs because generally delaying a signal entails less uncertainty than generating it. As the delay line does not have memory, the phase noise does not accumulate in DLL for more than a reference cycle. Lower phase noise accumulation is one of the main advantages of DLLs in the frequency/clock synthesis context. First order DLL has a first order loop transfer function unlike PLL small signal model which has a minimum second order transfer function. A wider loop bandwidth can be used since the transfer function is inherently stable. This allows the use of smaller loop filter capacitors and a faster acquisition time. Any correlated timing error (caused by a slow change of the control voltage) within the loop bandwidth is corrected by the loop. Larger loop bandwidth also helps to suppress more noise. Note that the loop just corrects the total delay of the delay line, i.e. the average delay of the delay cells and not the individual delay of each delay cell. Thus, any uncorrelated delay due to mismatch would be left unchanged which consequently generate spurs at the output of the DLL-based frequency synthesizer. This causes jitter (phase noise or spurs) in the DLL-based frequency synthesizers. A PLL resorts to a specific type of PD, the state-machine-based phase frequency detector (PFD), due to frequency acquisition constraints. On the other hand, a DLL can be implemented even using bang-bang control signal, a simple binary up and down instead of being proportional to the phase error magnitude.

Reference jitter propagation and limited phase capture range are the two main disadvantages of a conventional DLL. Thus, self-biased DLL [5] and digital-controlled half-replica delay line (DHDL) DLL [6] are proposed. The main idea of the self-biasing technique is to allow circuits to choose the operating bias levels in which they function

best. In a DLL this happens by adjusting the CP current to a linear function of the bias current of the delay cells (I_{bias}) i.e. varying the CP current with control voltage. This cancels the $\frac{1}{I_{bias}}$ dependence of the delay line gain and produces a constant ratio of the loop bandwidth to the operating frequency. In fact, bandwidth tracks the operating frequency and, thus, DLL can work over a wider range of frequency at the expense of jitter performance degradation. In practice, the non-ideality results in a charge leakage which causes voltage ripple on the loop capacitor.

In the DHDL approach, a larger CP current is used in the initialization state to reduce the lock time, and a smaller CP current is used in the lock state, to improve jitter performance.

2.3.4 DLL-based Frequency Synthesis

Traditionally PLLs have been used for clock/frequency synthesis. Recently there is a growing interest in using DLL due to its distinct characteristics. DLL is more stable than higher order PLLs and requires only one capacitor in its first-order loop filter. It offers better jitter performance due to lack of jitter accumulating and also easier integrability. One of the distinguishing characteristics of DLL is the easy access to the different phases of the input signal (different delayed versions of the reference signal). This unique characteristic is used in almost all the DLL applications. In clock/frequency synthesis, this accessibility is used to generate the multiplication factor. The block diagram of the DLL-based clock/frequency synthesizer is shown in Figure 2.3.5. The synthesizer consists of a DLL and one extra functional block - the edge combiner. In the edge

combiner, multiple reference signal phases from DLL are combined using digital logic to produce the synthesizer output.

2.3.4.1 Jitter and Spurs in DLL-based Synthesis

Random timing errors accumulate in an oscillator because the timing jitter at the end of each oscillation period alters the starting point of every following oscillation. The random timing error of the output is the sum of the timing errors of all previous oscillations. In contrast, for a DLL-based synthesizer, the output waveform is generated by the evenly spaced delayed versions of one reference edge. Thus, when a new reference edge appears at the input of the delay line all previous timing errors are reset. Therefore, the random timing errors accumulate only within one reference cycle. This results in an excellent long-term jitter performance and equivalently low close-in phase noise performance for a DLL-based frequency synthesizer.

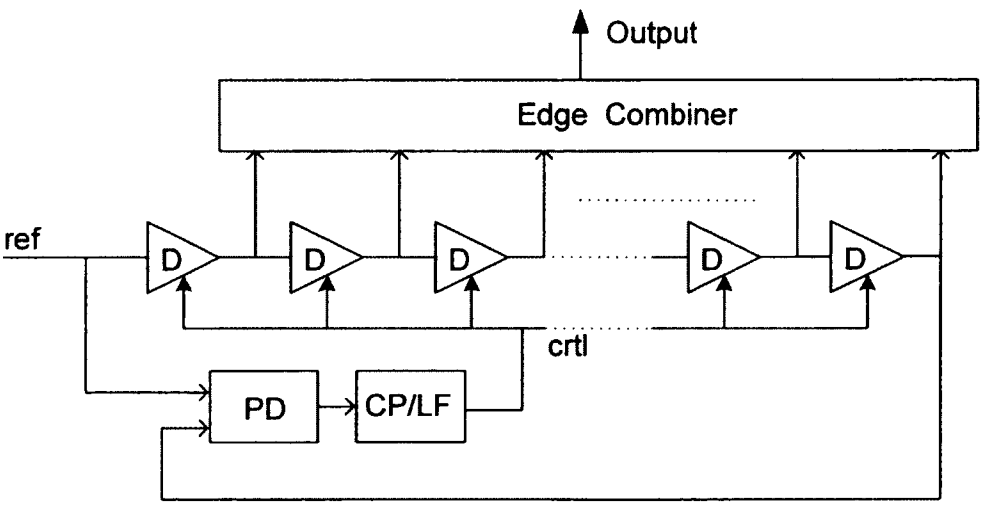


Figure 2.3.5: DLL-based frequency synthesizer

Figure 2.3.6 shows the output power spectrum of a PLL-based and a DLL-based synthesizer. Phase noise is a low frequency noise modulated by the carrier frequency. Occurrence rate of timing errors in the time domain determines offset frequency of an equivalent phase noise in the frequency domain.

Unlike oscillators, timing errors in DLL-based synthesizers are not correlated when they happen in more than one reference cycle apart. This leads to an almost flat close-in phase noise for offset frequencies less than f_{ref}. This low close-in phase noise performance is limited by the inherent phase noise of the reference signal and the jitter contributed by the delay line and edge combiner. The close-in phase noise spectrum may also impact by 1/f noise of circuit components.

The systematic and periodic fluctuations in the time domain introduce undesired components in the frequency spectrum. In a DLL-based synthesizer, the major sources of spurious tones are delay stages mismatch and static phase error. If one of the delay elements has slightly different delay, the corresponding period in the output waveform is shorter or longer depends on the type of the mismatch. Since the DLL is locked to one

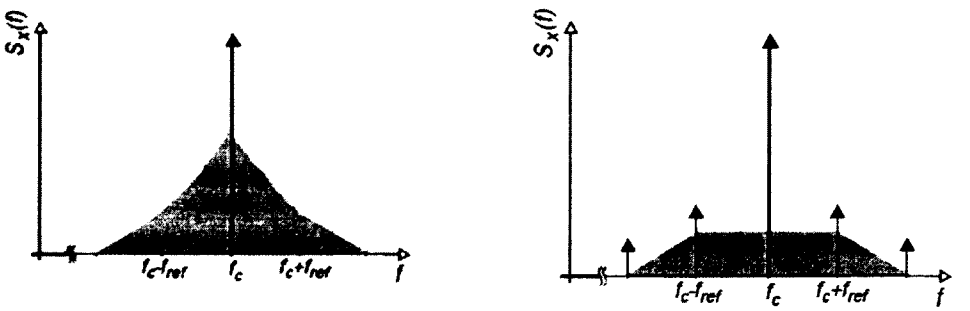


Figure 2.3.6 Out power spectrum of PLL-based synthesizer (left) and DLL-based synthesizer (right)

period of reference signal, the timing error due to this mismatch repeats every reference frequency. Therefore, in the output spectrum, spurious tones appear at f_{ref} and harmonics of f_{ref} away from f_{out}.

As the DC gain from PD to the loop filter is limited, a phase difference at the input of PD is required to sustain the desired control voltage. This phase difference is known as static phase error (this phenomenon is also known as PD dead zone). In DLL-based synthesizers, static phase error leads to a phase difference between the reference signal and the delay line output signal, which translates into the period change of one of the output waveform cycles - i.e. the one right before the reference signal starts its next cycle. The rate of occurrence for the dead zone is f_{ref} hence it causes spurious tones at f_{ref} and harmonics of f_{ref} away from f_{out}. Note that the closest spurious tone to the output frequency (f_{out}) is located at f_{ref} offset from f_{out} [44].

Increasing the DC loop gain is a common way to minimize the static phase error. However, a mismatch between CP currents and other non-idealities still cause static phase errors. A dead zone exists in PLL-based synthesizers but it only affects the phase relation of reference and output signals, which, as explained in the rest of this chapter, in a DLL-based synthesizer causes spurs at the frequency offset of f_{ref} (and harmonics off_{ref}). These spurs are normally suppressed by the input filter and it is not an issue in most of the wireless applications [44]. But its equivalent timing error reduces transmitter and receiver eye width in serial link applications.

2.3.4.2 Edge Combiner

Figure 2.3.7 and Figure 2.3.8 show two types of edge combiners. Consider the case when a multiplication factor of N is desired and, thus, N delay elements are in the delay line. The waveform of the edge combiner output has a period equal to $\frac{2\pi}{N} = \frac{T_{ref}}{N}$. The key idea in the Global edge combiner (shown in Figure 2.3.7) is to find two delayed version of the input with a phase difference of $\left(\pi \mp \frac{\pi}{N}\right)$, and make the half period of the target signal by combining them.

In lock, each buffer delay cell induces $\left(\frac{2\pi}{N}\right)$ phase shift. The two delayed versions of the input can be achieved from outputs of the i_{th} and the i_{th}+n delay cells if n satisfies the following equation:

$$n \times \frac{2\pi}{N} = 2k\pi \pm \left(\frac{\pi}{N}\right) \qquad (2.2)$$

where k is an integer and N is the number of delay elements in the delay line.

Global edge combiner, uses (combines) edges from different locations of delay line. To be exact in the Global edge combiner the output of the i_{th}delay cell is logically ANDed to the $\left[i + \left(\frac{N+1}{2}\right)\right]_{th}$ delay cell output for $i < \frac{N-1}{2}$ and it is logically AND to $\left[i - \left(\frac{N-1}{2}\right)\right]_{th}$ delay cell output for $i \geq \frac{N-1}{2}$. The outputs of these ANDs logically OR together to generate the output signal. In the Local edge combiner implementation (shown in Figure 2.3.8), each delay elements is made of two inverters and thus provides two phase of the clock with $\left(\pi - \frac{\pi}{N}\right)$ phase difference locally.

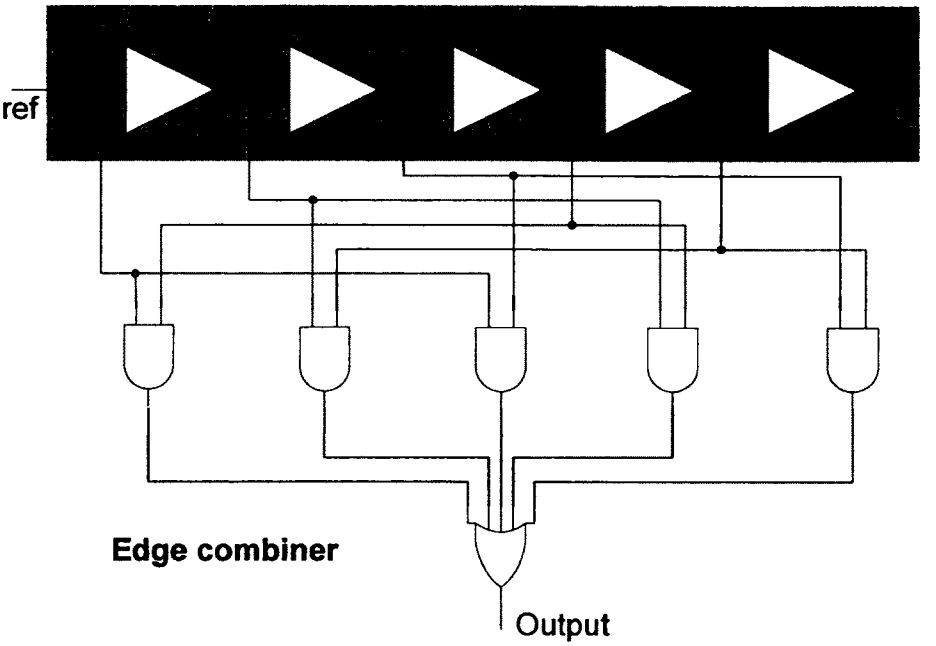


Figure 2.3.7: Global Edge-Combiner

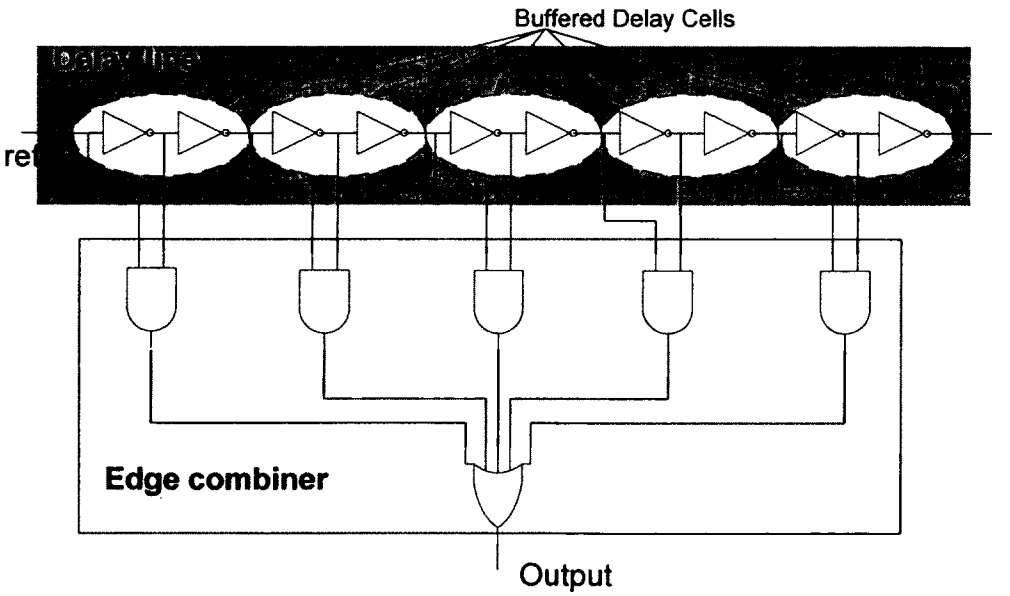


Figure 2.3.8: Local Edge-Combiner

The output of the first inverter in the delay element is logically ANDed with the second one to generate one period. Then, the outputs of these ANDs are logically ORed together to generate the output signal.

2.3.4.3 Operation Range and False Locking

Theoretically, the operating frequency of the DLL can vary from $\frac{1}{N \times d_{max}}$ to $\frac{1}{3 \times d_{min}}$ where d_{max} and d_{min} are the maximum, and the minimum delays of the delay cell, respectively. N is the number of delay cells used in the delay line. In lock, the total delay of the delay line is equal to one period of the input signal.

When the initial delay of the delay line is shorter than 0.5Tinput or longer than 1.5Tinput, DLL will try to lock to zero delay or the harmonics of the input signal. This attempt of the DLL to lock to an incorrect delay is called false locking. In other word the period of the input signal (reference clock) has to be in the range of:

$$\left(Max\left(N.d_{min}\,,\,\frac{2}{3}.d_{max}\right) < T_{input} < Min(2.d_{min}\,,\,d_{max})\right) \qquad (2.3)$$

Many different techniques are introduced to prevent false locking. One is using a phase frequency detector (PFD) [7] with a capture range of $(-2\pi, 2\pi)$, which is wider than other PDs. DLL tries to lock to a zero delay if the PFD used without any control circuit [8]. In high frequency operation, the initial timing gap between the reference clock and the VCDL output could be more than two clock cycles and the harmonic locking could occur. An analog DLL, using a replica delay line, has been developed [9] to solve the narrow frequency range problem of a conventional DLL. A replica delay line converts the delay of one delay cell to a ratio of the reference clock. This ratio is made by sizing the

charge pump current of a replica delay line. A simple replica delay circuit is shown in Figure 2.3.9

In [10], a digital in-lock-range detector is proposed and used with a modified PD to prevent false locking. When a DLL is out of the lock range (i.e. (2.3) is not satisfied) the in-lock-range detector sends a control signal to the PD. This control signal stops the usual PD operations and directly controls the CP. This signal increases or decreases the delay of the delay cells until the DLL returns to the lock range. Then, the PD restarts its usual operation and the DLL loop tries to lock to the correct period. By using an in-lock-range detector, the DLL does not require the delay control voltage to be set in power-up and can recover from the missing reference clock frequency signal. The DLL also has a wider delay range and thus, it can accommodate a wider range of reference clock frequency.

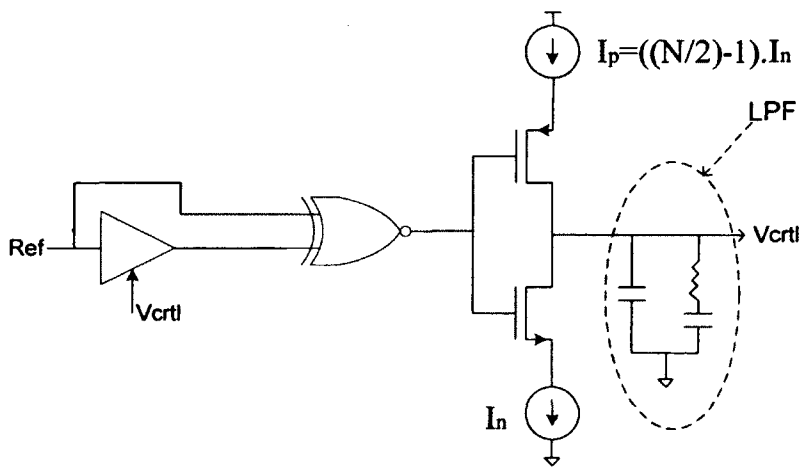


Figure 2.3.9: Replica delay line (for N stage delay line)

The DLL developed in [11] uses a stage selector for fast locking and wide range operation. This topology requires an additional VCDL, which increases the area. A stage selector is in fact a phase selecting circuit that can automatically decide the number of delay cells used in the DLL. This enables the DLL to operate in a wide frequency range.

In [12] the control voltage is initially set to its maximum value that causes the VCDL to have its minimum delay. Then the phase selector chooses the largest number of delay cells so that the total delay of delay cells is smaller than the reference period (T_{ref}). Then, the PD starts to lock the created loop to T_{ref}. This topology prevents false locking but it is not proper for synthesis application because these schemes usually require a delay line with predetermined number of delay elements in order to generate desired multiplication factor. Although, it is possible to set the number of delay cells to the desired value by making small changes in this topology, incrementing the number of delay cells one by one, is a very slow process. Note that adding each cell to the delay line requires one settling time for locking.

2.4 General Characteristics of a Frequency Synthesizer

Frequency accuracy is the most important characteristic of a synthesizer. In practice, usually integrated synthesizers have to meet very stringent frequency specifications, restrictive phase noise and spur requirements. In the indirect frequency synthesis approach, the reference frequency is usually provided by a crystal oscillator. The output of this oscillator has high accuracy in oscillation frequency but it is also sensitive to temperature and aging of the crystal. Thus most of the frequency errors are a result of the

other non-ideal components of the synthesizer. The issue of the phase noise is the main obstacle in integrated synthesizers due to lack of the high quality inductors.

In wireless applications, close-in noise degrades SNR at the output of the demodulator and high-frequency noise sidebands also degrade adjacent channel selectivity and limit the dynamic range. In addition to phase noise, the output of a synthesizer may contain undesired sidebands (spurs). The maximum tolerable level of spurs is a system requirement and has to be fulfilled by synthesizer performance. In an integer-N architecture, step size is tied to the reference frequency. A smaller frequency step size requires a lower reference frequency. Furthermore, the loop bandwidth should be sufficiently smaller than the reference frequency (usually one-tenth) in order to reduce the reference feed-through.

A narrower loop bandwidth results to a slower frequency switching (i.e. longer switching time). Therefore, there are trade-offs among frequency step size, switching time and maximum reference feed-through. Narrower loop bandwidth reduces the phase noise at a fixed offset. The synthesizer multiplication factor can be varied by changing the divider modulus in a PLL-based architecture or by increasing the number of the delay cells in a DLL-based counterpart. The VCO tuning range and the edge combiner frequency response limit the multiplication factor range. Increasing the divider modulus also increases the acquisition time and makes PLL-based synthesizers more sensitive to low frequency noise. Sensitivity to substrate noise (coming from surrounding digital circuitry) and supply noise (caused by any switching) are considered critical in a practical implementation. The other factors like input frequency (reference frequency), power consumption, area and cost also have to be considered in a design.

2.4.1 Phase Noise

Phase noise has a great impact on the overall system performance. It can be described as short term random frequency fluctuations of a signal. In the time domain a sinusoidal signal with phase noise can be written as:

$$V_{(t)}=(A+a(t))Sin(2\pi ft+\varphi(t)) \qquad (2.4)$$

Where $\varphi(t)$ is a function of time and represents the phase noise of the sine wave (a(t) represents the amplitude noise). The time domain phase fluctuation results in sidebands close to the operating frequency in the output frequency spectrum. One of the most common fundamental descriptions of phase noise is the one sided spectral density of phase fluctuations. The term power spectral density describes the energy distribution as a continuous function, expressed in units of energy per unit bandwidth. The phase noise of a synthesizer is best described in the frequency domain where the spectral density is characterized by measuring the noise power density on either side of the center frequency of the output signal. Single sideband phase noise is specified in dBc/Hz at a given frequency offset from the carrier, where dBc is the level in dB relative to the carrier. The phase noise of an oscillator at a given offset is derived from the ratio of the power in a 1Hz bandwidth at the offset frequency to the total power of the carrier (see following equation).

$$\mathbf{L(\Delta\omega) = 10Log[\frac{P_{noise}(\omega_0+\Delta\omega)}{P_{carrier}}]} \qquad (2.5)$$

Where P_{noise} is the noise power in unit bandwidth at the offset frequency $\Delta f = \frac{\Delta\omega}{2\pi}$ from the output frequency ω_0. The frequency spectrum of an ideal sine wave and a sine wave with phase noise is shown in Figure 2.4.1

2.4.2 Jitter

Phase noise and jitter are two different ways of quantifying the same phenomenon. Jitter is a measurement of the variations in the time domain, and essentially describes how far the signal period has wandered from its ideal value. There are two main types of jitter: deterministic and random. Deterministic jitter is created by identifiable interference signals. It is always bounded in amplitude, has specific (non random) causes and cannot be analyzed statistically. There are four main sources of deterministic jitter: crosstalk between adjacent signal traces; electromagnetic interference (EMI) radiation on a sensitive signal path; noise from power layers of a multi-layer substrate; and simultaneous switching of multiple gates to the same logic state. Random jitter describes timing variations caused by less predictable influences. Temperature affects the mobility of semiconductor crystal material and causes random variations in the carrier flow. Semiconductor process variations such as non-uniform doping density can also cause jitter.

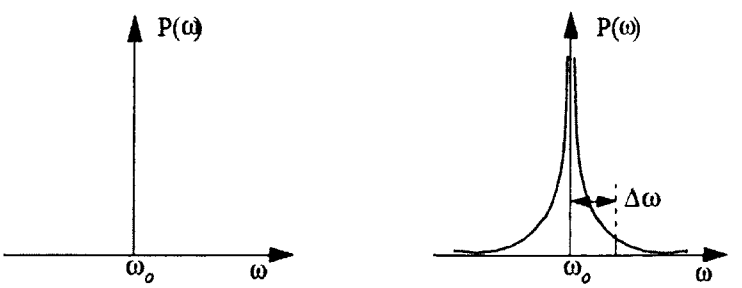


Figure 2.4.1: Power spectrum for ideal and practical output

Random jitter can be typically characterized using Gaussian distribution statistics. This normal distribution yields two common jitter specifications. One is the Peak-to-peak jitter or the distance from the smallest to the largest measurement on the normal curve. Theoretically, this value increases up to a value of infinity with the number of samples taken and, thus, might not be a useful measure. The other one is root mean-squared (RMS) jitter, the value of one standard deviation of the normal distribution. It is a more meaningful measurement since RMS jitter changes very little as the number of samples increases. However, it is only valid in pure Gaussian distributions where no deterministic jitter exists. Multiple sources of random jitter add in an RMS fashion. To obtain total jitter, a peak-to-peak value is needed to add random jitter to deterministic jitter. There are two ways of expressing jitter. One is timing jitter which is the uncertainty in the zero crossing point of any other periodic signal (equivalent to the edge position of a square wave).

$$t(n) = nT + J_t(n) \tag{2.6}$$

The other one is period jitter which quantifies the uncertainty in the period of a periodic signal.

$$T(n) = T + J_P(n) \tag{2.7}$$

where in Equation (2.6) and Equation (2.7) T is the period, n is the cycle index and J is the jitter.

2.4.3 Phase Noise to Jitter Conversion

Phase noise and jitter are two definitions of the same phenomenon in frequency and the time domain, respectively. The integral of phase noise over the band of interest results in the power level of the phase modulating noise in this band. Then, the RMS jitter can be calculated from this noise power as follows:

$$N = \int_{\omega_1}^{\omega_2} L(\omega) d\omega \tag{2.8}$$

$$J_{RMS} = \frac{\sqrt{10^{\left(\frac{N}{10}\right)}}}{\pi \times f_c} \tag{2.9}$$

where N is the noise power, f_c is synthesizer output frequency and J_{RMS} is in unit interval (UI). Unit interval is usually one second as integration of L(ω) gives the phase error and dividing it to phase speed ($\pi \times f_c$) would result to standard time unit of one second.

Chapter 3 PLL-Based Fractional-N Frequency Synthesizers

3.1 Introduction

The frequency synthesizers discussed in the previous chapter are referred to as integer-N frequency synthesizers because the output frequency is always an integer multiply of the reference frequency. The reasons for this limitation can be found in the edge-combiner topology and the divider structure in DLL-based and PLL-based frequency synthesizers, respectively. The divider modulus inherently is an integer number and the edge combiner generates an integer number of cycles in each reference period. There are several techniques to make fractional-N PLL-based frequency synthesizers. In this chapter these techniques will be discussed in detail. However, DLL-based fractional-N frequency synthesizer has been introduced and patented as the result of this Ph.D research [82]. The challenge of designing a DLL-based fractional-N frequency synthesizer is discussed in Chapter 4. In addition a few architectures for DLL-based fractional-N frequency synthesizer are proposed accordingly.

The motivation behind fractional-N PLL-based frequency synthesizers is discussed in the beginning of this chapter, followed by a review of their advantages and disadvantages. The applications of these synthesizers and their implementation techniques are described.

At the end of this chapter, simulation issues and a brief explanation of the methods for faster simulation are presented.

3.2 PLL-Based Fractional-N Frequency Synthesizers

The phase locked loop acts as a low-pass filter for reducing reference signal noise and low frequency path noise. However, it appears as a high-pass filter for VCO noise. As the VCO is the main contributor to the output phase noise, frequency synthesizer designers tend to use a wider loop bandwidth to suppress as much of the VCO noise as possible. However, because of stability concerns and suppressing the reference feed-through, the loop bandwidth must be approximately one order of magnitude smaller than the reference frequency. Thus, to have a higher loop bandwidth, a higher reference frequency must be used which in an integer-N frequency synthesizer translates into a larger channel spacing. The trade-off between the loop bandwidth and the channel spacing in PLL-based integer-N frequency synthesizers is relaxed in fractional-N architectures due to the capability of having the channel spacing equal to a fraction of the reference frequency.

3.2.1 Advantages and Drawbacks

In fractional-N frequency synthesizers the output frequency can be a fractional ratio of the reference frequency. This means that the frequency resolution (channel spacing) is finer than the reference frequency. Given the same channel spacing, a fractional-N synthesizer can be designed with a higher loop bandwidth than that of an integer-N synthesizer. Higher loop bandwidth results in faster frequency switching and thereby dynamic bandwidth techniques can be used more efficiently [13, 14]. In a dynamic bandwidth approach, the loop bandwidth is made higher when the PLL is outside of the

lock-in-range to obtain a faster settling time during the transient mode. The output in-band phase noise can be calculated as:

$$PN_{noise} = PLL_{noise} + 20logN \qquad (3.1)$$

where PLLnoise is the phase noise contributed from the PLL and N is the divider modulus. A higher reference frequency results in a higher comparison frequency that in turn relaxes the PLL requirements in terms of noise reduction and reference spur attenuation. For a given channel spacing and a target output phase noise, the PLL noise requirement is smaller for the fractional-N architecture when compared to its integer-N counterpart due to the smaller divider modulus. The reference spur is also less sensitive to leakage current and non-ideal effects of the charge-pump [15].

There are several approaches to designing fractional-N synthesizers but all of them are more complex as compared to their integer-N counterparts. Fractional-N synthesizers are inherently have more spurious output than the integer-N ones and may have worse phase noise performance due to quantization issues. Wider loop bandwidth imposes more stringent requirements on in-band phase noise, and also increases the reference frequency, PD noise and the discrete spurious levels [16, 17].

3.2.2 Applications

There is an emerging application in new radio systems called TETRA, in which the channel spacing and the switching time specifications cannot be met with ordinary integer-N synthesizers. The high resolution of the fractional-N architecture can be used for automatic frequency control (AFC), doppler correction or other features which require

fine frequency tuning. This architecture can also be used to relax the trade-offs in conventional integer-N synthesizers.

3.2.3 Architectures

The challenge of designing a fractional-N frequency synthesizer involves making trade-offs among phase noise, frequency switching speed, loop bandwidth, frequency resolution, tuning bandwidth, and power consumption. The frequency multiplication factor is achieved by manipulating the divider modulus (which is inherently an integer) so that the average division ratio is the desired fractional ratio. It is implemented by switching between the moduli of a multi-modulus divider (a divider with two or more moduli). This switching strategy for the modulus dividers provides five fractional-N synthesizer techniques: pulse swallowing, amplitude compensation, phase compensation, random jittering, $\Delta\sum$ modulated jittering, and phase interpolating $\Delta\sum$ modulated jittering.

3.2.3.1 Pulse Swallowing

Pulse swallowing fractional-N frequency synthesizers (Figure 3.2.1) are similar to conventional PLL-based integer-N frequency synthesizers with a dual modulus divider.

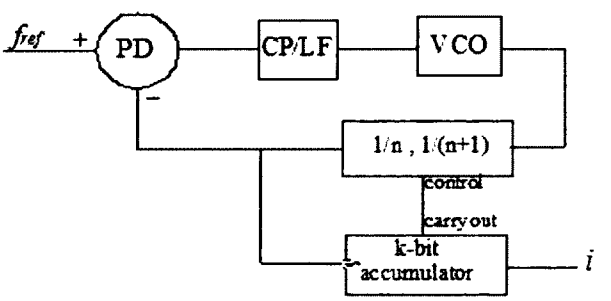


Figure 3.2.1: Pulse swallowing fractional-N synthesizer

The condition of overflow in the accumulator is used to shift the divider modulus from n to n+1. For a k-bit accumulator the average division factor N can be controlled by accumulator input i as indicated in the following formula:

$$N = n + \frac{i}{2^k} \qquad (3.2)$$

On every cycle of divider output, i is added to accumulator contents A. Thus, the new value of the accumulator would be A+i unless the accumulator overflows. In this case, the value assigned to the accumulator is $A+i-2^k$. Due to overflow, a carry output is generated. It is used to switch the divider moduli. This is a simple approach that requires only one additional accumulator in the hardware. A larger accumulator with a greater word length can provide a higher frequency resolution.

The main drawback of this approach is spurious frequencies generated in the synthesizer output spectrum. These spurious tones result from the train of zeros and ones appearing on the carry output. As i approaches zero or 2^kwhile the fractional ratio N approaches integers n and n+1, the train of zeros and ones becomes longer which in turn creates stronger spurious tones.

3.2.3.2 Amplitude and Phase Compensation

This approach uses the spur reduction technique to suppress the spurious tones as seen in the pulse swallowing approach. There are two types of spur reduction techniques used for this purpose; the first one compensates for the voltage error at the PD output that causes the spur (amplitude compensation technique). The second type compensates for the phase error at the PD input (phase compensation technique).

Figure 3.2.2 shows the amplitude compensation technique applied to PD output using a digital to analog converter (DAC). This approach is more effective with a sample and hold PD. When a sample and hold PD is used, the DAC must correct the PD output voltage to match its dc voltage for one reference clock period. The DAC uses the value of the accumulator that contains the information of the spurious beat tone to predict and compensate for the phase error.

The main drawback of this approach is the complexity of the DAC and its sensitivity to the process, supply voltage, and temperature (PVT) variation. The spurious rejection of this method is limited to 45dB [28].

The phase compensation method (Figure 3.2.3) [15] is applied to the PD input signal by the aid of a DLL. The settling time of DLL should be much smaller than that of the PLL. The number of stages, m, in the DLL determines the delay of each delay. The delay of each delay cell is T_{ref}/m

If n is the smaller divider moduli, for n+i/m fractional values, the instantaneous phase error at the input of the PD can be corrected by sending the i_{th}delay cell output to the phase detector. The phase correction of (1- i / m)$\times T_{ref}$ is employed when the divider moduli changes to n+1. In this topology DLL has to function at the output frequency of the synthesizer. The same phase compensation can be achieved by using a DLL with delay cells and an input is derived from a divide by k circuit.

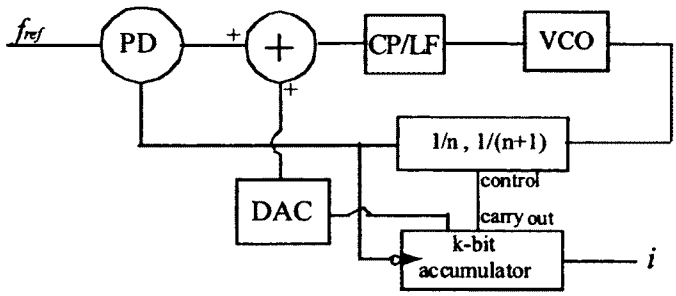


Figure 3.2.2: Amplitude compensation approach

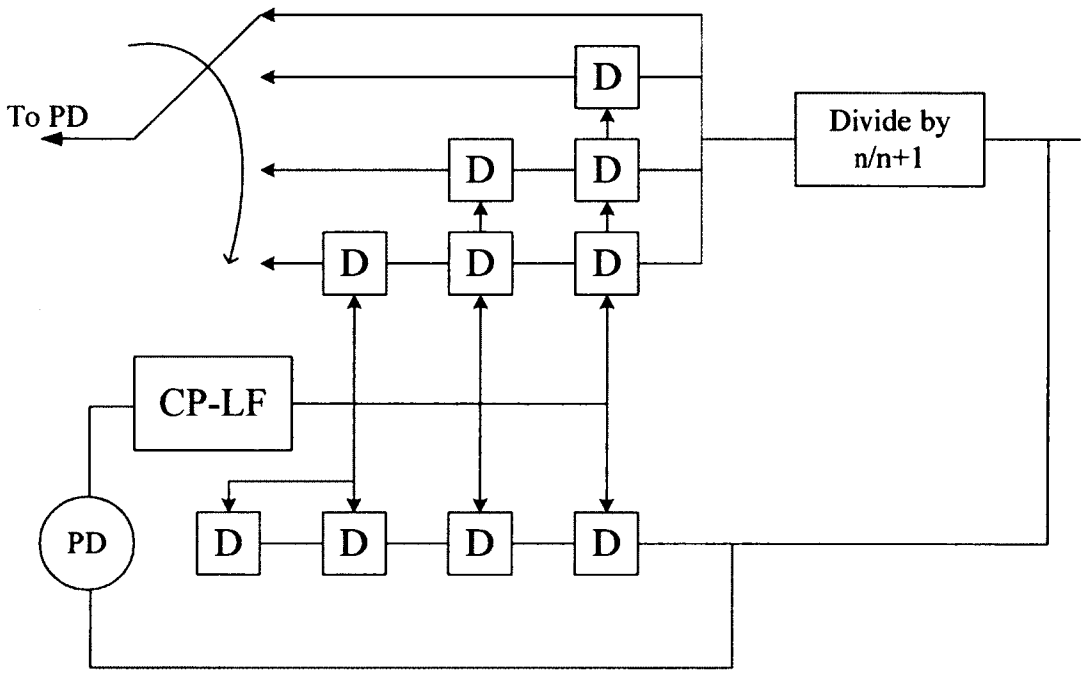


Figure 3.2.3: Phase compensation approach

In this new configuration the DLL operates at a lower frequency but requires more delay cells. The main disadvantage of this approach is that the spurious tone can compensate completely for only n+i/m fractional values. In addition, implementing a large value of m is not practical. This limits the resolution of this kind of synthesizers.

3.2.3.3 Random Jittering

The random jittering spur reduction technique uses a random sequence generator to randomize the division modulus and thus, to convert the output spurs to jitter [18]. Figure 3.3.1 shows a typical block diagram of such an implementation. A comparator is used to force the average of the divider moduli to the desired fractional ratio. The resolution depends on k, the number of bits of the random number generator, and the comparator. The average divider modulus is controlled by i - the threshold of the comparator. The comparator output is one bit that determines the divider modulus. The main drawback of this approach is that the output spectrum exhibits $1 / f^2$ phase noise near the output frequency.

3.2.3.4 ΔΣ Modulated Jittering

In this approach, an oversampling ΔΣ modulator is used to interpolate a fractional ratio with a coarse integer divider. Figure 3.3.2 presents a typical ΔΣ fractional-N frequency synthesizer. The noise shaping ability of ΔΣ modulator is used to shape the phase noise resulting from quantization and randomization to a higher offset frequency. The ΔΣ modulated jittering method can generate an arbitrarily fine frequency with digital modulation. When compared to the DAC method, this one is less sensitive to analog mismatch and PVT variations. It does however, have a relatively higher complexity and power consumption.

3.3 PLL-based ΔΣ Fractional-N Frequency Synthesis

This kind of fractional-N frequency synthesizers is more popular for an integrated implementation mainly because it offers a good phase noise performance with an

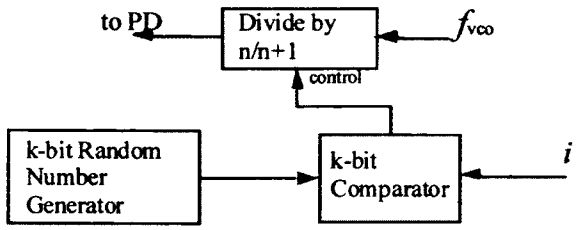


Figure 3.3.1: Random jitter approach

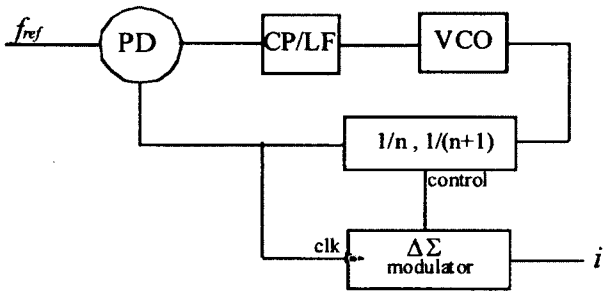


Figure 3.3.2: ΔΣ fractional-N synthesizer

acceptable complexity. The ΔΣ modulators were mainly employed in oversampling converters until Riley et. al. [19] used their noise shaping ability to improve the random jitter approach and remove the $1/f^2$ phase noise. The natural high pass transfer function of ΔΣ modulators pushes the close-in phase noise to higher frequencies where the low-pass loop filter can remove parts of this high frequency noise which are outside of its bandwidth. These synthesizers have recently been used commercially [17, 20, 21] to ensure small frequency step sizes, reduced output phase noise, reduced spurious tones, and flexibility in design.

3.3.1 Fundamental Issues

The design of ΔΣ fractional-N frequency synthesizer involves many trade-offs. The main one is between the in-band phase noise (ΔΣ quantization noise and other circuit noise added in the loop filter) and VCO noise.

The VCO noise can be suppressed by increasing the loop band-width. Decreasing loop band-width can reduce both in-band phase noise and quantization noise. In-band phase noise has contributions from the reference signal path, divider, phase detector and charge pump.

The optimum trade-off for each application would be different. For example a QAM constellation with 1024 points requires very low in-band noise while for GSM and similar applications, which are sensitive to interference, the out-of-band phase noise is of significant importance.

The source of in-band noise can be jitter in digital logic, noise folding due to nonlinearity, charge- pump current noise, offset current noise and reference noise. Digital logic jitter is the major contributor to in-band noise [22], which includes contributions from the divider, PD, and reference path. Non-linearities due to the PD dead-zone and charge-pump modulate high frequency quantization noise into signal bandwidth. This phenomenon is called noise folding and can be suppressed by controlling the distribution of the quantization noise and of the PD operating region [22].

3.3.2 ΔΣ Modulators

In fractional-N synthesizers, the input of the ΔΣ modulator is usually a digital word representing the desired fractional value. The output of a ΔΣ modulator is a stream of

integer numbers used to control the divider modulus. This output stream controls the divider modulus so that the VCO output frequency be a fractional ratio of the reference frequency. Over time, the average of the ΔΣ modulator output converges to the desired fractional ratio. All of the ΔΣ modulators induce some quantization noise but the value of this noise is larger in single bit modulators. Multi-bit ΔΣ modulators might be used to reduce the quantization noise. The order of the ΔΣ modulator is equal to the number of integrators in the structure. Each integrator introduces a zero at the origin in its transfer function, and thus shapes the noise spectrum in the frequency domain. Converting the frequency to phase removes the zero from a first order ΔΣ modulator causing it to fail to randomize the quantization error and thereby prevents it from removing spurious frequency components from the synthesizer output spectrum [19]. By selecting higher order ΔΣ modulators, the spurious energy is spread out and shaped to resemble high frequency noise which is removed by the low-pass nature of the loop filter. The low frequency components of quantization nonlinearity error are more effectively filtered by higher order ΔΣ modulators. The output noise spectral density of the higher order modulators increases at a greater rate per unit frequency resulting in a greater SNR in the base-band at the expense of increased out-of-band noise. When higher order modulators are used, the PLL requires extra poles in the loop filter to suppress the quantization noise at high frequency. In practice, both in-band and out-of-band noise affect the synthesizer performance. However, the higher frequency noise is difficult to suppress with a finite number of PLL poles [20]. Second and third order ΔΣ modulators are practically used in fractional-N synthesizers [16, 17, 19, 20, 28].

3.3.2.1 ΔΣ Modulators Architectures

The choice of appropriate ΔΣ modulator structure for fractional-N synthesis involves many factors including noise shaping, spurious content of the output spectrum, output levels, loop filter order, and circuit complexity. Both analog and digital implementation of these architectures is possible but the digital implementation is more common in fractional-N synthesizers. In a digital implementation, an accumulator acts as an integrator and comparator. As the accumulator has a feedback path, the accumulator can be considered as a compact first order ΔΣ modulator [28]. Digital ΔΣ modulators do not have any non-idealities; furthermore there is not an overload problem as far as they are stable [79,80,81]. Cascade digital modulators do not suffer from mismatch or noise leakage from the input stage (unlike their analog counterparts). Multi-bit quantizers on their part, do not suffer from non-linear effects. High order modulators can be implemented using interpolative and MASH (Multi-stage noise shaping) architectures.

MASH architecture uses a cascade of lower-order structures to construct high-order modulator [29]. The MASH architecture is usually constructed by a cascade of first order modulators or a combination of first and second order modulators. A MASH modulator produces a multi-bit output that controls a multi-modulus divider. In general, a multi-bit modulator can achieve more desirable noise shaping for frequency synthesis. A programmable counter can serve as a multi-modulus divider. However, such a counter poses certain difficulties for a very high speed implementation. An estimate of the hardware complexity of a MASH modulator can be found in [30]. MASH offers simpler high order architecture without any stability problems and tends to generate wide-spread high frequency bit pattern that imposes more stringent requirements for PD design.

Intensive switching of the MASH ΔΣ modulator increases the high frequency noise and causes larger instantaneous phase error. Fourth-order MASH provides a higher order of noise shaping (-80 dB/dec) but has almost twice the complexity of a third-order MASH and consumes more power. As shown in Figures 3.3.3 and 3.3.4, third-order MASH can be realized as MASH1-1-1 and MASH1-2 which are cascades of three first order modulators or cascades of one first-order and one second-order modulators, respectively (Ritchie [31]). MASH 1-1-1 and MASH 1-2 exhibit the same order of noise shaping. However MASH 1-2 can be designed to have four output levels instead of eight as with MASH 1-1-1 [30]. The big disadvantage of MASH 1-2 is that it only allows the input to operate about 75% of the whole fractional range [30].

Single-loop (also called interpolating or single-stage) ΔΣ modulators introduce less phase noise and can provide either a single-bit or a multi-bit output. However, they are subject to instability and a smaller input range. The latter can be eliminated with a multi-bit quantizer in digital implementation.

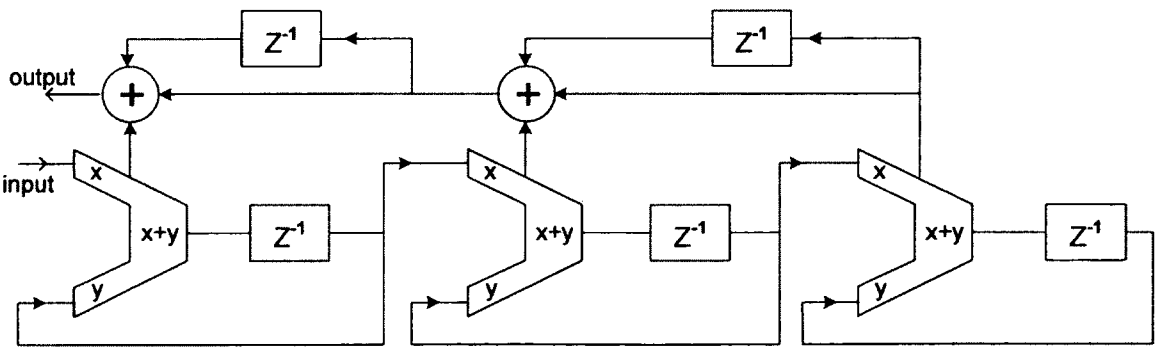


Figure 3.3.3: A digital MASH 1-1-1 ΔΣ modulator

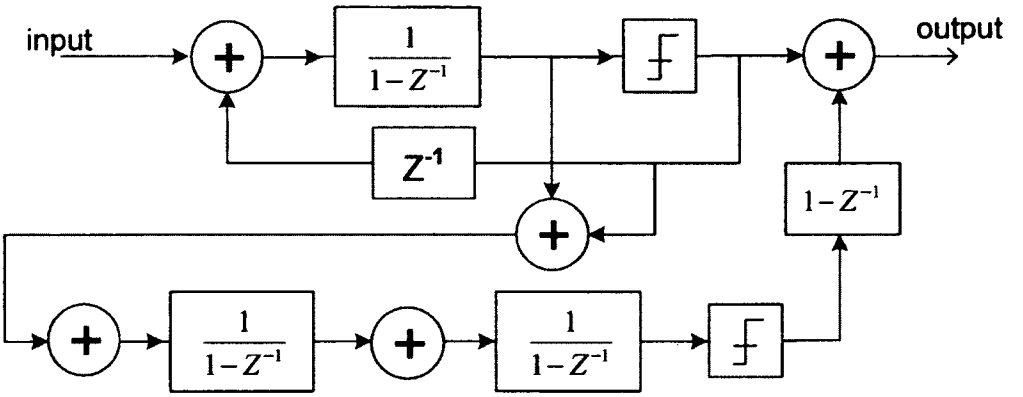


Figure 3.3.4: A digital MASH 2-1 ΔΣ Modulator

The quantizer output of a typical third-order single-loop multiple-feed-forward ΔΣ modulator (Figure 3.3.5) is limited to three levels and the feed-forward branches can be truncated to reduce its complexity.

Another version of a single-loop ΔΣ modulator uses multiple-feedback [32] (Figure 3.3.6). In order to obtain a reasonably stable input range for the multiple-feedback ΔΣ modulator, a large number of quantization levels is required (nine in the case of third-order modulator).

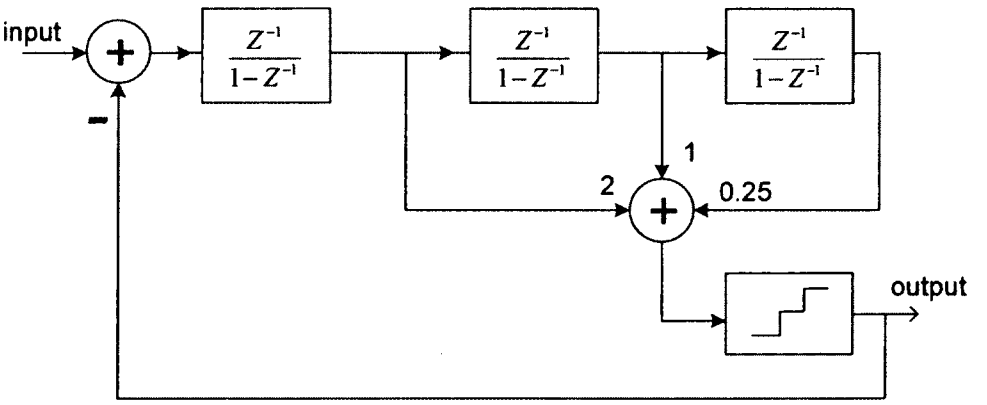


Figure 3.3.5: Single-stage 3rd-order feed forward ΔΣ modulator

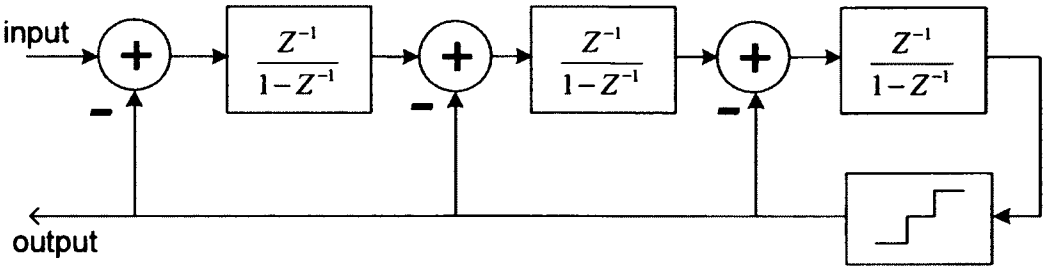


Figure 3.3.6: Single-stage 3rd-order feedback ΔΣ modulator

The word-length of the adders before the accumulators is much shorter than the accumulators themselves [33]. A tone free output can be achieved at the price of high number of output levels. However, the output levels are more concentrated than those of MASH1-1-1 [34]. Table 3.3.1 [34] shows a performance comparison of third-order ΔΣ modulators.

The wide-spread output pattern of a MASH modulator makes the synthesizer more sensitive to the substrate noise coupling since the turn-on time of the charge-pump in the locked condition increases. This can be reduced by limiting the output range of the modulator [20]. The smaller on-time of the CP in a single-loop modulator makes it less sensitive to noise coupling from the substrate and power supply. Due to non-linear mixing in PD and CP, noise at fref/2 folds back to a lower frequency similar to that of multi-bit ΔΣ ADCs. For a single-loop modulator, noise at fref/2 is much lower and therefore, its noise leakage due to non-linearity is also lower. Although the ideal in-band phase noise is lower for the MASH ΔΣ modulator, due to its higher phase error

Table 3.3.1: Performance Comparison of 3rd-order ΔΣ modulators topologies

Architecture	MASH 1-1-1	MASH 2-1	Single Stage feed forward	Single Stage feedback
Noise Shaping	Good	Fair	Fair	Good
Spurious tones	Very tonal	Some tones	A few tones	Almost tone free
Output levels	8(-3~4)	4(-1~2)	3(0~2)	9(-4~4)
Working Clock	f_{out}	$0.5\times f_{out}$	$\approx f_{out}$	$0.33\times f_{out}$
Stable DC input range	0~1	0.125~0.875	0.263~1.678	-2.5~2.5

introduced, only a small non-linearity is enough to increase the in-band noise more than what is expected from a single-loop modulator. Single-bit high-order modulators have a dead-band problem due to the limited input range of the quantizer in their synthesizer applications. The non-ideal effects at the band edges can be reduced by extending the input range with a multi-level quantizer [20].

3.3.2.2 Implementation Considerations

The division modulus is modulated by the ΔΣ output, by the desired mean value as well as by the shaped high frequency quantization noise. The quantization noise is $\Delta^2/12$ where $\Delta = (\Delta N)/(2^b - 1)$, where ΔN is the modulus range and b is the effective number of ΔΣ output bits. The roll-off of the ΔΣ modulator noise at the output of synthesizer can be calculated to be -20(m-n+1) dB/dec where n is the order of the ΔΣ modulator and m is the order of Butterworth filter used as the loop filter [35, 36]. The main advantage of the ΔΣ fractional-N synthesizer is the decoupling of the choice of the reference frequency and the PLL bandwidth. Thus, to make sure that the ΔΣ modulator does not corrupt the rms phase error, the dynamic range of ΔΣ modulator must be higher than that of the frequency synthesizer [20]. The dynamic range of a frequency synthesizer is defined as

the ratio of the largest possible frequency change to the smallest one. The largest frequency change is the full frequency range of the modulator $\Delta N.f_{ref}$ where ΔN is the modulus range. The smallest frequency change will be determined by the frequency noise which is the frequency translation of the in-band phase noise. For in-band phase noise of $10\log A_n$dBc/Hz, an equivalent frequency noise can be calculated as:

$$\Delta f = \sqrt{\frac{2.A_n.(BW_n)^3}{3}} \qquad (3.3)$$

where BW_n is noise bandwidth, the phase noise is A_n inside this band and is zero outside it. The relation between the in-band phase noise A_n and the rms phase error $\Delta\phi_{rms}$ is:

$$A_n = \frac{(\Delta\emptyset_{rms})^2}{2BW_n} \qquad (3.4)$$

Considering $f_c \approx BW_n$, the dynamic range of a PLL can be written as:

$$DR = \left(\frac{\Delta N.f_{ref}}{\Delta f_n}\right)^2 = 8 \times \frac{\Delta N^2}{\Delta\emptyset_{rms}^2} \times \left(\frac{f_{ref}}{2f_c}\right)^2 \qquad (3.5)$$

where f_c is the loop filter band-width. Comparing this with the dynamic range of a MASH modulator, the following non-equality should be satisfied [20]:

$$f_c < \left[\frac{3}{8} \times \frac{2n+1}{(2\pi)^{2n}} \times (\emptyset_{rms})^2\right]^{\frac{1}{2n-1}} \times f_{ref} \qquad (3.6)$$

where n is the order of $\Delta\Sigma$ modulator and $\Delta\Phi_{rms}$ is the rms phase error of frequency synthesizer. An approximation of PLL output phase noise [36] in the range of $f_c << f < f_{ref}$ is:

$$10\log(S_\theta(f)) \approx 10log\left(\frac{(2\pi)^{2n}}{12} \times \left(\frac{1}{f_{ref}}\right)^{2n-1} \times f_c^{2m}\right) + 20(n-1-m)\log(f) \qquad (3.7)$$

In fractional-N synthesis application, the ΔΣ input is a constant number, thus the output sequence may not be long enough to be of practical use. The periodic nature of a short sequence causes spurious tones in the synthesizer output. A simple way to achieve a longer output sequence is to increase the bit-length of the input, but this increase also raises modulator complexity and the power consumption. To reduce the fractional spurs resulting from the limited output sequence, some perturbations are imposed in ΔΣ output sequence by applying dithering signal from a pseudo-random generator to the input. In [30], the carry-in input of the adder is used in a feedback path to randomize the input. In [37], a 14 dB suppression of spurious tones is achieved by using the 3 output bits of the MASH modulator as a dithering signal to replace the least significant bits of the modulator.

3.3.2.3 Simulation Issues

The high output frequency of the synthesizer imposes the use of a high simulation sample frequency. However, the overall dynamics of the loop typically correspond to a much lower bandwidth. On the other hand, the fractional-N synthesizer has a non-periodic behavior in steady-state which prevents the use of methods developed for periodic steady-state conditions [23]. In [24], two approaches to speed up the simulation are discussed. First, the area conservation principle is used to convert continuous time phase-frequency detector (PFD) output to discrete time sequence. This conversion allows the use of uniform step size for simulation. Second, the VCO and the divider are assumed to be used as one block that allows much smaller sampling period in the simulation.

3.4 Other Types of Fractional-N Synthesizers

The other types of fractional-N synthesizers use the different phases of the reference signal to generate a fractional multiplication factor. Three architectures are discussed in this section. The first one uses a multi-phase oscillator to reduce the phase jump in PLL-based ΔΣ fractional-N architecture. The second one uses a delay locked loop and a switch to change the period of the reference signal. This technique is known as Period Synthesis (PS). The third one injects the reference rising edge into a multi-phase oscillator to reduce accumulated jitter and its correspond phase noise. This technique is known as multiplying DLL (MDLL).

3.4.1 Multi-phase ΔΣ Fractional-N Synthesizer

Although a multi-modulus divider can achieve any arbitrary fractional ratio, the minimum phase jump at the divider output is still equivalent to one VCO period. In [23, 24] a finer resolution was achieved by interpolating phases in an analog way using multi-phase VCOs, to make the phase jump smaller than one VCO period at the divider output. However, the phase mismatch (i.e. the phase inaccuracy of the multi-phase VCO outputs) gives rise to spurs at a fixed offset frequency when output phases are sequentially selected. In [25] ΔΣ modulation is used in combination with phase interpolation to eliminate spurs. A smaller phase jump at the divider output decreases the equivalent quantization step size and consequently the equivalent quantization noise of the ΔΣ modulator. Thus, lower phase noise is achieved compare to using multi-modulus divider. A block diagram of multi-phase ΔΣ fractional-N synthesizer is shown in Figure 3.4.1.

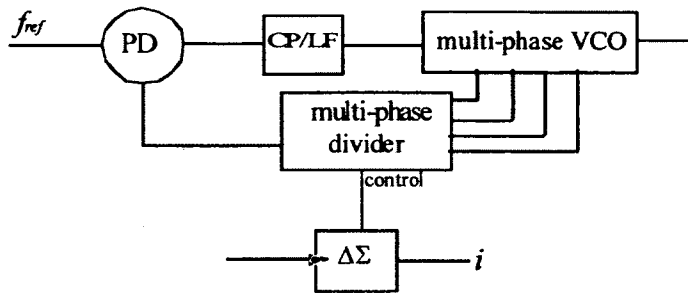


Figure 3.4.1: Multi-phase fractional-N synthesizer

3.4.2 Period Synthesis (PS)

Period Synthesis (PS) is used to generate a signal with an arbitrary period (T_{new}) from an input signal with a fixed period (T_{ref}). The resulting frequency multiplication factor can be a fractional number between 0.5 to 2. Therefore PS can be categorized as a fractional-N synthesizer. Figure 3.4.2 presents only the basic concept of the period synthesis technique. The input signal with period T_{ref} passes through a delay line where each delay element delays the reference signal by t_{dc}. A multiple-input single-output (MISO) switch connects only one of the delay elements to the output at any given instant. The process begins by passing the reference signal to the output. In the next cycle, the MISO switch connects the next delay cell to the output and, thus, the next rising edge comes with t_{dc} delay. If this process repeats every cycle, the output signal period will have a period of $T_{new}=T_{ref}+t_{dc}$. By switching the output to the previous delay at every cycle, the output period would change to $T_{new}=T_{ref}-t_{dc}$. Choosing the correct switching time is of critical importance to prevent a glitch in the output signal. In general, the proper switching time is the time when the outputs of both the current delay cell and the target delay cell (next or previous one) have the same voltage level. As it can be observed, the output signal does not have a 50% duty cycle.

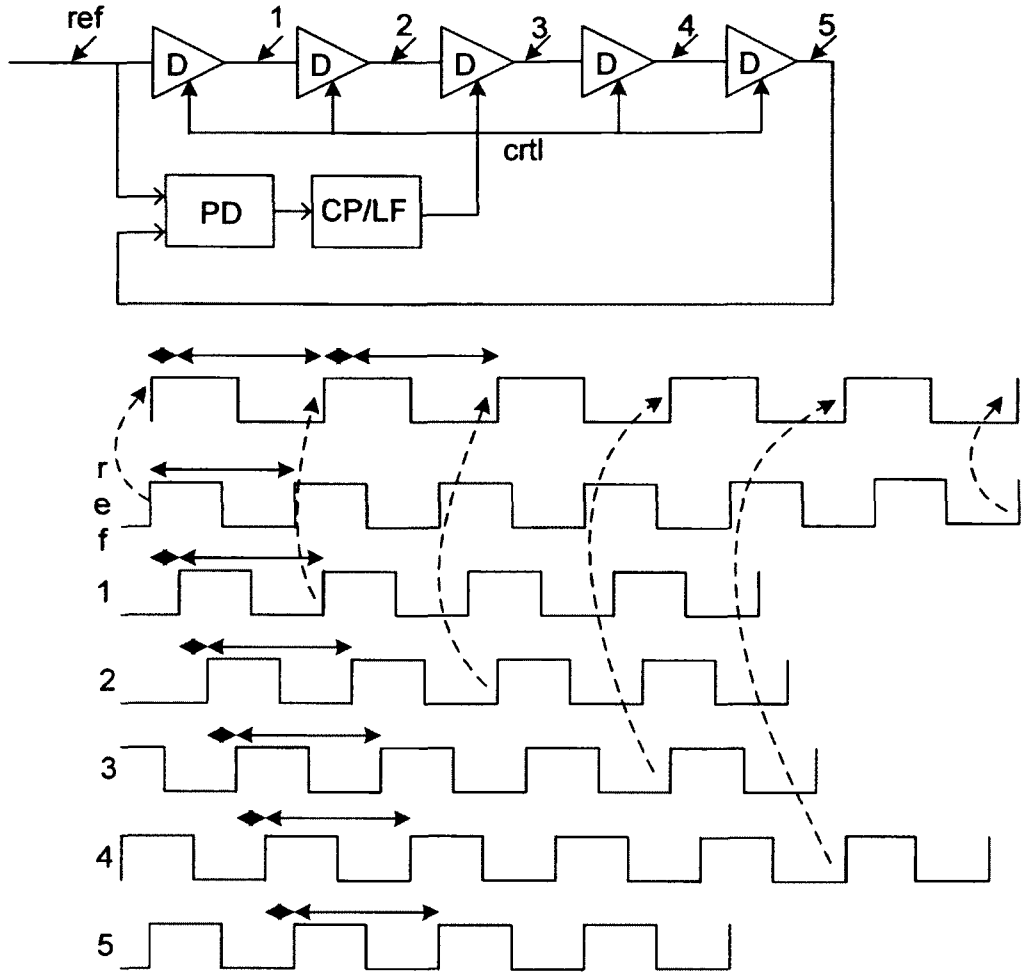


Figure 3.4.2: Concept of the period synthesis techniques

A simplified topology of a period synthesis circuit with five stages VCDL is shown in Figure 3.4.3 A Barrel shift register [43] controls the MISO switch which is made of a series of pass logic switches. The clock of the shift register provides by the period synthesis output. The resolution of this period synthesis circuit with five delay stages is 0.2Tref. The period T_{new} can also be controlled by the step size, or the number of delay cells that the switch jumps to at every cycle.

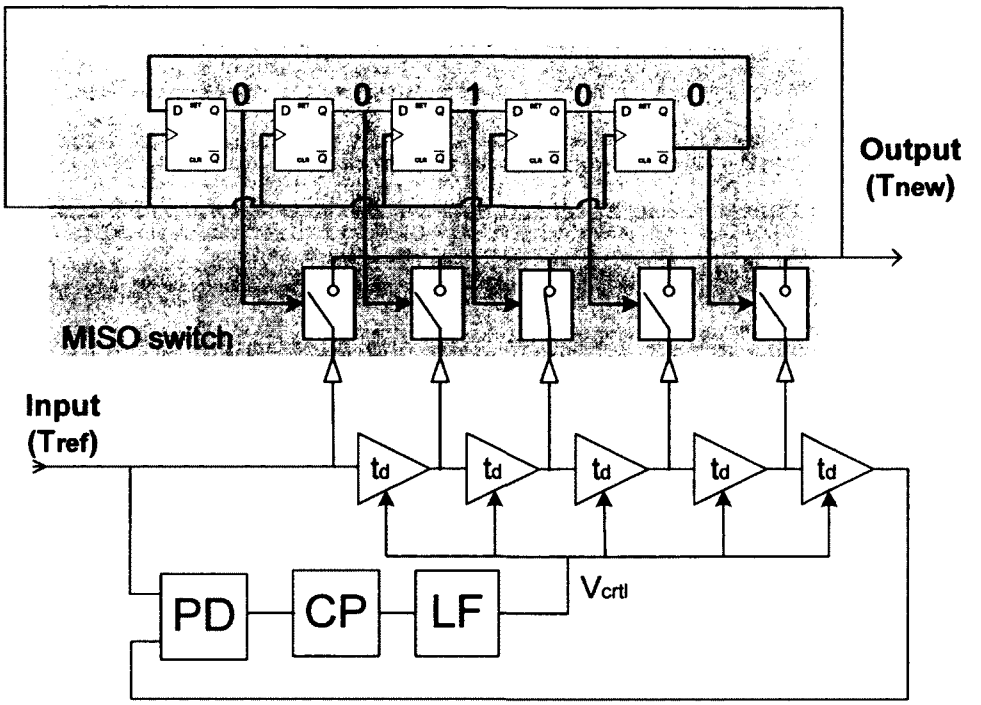


Figure 3.4.3: Five stages period synthesizer

For example, if the MISO switch connects the second next delay cell to the output at each cycle, then, the output period would be $T_{new}=T_{ref}+2t_{dc}$.

In general, the output period can be adjusted to $T_{new}=T_{ref}\pm(i.\ t_{dc})$ by switching the output to the ith following or previous delay element. The target periods of $Tnew=0.6T_{ref}$, $0.8T_{ref}$, $1.2T_{ref}$, $1.4T_{ref}$, $1.6T_{ref}$, $1.8T_{ref}$ can be generated by modifying the MISO switch in the period synthesis circuit (Figure 3.12) to adjust i=-2, -1, 1, 2, 3, and 4, respectively. The switching window disappears when Tnew<(Tref/2) although it still might be practically possible to generate periods equal to $0.2T_{ref}$, $0.4T_{ref}$. The resolution of Tnew is directly proportional to the resolution of t_{dc}. Producing a smaller t_{dc} requires a longer

delay line and a bigger switch which contributes to an overall delay and jitter of the circuit. A Period synthesis circuits made of a direct frequency synthesis scheme combined with a digital DLL, are proposed in [41] [42].

3.4.3 Multiplying DLL (MDLL)

Although MDLL is originally proposed as an integer-N frequency multiplier [46], it is modified to work in a fractional-N mode with low resolution [47]. A MDLL [46] (Figure 3.4.4) can be considered as a PLL with a ring oscillator that the reference feed into it every few cycle to reset the accumulated jitter and help to reduce the close-in phase noise (Figure 3.4.5). The working process of a MDLL illustrated in Figure 3.4.5 is as follow: At the beginning of the process the switch is in position A and the reference is fed into the delay line. Immediately after that the switch turns in position B, providing a feed-back path between the input and output of the delay line and converting it into a ring oscillator. The next time when the rising edge of the reference signal is almost aligned with the feedback path signal, the switch goes back to position A and feeds the reference into the delay line (ring oscillator) and, thus, then process repeats. Note that if there is not any noise in the system, the two signals would be perfectly aligned. In lock condition, misalignment is a result of the accumulated jitter which is eliminated by inserting the reference rising edge. When the switch is in position B, the MDLL works exactly like a PLL. As shown in Figure 3.4.4, the ring oscillator output (feedback path signal) feeds through a divider and the divider output goes to the PD. It is connected to a CP and a LF to provide proper control voltage for the delay cells.

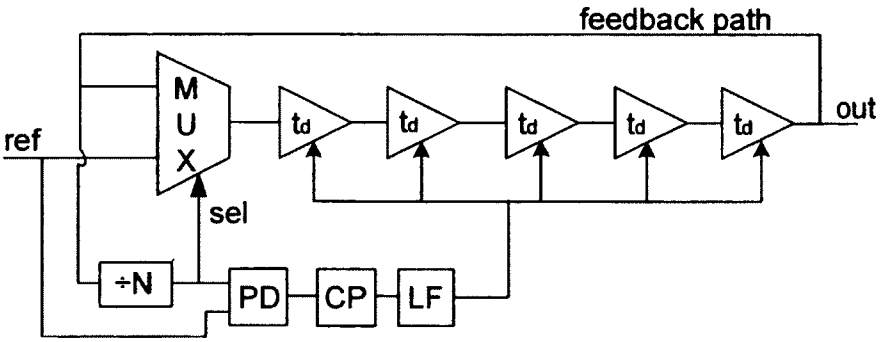


Figure 3.4.4: A multiplying DLL (MDLL)

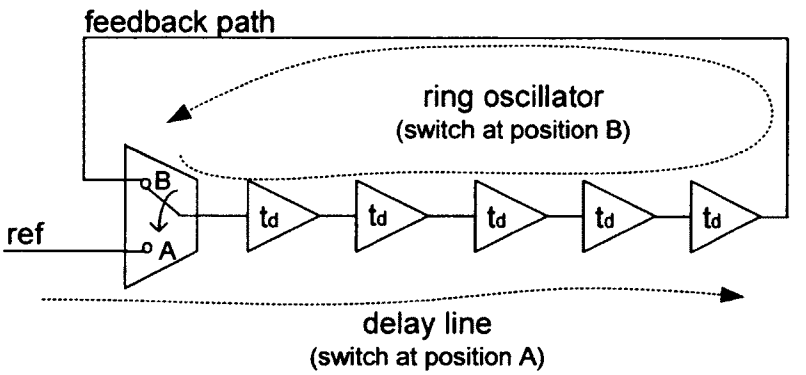


Figure 3.4.5: Changing mode in MDLL: when switch is at position 'A' it is a delay line, when switch is at position 'B' it is a ring oscillator

Chapter 4

DLL-based Factional-N Frequency Synthesizer; Proposed Architectures

4.1 Introduction

An integer-N DLL-based frequency synthesizer can be fully integrated and offers a superior phase-noise performance compared to its integrated counterparts [38]. As discussed in Chapter 2, to maintain this superior phase-noise performance, a DLL-based frequency multiplier (synthesizer) employs a short delay line (usually less than 30 cells) and a high reference frequency. The structure of the edge-combiner imposes limitation of having only integer multiplication factor. The integer multiplication factor together with the high reference frequency results in a very low frequency selectivity (channel spacing or frequency resolution).

On the other hand, in most of the wireless standards (Table 4.1.1) the channel selectivity is a very important issue. In the commonly used transceiver architectures, the channel selection is performed in RF band by changing the local oscillator frequency. Also in wire-line transceivers including high speed back planes and chip to chip interconnects, working in different data rates and using a range of input reference clocks are extremely desirable (Table 4.1.2). In this chapter several novel fractional-N DLL-based synthesizers

are proposed to provide the required frequency selectivity. This architecture benefits from the low close-in phase-noise performance of the highly integrated DLL-based synthesizer and the small channel spacing of the fractional-N architecture.

Table 4.1.1: Summary of the wireless communication standards

	AMPS	IS-54	GSM	DECT	802.11b
Origin	EIA/TIA	EIA/TIA	ETSI	ETSI	IEEE
Access	FDD	FDM/FDD/ TDM	FDM/FDD/ TDM	FDM/FDD/ TDM	FH/FDM
Modulation	FM	QPSK	GMSK,diff	GFSK	(G)FSK
RF Channel Frequency (MHz)	824-848 (Tx) 869-893 (Rx)	824-848 (Tx) 869-893 (Rx)	890-915 (Tx) 935-960 (Rx)	0: 1897.344 9: 1811.792	2400-2500
Number of Channels	833	833	124	10	75
Channel Spacing	30 kHz	30 kHz	200 kHz	1782 kHz	5000 kHz
Δf/f	0.08	0.08	0.0756	0.046	0.0408

Table 4.1.2: Summary of Serial link protocols

Protocol	Data Rate (Gbps)	Multiplication factor (1G ref.)	Output frequency	Division factor
sRIO	3.125	6.25	6.25	2
sRIO	1.25/2.5	6/5	6/5	4/2
SATA1/2	1.5/3	6	6	4/2
PCIe1/2	2.5/5	5	5	2/1
PCIe 3	8	8	8	1
CEI6-SR/MR	6.25	6.25	6.25	1
XAUI	3.125/6.25	6.25	6.25	2/1
XAUI-T	3.75	3.75	3.75	1
FC1/2/3	1.0625/2.125/4.25	4.25	4.25	4/2/1
SGMII	1.25	5	5	4
QSGMII	5	5	5	1
GE	1.25	5	5	4

4.2 Design Challenges

As mentioned in Chapter 2, VCDL provides the equidistant delayed versions of the reference. The rising (or falling) edges of these delayed versions are combined in the edge combiner block to generate the desired output frequency. Each rising (or falling) edge of the reference frequency resets all the accumulated jitter and helps to improve phase noise performance of synthesized clock/frequency. Any phase difference between the reference signal and VCDL output, so called in-lock error, introduces a timing error in the output signal that translates to spurious tones at the output spectrum.

There are several architectures for fractional-N PLL- based synthesizers introduced in Chapter 3. However, in a DLL-based synthesizer, any attempt to change the delay of delay cells in order to achieve a fractional multiplication ratio forces the DLL out of the lock. This results in an unavoidable increase in the in-lock error and thus, a dramatic degradation of performance. This also increases the level of the spurious tones at the output spectrum.

The output of the DLL-based synthesizer is presented in Figure 4.2.1. For Global edge combiner (see Chapter 2) the in-lock error results in a duty cycle change for all the cycles and a change of the period per every N cycle. For Local edge combiner (see Chapter 2) the in-lock error results in a change of the period of one cycle in every N cycle. In general, a period change happens every N cycles where N is the number of delay elements in the VCDL. In spite of the large amount of timing error and unwanted jitter and spurs, a component of the desired frequency also appears in the output spectrum. Let's consider the multiplication factor as $N+\alpha$ where N is the integer part of

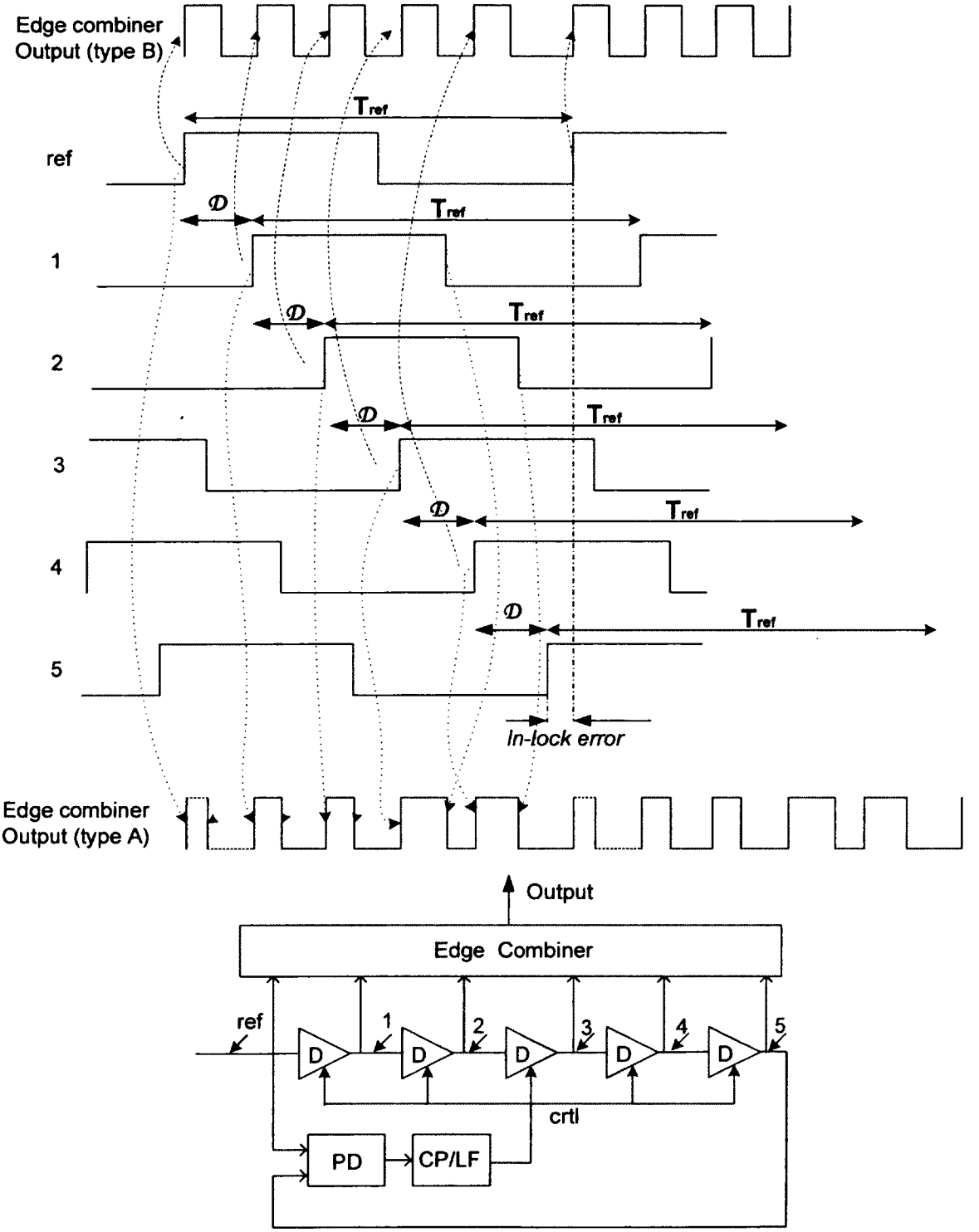


Figure 4.2.1: The effect of in-lock error on edge combiner output

the multiplication factor (that indicates the number of delay cells in delay line) and $0<\alpha<1$ is the fractional part of the multiplication factor. The value of the in-lock error is dependant on the fractional part of the multiplication factor. The in-lock error can be calculated as follows:

$$\text{In-lock-error} = \frac{\alpha}{N+\alpha} \times T_{ref} = \frac{N.\alpha}{N+\alpha} \times t_d = \alpha \times t'_d \qquad (4.1)$$

where $t_d = \frac{T_{ref}}{N}$ is the delay of one delay cell when DLL is in integer mode and $t'_d = \frac{T_{ref}}{N+\alpha}$ is the delay of a delay cell when DLL is in fractional mode. Figure 4.2.2 shows the value of architectural in-lock error versus the desired fractional multiplication factor for VCDLs with a different number of delay cells. As observed there is a considerable in-lock error for most of the range.

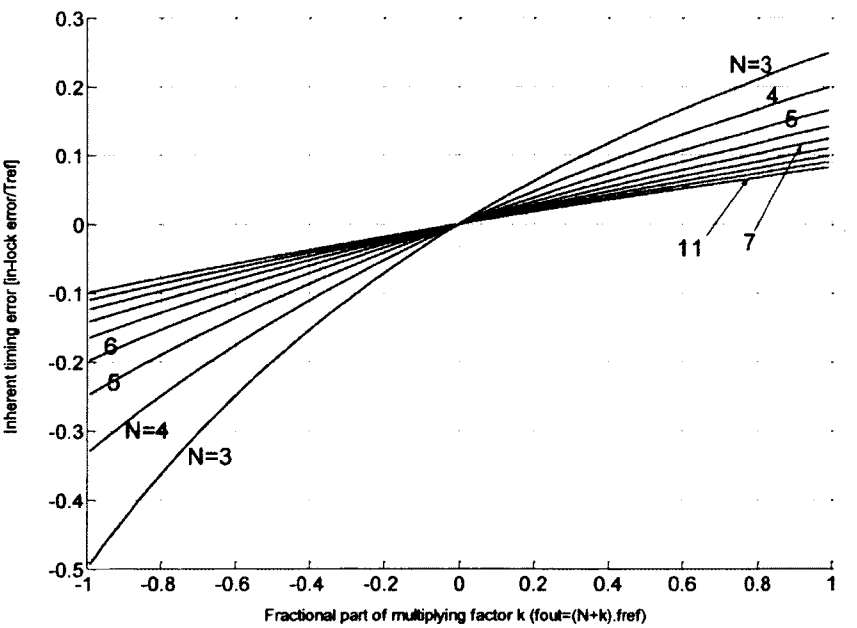


Figure 4.2.2: In-lock error vs. fractional ratio (unit interval = reference period)

4.3 ΣΔ DLL-based Fractional-N Architecture; Options and requirements

Figure 4.3.2(a) presents the proposed dual loop architecture for generating a fractional multiplication ratio using DLL-based synthesis techniques. The main idea is to change the input reference frequency by period synthesis techniques and then use a DLL multiplier to increase the frequency by an integer factor. This method is similar to the conventional method of generating a fractional-N ratio using a cascaded frequency divider and frequency multiplier (Figure 4.3.2(b)). Conventional method of using division is not suitable for DLL-based synthesis for the following reasons:

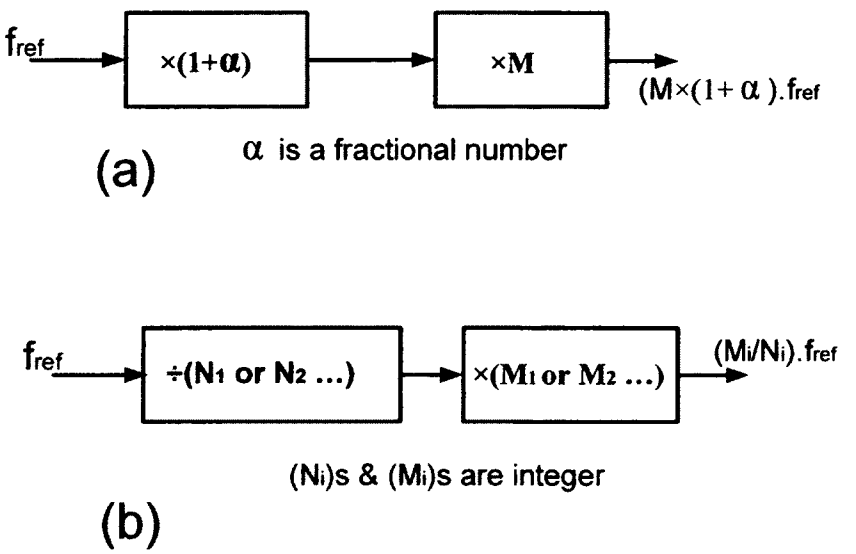


Figure 4.3.1: The Dual loop (a) and the conventional (b) method for fractional-N synthesizer

The reference frequency decreases by division which increases the required multiplication factor in the second stage for a certain target fractional multiplication factor. A higher multiplication factor requires a longer VCDL which causes more jitter. Note that the division in this process does not reduce the jitter as the input reference is already clean.

For example, consider the AMPS standard with channel spacing of 30kHz. In order to provide the desired channel spacing for AMPS when reference frequency comes from a 30MHz crystal oscillator a division by 1000 is required. This increases the required multiplication factor in the second stage. As a result, the required number of delay cells in the VCDL increases by an impractical factor of 1000. In addition, to cover the range of desired target frequency, the second stage has to provide a multiple integer multiplication factor which increases the complexity in DLL-based frequency synthesis.

The next couple of sections are dedicated to period synthesis techniques and architectures. The conventional period synthesis techniques and modified variable length alternative method will be discussed in detail and it will show that using $\Delta\Sigma$-based period synthesis is inevitable to achieve the required resolution. $\Delta\Sigma$ -based period synthesis and the nature of their induced timing errors are described in section 4.4.2. A few techniques to reduce or eliminate those timing errors will be introduced in following sub-sections. In section 4.5, single loop DLL-based fractional-N architecture will be introduced which benefits from combining the period synthesis and multiplication loops. Section 4.6 covers alternative proposed DLL-based fractional-N synthesizer architectures that can be implemented using combination of the same techniques.

4.4 Period Synthesis Architectures (PS)

The Period Synthesis (PS) circuit (see sections 3.4.2) is used to generate a signal with an arbitrary period (T_{new}) from an input signal with a fixed period (T_{ref}). It can generate a fractional-N frequency multiplication factor between 0.5 to 2. The conventional PS techniques use a digital DLL with fixed number of delay cells and, thus, exhibit limited resolution. In general, their output period is $T_{new} = T_{ref} \pm (i \, . t_{dc})$ where i is an integer and i<N (N is the numbers of delay cells in the VCDL and t_{dc} is the delay of one delay cell). In this section two new architectures for improving PS resolution are proposed. The first one utilizes an analog DLL with a variable number of delay cells, the second one uses a $\Delta\Sigma$ modulator.

4.4.1 Variable Length DLL Period Synthesis Circuit (VL-PS)

This proposed period synthesis circuit is using an analog DLL with a variable number of delay cells to increase the resolution of the output period [45]. Figure 4.4.1 illustrates the topology of the proposed period synthesis circuit which includes a DLL with an adjustable number of delay cells and two MISO switches. The first one (S1) to determine the number of delay cells in the loop; and the second one (S2) is used to generate an output signal with the period T_{new}. S1 is used to form a delay locked loop with a required number of delay stages (N) by connecting the N_{th} delay cell output to the phase-detector (PD). The full DLL part of the period synthesis topology is shown in Figure 4.4.2. To avoid false locking a flexible in-lock-range detector is required to take control of the charge pump and push the DLL into the lock range. The flexible in-lock-range detector must have the capability of adopting itself to a variable number of delay cells. A flexible in-lock-range-detector designed for the PS architecture is shown in Figure 4.4.3 Delay

cell outputs are sampled at the reference clock rising edges and analyzed by the in-lock-range detector. Depending on the number of delay stages in VCDL, a look-up-table and two MISO switches select the proper "under" and "over" signals. These signals overwrite the PD outputs and take control of the charge pump, if the delay is within the lock range, the control returns to the PD. The phase detector and the combinational logic are shown in Figure 4.4.4 The PD is armed when both the reference and delayed signals are low. The PD generates proper "up" or "down" signals corresponding to the arrival of the rising edge of the reference or delayed signal, respectively.

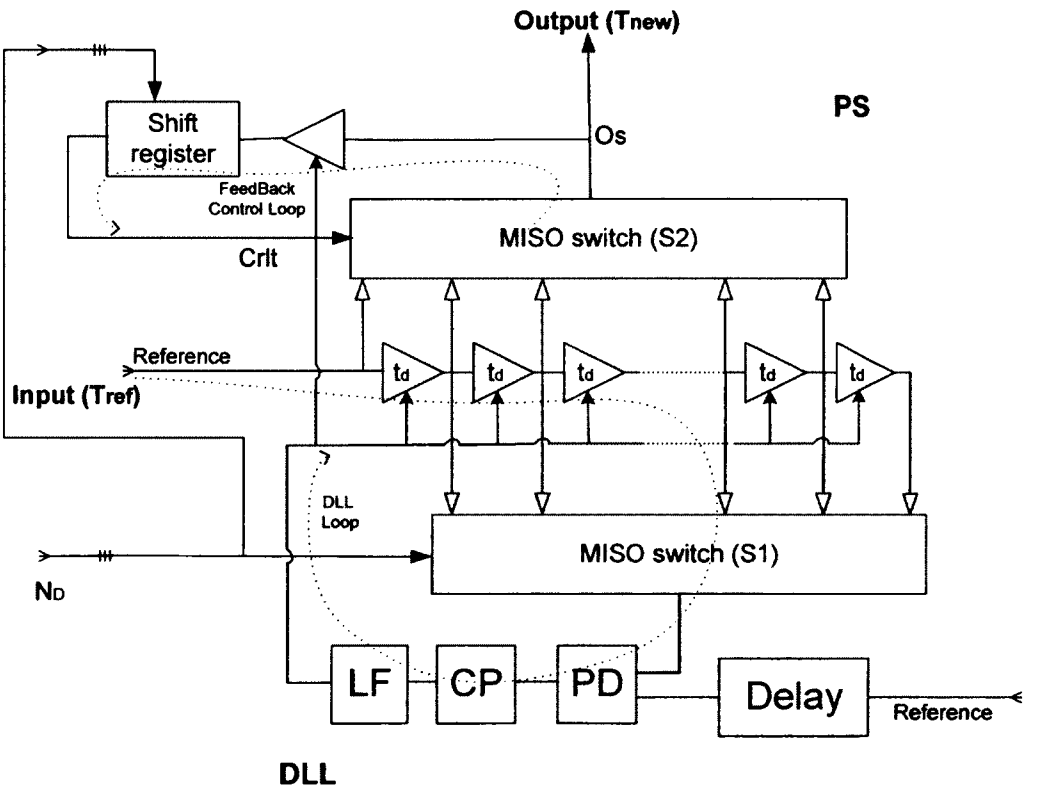


Figure 4.4.1: Proposed period synthesis circuit

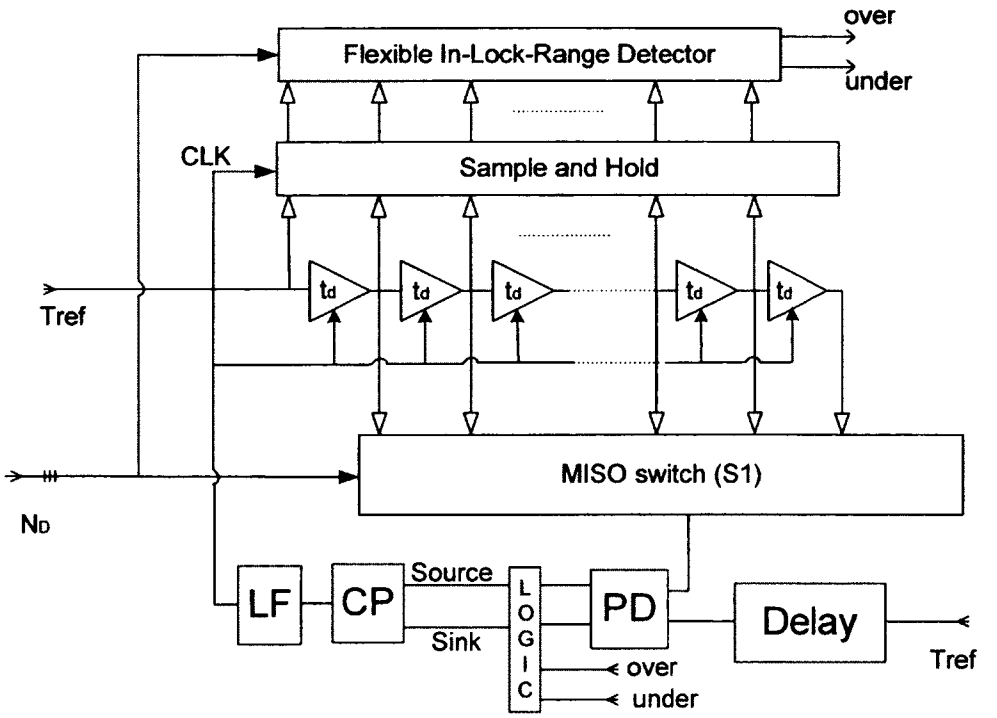


Figure 4.4.2: Proposed DLL with adjustable number of delay cells

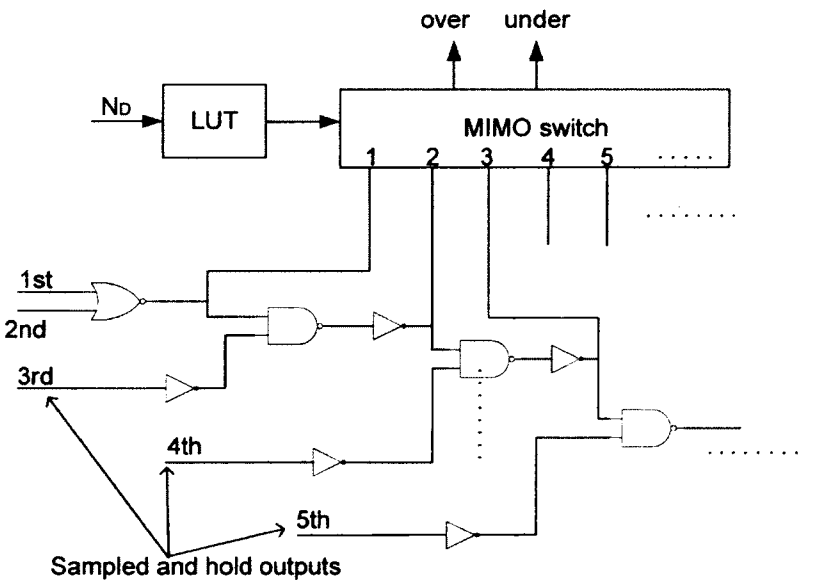


Figure 4.4.3: Flexible in-lock-range detector

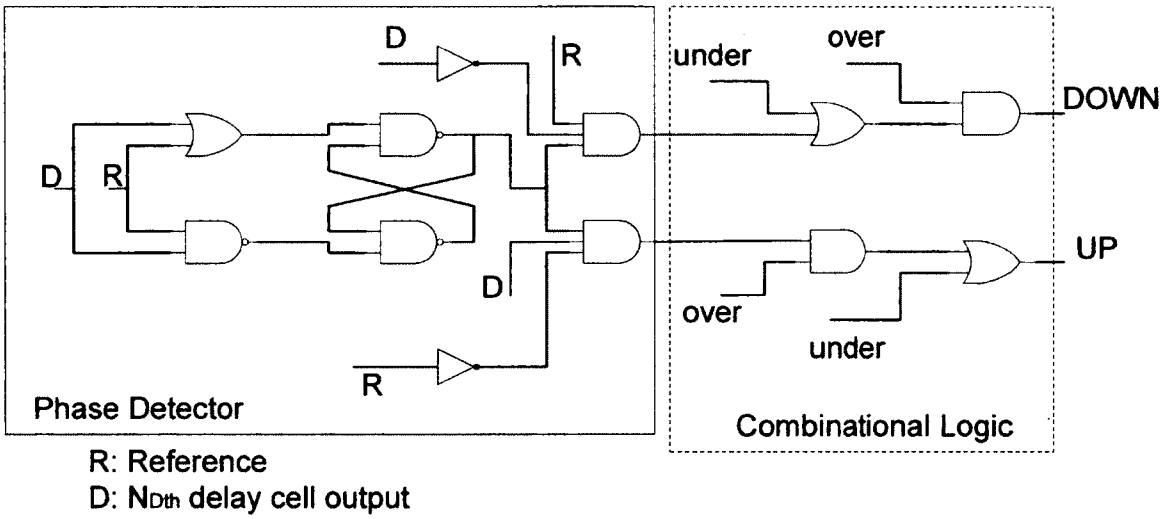


Figure 4.4.4: Phase detector (PD) and combinational logic

The "under" and "over" signals and the PD outputs form the inputs for a simple combinational logic block which provides the main "Source" and "Sink" signals that control the charge pump (CP). Note that using a single-ended charge pump topology the "Source" and "over" signals are active when they are low. Considering that the "under" and "over" signals are updated at every reference period, the delay of the critical path of the in-lock-range detector should be smaller than T_{ref}. The critical path becomes longer as the number of delay cells used in the DLL increases which could become a limiting factor at high operating frequencies when the number of delay stages in VCDL is a large value.

Further examination of the feedback control loop (Figure 4.4.1) reveals a race between the S2 control signal and i_{th} next/previous delay cell rising edge. This means that in order to prevent skipping a rising edge or sending one rising edge twice to the period

synthesizer output, the feedback control loop delay (S_t) should be within a certain range. For forward stepping the S2 switches to the next delay cell, S_t should be in the range of $t_{dc} < S_t < T_{ref}/2$. For backward stepping, the S2 switches to the previous delay cell and St should be in the range of $S_t < (T_{ref}/2) - t_{dc}$. The delay cell in the feedback control loop guarantees $t_{dc} < S_t$ while the upper limit for S_t is technology dependent and has to be established for high frequency applications. Any delay mismatch in the signal paths originating at VCDL and ending at the S2 output would cause an output spur (The same way any delay mismatch in delay cells induced delay (inside VCDL) generates spurious tone at the output spectrum). Therefore, it is important to ensure that all these paths have the same delay. The symmetry of the layout and usage of dummy cells allows for delay time equalization.

The inherent architectural error of generating $T_{new} = T_{ref} \pm D$ can be written as D - (i. t_{dc}). The normalized value of this error (ratio of inherent architectural error to reference period) for an 18 stage period synthesis circuit is shown in Figure 4.4.5. The solid line stands for the inherent architectural error for proposed variable lenght architecture and the dashed line shows the error for architectures with a fixed number of delay elements. Note that architectural error is directly related to the resolution of the synthesized period. The number of target periods that can be generated without any systematic error is increased by using variable length architecture. In addition, considerable error reduction is achieved for the entire range of target periods. Figure 4.4.6 shows the inherent error of the target frequency for the range of realizable multiplying factors. A higher reduction in frequency error is observed for larger multiplying factors.

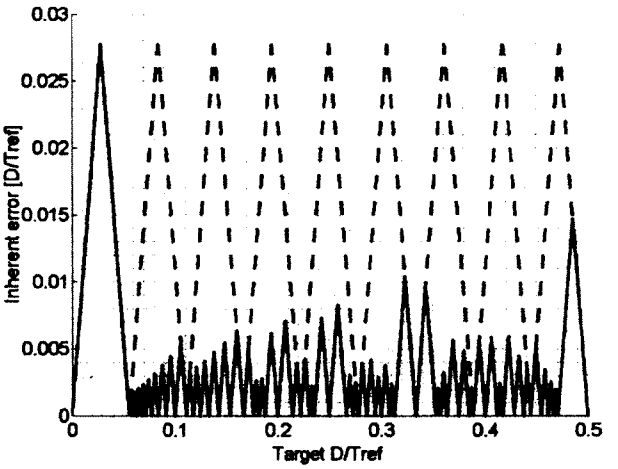


Figure 4.4.5: Inherent timing error of an 18 stage period synthesis circuit with, proposed (solid line) and conventional (dashed line) architectures

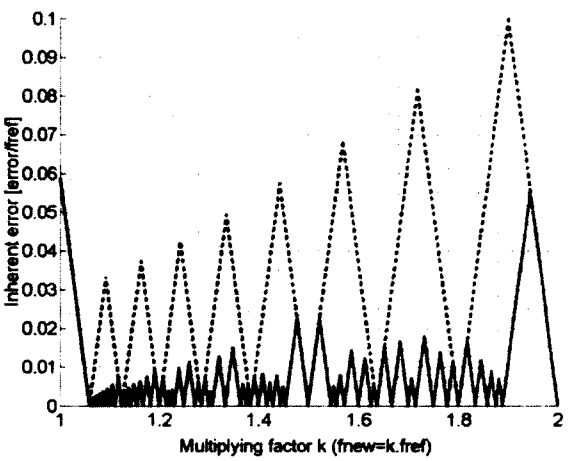


Figure 4.4.6: multiplication factor error of an 18 stage period synthesis circuit with, proposed (solid line) and conventional (dashed line) architectures

Note that this PS architecture can generate periods in the range of $0.5\times T_{ref}$ to $1.5\times T_{ref}$ (excluding $0.95\times T_{ref} < T_{new} < 1.05\times T_{ref}$).

4.4.2 ΔΣ-Based Period Synthesis Architecture (ΔΣ-PS)

Although the proposed period synthesis architecture based on an analog DLL with variable number of delay cells can generate a target period (frequency) with a higher resolution, it cannot be used as the first stage of the proposed method in the majority of applications for the following three reasons.

1) For many standards this resolution is not sufficient.
2) Unlike the channels in many communication standards, the target periods (frequencies) that can be generated without systematic timing error are not distributed uniformly in the frequency domain.
3) In most of the telecommunication standards the bandwidth to centre frequency ratio is small (typically less than 0.1).

This prevents the use of the entire output frequency range without changing the second stage integer multiplication factor (Figure 4.3.1a).

In general, period synthesizers change the reference period by $\left(\frac{i}{N}\right) \times T_{ref}$ where $(\frac{T_{ref}}{N})$ is the delay provided by the DLL and i is the number of delay cells jumped in each switch. Period synthesis approaches discussed in previous section, use an analog DLL with a variable number of delays and extra hardware in the barrel shift register makes it possible to change both i and N. This increases the frequency resolution and decreases the

timing error. However, i and N are both integer and a fine resolution is not achievable with a practical number of delay cells.

The ΔΣ-based period synthesis architecture proposed in this section is designed to provide a very high resolution fractional multiplication factor for the first stage. Using ΔΣ modulation in PLL-based fractional-N synthesizers is discussed in detail in Chapter 3. The use of this technique within the DLL structure is addressed in this section. Figure 4.4.7 shows the simple version of a ΔΣ -based DLL. In the conventional integer DLL, the output of the fifth or sixth delay stage would be connected to the phase detector which at lock condition, force the delay of each cell to be $\frac{T_{ref}}{5}$ or $\frac{T_{ref}}{6}$, respectively. The first order ΔΣ modulator generates a bit stream with the average of its input, noted by β ($0 < \beta < 1$) which controls the switches. Depending on the ΔΣ modulator output value, the output of the fifth or sixth delay cell is connected to the phase detector. As the average of the bit stream approaches β, the delay of the delay cells is eventually set to $\frac{T_{ref}}{5+\beta}$. Theoretically, the delay of each delay cell can be set with an extra fine resolution within the range of $\frac{T_{ref}}{5}$ to $\frac{T_{ref}}{6}$. The quantization noise is more severe for values of α close to 0 or 1. Using a higher order multi-bit ΔΣ modulator can reduce the quantization noise and increase the range of target delays. Figure 4.4.8 illustrates a ΔΣ-DLL with 3-bit ΔΣ modulator. In theory, the delay of the delay cells can be set to any value in the range of $\frac{T_{ref}}{3}$ to $\frac{T_{ref}}{11}$. Higher order modulators also have better noise shaping, therefore less quantization noise is expected in the middle of the range. The ΔΣ -based period synthesis architecture used in conjunction with a DLL can generate a target period with a very fine resolution.

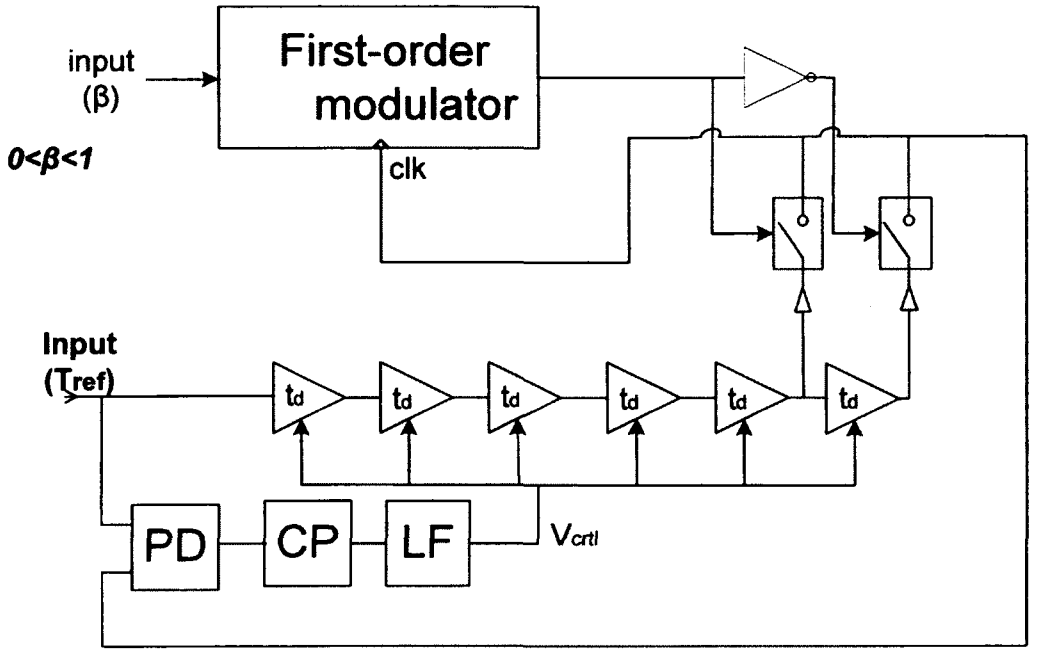


Figure 4.4.7: First-order delta-sigma DLL

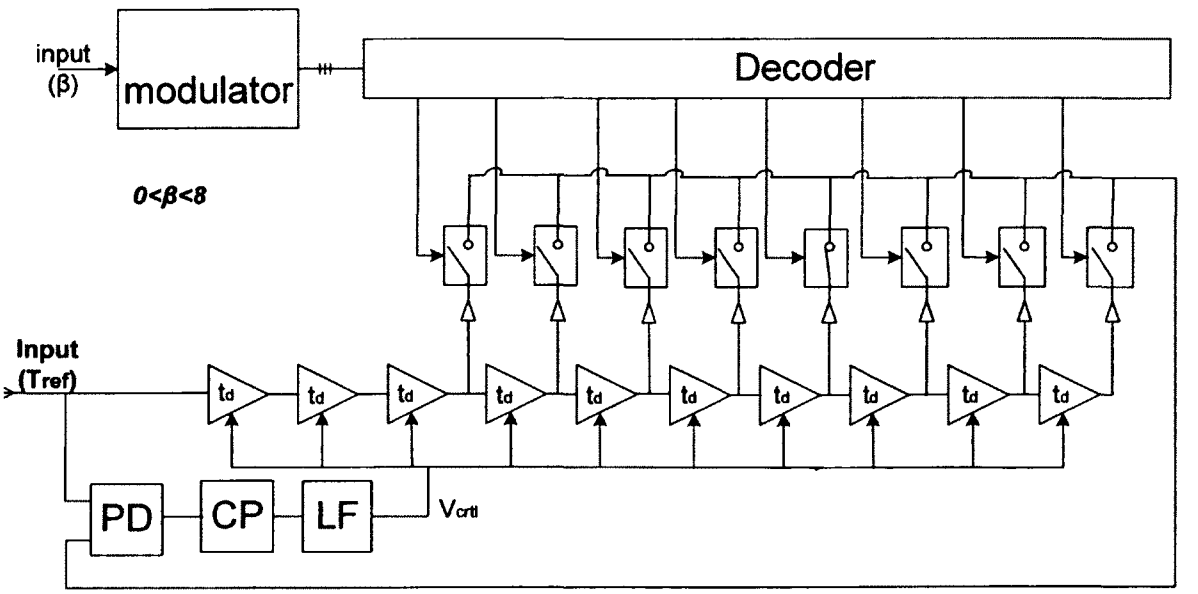


Figure 4.4.8: High-order multi-bit delta-sigma DLL

In general, by implementing a ΔΣ -based DLL in a period synthesis, a target period T_{new} can be achieved; where:

$$T_{new} = T_{ref}\left(1 - \frac{1}{N+\beta}\right) \times T_{ref} = \left(\frac{N+\beta-1}{N+\beta}\right) \times T_{ref} \quad (4.2)$$

and the output frequency f_{new} can be written as:

$$f_{new} = \left(1 + \frac{1}{N+\beta-1}\right) \times f_{ref} = (1+\gamma) \times f_{ref} \quad (4.3)$$

$$\gamma = \frac{1}{N+\beta-1} \quad (4.4)$$

Equation 4.4 shows the relation between the fractional part of the delay division factor and frequency multiplication factor for $\gamma > 0$. Note that β and γ are fractional part of period multiplication factor and frequency multiplication factor for PS output, respectively.

4.4.2.1 Timing Error in ΔΣ-based Period Synthesizer

When a conventional DLL is in lock condition, the output of the last delay cell is aligned to the reference signal. Therefore, in the conventional period synthesis process, switching to the last delay cell output is equivalent to switching to the input of the delay line and vice versa. After the period synthesis circuit connects the last delay cell to its output; PS connects the first delay cell to its output and repeats the process without introducing any timing error. In a ΔΣ DLL, the delay of the delay cells might be a fractional division of the reference. In this case, none of the delay cell outputs is aligned with reference and it is not possible to switch back to the other side of the VCDL without any timing error. This timing error has the same nature and value of the in-lock error discussed in the

beginning of this chapter. Timing error versus the fractional part of the frequency multiplication factor (β) is shown in Figure 4.2.2. In special cases, for example when $\beta = 0.5$ the timing error can be eliminated by doubling the length of VCDL (the number of delay cells in VCDL). In this case, the output of the delay line is aligned with the second harmonic of the reference. By using the concept of harmonic locking, the timing error can be minimized by making a proper choice of an effective length for VCDL (the number of the delay cells in VCDL). Figure 4.4.9 shows the number of delay cells needed to achieve a timing error of less than $0.01 \times T_{ref}$ for a fractional division of $\frac{T_{ref}}{10+\beta}$. A timing error less than 1/(max number of delay cells) can be achieved by choosing a proper returning point in the period synthesis process. An interesting region is $-0.1 < \beta < 0.1$ where the period synthesis demonstrates timing error of less than $0.01 \times T_{ref}$ without any extra delay cell.

RMS Jitter less than 0.1 UI can be achieved if the multiplication factor of the following stage is less than 10. Note that the equivalent range of the multiplication factor (β) is less than 0.002 (for N=10). Given the discussion above, it seems that compensating for the timing error becomes inevitable in a practical implementation of DLL-based fractional-N synthesizer. The following section discusses the possible solutions for this problem.

4.4.2.2 Timing Error Compensation Method

As shown in the previous section, a timing error is inevitable when the period synthesis circuit switches back to the opposite end of the line. This error is due to the fact that in a ΔΣ-DLL the reference is not aligned with the output of delay line. Figure 4.4.10 shows a circular timing diagram for two four stage delay lines. The first one is in the main DLL

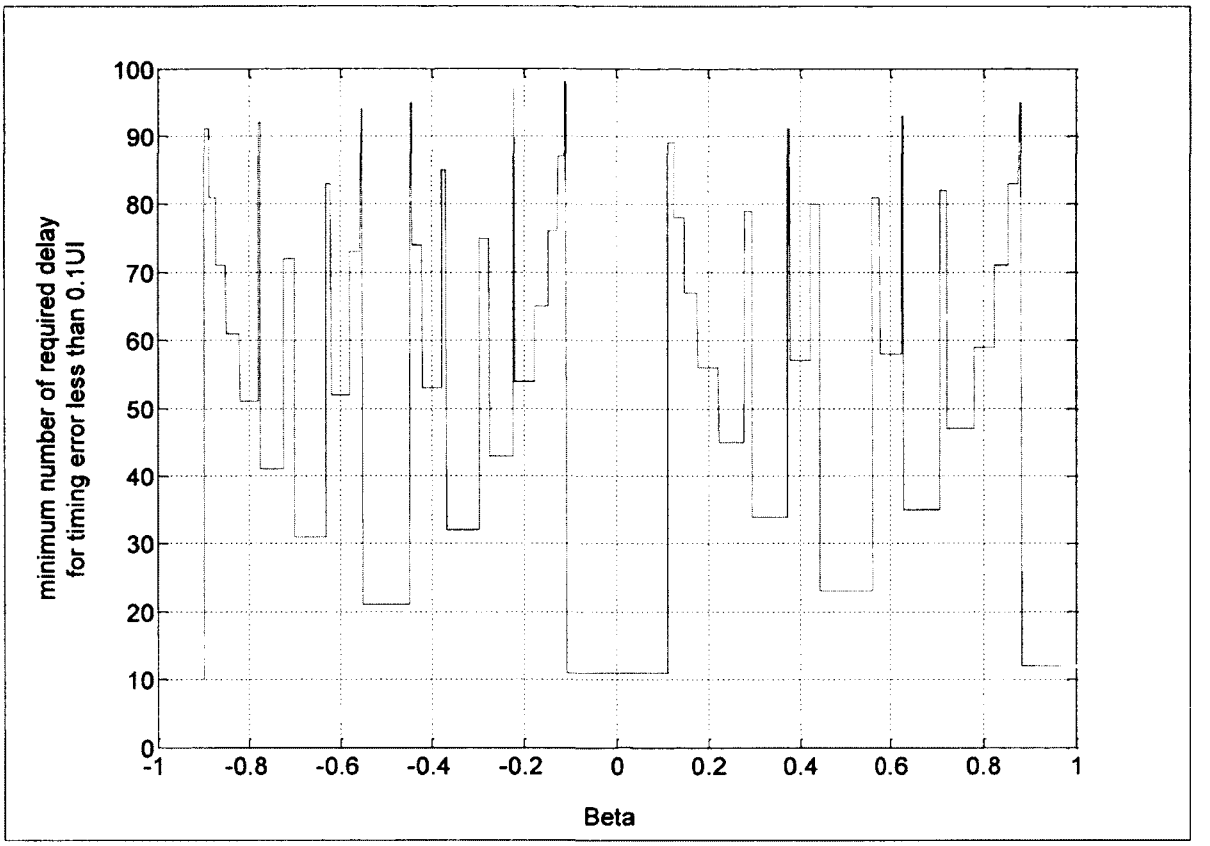


Figure 4.4.9: The minimum number of delay cells that the output is aligned with reference with less than 0.01 T_{ref}

and the second one is in an auxiliary DLL for error correction. The hollow circles show the phase of the output of each delay cell when the main DLL is in integer lock condition (integer mode), i.e. the output of the delay line is aligned with the reference. In this condition, switching back to the other end of the delay line does not generate any error. The shaded circles show the output phase of each delay cell when the DLL is in a fractional mode, i.e. the output of the delay line is not aligned with the reference. In this case switching to the other end of the line generates an error. This error has the same nature of the in-lock error but usually with much higher amplitude. The value of the timing error can be formulated as:

$$t_e = (N \times t_{dc}) - T_{ref} = \frac{-\beta}{N+\beta} \times T_{ref} \qquad (4.5)$$

Note that the timing error t_e occurs every N_{th} reference cycle where N is the number of delay elements in the VCDL. In theory this timing error (t_e) can be eliminated by using a second line (Figure 4.4.10). The reference of the second line has to be aligned with the output of the main delay line. The delay cells of the second delay line should have the same delay as of the delay cells of the first delay line. The PS circuit is switched to the second delay line without generating any error (instead of switching to the other end of the same line). There are many ways to implement a second delay line. A simple method is to implement it right after the first delay line. This results in a longer delay line for the PS circuit. Worst case timing error can be reduced to t_e / M by using a delay line that is M times longer and by choosing a proper switching time. In this case a timing error occurs every N×M cycles. In a PS circuit, using a longer delay line requires a large MISO switch which is impractical for large M. In addition, a longer delay line is more jitter-prone and degrades the performance of the PS output.

Another way to compensate for a timing error is to alter the period of the PS input, which in this case is the reference period (Figure 4.4.11). The set of the phase detector, charge pump and loop filter (PD-CP-LF) works to align the output of the delay line by adjusting the delay of the last delay cell to t_e . This way the timing error is detected and replicated to all delay cells of a second delay line. Note that the delay of delay cells of the primary VCDL are set to t_d by using a ΔΣ modulator as previously discussed.

Likewise the original period synthesis technique, the input period changes every N_{th} cycle by switching S3 to the next/previous delay cell. As the delay cells in the second VCDL

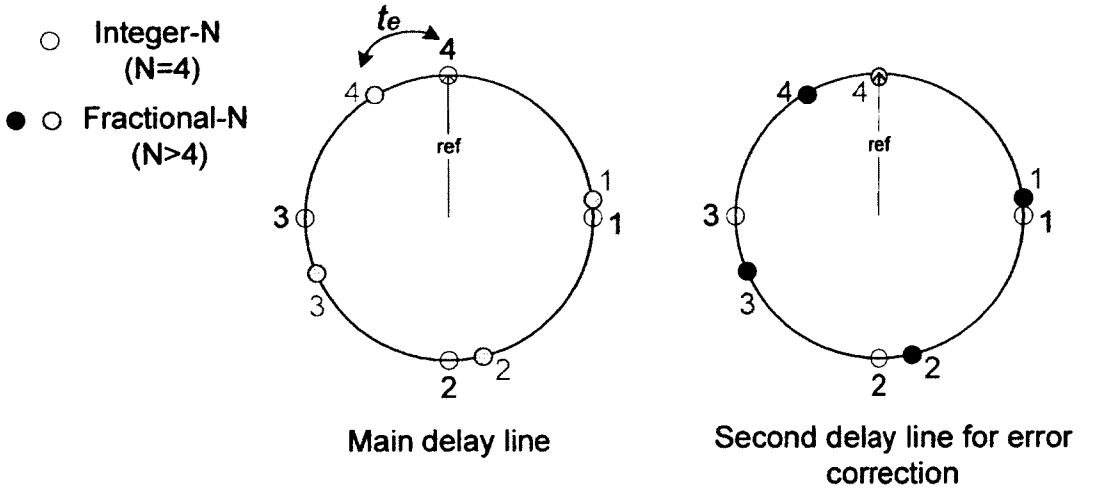


Figure 4.4.10: The circular timing diagram for ΔΣ -PS; its timing error and theoretical solution: a second delay line

have a delay equal to the timing error t_e, it provides the mean for delaying the reference signal by t_e and removing the timing error. In other words changing the input period by t_e, timing error due to fractional-N multiplication factor in ΔΣ loop is eliminated. Note that the timing error still occurs when S3 switches to the other end of the second VCDL. However, this new timing error occurs every N×M reference cycles with a worst case value of t_e /M where M is the number of delay cells in the second delay line. The overall delay of the second delay line should be almost equal to the reference period in order to minimize the timing error. The delay of the MISO switch does not introduce any error when the reference itself passes through the same switch (it subjects to almost the same delay). This method reduces the occurrence and amplitude of the timing error with a factor of 1/M by using only M+N delay cells. Considering the fact that the longer delay line is more jitter-prone, this new topology is expected to have a better jitter performance than using a single line with N×M delay cells.

Figure 4.2.2 shows the value of the timing error t_e . As it can be observed, the error t_e is small and negative in some ranges. The above method cannot be used when t_e is very small. Although it can be used for an acceptable wide range of multiplication factors, the phase-noise and jitter performance degrade by increasing the number of delay cells in the clock path. The timing error can be further reduced by repeating this technique. In general, the minimum error that can be corrected with this method is equal to the minimum delay of a delay cell.

4.4.2.3 Timing Error Correction Method

This section discusses another error correction technique which can be used for correcting even negative and/or small timing errors. It also uses a finite number of delay cells for high a resolution correction. The main idea is to correct the timing error by the means of changing the delay of one (or a few) delay cell(s) as shown in Figure 4.4.12.a for a five stage period synthesis.

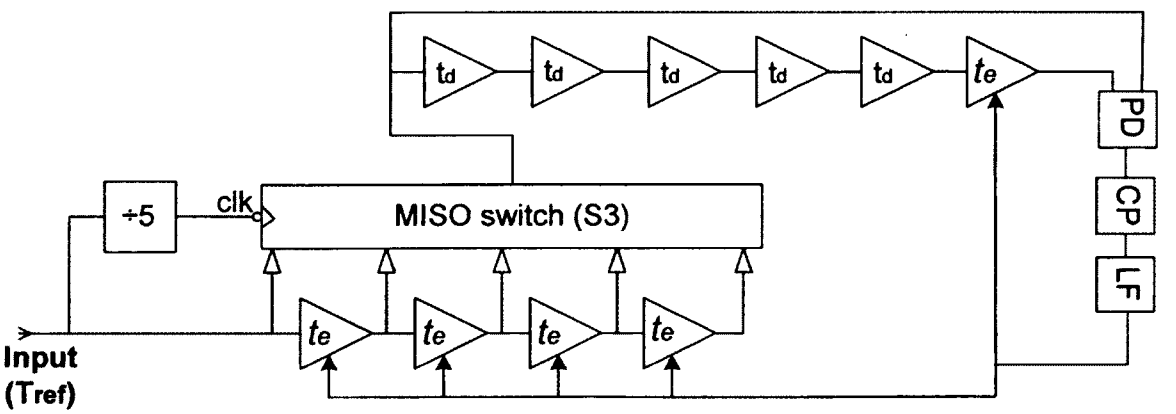


Figure 4.4.11: The timing error compensation circuit

The delay of the first stage is changed to $t_c = t_d + i \times t_e$ where i is incremented each time as the period synthesis circuit switches the output to the other end of the line. In this particular example (with a five stage VCDL period synthesis circuit) the timing error occurs when the circuit switches from point 5 to 1 or vise versa. At this point, the delay of the first cell must increment or decrement by $|t_e|$. The correcting delay t_c is generated in two steps. In the first step, a delay $t_{de} = t_d + t_e$ is produced by a sense circuit using a duplicated delay line (Figure 4.4.12.b). Then the delay t_e is copied to the delay cell (x) and corrects the error the first time the period synthesis circuit switches from 5 to 1 (or from 1 to 5) (Figure 4.4.13).

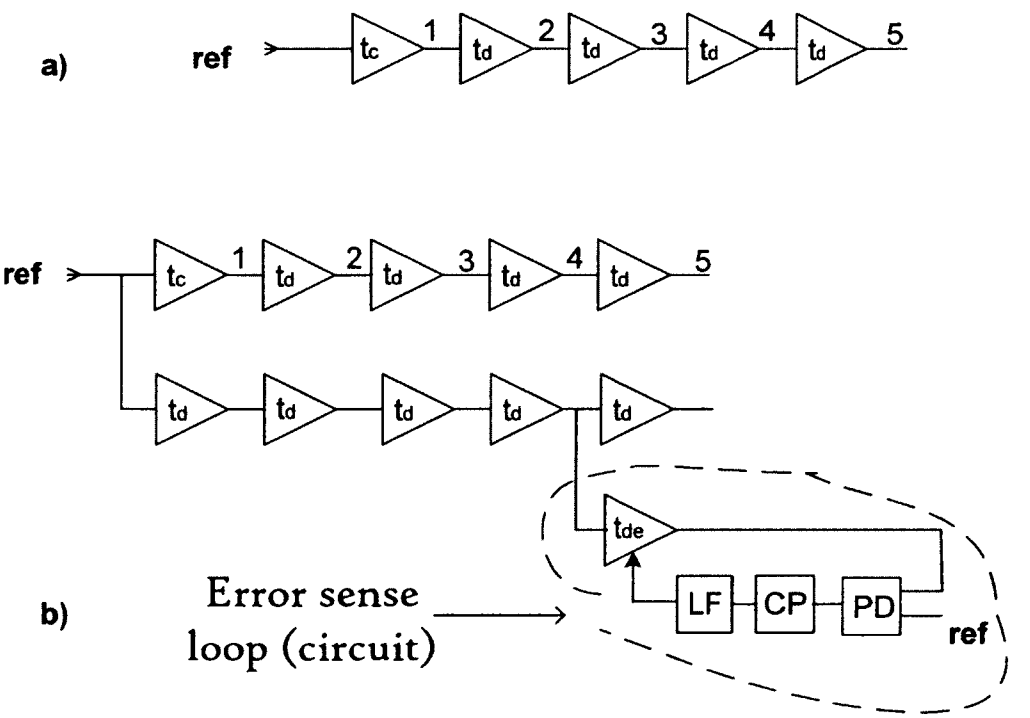


Figure 4.4.12: The timing error correction a) By changing the delay of the first delay cell b) Error sense circuit

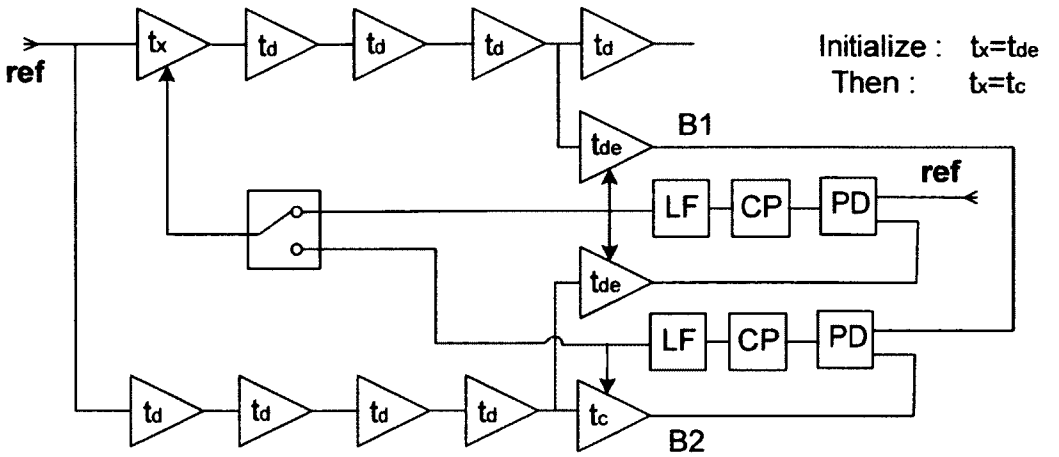


Figure 4.4.13: The timing error correction circuit

It is also copied to another delay cell at the end of the main delay line to provide a reference point (B1) which is used to generate the correcting delay t_c.

Then another set of sense circuitry compares the rising edges on B1 and B2 and generates the correcting delay t_c. The delay t_c is always equal to $t_x + t_e = t_d + k \times t_e$ (integer k) and therefore, the incremental delay is provided for the next time. From this point, the timing error is corrected by copying the delay t_c to the delay cell x every time when the period synthesis circuit switches to the other end of the line.

There are two main critical issues in implementing such error correction techniques:

1) Copying the delay of one delay cell to another identical delay cell with a good resolution at a specific instant

2) Keeping this delay constant until the next updated delay value copies to the cell

Delay cells are very sensitive to changes in the control voltage; and copying a voltage value is susceptible to noise. On the other hand, many of the delay cells, in fact, are controlled by a current which is easy to add and subtract in a node. Figure 4.4.14.a shows a typical delay cell. The delay of the delay cell is proportional to the current $I = I_1 + I_2$ which passes through it. I_1 and I_2 are controlled by two independent sets of circuitry. In this case the set of the phase detector, charge pump and loop filter (PD-CP-LF) in the period synthesis circuit (the one that is controlled by a ΔΣ modulator) generates I_1 current which is used to set the delay of the delay cells to t_d. The error sense circuit generates a current I_2 which is meant to cause a slight change in the value of delays t_{de} or t_c. If all of the delay cells are identical, then by applying I_2 the delay of any delay cell can be changed to t_c . Therefore, the delay of the cell in the error sense loop is copied to the target delay cell by copying this current. The circuit shown in Figure 4.4.14.b can be used to copy a current at a certain instant which in fact is a current mirror with a control switch. As mentioned earlier, the delay t_c must be copied to delay cell x in a particular

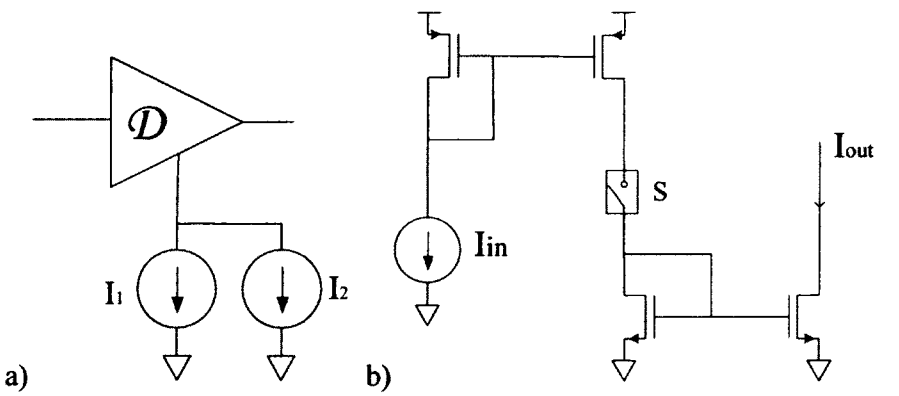
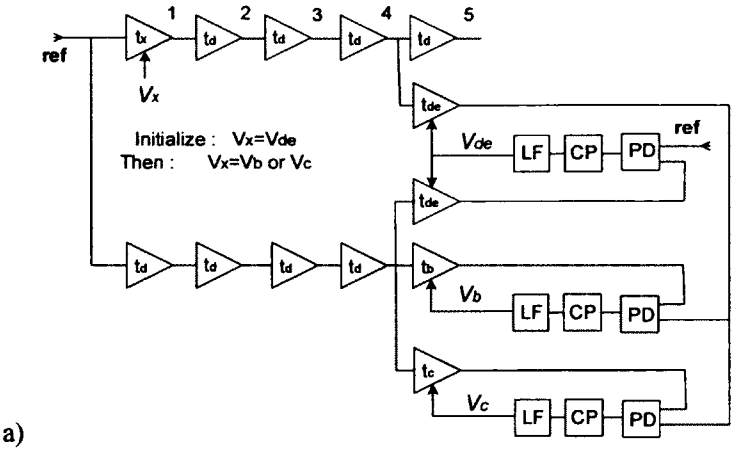


Figure 4.4.14: Delay cell and switched current mirror to copy the delay

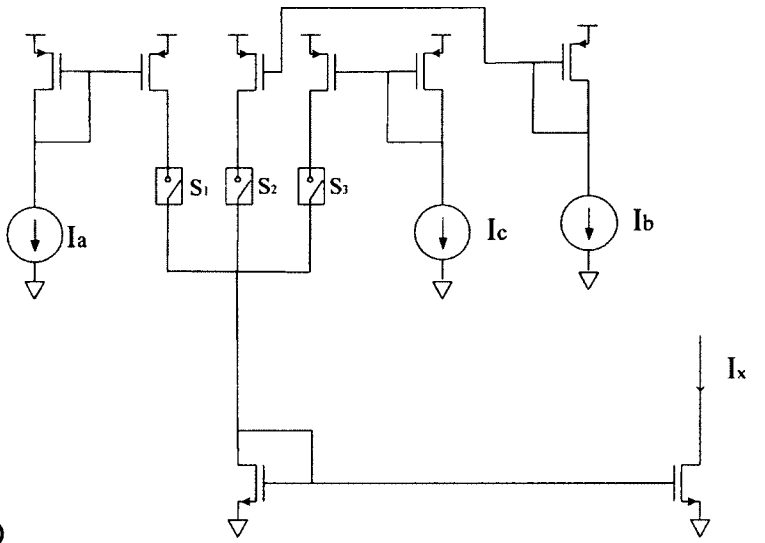
instant. This affects the reference point B1 and enforces the sense loop to align B2 by changing t_c. It is desirable that t_x (the delay of the delay cell x) hold its value until the next time when the new t_c is copied to it. Using the circuit in Figure 4.4.14.b, the current I_{in} can be copied to the output by closing the switch. However, as long as the switch is closed, I_{out} follows the change of I_{in} which in this case is not intended. On the other hand, by opening the switch I_{out} falls to zero which is also not desired.

The modified architecture and its modified switched current circuit are shown in Figure 4.4.15. At the beginning all of the delay cells have the delay of t_d. Every time when the period synthesis circuit switches to the other side of the main delay line, the delay of the first delay cell is modified by switching to a new current.

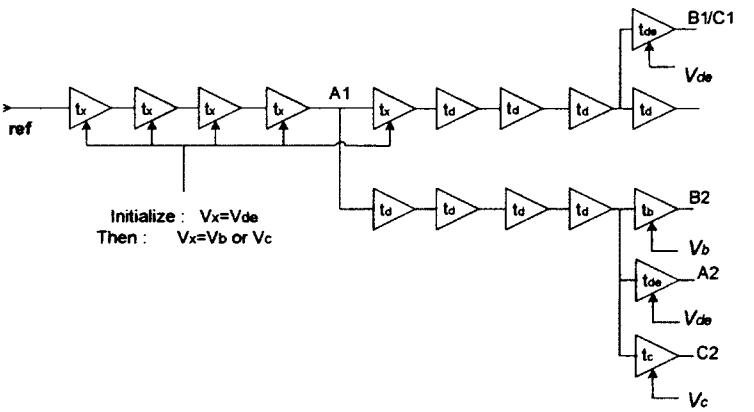
In the first step, by closing switch S1, the current of I_a which corresponds to the control voltage A (generated by the comparison of the phases on points A1 and A2) is copied to I_X and consequently changes the delay of the first delay cell x. After that, at right switching time, by closing one of the switches S2 or S3, I_X is set to currents $I_b$or I_c (correspond to control voltage B or C generated by comparison of the phases on points B1 and B2 or C1 and C2). At any point in time, only one of the switches S1 or S2 or S3 is closed. Thus I_X keeps its value until it is updated by switching to a new current. As illustrated in Figure 4.4.15, delays t_b and t_c are both set to $t_d + k \times t_e$; one is used for odd values of k and the other for even values of k. When I_X=I_b (i.e. S2 is closed), the corresponding phase detector should be inactive in order to keep the current value and delay constant. Therefore, cell x has a constant delay equal to t_b from the instant when the switch S2 is closed until a next value is copied to this cell.



a)



b)



c)

Figure 4.4.15: Complete error correction architecture and its current switch circuit

At the same time, delay t_c is adjusted to a new value and the corresponding drive current I_c is prepared to copy to I_X and changed the delay of the cell x to a new value.

The resolution of this error correction architecture is directly related to the sensitivity of the PD and response time of the PD-CP-LF set, the current mirror, and delay cells. By a proper design, the timing error can be reduced to the order of the in-lock-error in a conventional (integer-N) DLL-based synthesizer.

Note that the delay cell x, must follow the timing error for a full range of one reference period. This range can be realized by using several delay cells; i.e. the new current is applied to each delay cell one at a time. Using several delay cells also provides more time for the sense loop to lock to the error value.

4.5 Single loop DLL-based fractional-N Synthesizer; Proposed Architecture

Dual/Multi loop solutions for the DLL-based fractional-N synthesizer have been explored in previous sections. A ΔΣ modulator has been used to achieve required resolution for period synthesis. A couple of solutions have been presented to address inherent timing error associated with the ΔΣ -based period synthesis. Both proposed error compensation and error correction schemes required delay locked loops. Therefore any complete dual/multi loop solution with error correction/compensation involves two or more simultaneous loops. On the other hand the relation between synthesizer multiplication factor and the ratio on T_{new} to T_{ref} is not straight forward. The period synthesis generates $T_{new}=(1+\beta).T_{ref}$ and subsequent DLL stage multiply the frequency by M so $T_{out}=T_{new}/M$,

the general fractional-N multiplication factor of synthesizer include integer part of N and fractional part of is equal to Q where:

$$Q = \frac{T_{out}}{T_{ref}} = \frac{(1+\beta)}{M} = (N + \alpha) \qquad (4.6)$$

As can be observed a floating point operation or a look up table is necessary to calculate β from α which add to the complexity of a multi-loop implementation. The high complexity of multi-loop architectures may not be appealing for some practical implementations. As an example clock synthesizer power and area are critically important in high-speed interconnect applications and synthesizer performance can be compromised to some extend for a better form factor, lower power and less design risk and effort.

In this section a novel single loop DLL-based fractional-N architecture is proposed. Single loop architecture has following advantages:

1) It involves only one loop, a ΔΣ -controlled DLL which is called the Master loop.

2) Period synthesis and frequency multiplication are both using slave delay lines; a slave delay line is controlled by Master loop control signal without being inside the loop.

3) There is a simple relation between period synthesis output period and synthesizer multiplication factor.

The single loop architecture (shown in Figure 4.5.1) can be divided into three sections: Master loop, period synthesis (PS) and frequency multiplier. It includes three delay lines (one in each section) that share a common control signal.

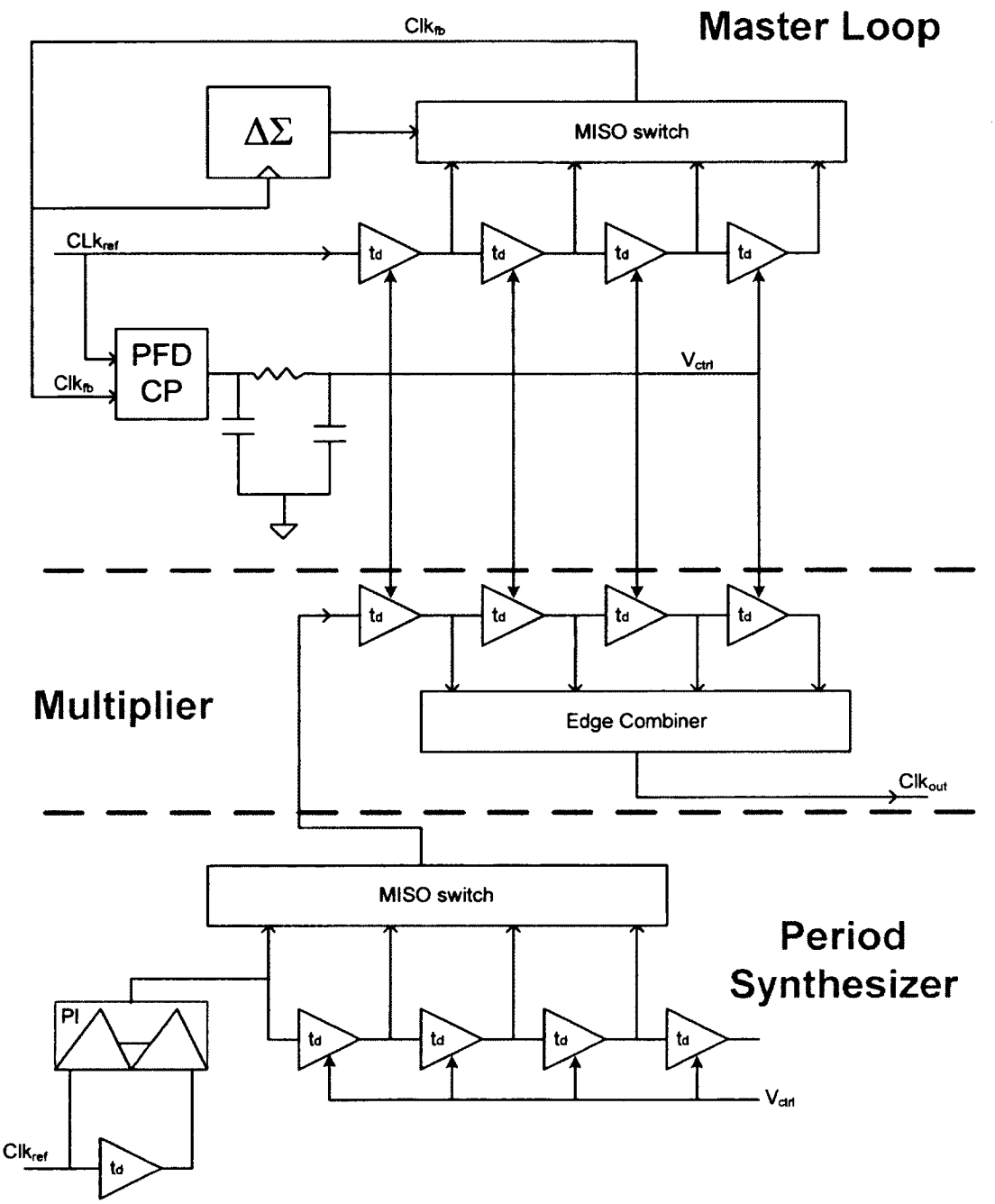


Figure 4.5.1: Block diagram of single loop DLL-based fractional-N frequency synthesizer

All the delay lines use the non-inverting delay cells consists of two inverting delay element. The first delay line (master delay line) is inside a delay locked loop. This delay locked loop sets the delay of all three delay lines. The ΔΣ modulator incorporated inside the master loop gives the ability of setting the arbitrary delay with a good resolution. The details of the master loop operation will be discussed in the next subsection. The second delay line is connected to an edge-combiner to generate the desire output frequency. Edge combiner mixes multiple equidistant phases of the clock and makes a higher frequency signal. The third delay line is used with accompany of a phase interpolator to form a period synthesis. It aligns the second delay line input and removes timing error.

4.5.1 Master loop

When the conventional delay locked loop (described in Chapter 2) is in lock position, the output of the delay line and the reference clock are aligned and the delay of the delay line is equal to one period of the reference clock. The delay of each delay element is $t_d=T_{ref}/N$ where N is the number of delay elements in the delay line and T_{ref} is the period of the reference clock. Then the edge combiner gets these N equidistant phases of the clock and generates a signal with period t_d that has frequency of $N.f_{ref}$. Note that both in the DLL and in the edge combiner, N is inherently integer. The first step to make a fractional DLL frequency synthesizer is to set the delay of the delay elements to an arbitrary delay of $t'_d=T_{ref}/Q$, where Q is a fractional number. Q has been shown as $Q=N+\alpha$ where N is integer and $-1 < \alpha < 1$. The role of the master loop in this design is to generate t'_d. A typical diagram of a master loop is shown in Figure 4.5.2. For setting the delay of delay elements to t'_d when $N<Q<N+1$ ($\alpha>0$), the outputs of the N^{th} and the $(N+1)^{th}$ delay cells would send to phase detector with the probability of α and $(1-\alpha)$, respectively. It is

equivalent to changing the divider index in a PLL-based fractional-N frequency synthesizer [78] with one exception; here the delay is modulating not the frequency. The same discussion can be made if N-1<Q<N and α<0. A ΔΣ modulator is used to select the appropriate connection at any point in time. ΔΣ modulator generates a sequence of the quasi-random binary output with the average of its input; here the fractional part of the target multiplication factor. The output of the ΔΣ modulator also has a high pass characteristic that pushes the spurs results from switching to a higher frequency that can be suppressed by the loop bandwidth. Extra attention is required to make sure switching happen when both delay cells are at the same state (both must be low or high). Otherwise a glitch might appear at the input of the phase detector and be interpreted as a false edge.

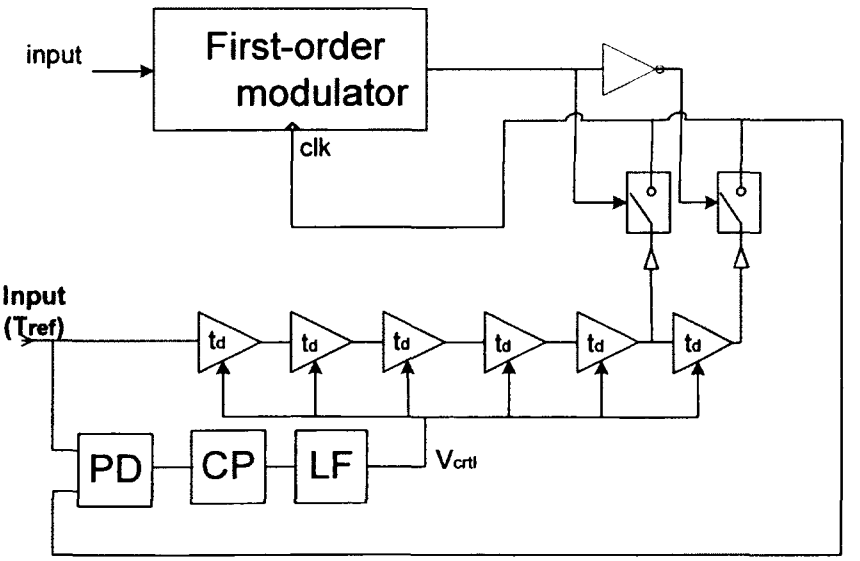


Figure 4.5.2: A typical Master Loop with a first order ΔΣ modulator

4.5.2 Frequency multiplier; Main delay line and Edge-combiner

The main delay line shares the control voltage with the master delay line and thus each of its delay elements impose t'_d delay. By applying the reference clock to the input of main delay line and giving the outputs of it to the edge combiner a clock with frequency of $Q.f_{ref}$ is generated with only one problem, one of every N cycle of clock has a different period (Figure 4.2.2). This is because the reference rising edge is no longer aligned with any of main delay line outputs. Or in other words $T_{ref} \neq N \cdot t'_d$. The timing error is equal to:

$$t_e = T_{ref} - T_{new} = T_{ref} - N.\ t'_d = T_{ref} - N.(T_{ref}/Q) \qquad (4.7)$$

$$t_e = \alpha.\ t'_d = (\alpha N/N+\alpha)t_d \qquad (4.8)$$

To remove this timing error the input of the main delay line has to be shifted by t_e in each period. Simply put, the period of the input signal of the Slave delay line should set to $T_{ref}+t_e$. This can be done by using period synthesis techniques. However, the fact that the ratio of timing error to the delay of one mater loop delay cell is equal to the fractional part of multiplication factor ($\frac{t_e}{t'_d} = \alpha$) helps to have a simpler and more robust period synthesis scheme without any loop. This period synthesis scheme will be explained in next sub-section.

4.5.3 Period synthesis for timing error compensation

Figure 4.5.1 shows a period synthesis circuit consists of a delay line and a phase mixer (interpolator). This period synthesis is employed to shift the reference clock edge by t_e and compensate for the timing error in the main delay line. In fact the output of the period

synthesis has a period of $T_{ref}+t_e$. The delay line shares its control voltage with the master loop and thus each of its delay elements imposes the same delay, i.e. t'_d. The phase mixer interpolates between the two phases of the clock with t'_d timing distance based on its digital control input. A phase mixer can be designed to have over 100 setting (steps), allowing it to generate timing resolution as small as $t'_d/100$. The new desire period T_{new} ($T_{new}=T_{ref}+t_e = T_{ref}+\alpha.t'_d$) is generated by incrementing (or decrementing depends on value of α) the setting of the phase mixer at each reference period. As α is the fractional part of the multiplication factor, this architecture can theoretically generate any fractional frequency with the step of the 1/100 (1/PIsteps) without any timing error. It also can generate finer resolution with a maximum timing error smaller than $t'_d/200$ ($t'_d/2$PIsteps). This error happens only one out of N cycles in the output frequency. The delay line is used to extend the range of delay variation to one reference cycle. In this case switch and phase mixer setting sets back to its initial condition. The whole cycle of compensation circuit switch and setting is as follow:

Cycle starts when the multiple input single output (MISO) switch connects the interpolator output to the main delay line input. At each reference cycle, interpolator setting increment/decrement by m where m=ROUND(α.PIsteps). The interpolator setting is control with an accumulator. In the event that the accumulator overflows, MISO switch connects the next delay elements output to the input of the main delay line. This process continues until the last delay elements connect to the input of main delay line. When overflow happens in this situation, MISO switch connects the phase mixer output back to the main delay line input and adds 2×m to the accumulator to account for the timing difference of these two points.

4.6 Alternative Solutions; Other Proposed Architectures

Some alternative solutions and techniques have been considered for a DLL-based fractional-N frequency synthesis which results in a few novel architectures. A brief explanation of these novel architectures are presented in this section.

4.6.1 ΔΣ Multiplying DLL Period Synthesis (ΔΣ -MDLL-PS)

The proposed period synthesis (PS) architecture using Multiplying DLL (MDLL) is shown in Figure 4.6.1 The target delay is generated by a ΔΣ -DLL (Figure 4.5.2). This ΔΣ-DLL acts as a master loop and the MDLL act as a slave loop. All delay cells in the ΔΣ-DLL and MDLL impose the same delay to their input signals. After applying the reference to the MDLL delay line, MUX connects the output of the delay line to its input and converts it to a ring oscillator. Note that the delay of each delay cell and, thus, the frequency of the ring oscillator is controlled by the master loop. In addition, when the master loop imposes a delay $t_d$where $t_d \neq \frac{T_{ref}}{N}$ for N integer, the next reference rising edge is not aligned with the delay line output (feed back path of the ring oscillator). However, switching to the other end of the line (from 1 to 6 or 6 to 1) does not generate any timing error as the delay line is converted to a ring oscillator. The MISO switch works in the same way as in a ΔΣ -DLL. It is better to insert the reference signal to MDLL in order to reset the accumulated jitter. However, this has to be done without introducing a timing error or at least by introducing an acceptable timing error. In order to do this, a phase detector is used to sense the phase difference between the reference signal and feedback path signal. If they are almost aligned (the timing error is less than the threshold) the MUX switches to the delay line position (position A in Figure 3.4.5)

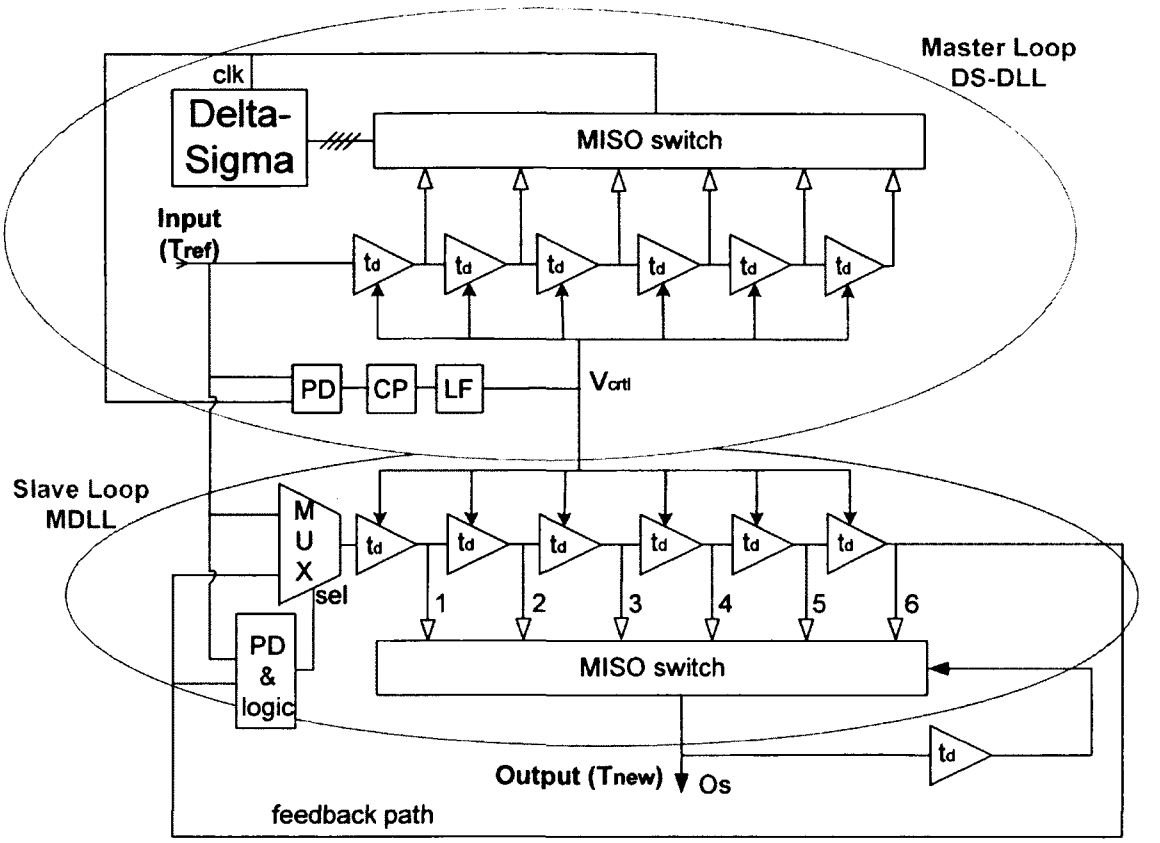


Figure 4.6.1: Proposed ΔΣ multiplying DLL period synthesis circuit (ΔΣ-MDLL-PS)

and applies the reference to the input of the delay line. This process is repeated afterwards. It is assumed that the MUX does not have a delay in Figure 4.6.1. However, in practice, the MUX delay is comparable to the delay of the delay cells and, thus it may introduce a non-negligible timing error. To solve this problem, the value of the timing error has to be estimated at least one MUX delay before the reference rising edge arrives. Considering the fact that all delay cells have the same delay, the timing error can be estimated by comparing the phase of the signals at the middle of the master and slave loops (Figure 4.6.2).

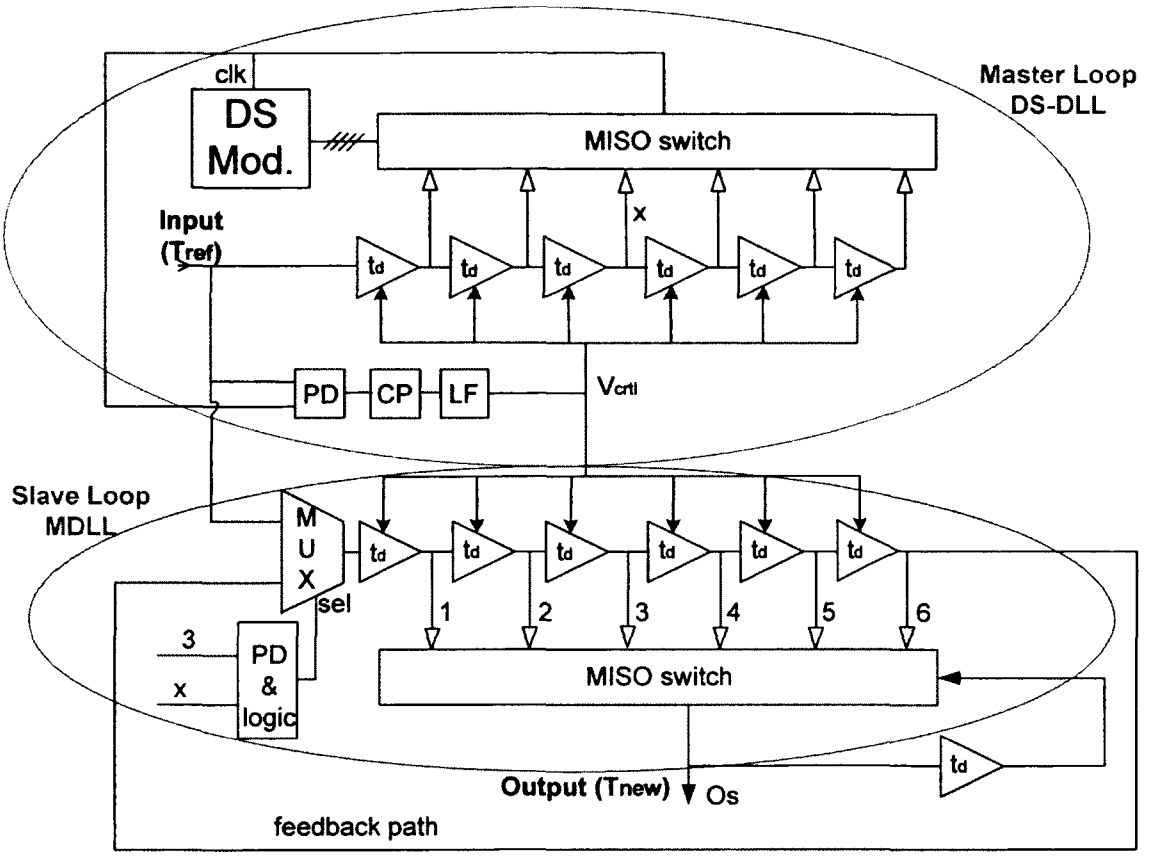


Figure 4.6.2: Modified (ΔΣ -MDLL-PS) for MUX delay

Note that the reference signal can apply every M cycle to MDLL with a timing error less than the threshold. The value of M versus β for a 6 stage MDLL and a threshold value of 0.01 UI (maximum timing error of 0.01 unit interval) is shown in Figure 4.6.4a. For further reduction of the accumulated jitter, the reference signal has to be more frequently applied to the Slave loop. To achieve that, the MUX delay is incorporated into the delay of the delay cell. Figure 4.6.3 shows the modified architecture where the delay of the MUX is incorporated in the total delay of the delay element. Note that in this new architecture, the minimum delay of a delay element is equal to $t_{d_{min}} + t_{d_{mux}}$ where $t_{d_{min}}$

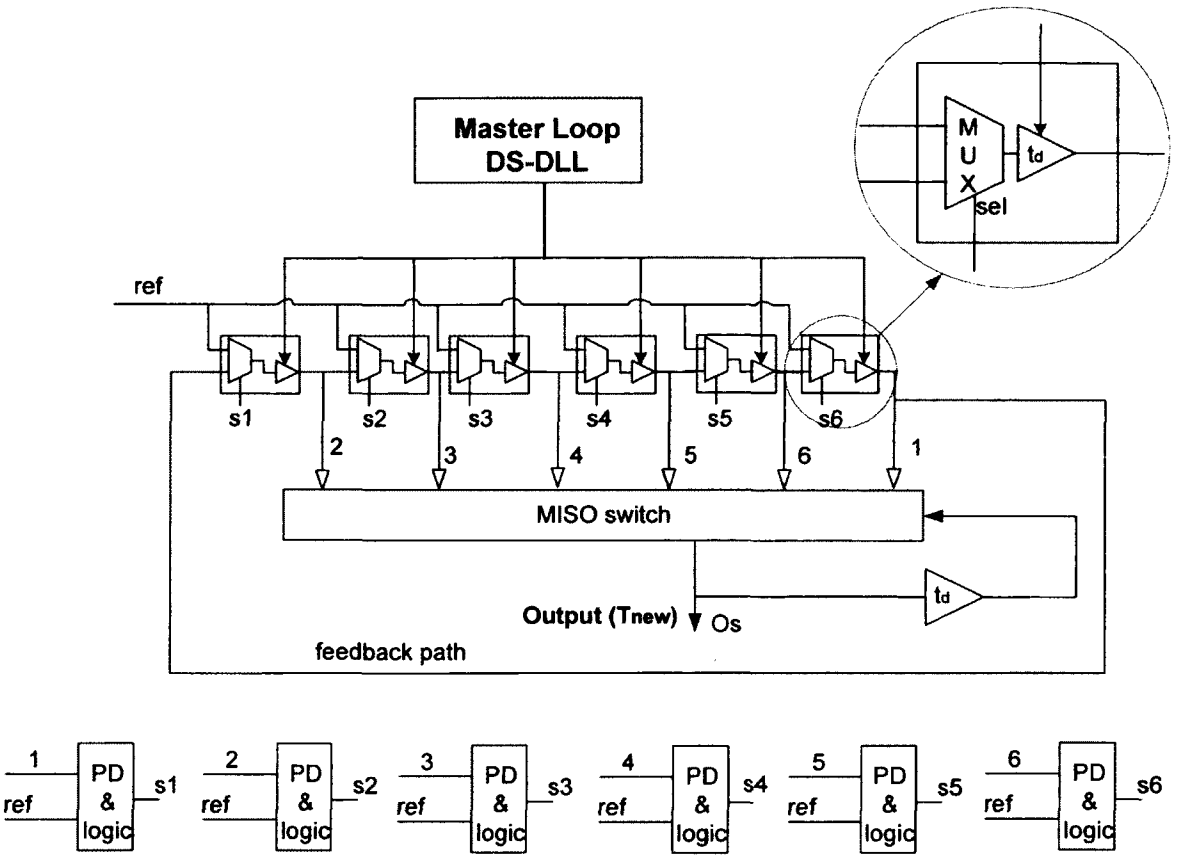
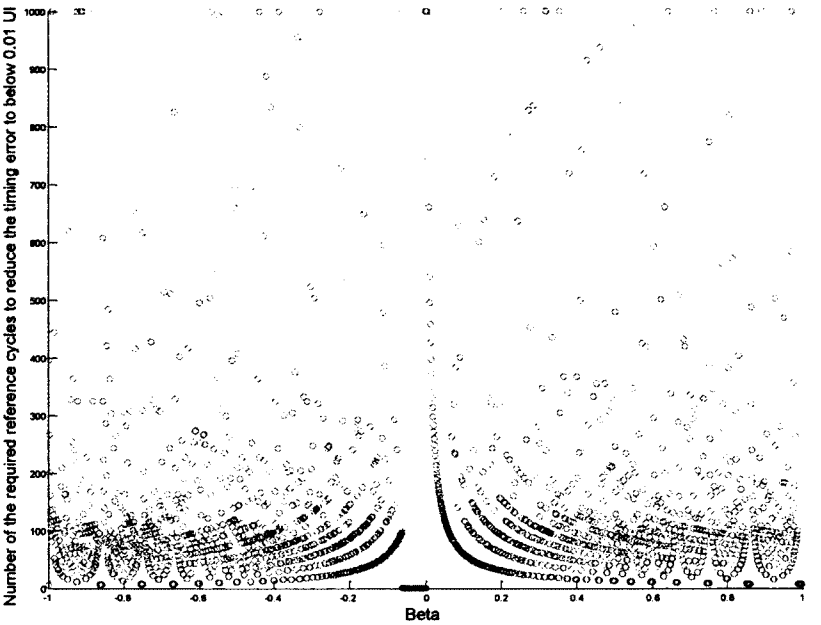
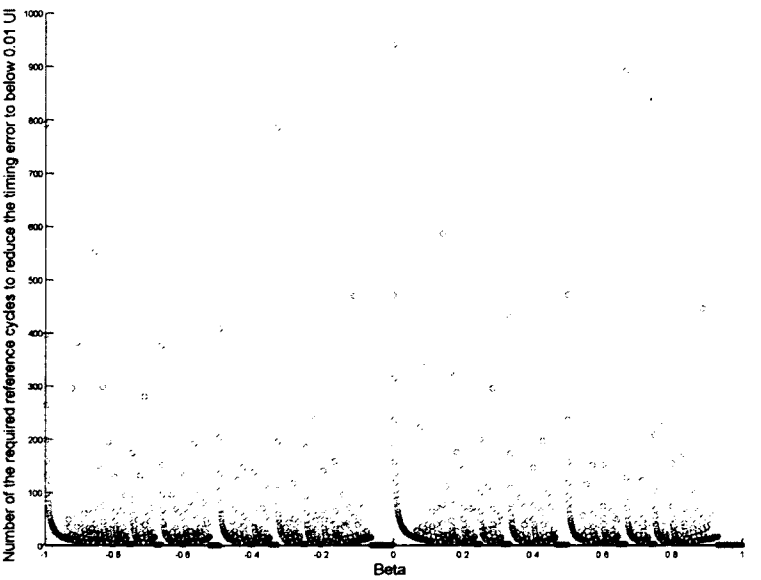


Figure 4.6.3: Modified (ΔΣ-MDLL-PS) with a MUX incorporated in the delay element. The reference signal can inserted in any point of the delay line (ring oscillator).

is the minimum delay of a delay cell and $t_{d_{mux}}$ is the MUX propagation delay. Therefore, it is more practical to work in high α - the fractional part of frequency multiplication factor (Figure 4.2). In addition, the reference signal can be applied to any point of the delay line (ring oscillator), if the corresponding timing error is less than the threshold. The value of M (the number of cycles that it takes for the timing error to acquire a value smaller than the threshold) versus β for a 6 stage MDLL and a threshold value of 0.01 UI is presented in Figure 4.6.4b.



(a)



(b)

Figure 4.6.4: Number of cycles required to apply the reference signal with a timing error less than 0.01UI a) For a MDLL with a simple delay cell (left) b) For a MDLL with a MUX incorporated in each delay element (right)

It can be observed that a smaller number of cycles is required to achieve a timing error smaller than a certain threshold (in this case 0.01 UI) when the reference is applied to the delay line.

4.6.2 Simple ΔΣ Fractional-N DLL-based Synthesizer

The simple (single loop) ΔΣ fractional-N DLL-based synthesizer (Figure 4.6.5) is similar to a ΔΣ-DLL-PS in which the MISO switch is replaced by an edge combiner. This architecture is simple and able to generate output frequency with a high resolution. However, there is an in-lock error result from the fractional-N multiplication factor. The fact that the delay line output is not aligned with the reference signal generates a strong spur at the reference frequency offset from the output frequency.

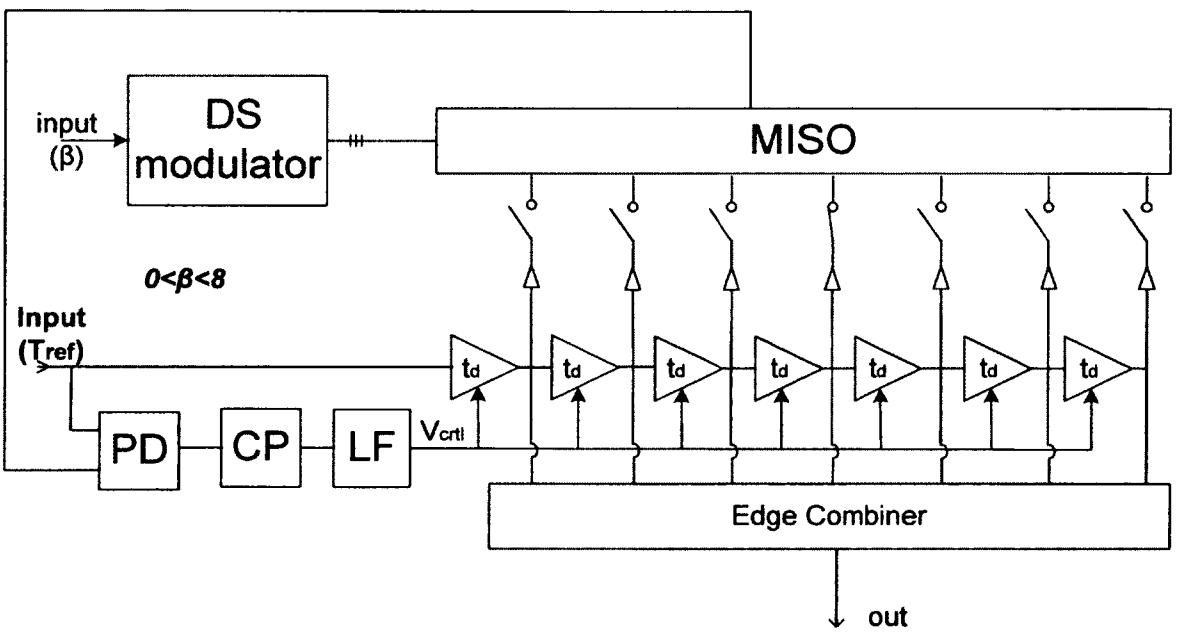


Figure 4.6.5: Single loop DLL-base fractional-N frequency synthesizer with no error compensation

In some applications with high reference frequency this spur may be way out of the interested band to be important or could be suppressed to acceptable level by a sharp narrow-band band-pass filter. For example, a super-heterodyne (or direct-conversion) transmitter with a narrow-band band-pass filter in front of the power amplifier (PA) (Figure 4.6.6) or a narrow-band low-power low-range transceiver for use in rural area may be a suitable application for this architecture.

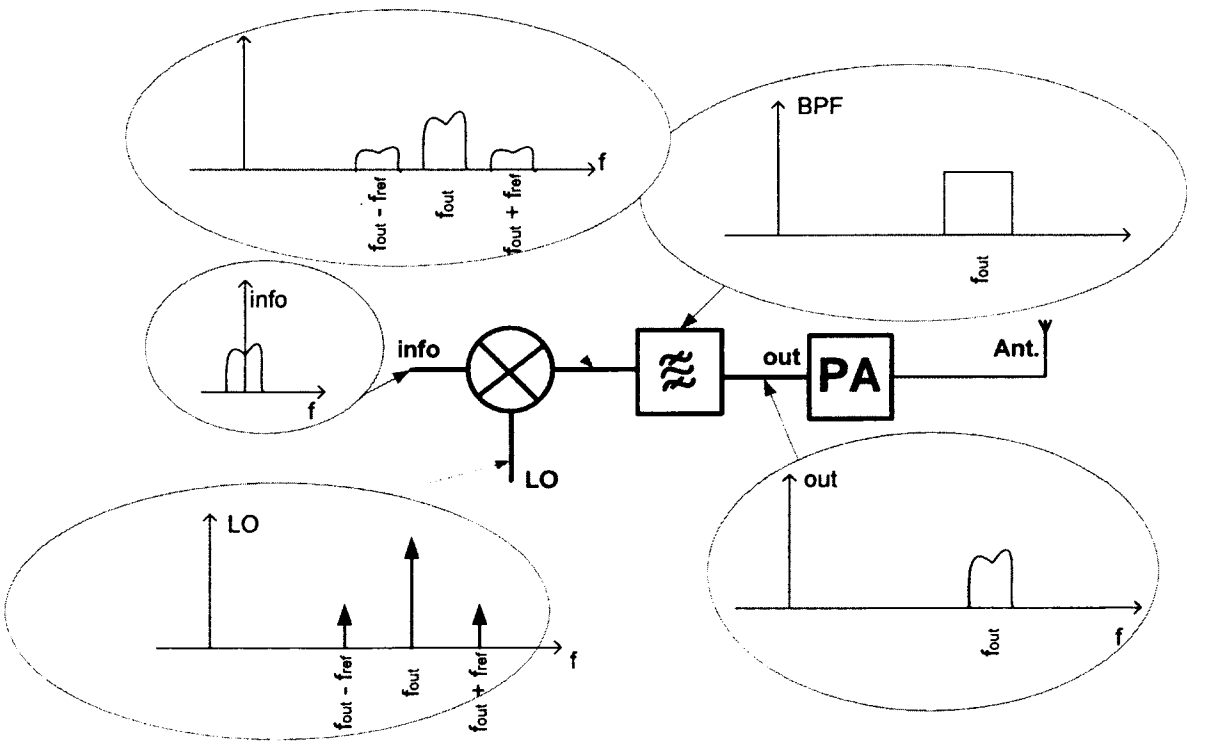


Figure 4.6.6: Transmitter; an example to show the possible application for single loop ΔΣ fractional-n DLL-based synthesizer

4.6.3 ΔΣ Fractional-N MDLL-based Synthesizer

ΔΣ fractional-N MDLL-based synthesizer (Figure 4.6.7) is similar to a ΔΣ -MDLL-PS in which the MISO switch is replaced by an edge combiner. Note that this architecture has major differences from the MDLL reported in [46]. This architecture does not have a divider; frequency multiplication is achieved by the means of an edge combiner. In start-up, the reference edge passes to the delay line through the MUX. Then, the MUX connects the delay line output to its input and converts it to a ring oscillator. Only when the phase difference between the feed back path signal and the reference signal is smaller than a threshold the MUX switches back and passes the reference to the delay line (the MDLL operation in this regard is similar to an injected lock PLL). The timing error of this architecture is small and occurs much less frequently. The main disadvantages of this architecture are that the close-in spurs results from the timing error and that the jitter accumulates when the MDLL operates in ring oscillator mode.

4.6.4 Multi-loop DLL-based frequency Synthesizer

Multi-loop architecture shown in Figure 4.4.12 is made of a ΔΣ-DLL (just like master loop described in section 4.5.1) and timing error correction scheme explained in chapter 4.4.2.3. Timing error correction is performed by a combination of loop-based period synthesis scheme (explained in section 3.4.2 and 4.4) and switched current control delay line. Timing error is detected with the aid of error detector loop when the total delay of delay line is smaller than the reference period. Loop forces all the delay cells inside period synthesis delay line to have delay of t_e; equivalent to the difference of the reference period and the total delay of master loop delay line.

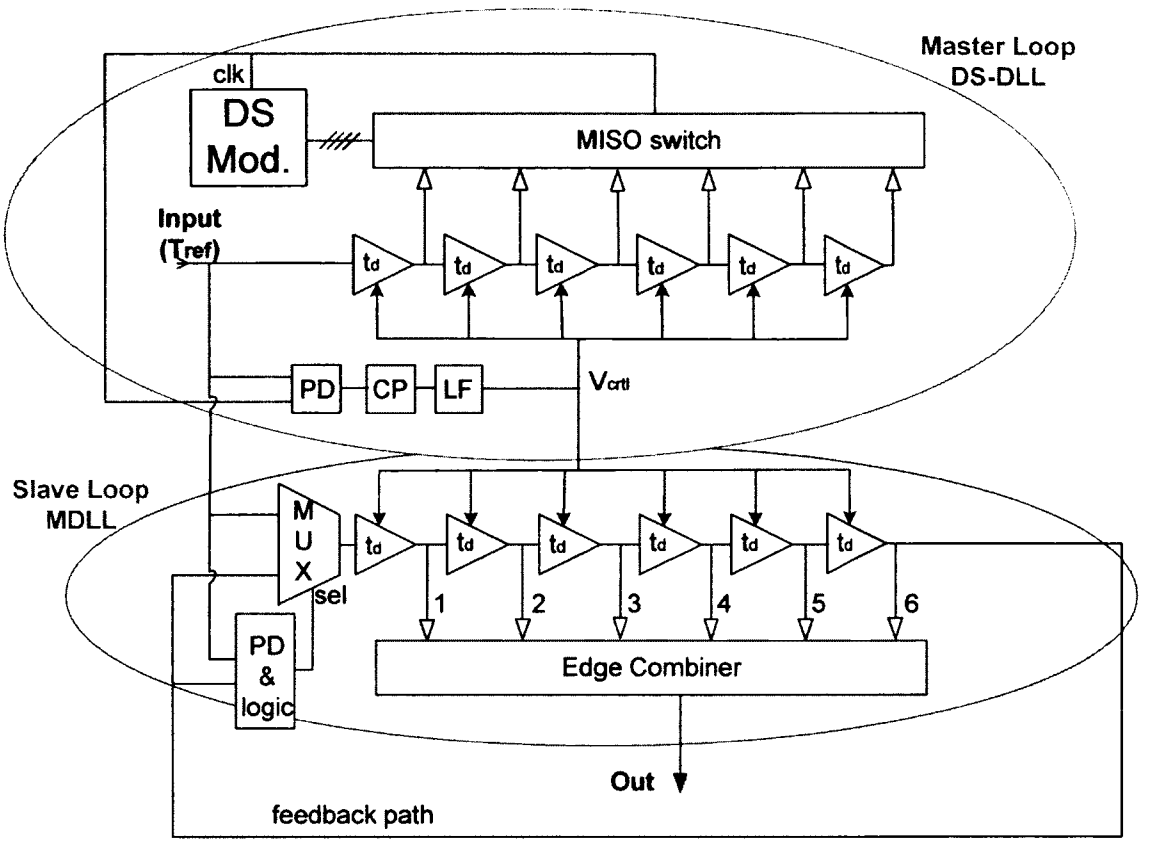


Figure 4.6.7: Single loop ΔΣ fractional-n MDLL-based synthesizer

Period synthesis can remove the timing error for every cycle except the time that it is switched to the other side of period synthesis delay line. Another error correction scheme using multiple loops is used to remove this timing error as it discussed in section 4.4.2.3. A slave delay line with the input coming from period synthesis and a Local edge combiner performs frequency multiplication exactly the same as main single loop proposed architecture (section 4.5.2).

4.6.5 Comparison of proposed architectures

Proposed architectures for DLL-based fractional-N frequency synthesizers will be compared in this chapter and advantages and associated challenges will be discussed. Four main DLL-based fractional-N frequency synthesizers architectures proposed in this chapter are:

1. Single loop architecture with PI based period synthesis error correction scheme (Implemented architecture introduced in section 4.5)
2. Multi loop architecture with loop-based period synthesis and switched delay error correction scheme (section 4.4.2.3 and summarized in 4.6.4)
3. Simple single loop ΔΣ-DLL without no error correction (section 4.6.2)
4. ΔΣ fractional-N MDLL-based synthesizer (section 4.6.3)

Table 4.6.1 shows the summary of features and issues for all the architecture. For simplicity these architectures are called by their order in the above list in the rest of this section. For example, architecture one refers to implemented single loop topology with PI based period synthesis and architecture two refers to multi-loop architecture.

Architecture 3 is the simplest one which doesn't have any timing error correction and only includes a ΔΣ-DLL. It can achieve desired fractional-N multiplication factor with a spurious tone f_{ref} away from desired output frequency. Synthesizer can be built very compact and low power with low close-in phase noise. By using high frequency reference the spurious tone can be far from communication channel. In remote area, or short range application, spurious tone may fall in part of spectrum that is not used.

Table 4.6.1: Comparison features of proposed DLL-based fractional-N frequency synthesizer architectures

Architecture Feature	Single Loop	Multi-loop	Simple ΔΣ-DLL	ΔΣ-MDLL
Section	4.5	4.4.2.3 and 4.6.4	4.6.2	4.6.3
Fractional delay generation method	ΔΣ-DLL	ΔΣ-DLL	ΔΣ-DLL	ΔΣ-DLL
Timing error correction method	Period Synthesis	Period Synthesis	None	Ring style feed back + edge injection
Period Synthesis method	PI based	Loop based + delay switching	N/A	N/A
Theoretical frequency range	$(N-1)f_{ref} < f_{out} < (N+1)f_{ref}$ (*)	$(N-1)f_{ref} < f_{out} < N.f_{ref}$	$(N-0.12)f_{ref} < f_{out} < (N+0.12)f_{ref}$	$(N-1)f_{ref} < f_{out} < (N+1)f_{ref}$
Best perform range of fractional multiplication	$0<\alpha<0.25$	$0.25<\alpha<0.75$	$0.1<\alpha<0.1$	when $\frac{K}{\alpha}$ is small integer (**)
Major jitter contributor	Phase interpolator	Period Synthesis switch	In-lock-error	Accumulated jitter in ring mode
Complexity	Medium	High	Low	Medium
Close-in phase noise	Low	Low	Low	high
Spurious tone level	Low (***)	Low (***)	High (***)	Very low
Spurious tone frequency	f_{ref} away from carrier	f_{ref} /N away from carrier (****)	f_{ref} away from carrier	N/A
Frequency resolution limiter	PI stepsize (*****)	ΔΣ modulator	ΔΣ modulator	ΔΣ modulator
Draw backs	PI INL contribution to jitter	High Complexity, delay switching noise	High level of spurs	Higher close-in phase noise level
Advantages	Area and power efficient, good immunity to PSS, suitable for digital implementation, low close-in phase noise	Low close-in phase noise, low jitter, high frequency resolution	Low complexity, low close-in phase noise, area and power efficient	Low total jitter, area and power efficient, wide frequency range coverage with acceptable complexity

* Consider both forward and backward stepping
** K is an arbitrary integer number
*** M dependent (M: multiplication factor)
**** N is number of delay cell in the period synthesis DLL
***** PI is not limited factor if $jitter=0.5\times PIstepsize$ can be tolerated

Architectures 1 and 2 use period synthesis technique to remove timing error and suppress the spurious tone. Loop-based period synthesis scheme in architecture 2 has higher complexity than PI-based period synthesis of architecture 1.

Architecture 4 use a combination of ΔΣ-DLL and MDLL to avoid timing error and eliminate spurious tone for fractional-N multiplication factors. Reference inserted to MDLL less often only when it is aligned to feed back. Therefore architecture 4 has higher close-in phase noise compare to the other architectures.

Architecture one is chosen for implementation as it is the lowest complexity single loop topology with timing error correction which offers both low close-in phase noise and low spurious tone level. Design considerations for DLL-based fractional-N frequency synthesizer with a focus on architecture one will present in next chapter. Circuit topologies of all the building blocks are investigated and corresponding implementation issues are presented.

Chapter 5 Design Consideration

In this chapter the practical implementation of a fractional-N DLL-based frequency synthesizer is discussed in detail. All aspects of the design from system level analysis to circuit implementation and layout techniques will be explained as well as the corresponding trade-offs and the technical decisions. Chapter begins by explaining the special consideration for the fractional-N DLL loop and showing a detailed analysis of such a feedback system. System level requirements for a practical design are explained in the second section. Sections three through six are dedicated to circuit level implementation and layout consideration for different building blocks of the fractional-N DLL frequency synthesizer. In section seven the whole frequency synthesizer and interaction between the blocks will be presented.

5.1 Loop analysis for Fractional-N DLL

A general schematic of a fractional-N DLL is shown in Figure 4.5.2. Many variation of the loop can be built as each component of the loop can adopt different topologies. Never the less, each component has its related parameters, and DLL can be analyzed as a feedback system. A delay line can be expressed by its delay gain as $G_{dl}=\Delta t_d/\Delta V_{cp}$. Charge pump gain is expressed by its current I_{cp}. The relationship between the charge pump current and its voltage output is depended on the loop filter structure. Loop filter order, bandwidth and the value of integration capacitor all play a significant role in loop

dynamics. The order of the $\Delta\sum$ modulator and its topology is another big factor in the dynamics of the fractional-N DLL. Different topologies of $\Delta\sum$ modulators are discussed in Chapter 3. Switching induced noise ($\Delta\sum$ modulator quantization noise), results from switching different clock phases to the phase detector based on the output of a $\Delta\sum$ modulator, is a major performance measure for fractional-N DLL. In general the following multi-dimensional trade-offs have been found in a fractional-N DLL. Switching-induced jitter is reduced with lower loop bandwidth at the expense of locking time increase. The ratio of the charge pump current I_{cp} to the value of the total integration capacitor in the loop filter is one of the factors affecting the switching induced jitter. A lower charge pump current reduces the switching induced noise and loop gain as well as increasing the impact of capacitor leakage and charge pump current mismatch. The same loop bandwidth with a larger capacitor shows less switching induced jitter and it is recommended as far as capacitor area is within the budget and capacitor leakage is in control. A higher order loop filter helps reduce the switching induced jitter in a cost of raising stability issues. A DLL with a first order filter is naturally stable. Second order loops require careful stability analysis, and higher order loops increase the risk of instability due to circuit variations. More switching induced jitter is generated when the fractional multiplication factor is close to an integer. Using higher order $\Delta\sum$ modulators helps reduce the jitter but adds complexity. Feedback style $\Delta\sum$ modulators induce less switching noise compared to the same order MASH topology counter parts.

5.2 System level requirements for Fractional-N DLL

For a practical implementation of this frequency synthesizer one needs to target an output frequency range. Frequency range of 4 GHz to 5 GHz has been chosen to cover IO standards like GDDR5, PCIe Gen2 directly and IEEE 802.ll wireless standard, PCIe Gen 1 and DDR4 by means of a divider. The period of a 5 GHz signal is 200ps. In a DLL-based frequency synthesizer with Local edge combiner (explained in Chapter 2), the delay of the buffer delay cell is equal to the period of the edge combiner output. Synthesizer multiplication factor would be equal to the number of the buffer delay cells or half of the number of the inverter delay cells. It is obvious that the number of the inverter delay cells should always be an even number. An inverter delay cell with a typical delay of 100ps or a buffer delay cell with the delay of 200ps required for 5 GHz target frequency. Using a 1000MHz reference clock, the DLL-based synthesizer must provide the multiplication factor around 5 and thus requires a delay line with 5+1=6 buffer delay cells or 10+2=12 inverter delay cells. An extra buffer delay cell is used by $\Delta\Sigma$ modulator to make fractional delay.

The channel spacing required for IEEE 802.11b wireless standards is 5MHz which translates to a change of the multiplication factor by 0.2%. This fine channel spacing might not be necessary for many high speed IO applications. In most of the backplane or chip to chip IO blocks, generating different clock frequencies which might not be the integer multiplies of the reference frequency is desirable. The requirement of different clocks might be a system level necessity to comply with different protocols. The ability of changing the clock gives flexibility to operate in a lower speed either to be backward compatible to older communication standards, communicate with older technology parts,

saving power when less aggregating bandwidth is required, or opening the eye by moving away signal frequency contents from notch of a poor communication channel or bad part of a curve in a super skewed process corner. It also can help to reduce crosstalk when two different IOs work side by side and enables reuse of IO blocks in many different products with different protocols which are offered in the same fabrication technology.

5.3 Delay Line Structure and Tradeoffs

A delay line is made of series of delay elements/cells. The delay of each delay cell is in general controlled by a signal. The control signal can be an analog voltage, a current or even a digital word. The delay of a delay element regardless of its topology depends on the CMOS devices speed (F_t), the drain saturation current (I_{dsat}) and the associated diffusion and parasitic capacitances. Thus the delay element must be designed for the target operation frequency at any particular fabrication technology. Process, voltage and temperature (PVT) variation also impact the delay severely. The delay element should have a sufficient delay range to cover the target frequency range and PVT variations. The superposition of the target frequency range and PVT variations at times may force the designer to use multiple delay control mechanisms to extend the delay range of a cell. Power supply induced jitter is among the primary factors which impact the choice of a delay line topology. Popular jitter reduction techniques include use of voltage regulator for single ended delay cell topologies or the use of differential delay cell topologies. Differential topologies show superior jitter performance compared to their single ended counter parts. Usually the users of generated clock (transmitters and samplers in a typical high speed serial I/O) need single ended CMOS level (rail to rail) clock and thus a power

hungry and jitter prone differential to single ended converter circuit is required to convert low swing differential clock to rail to rail clock with reasonably sharp transitions. This differential to single ended converter some times can change the power/performance/complexity balance in favor of single ended topologies. Power supply sensitivity can be presented by alpha number defined as the ratio of the percentage of the delay change to the percentage of the power supply change.

$$\text{Alpha}=((d1-d2)/d2)/((V1-V2)/V2) \qquad (5.1)$$

Power consumption and sensitivity to the local variation are among other factors that need to be considered to choose a delay line topology for fractional-N DLL. Delay cells of a delay line may have different delays due to the local variation of CMOS transistors [44]. This delay difference will contribute directly to the spur level of the generated clock at the edge combiner output. Local variation can be reduced by using bigger feature size and larger transistor which would result to a higher power consumption for certain target output frequency. These factors are discussed in following sub-sections and supporting simulation results are presented for delay line topologies.

5.3.1 Voltage control single ended topology

The simplest delay cell is an inverter which its delay is controlled by its supply voltage. Figure 5.3.1 shows the single ended delay cell and a typical delay line made of it. A higher supply voltage gives a larger overdrive to the gate of both pull-up and pull-down devices and thus increases their currents. As a result, the load capacitance (parasitic and gate capacitors) charges and discharges faster and therefore delay decreases. As the

overdrive voltage has a nonlinear relation with the current, the delay of this delay cells doesn't change linearly with control voltage (Figure 5.3.2).

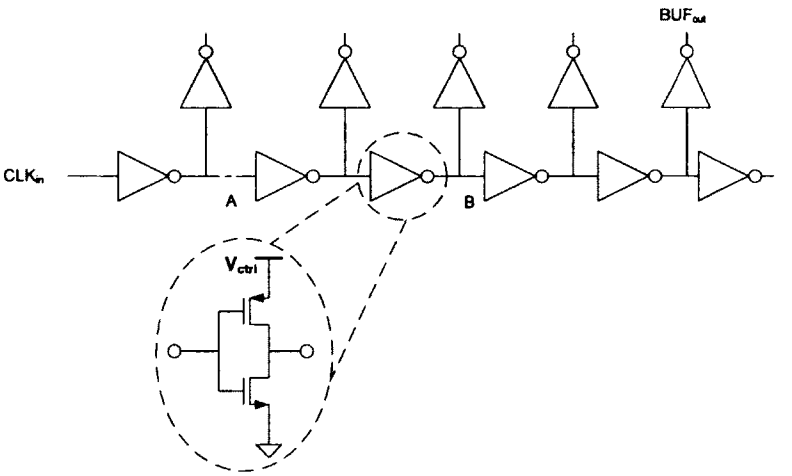


Figure 5.3.1: Typical delay line made from simplest voltage control delay element; an inverter. Delay of an inverter can be controlled by supply voltage

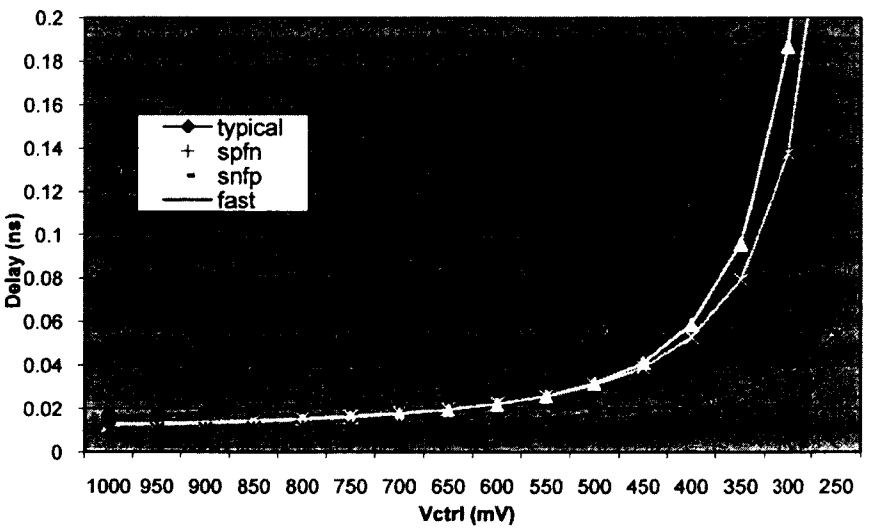


Figure 5.3.2: Delay of a buffer delay element made of two inverter versus the control supply voltage. Cross skew corners of spfn (slow pmos and fast nmos) and snfp (slow NMOS and fast PMOS) are also presented.

Also as it can be seen delays versus voltage curves are greatly different in different process corners. This is the case for all of the sub 90nm CMOS technologies and would be more pronounced as the feature size gets smaller. One of the main challenges for using inverter delay cell in current sub 90nm technologies is to achieve the target delay in all process corners. This usually is done by adding another delay control mechanism, like switching capacitors, to compensate for process variation. This type of the delay line will be described in section 5.3.3. The power supply sensitivity (PSS) presented by alpha number (Figure 5.3.3) is a function of the control signal. PSS is higher in lower control voltages and in slow process corner where the delay line gain is extremely high. Using a regulator for this delay line would decrease the overall PSS.

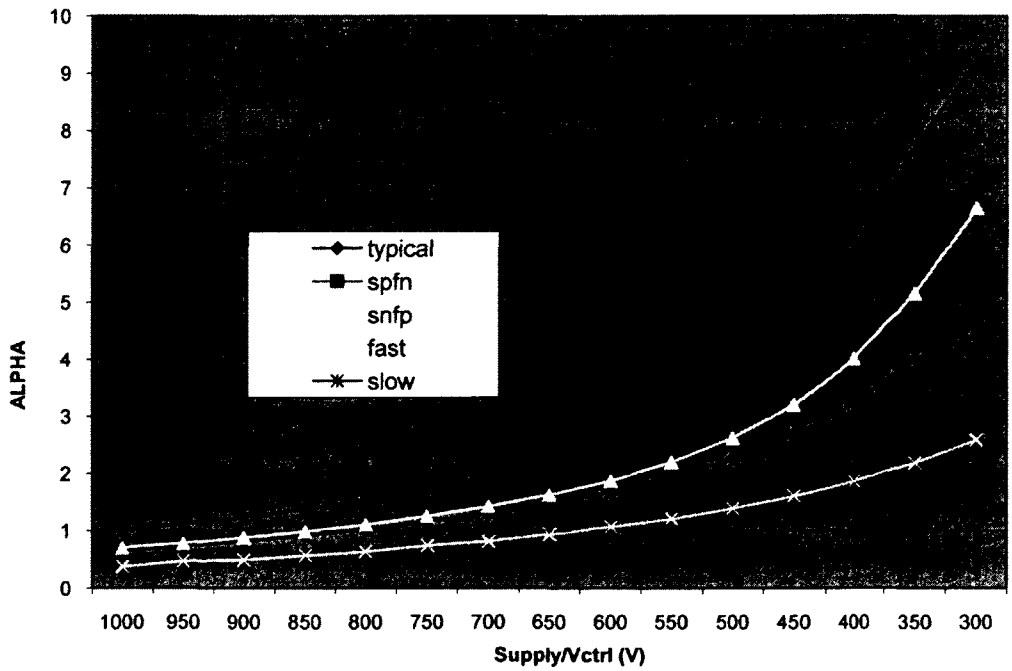


Figure 5.3.3: Power supply sensitivity of an inverter base VCDL based on ALPHA number. ALPHA has been defined in equation 5.1.

5.3.2 Current control single ended topology

To have a more linear and controlled delay the pull-up and pull-down current can be controlled directly as shown in Figure 5.3.4. This topology is different from a double rail regulated delay line [48] in which both supply and ground rail is regulated. One advantage of current control delay line is its rail to rail output which makes a simple connection to an inverter buffer possible. On the other hand the delay variation with the process corner is much less. Figure 5.3.5 shows the delay of a typical current controlled delay line (CCDL) with control current. If eventually a voltage signal is supposed to control the delay of the delay cell, then the nonlinearity is just pushed to the voltage to current converter (Figure 5.3.6). CCDL topology needs more voltage headroom due to additional stack transistors. This might be considered a draw back for the low supply voltages of today's submicron technologies.

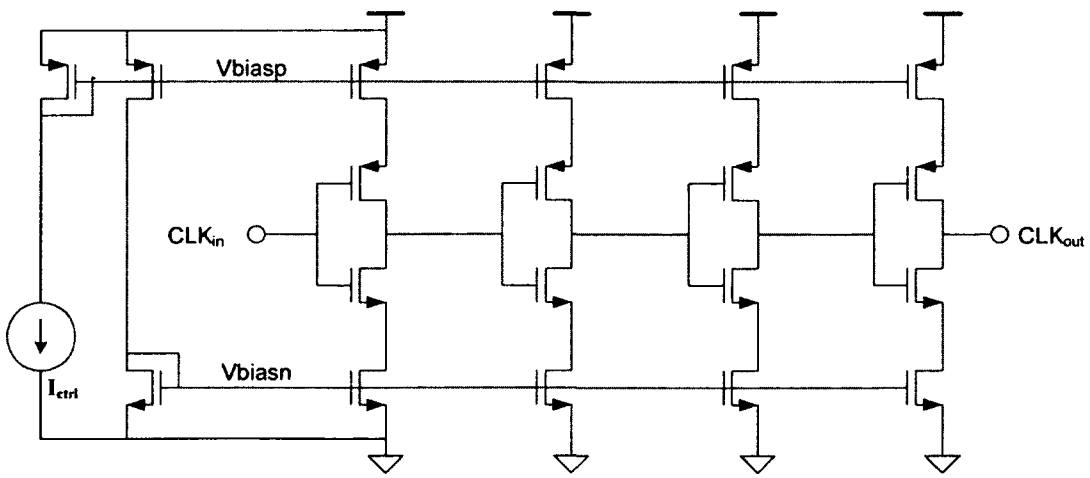


Figure 5.3.4 Four stages Current Control Delay line (CCDL)

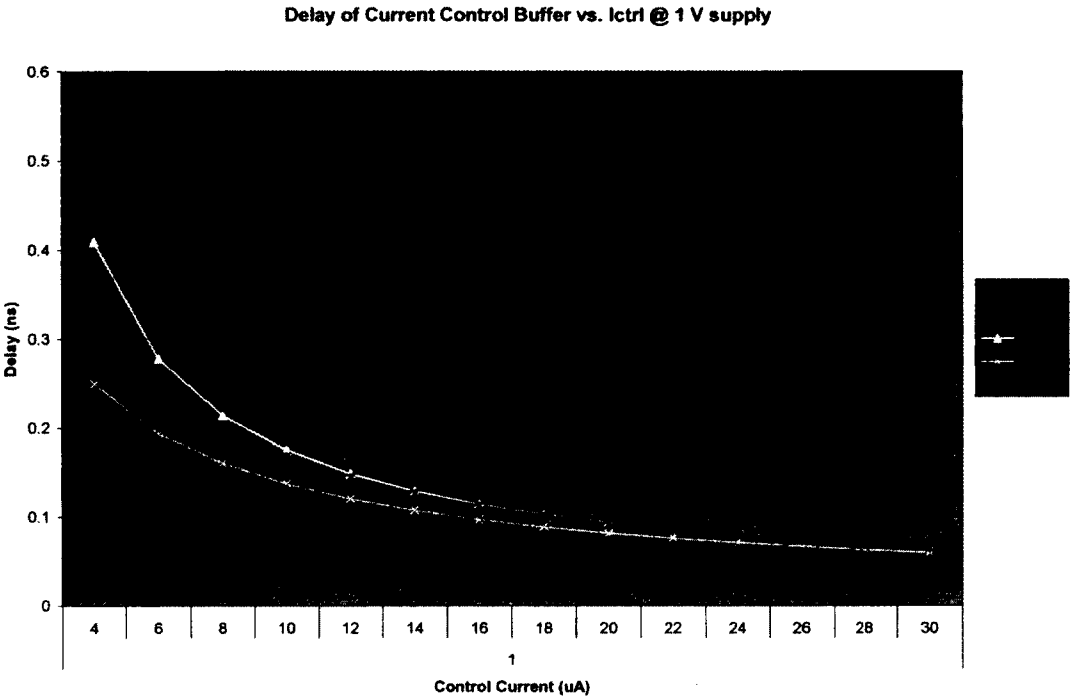


Figure 5.3.5: Delay of Current Control Buffer Cell (CCBC) versus its control current (Ictrl) at 1V supply voltage; there is less process variation at high current

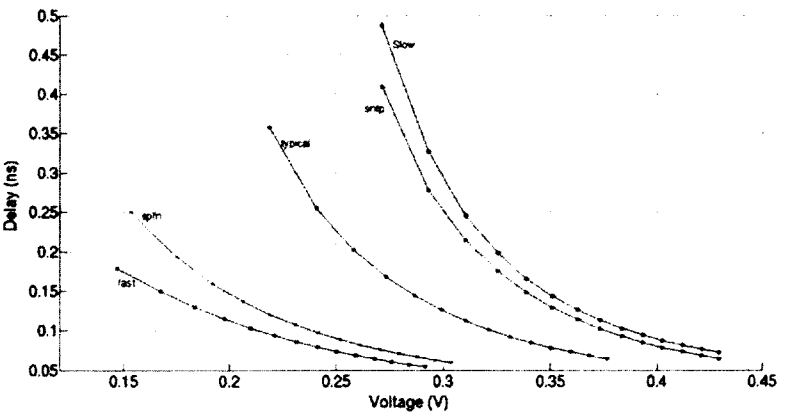


Figure 5.3.6: Delay of Current Control Buffer Cell (CCBC) versus equivalent bias voltage again at 1V supply voltage

Power supply sensitivity (PSS) for current control delay line (CCDL) as equivalent ALPHA number is shown in Figures 5.3.7 and 5.3.8. In general CCDL has much better PSS compare to inverter base VCDL as it has smaller ALPHA number for all the range. The interesting characteristic of the CCDL is that for range of interested delay it has negative alpha number. It means the delay of the delay element reduces/increase when supply voltage increase/decrease. This characteristic can be used to made delay cells with better PSS i.e. smaller absolute value of ALPHA can be achieved by interleaving current control delay cells with inverters as it is shown in Figure 5.3.9. Delay and PSS for such this delay line in terms of ALPHA number is shown in Figures 5.3.10 and 5.3.11. As it can be seen very good PSS can be achieved by a careful design.

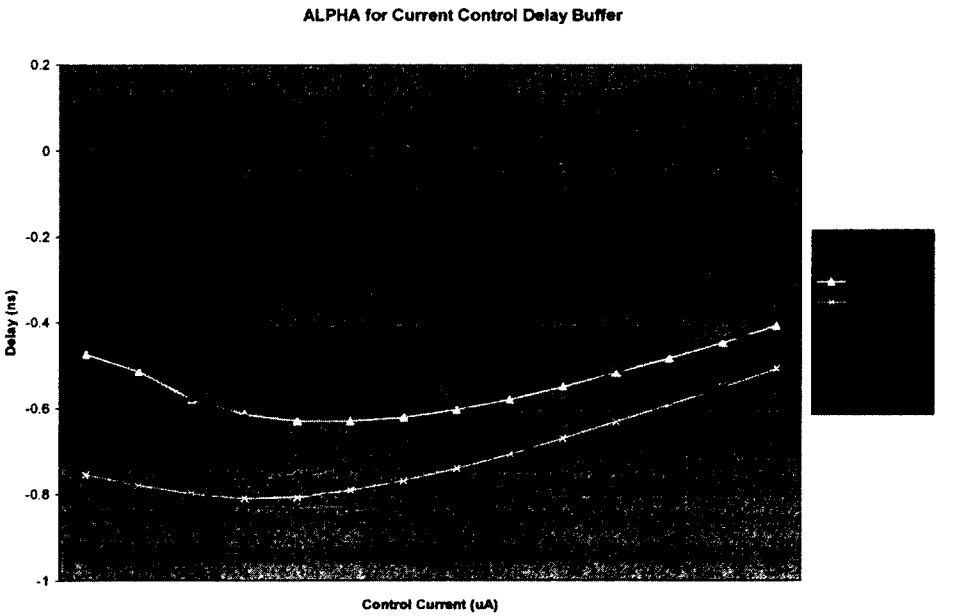


Figure 5.3.7: ALPHA of CCDL versus equivalent its control current for typical and cross skew process corners at different supply voltages

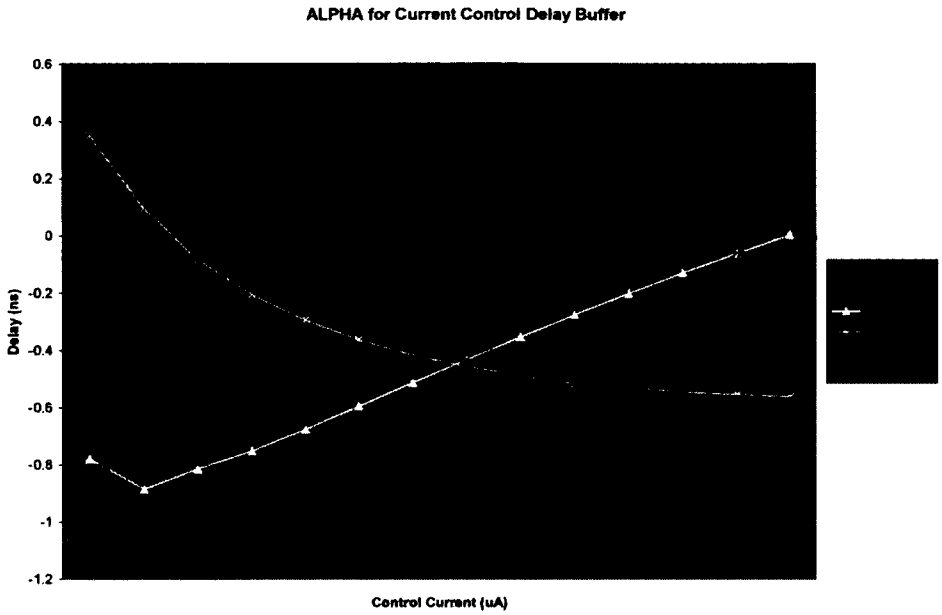


Figure 5.3.8: ALPHA of CCDL versus equivalent its control current for fast and slow process corners at different supply voltages

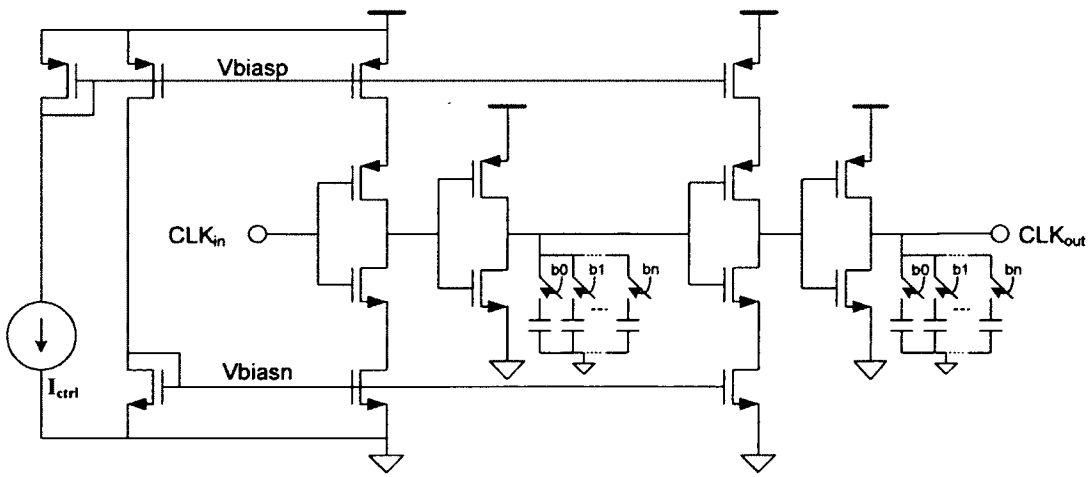


Figure 5.3.9: CCDL with inverter interleave (CCDLII)

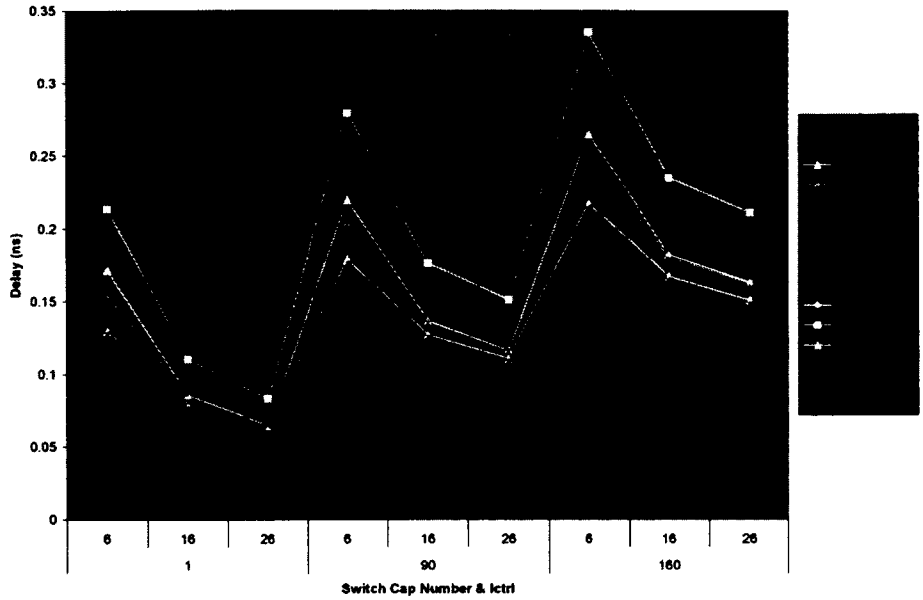


Figure 5.3.10: Delay of Current Control with Inverter Interleaved (CCDLII)

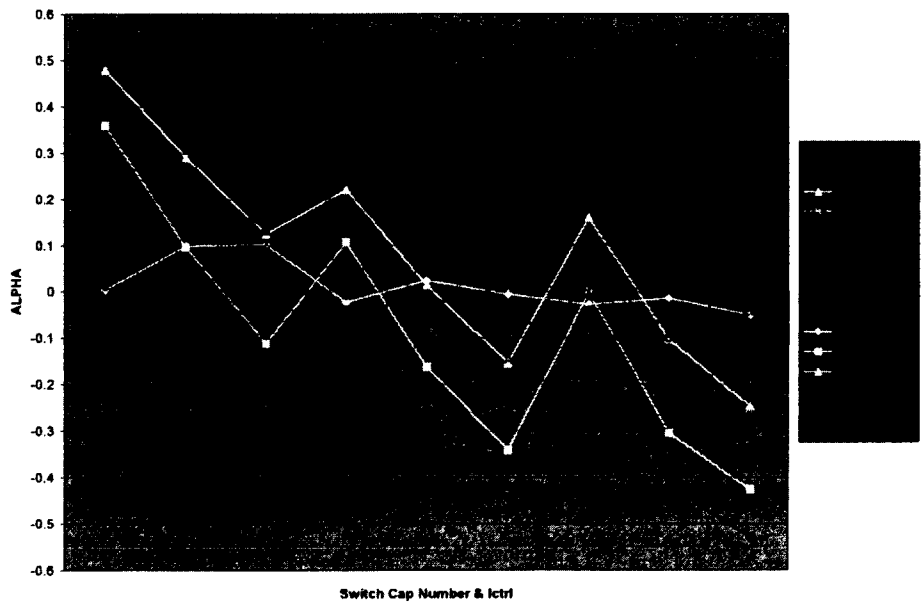


Figure 5.3.11: ALPHA for Current Control with Inverter Interleaved (CCDLII)

5.3.3 Switch capacitor single ended topology

Another way of altering the delay of a delay cell is changing its load capacitance as shown in Figure 5.3.12. Switching in or out the capacitors would change the load capacitance and thus changing the delay of the delay element. The control signal for this delay element is a digital word. Using thermometer code with equal size capacitors can ensure a monotonic delay change even with the local variations of the on chip components. These types of delay cells don't have enough resolution for the fractional-N architecture. But a combination of course switched capacitors control and voltage or current control seems to be a good candidate for fractional-N DLL application (Figures 5.3.9 and 5.3.12).

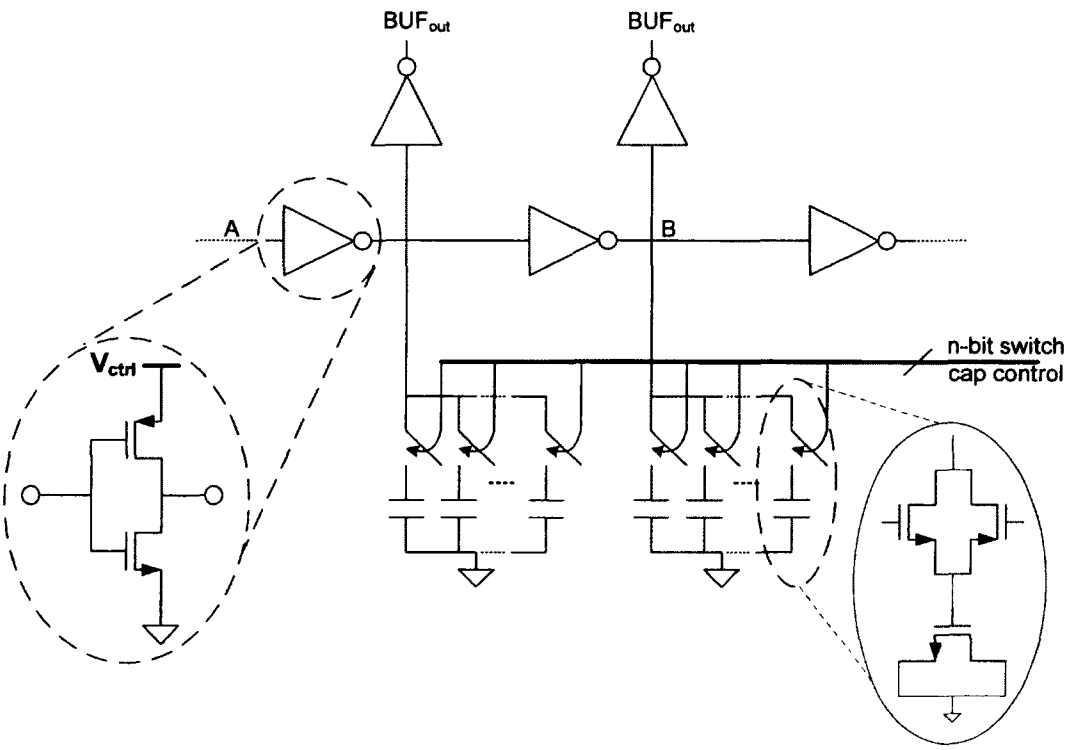


Figure 5.3.12: Switched Capacitors single ended delay element; delay is controlled by switching the caps using analog pass gates; caps can be device capacitors as it has shown in the figure, or metal finger cap or any other low leakage available capacitor

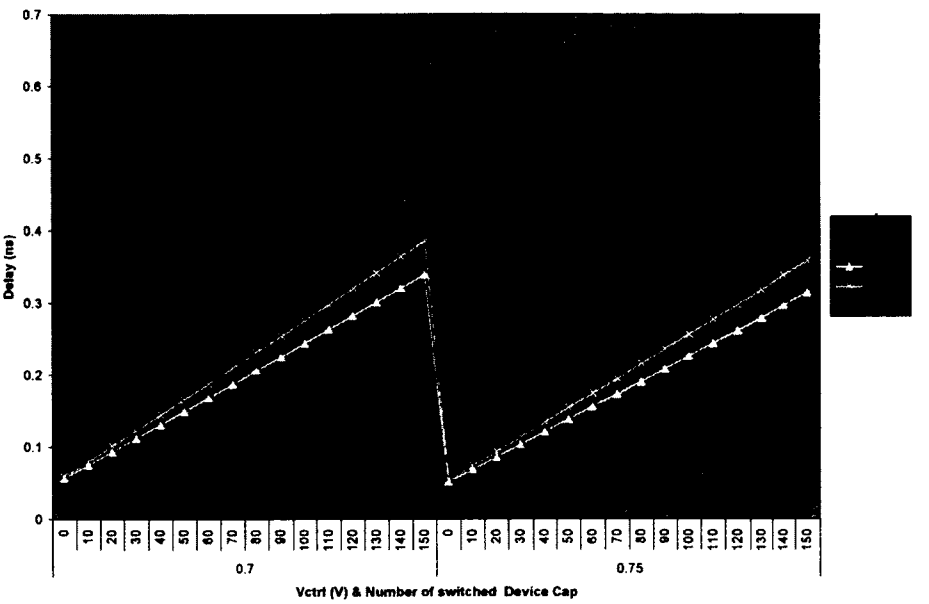


Figure 5.3.13: Delay of buffered delay versus number of the switched device capacitors for two control voltages; any target delay between 100ps and 200ps can be achieved for all process corners.

A course switch capacitor delay control is usually used to compensate for process variations and to reduce the range of control voltage or current. The limited range of control voltage has a great impact on the charge pump design and relaxes the challenge of keeping its switching current sources in saturation. A fine switch capacitor delay control has smaller range and can be used to compensate for local and random variations or to lock the loop in a digital loop approach. The switched capacitor delay control offers good linearity. A combination of coarse switch capacitor (switch in a big cap) and fine switch capacitor (switch in smallest possible cap) can provide enough range and good resolution and is a good candidate for digital loops. ALPHA for the switched capacitor delay line is shown in Figure 5.3.3.c. PSS for this delay line is in the same order of the inverter base VCDL with the same V_{ctrl}.

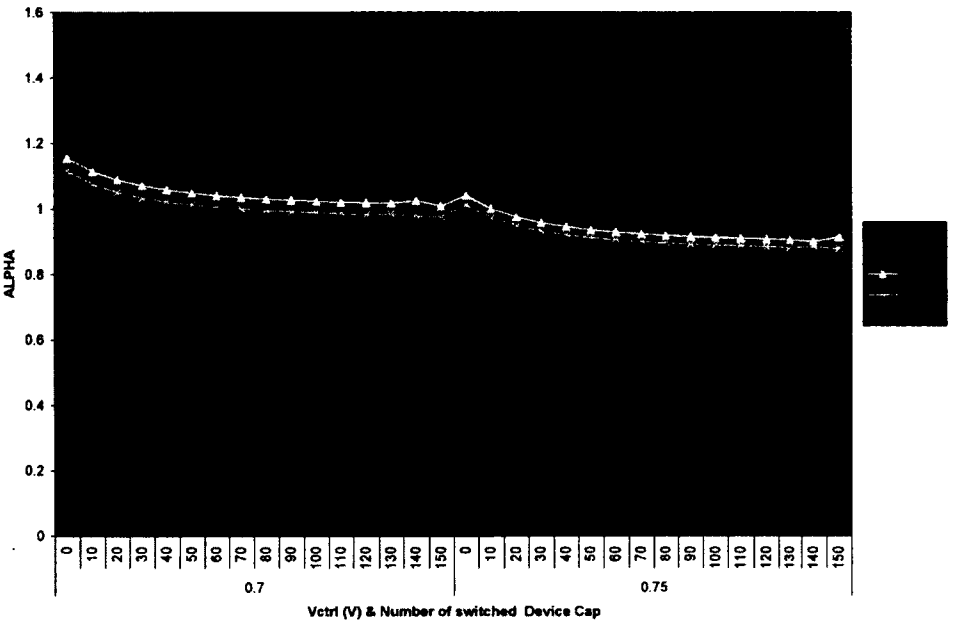


Figure 5.3.14: ALPHA for switched capacitors single ended delay line in Vctrl of interest

5.3.4 Differential topology

One important characteristic of the delay cells which has huge contribution to overall jitter of the synthesizer is their sensitivity to the supply noise. In this regard a differential delay cell shows a better performance due to its common mode voltage rejection. Three common differential delay cell topologies are shown in Figure 5.3.15. The delay of differential delay cells are mainly controlled by changing its current. The delay change of a typical differential delay cell (incorporated in delay line shown in Figure 5.3.16) versus its bias current is shown in Figure 5.3.17. Differential delay cell is designed in a way that current completely switches to the other leg in each half of the period. This would provide high swing and considerably fast transitions which more suitable for the delay line applications where the delay of a cell is much less than the reference clock period. Delay can also alternatively changed by switching capacitors or changing the active load

bias voltage. Again in deep submicron technology, process variation has huge impact on the delay of a differential delay cell and designers can use these alternative delay controls to compensate for the process corners and/or devices random variations. As it can be observed the variation of the delay is much smaller compare to single ended topologies. For a better comparison the ALPHA number for different corners are given in Figure 5.3.18.

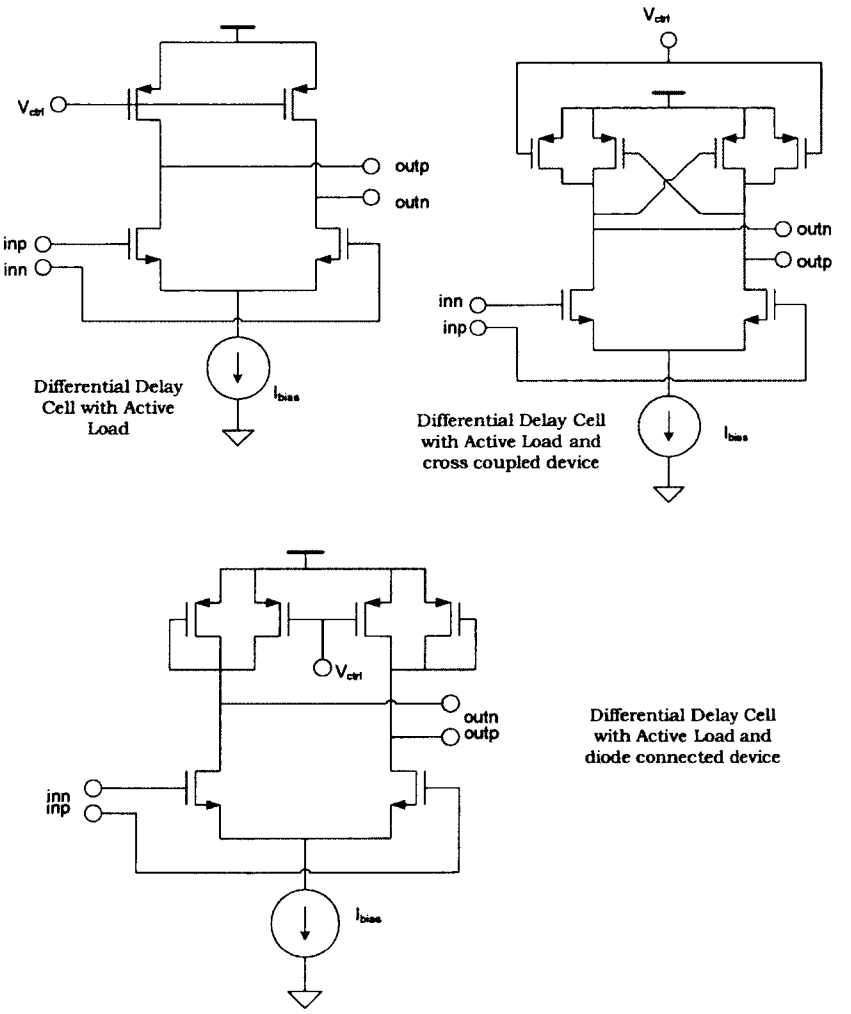


Figure 5.3.15: Three topologies for differential delay cells

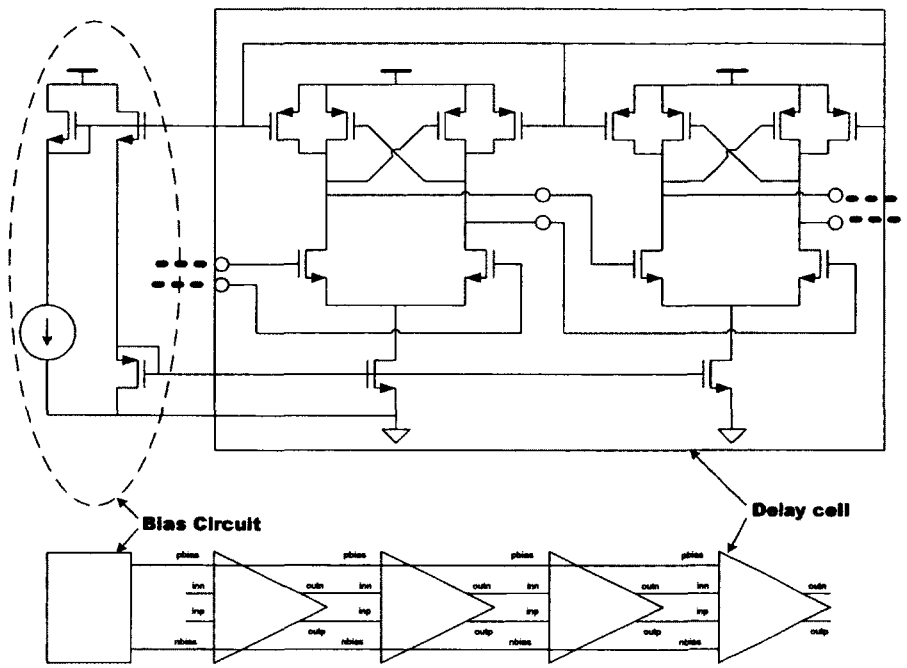


Figure 5.3.16: A typical buffer differential delay cell in the middle of delay line

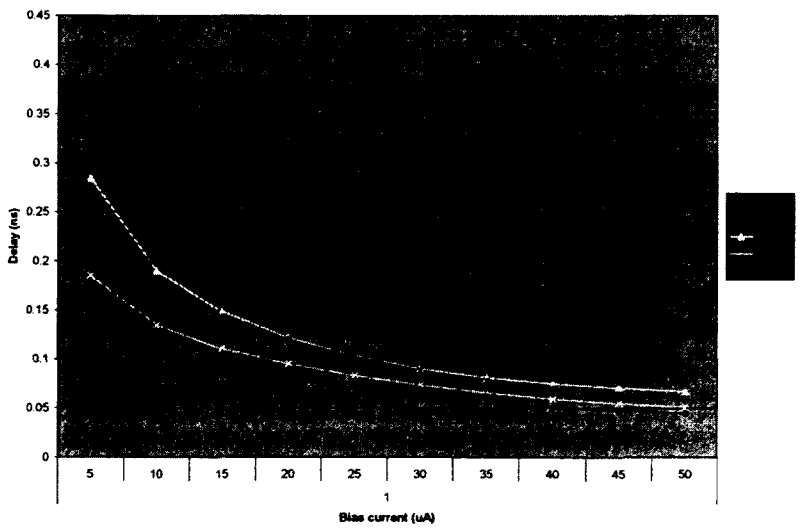


Figure 5.3.17: Delay of buffer differential delay cell in figure 5.3.16 versus its bias current; bias current is controlled the delay

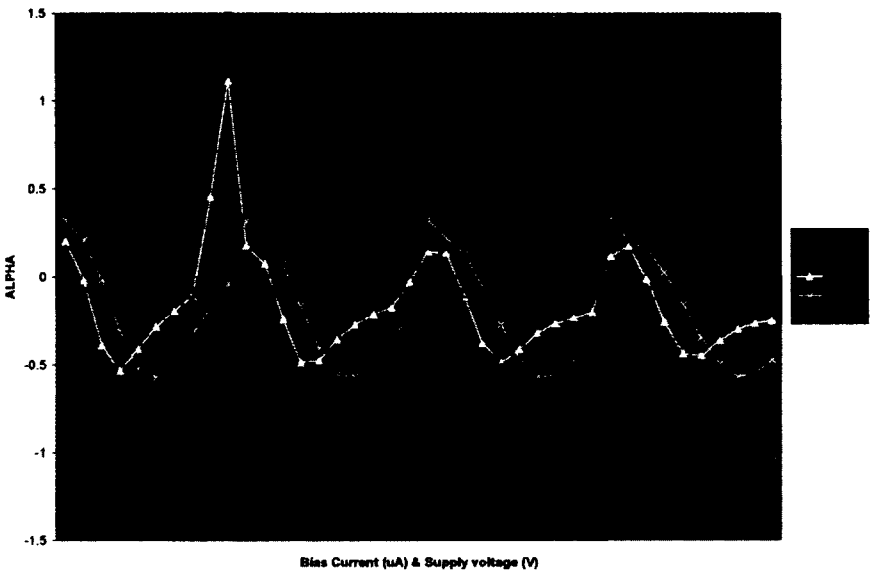


Figure 5.3.18: ALPHA for differential delay cell of figure 5.3.16

5.3.5 Impact of local variation on delay mismatch

Variation of the device characteristics can be categorized in two major sections, die to die variation and intra-die variation. Die to die is the process skew during fabrication which affects all the devices on one die to be slower or faster than their typical characteristics. Maximum variation in each direction is modeled as a process corner and used for the PVT (Process, Voltage, temperature) check simulations. Some times cross skew happens i.e. PMOS and NMOS transistors skew in two different directions. In this case die may have fast PMOS and slow NMOS devices or vise versa. The delay line simulation results have been presented for typical, slow, fast and cross skew corners in the previous sections.

The intra-die variation is in fact the variation of device characteristics on the same die which can be divided in two categories: local variation and mismatch (random variation). The local variation is the geometry dependence shift of device characteristics caused by fabrication process skew in an arbitrary direction. There are several layout techniques to alleviate the local variations and have better match devices; including keeping devices as close as possible, provide similar surrounding structures, and placing them in a common centriod configuration [49,50]. The random variation of device characteristics is the other source of mismatch for two identically drawn devices. Based on author personal experience with many sub 0.1 micron CMOS technologies, random variation is always a major mismatch factor. Especially for under 45nm, the random variation can be considered as the dominant mismatch factor. MOSFET mismatch has been investigated and modeled by many research groups in the industry and academia [51,52,57]. Conventionally, it has been modeled by normal distribution of the threshold voltage V_{th} or the transistor effective width W. The sigma for these normal distributions calculated from I_{dsat} measurement of a batch of specific size devices in a certain bias condition. This approach had several problems. The measured sigma is specific to the bias condition and the size of the devices in the batch and none of the V_{th} and W is independently changed by process variation. In fact variation of W and L is inversely proportional with the other dimension [51,52]:

$$\sigma_L^2 \propto \frac{1}{W} \qquad (5.2)$$

$$\sigma_W^2 \propto \frac{1}{L} \qquad (5.3)$$

The newer approaches are calculating the standard variations of process parameters by solving a set of equations built from measurements collected across many dies for many biases and geometries. The standard variation of parameters like flat band voltage V_{fb}, mobility, substrate dopant concentration (N_{sub}), length offset, width offset, short channel effect, narrow width effect, source drain sheet resistance and gate oxide thickness can be found with this method and fit into transistor models for SPICE or similar simulators. This information is usually incorporate into model files and therefore can directly be used in mismatch simulations of the devices with any arbitrary geometry and in any bias condition.

The random variation for the delay line topologies discussed in this section has been calculated by running Monte Carlo simulations. Table 5.3.1 shows the average delay and the sigma of the delay variation for a delay cell at typical process corner and fix voltage and temperature condition. As it can be observed, mismatch is significant in today sub 90nm technologies. The main technique to reduce mismatch is to increase feature size of transistors.

$$\sigma_p^2 \propto \frac{1}{LW} \qquad (5.4)$$

where p represents the process parameter of the interest. But geometry and bias inter-relationship might be less straight forward as in modern technologies devices can not have any arbitrary length. In some technologies the length is limited and quantized. On the other hand more gate overdrive voltage reduces the effect of V_{th} variation on I_d mismatch and thus increasing W for a current mirror might not be any help on decreasing the mismatch.

Table 5.3.1: Sigma of the delay variation due to the random mismatch for different types of delay elements

Delay line type	Switch Capacitors VCDL	CCDL	Differential CCDL
Average Delay (ps)	245	108	104
Sigma (ps)	46	29	15
Relative One Sigma Change (%)	18.7	26.8	14.4

For reducing current mismatch in the current mirror topologies usually increasing L is more effective. In the CMOS technologies where L is limited, series of transistors can be used to increase effective length. Although a series of n transistors with length L do not necessary have the same random variation of a transistor with length $n.L$.

5.4 Special consideration for Phase Detectors

Phase Detector (PD) circuit generates a signal proportional to the phase difference of its two input clocks. In the synthesizer context, these two input clocks are the reference clock and the feed back clock. There are many ways to implement a PD from Gilbert multiplier to digital circuits [53,54,55]. In modern charge pump base synthesizers Phase Frequency Detectors (PFD) are more popular. PFD is a state machine built from one or more memory elements. They offer unlimited pull-in range and the capability of frequency acquisition. Four popular types of PFDs will be presented in this section and advantages and bottle necks of each topology will be described briefly.

A typical conventional PFD is shown in Figure 5.4.1.a. [56]. Both reference and feed back clocks are latched to form a memory for this state machine. One problem of this PFD was observed in the simulations is the race between feedback path and the reset

path. In very fast CMOS technologies the reset path delay can be shorter than the feed back path delay. This doesn't let the reference clock latch properly when DOWN signal is active for a long time i.e. feed back clock is lagging a lot. The exact similar case can be observed when feed back clock leading reference for good chunk of the period. This has been addressed by adding an even number of inverter and increasing the delay of the reset path as it shown in Figure 5.4.1.b. The next drawback of using this PFD is nonlinearity of the UP and DOWN pulse width for small values of phase differences. Figure 5.4.2 shows UP pulses for small amount of the phase differences. This phenomenon doesn't cause any problem for non-fractional-N architecture but can cause a considerable locking error in the proposed architecture when the multiplication factor is set to a fractional number close to an integer. Tuning PFD for a better linearity reduces its sensitivity to small phase differences which more or less causes the same magnitude of locking error in the proposed fractional architecture.

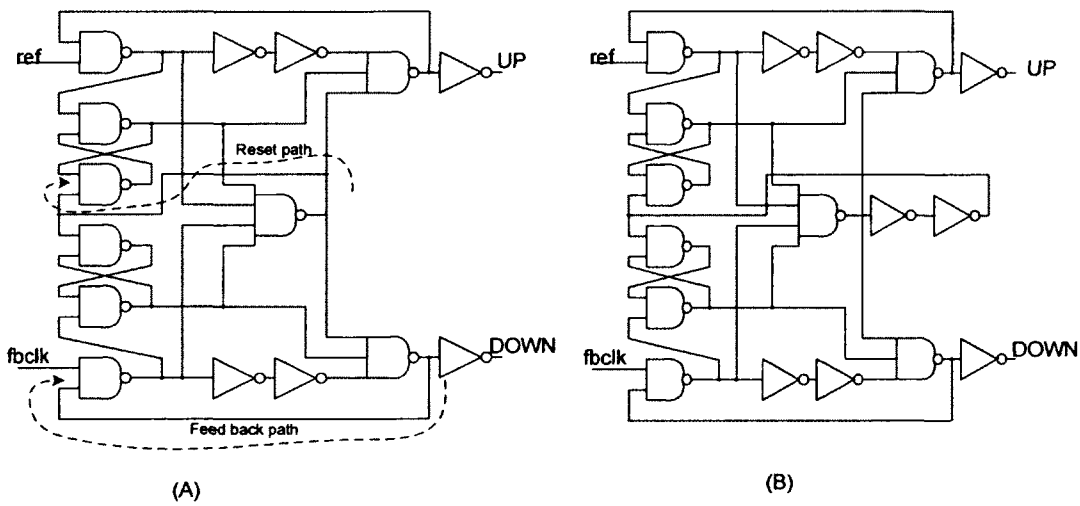


Figure 5.4.1: a) Conventional PFD; b) Modified conventional PFD for fast corners

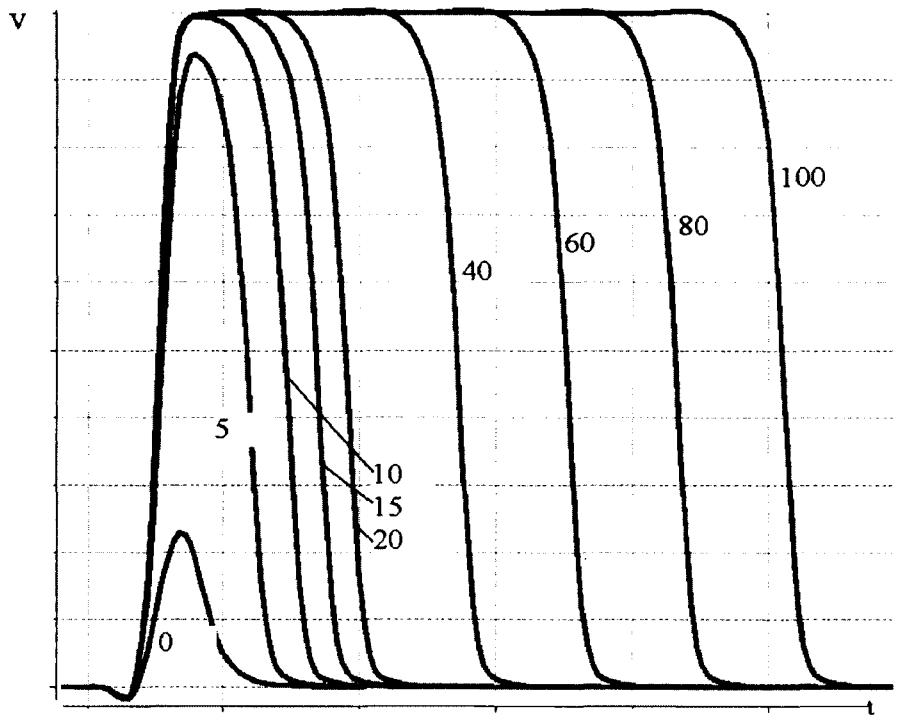


Figure 5.4.2: UP output of above conventional PFD for different values of phase differences

Wide varieties of recent PFD introduced in the literature are based on pre-charged latch topologies [58]. The clocked inverters are the building blocks of this topology. Four types of the clocked inverters are shown in Figure 5.4.3. With a combination of these clock inverters one can make pre-charged or non pre-charged latches. A typical pre-charged latch is shown in Figure 5.4.4.a. When CLK is low the first stage is charged to VCC and as a result Q (output) doesn't have any path to VCC or ground and thus would hold its previous value. When CLK goes high, output Q will follow the input. A typical non pre-charged latch is shown in Figure 5.4.4.b. When CLK is low Q would hold its previous value, and when CLK is high Q would follow the input. A positive edge triggered D-flip flop made of clocked inverters is shown in Figure 5.4.5.

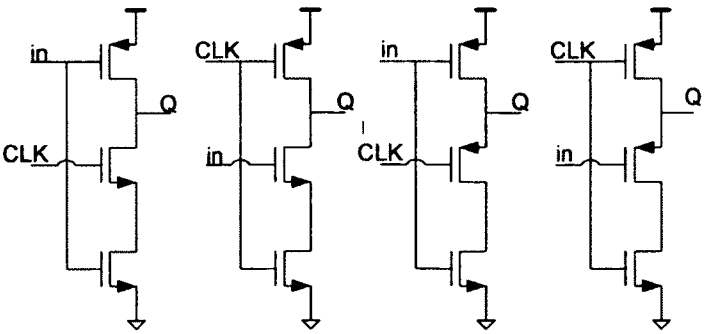


Figure 5.4.3: Four types of clocked inverters; from left to right NC, NC2, PC, PC2

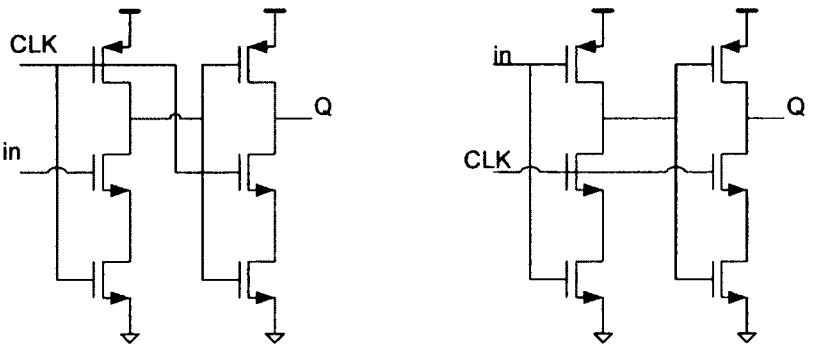


Figure 5.4.4: a) Pre-charged N latch (left); b) Non pre-charged N latch (right)

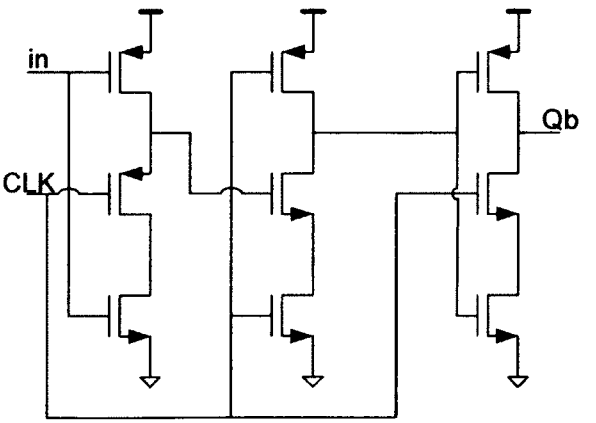


Figure 5.4.5: Positive edge triggered D-flip flop based on pre-charged stage

A nc-PFD [59] is shown in Figure 5.4.6.a. In the pre-charge mode when both the reference clock and the feedback clock are low, PMOS devices are on and nodes ncdnb and ncupb are charged to supply and consequently both DOWN and UP are low. The first rising edge of either the reference or the feedback clock would turn off its associated PMOS device. But nodes ncdnb and ncupb will stay charged until the delayed version of the other clock open NMOS middle switch. At this point both ncdnb and ncupb will be connected to the ground and thus both UP and DOWN would be high. When the reference/feedback clock goes low, the corresponding PMOS turns on and bottom NMOS turns off and so the DOWN/UP signal goes low. The other signal goes low when the other falling clock edge appears. This PFD rely on falling edge and so is sensitive to clocks duty cycle. Buffering the clocks separates UP and DOWN rising edges by two inverter delay. This cause an error which is considerable when there is a small phase difference between the clocks. A modified version of the nc-PFD without buffer is shown in Figure 5.4.6.b. NC-PDF generates two long pulses for UP and DOWN (Figure 5.4.7.a), one pulse with the pulse width of half of the clock period and other with the pulse width of the time both reference and feed back clocks are high.

Another PFD topology famous as pt-PFD [59] is shown in Figure 5.4.8.a. This type of PFD is built of two parallel pre-charged N-latch with cross coupled reset feed back. When any of the clocks are low, the output of the first stage will be high regardless of the status of the other clock or the output of the PFD.

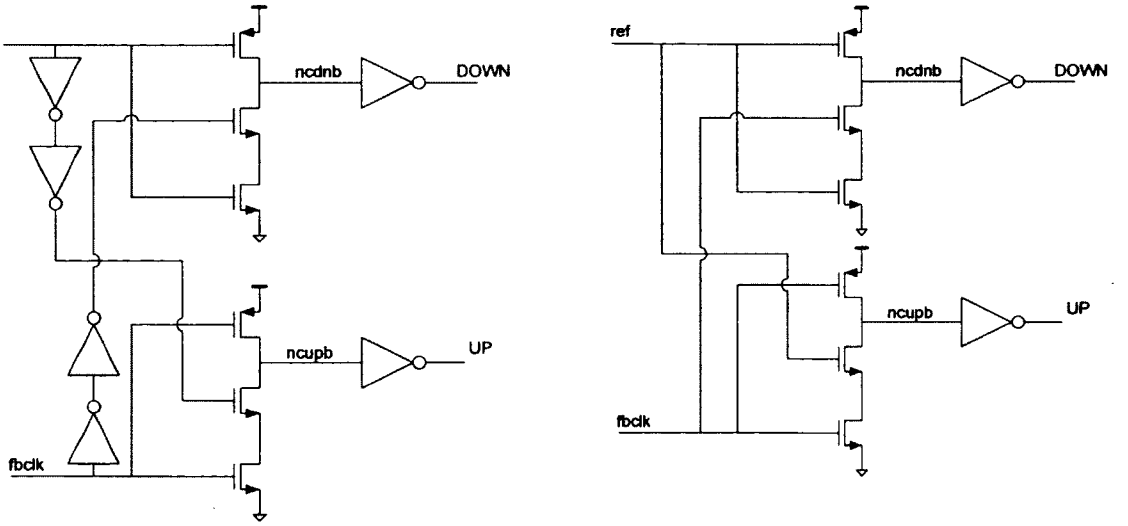


Figure 5.4.6: a) A typical NC-PFD (left) b) modified NC-PFD (right)

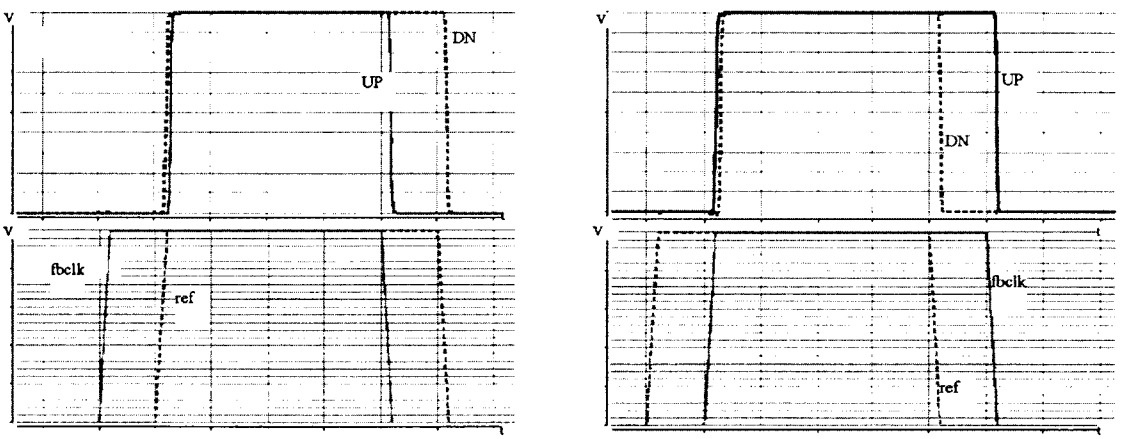


Figure 5.4.7: Reference clock, feedback clock, UP and DOWN signal for NC-PFD a) feedback clock lag 100ps (left graphs) b) Feedback clock lead 20ps (right graphs)

Let's assume at the initial state both of the clocks are low. Then the output of the first stage for both reference and feed back clock paths are charged to high. When the first stage is high and the clocks are low then the output of the second stage will hold its

current state, usually a high state results from the previous state at which the first stage is a low. This state would last until one of the clocks goes high, and consequently corresponding second stage goes low and its output goes to high. As the second clock's rising edge appears, the first stage of both paths goes to ground and pt-PFD initialized to its primary state. DOWN signals of a pt-PFD for phase differences of 0ps, 5ps, 10ps, 15ps, 20ps, 40ps, 60ps, 80ps, and 100ps are shown in Figure 5.4.8.b. There would be no UP pulse for any of these delay differences. As it can be observed delay differences less than 5ps can not be detected and the pt-PFD output pulse widths are not linearly proportional to phase differences less than 20ps. This might not be a major problem for the propose architecture depends on the target frequency range.

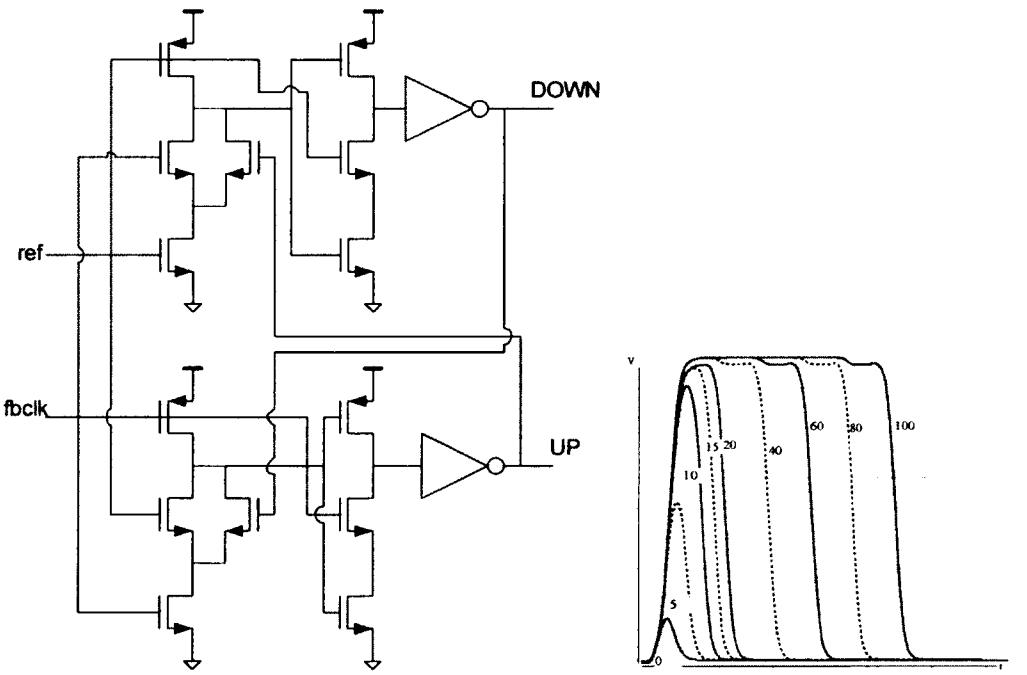


Figure 5.4.8: a) pt-PFD with cross coupled feedback reset (left); b) DOWN signal for 0, 5, 10, 20, ..., 100ps delay lag between reference and feed back (right chart)

A low blind zone PFD [60] (lbz-PDF) is shown in Figure 5.4.9. This topology has a feedback similar to the conventional PFD and uses the clocked inverter of Figure 5.4.4 to form the memory latches. When both reference clock and feed back clock are low, the first stage will be pre-charged to supply voltage and thus armed the next stage to get the rising edge. The first rising edge of the reference/feedback clock makes UP/DOWN active high until the other clock gets high. At this point both UP and DOWN will turn high until feedback reset the latches. After reset, UP, DOWN and reset turns low and PFD goes to its initial state. An lbz-PFD generates two identical UP and DOWN pulses when two inputs have zero phase (delay) difference. This would address the dead zone problem in a charge pump base loop. The conventional PFD and the pt-PFD do not generate any UP or DOWN pulses when the two inputs are in phase, situation corresponding to lock position in integer-N synthesizers.

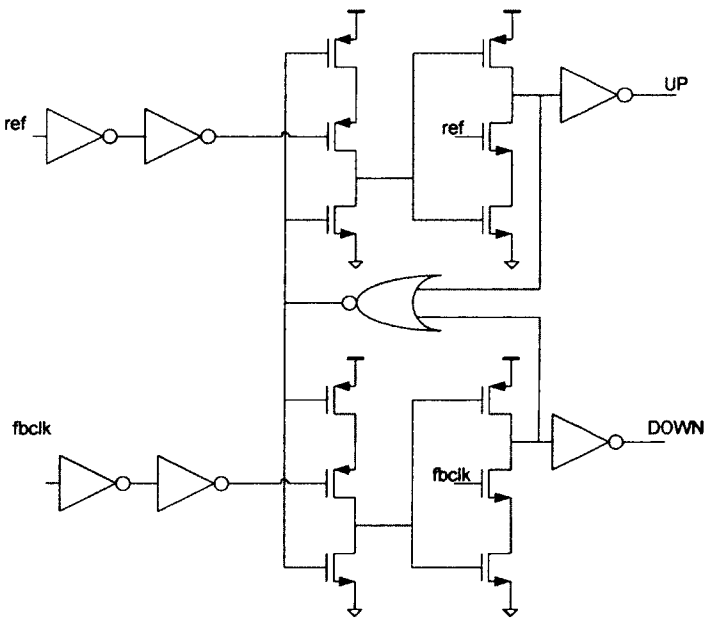


Figure 5.4.9: Low blind zone PFD with NOR gate feed back reset

The extra lbz-PFD fix-length pulses would also add to the width of the active UP and DOWN signals. Wider UP and DOWN pulses help lbz-PFD to have a higher sensitivity to phase difference between the reference and feed back clocks. DOWN and UP signals of a lbz-PFD for phase difference of 0ps, 5ps, 10ps, 15ps, 20ps, 40ps, 60ps, 80ps, and 100ps are shown in Figures 5.4.10.

Table 5.4.1 summarizes UP and DOWN pulse widths versus the input phase difference for PFDs discussed in this section. Table 5.4.2 shows the difference of the pulse width of UP and DOWN signals of lbz-PFD and NC-PFD. Note that for both lbz-PFD and NC-PFD there would be always UP and DOWN signals even in lock position in an integer-N synthesizer. Therefore the difference of the pulse widths determines the input of the charge pump and affects the dynamic of the loop.

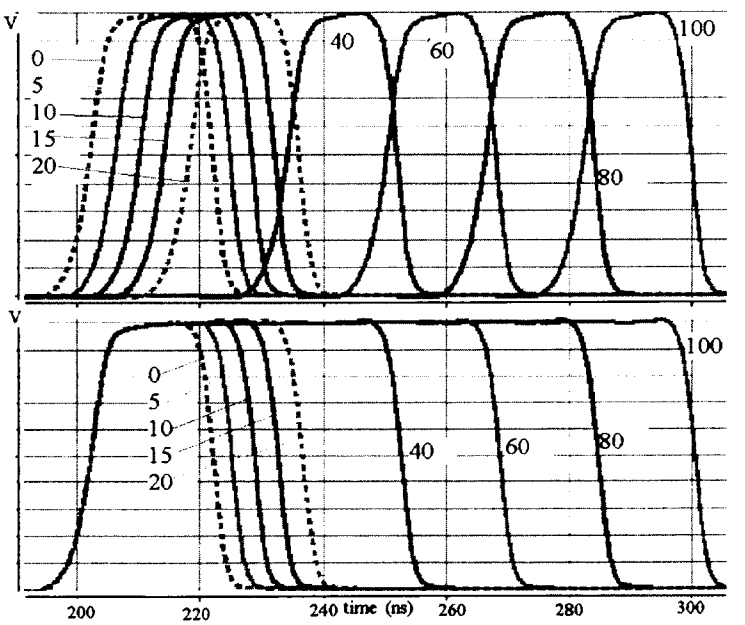


Figure 5.4.10: UP and DOWN signals of lbz-PFD (UP in top graph and DOWN in bottom graph) for 0, 5ns, 10ns, 15ns, 20ns, 40ns, 60ns, 80ns, 100ns delay between reference and feedback clocks

Table 5.4.1: UP and DOWN pulse width for different types of PFD vs. clocks delay difference; all the numbers are in nano-second

Delay difference	Conventional		Low blind Zone		Pre-charged (pt)		NC-PDF	
	DOWN	UP	DOWN	UP	DOWN	UP	DOWN	UP
-0.1	0.111694	No pulse	0.12123	0.0204	0.106185	No pulse	0.390086	0.497046
-0.08	0.091696	No pulse	0.10126	0.020395	0.086185	No pulse	0.41005	0.497071
-0.06	0.071697	No pulse	0.08129	0.02039	0.066184	No pulse	0.430014	0.497098
-0.04	0.051573	No pulse	0.06133	0.020396	0.046166	No pulse	0.449977	0.497129
-0.02	0.029481	No pulse	0.04151	0.020588	0.025827	No pulse	0.46994	0.497212
-0.015	0.023328	No pulse	0.03670	0.020805	0.020349	No pulse	0.47493	0.497318
-0.01	0.013604	No pulse	0.0321	0.021317	0.014306	No pulse	0.479909	0.497277
-0.005	No pulse	No pulse	0.02802	0.02254	0.00267	No pulse	0.484909	0.494818
0	No pulse	No pulse	0.02501	0.024997	No pulse	No pulse	0.489955	0.489955
0.005	No pulse	0.015365	0.02346	0.028692	No pulse	0.00267	0.494818	0.484909
0.01	No pulse	0.02315	0.02274	0.033206	No pulse	0.014306	0.497277	0.479909
0.015	No pulse	0.028916	0.02244	0.038025	No pulse	0.020349	0.497318	0.47493
0.02	No pulse	0.034217	0.02229	0.042925	No pulse	0.025827	0.497199	0.46994
0.04	No pulse	0.054676	0.02213	0.062815	No pulse	0.046166	0.497129	0.449977
0.06	No pulse	0.074694	0.02212	0.082769	No pulse	0.066184	0.497098	0.430014
0.08	No pulse	0.094689	0.02211	0.10273	No pulse	0.086185	0.497071	0.41005
0.1	No pulse	0.114685	0.02212	0.122692	No pulse	0.106185	0.497046	0.390086

Table 5.4.2: UP and DOWN pulse width difference for lbz-PDF and NC-PDF

Phase difference (ns)	UP and DOWN pulse difference	
	lbz-PDF	NC-PDF
-0.1	-0.10083	-0.10696
-0.08	-0.08087	-0.08702
-0.06	-0.06091	-0.06708
-0.04	-0.04094	-0.04715
-0.02	-0.02093	-0.02727
-0.015	-0.0159	-0.02239
-0.01	-0.01078	-0.01737
-0.005	-0.00548	-0.00991
0	-1.66e-05	0
0.005	0.00523	0.009909
0.01	0.010463	0.017368
0.015	0.015584	0.022388
0.02	0.020627	0.02726
0.04	0.040679	0.047152
0.06	0.060649	0.067084
0.08	0.080611	0.087021
0.1	0.100572	0.10696

5.5 Charge pump design for a never locked loop

Charge pump (CP) changes the timing information of the UP and DOWN signal to an analog control voltage. In a fractional-N DLL loop PD or PFD constantly generates UP and DOWN pulses even in the steady state mode of operation when the desired delay has been acquired. Unlike in an integer-N architecture, UP and DOWN pulse widths would have larger or smaller sizes regardless of whether the lock is acquired or not. As a result, the CP would be always active with a long ON time as it works in a never locked loop. Consequently a fractional-N DLL charge pump demands more stringent design requirements compare to integer-N counter parts due to its long on time at the lock

operation mode. All types of non-idealities need to be carefully addressed in a fractional-N charge pump design to reduce the induced jitter.

5.5.1 Source of errors in charge pump

A typical schematic of a charge pump is shown in Figure 5.5.2.a. A charge pump functions by sourcing current in and sinking current out of a capacitor and thus changes the total charges which is stored in it. By changing the charged stored in the capacitor, its output voltage would also change. This voltage is usually used as control voltage in the synthesizer loop. The main sources of errors in charge pump are leakage current and mismatch of the source and sink current paths. Leakage can happen in the capacitor or from the switches. Switch leakage can be due to the gate leakage in ON state or the drain current at OFF state. Depends on the fabrication technology and the type of capacitor these leakage currents can be quite significant. Never the less the leakage current causes phase/delay error in the loop. In a DLL the delay error due to the leakage current I_{leak} can be shown as:

$$t_e = (I_{leak}/I_{cp}).T_{ref} \qquad (5.5)$$

Where T is the period of the reference clock and I_{cp} is the charge pump current. This error can be reduced by increasing the charge pump current. Other major source of the error is the current mismatch. Any of the UP or DOWN signals coming from PFD opens one of the CP switches to either source current to the capacitor or sink current from it. The difference of the source and the sink current (Δi_{cp}) cause a delay error in DLL which can be calculated with following equation:

$$t_e=(\Delta i_{cp}/I_{cp}).t_{on} \qquad (5.6)$$

Where t_e is the delay error due to current mismatch, I_{cp} is charge pump current, t_{on} is the total CP on time and Δi_{cp} current mismatch between CP sink and source currents. There are few factors that contribute to the current mismatch. Depends on the output voltage, sink and source switches and current mirror devices may not have exact same bias conditions. Process corners especially cross skew corners and device mismatch can also aggravate CP current mismatch. As it can be seen in equation 5.6, a higher CP current will reduce phase/delay error. However, the total current consumption of CP is much smaller as the switches are ON for only a short period of time. Although in the fractional-N architecture the ON time (t_{on}) is longer than integer-N counterparts, still total CP power is smaller. A good approximation for CP current consumption is $(t_{on}/T_{ref}).I_{cp}$ plus the bias currents. Charge pump topologies are explained in the next section and the techniques to reduce the current mismatch will be discussed accordingly.

5.5.2 Charge pump topologies

Charge pump topologies are divided into the single-ended and differential categories. Figure 5.5.1 shows four different single ended CP topologies. Figures 5.5.1.a, b, and c show single ended CP topologies where the current switches are in drain, gate and source of the current mirror devices. Charge sharing can add some extra error when the switch is located at the drain as it is shown in Figure 5.5.1.a. Switching in gates (Figure 5.5.1.b) guarantees that current mirrors are in the saturation region [61]. However, for fast switching the mirror currents passing M1 and M2 needs to be high. This would increase CP current consumption. For switches in the source [62] M3 and M4 are in saturation but there is no need for a high current in M5, M6 or M7, M8 paths to speed up the switching. Figure 5.5.1.d shows a single ended CP topology where both UP and DOWN switches

are made of NMOS devices [63]. There is no current passing M3 when the UP switch is off. This increases the switching time unless a large current is used in both of the UP and DN paths.

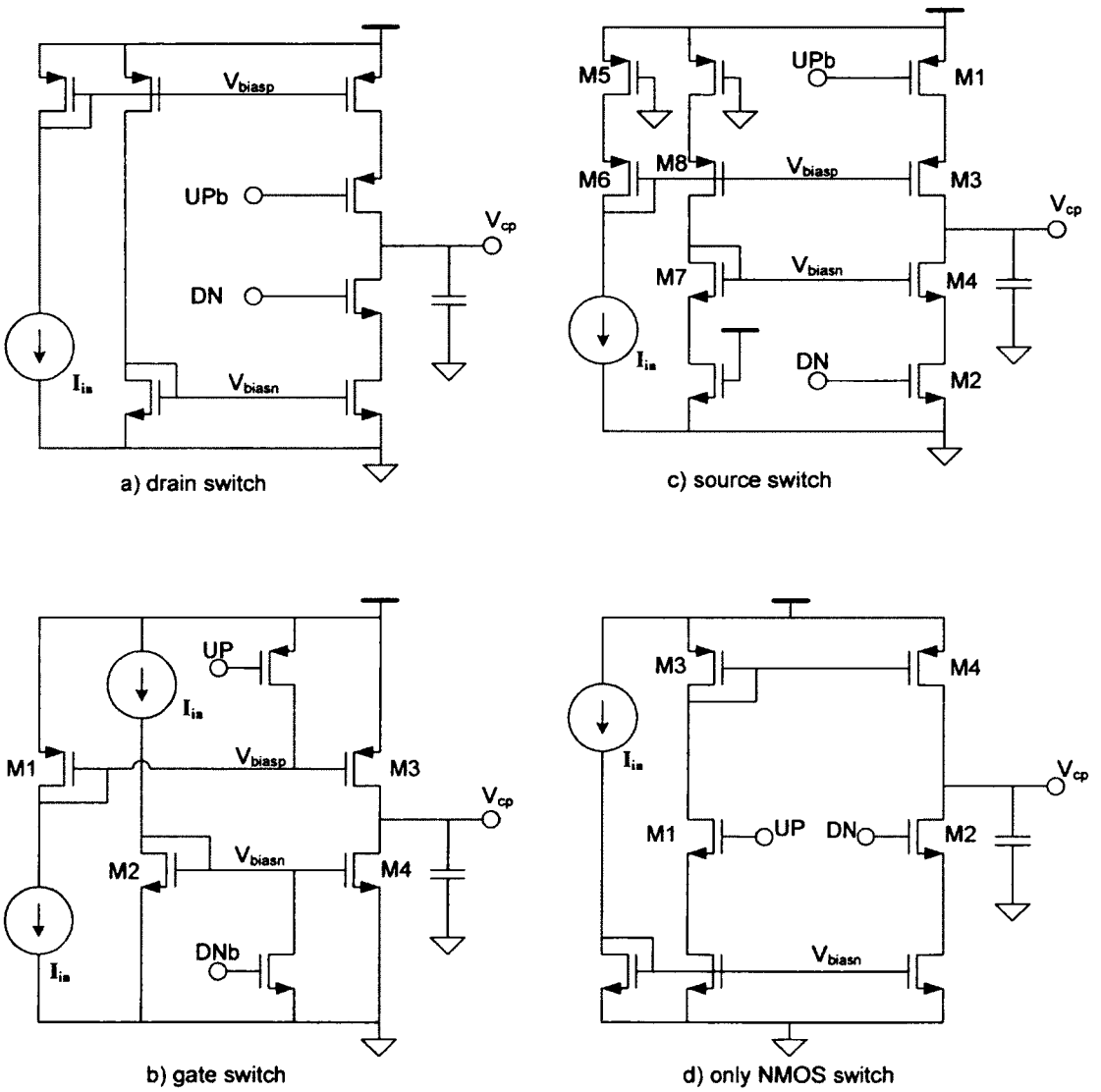


Figure 5.5.1: a) CP with current switch at drains; b) CP with switching current by gate control; c) CP with current switch at source d) CP with two NMOS switches

The differential topology can reduce the impact of the mismatch current on the performance of the CP. The source of mismatch in CP is the mismatch between NMOS and PMOS devices and short channel effect. A differential CP is shown in Figure 5.5.2. The mismatch between NMOS and PMOS transistors has less impact on the overall performance as it only affects the common mode variation of the output voltage. A common mode feedback circuitry (not shown in the Figure) can further reduce the impact and thus relaxes matching requirement between NMOS and PMOS comparing to the single ended CP. The mismatch between two same type NMOS or PMOS devices will appear as the common mode or differential error. If M1 source more current than M3 and M2 sink more current than M4 (means one side of CP is stronger than the other side) the error would appear as the common mode and can be corrected by proper common mode feedback. In this case still CP has no differential error and thus wouldn't generate any phase error. If M1 and M4 pass less or more current than M2 and M3, then mismatch

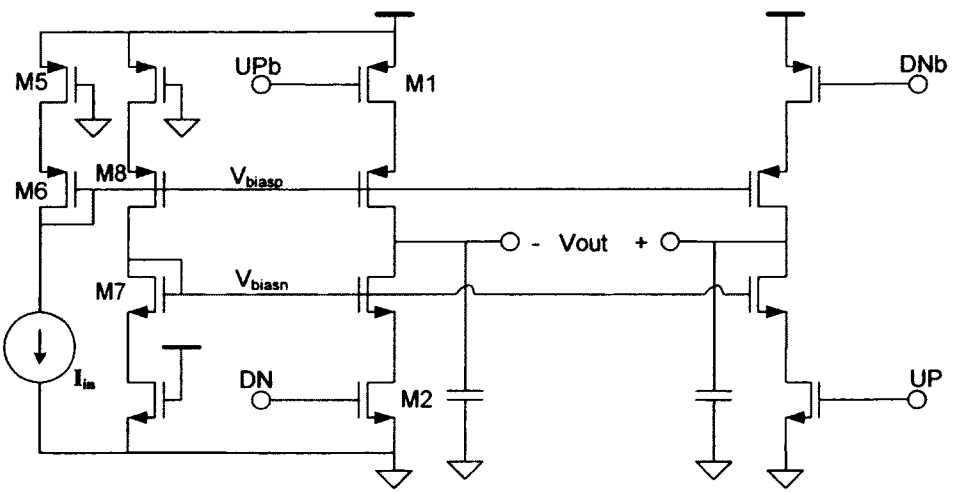


Figure 5.5.2: Differential CP with switch at source

creates a differential voltage error which consequently induces a phase error. This case can be easily generated due to short channel effect. Imagine the V_{out} positive has higher voltage than the V_{out} negative. In this case the V_{ds} of M1 and M4 are higher than the V_{ds} of M2 and M3. The additional inverter delay on the UPb and DNb path does not create any offset error due to the symmetry of the design. This topology is also less sensitive to cross skew corners and the leakage current. The current mismatch can be further decreased by using active current mirrors. To explain how an active loop can help reduce the CP mismatch, consider the source switching CP of Figure 5.2.1.c. Depends on the CP voltage V_{cp}, transistors M3 and M4 would have a different biasing condition comparing to the M7 and M8, the other side of the current mirror. These causes mismatch between the source and the sink current. Figure 5.5.3 shows the same charge pump where an active amplifier controls the gate voltage of PMOS devices and tries to adjust the voltage node N1 in a way that pairs M4, M7 and M3, M8 have more similar bias conditions. Note that due to the amplifier systematic error and the mismatch and other non-idealities and non-symmetry in the circuit a residue of current mismatch would be remained. Much more complex active CPs have been proposed in the literature which some have compensation for both of the sink and source currents [64,65,66,67].

5.5.3 Variable gain charge pumps and design consideration

As it is explained in section 5.1 CP current, I_{cp}, has a great impact on the switching induced jitter in a fractional-N DLL. On the other hand as it is explained in this section, the higher I_{cp} reduces the effects of many CP non-idealities, including the current mismatch and the capacitor leakage current. Also more I_{cp} and loop bandwidth would

help a faster lock acquisition. In general during transition, higher I_{cp} and loop bandwidth are desirable but at the lock position lower I_{cp} and bandwidth reduce the jitter and thus is more desirable. Therefore a variable gain charge pump topology looks appealing for the fractional-N DLL. However, such a solution involves many design considerations. A low I_{cp} needs to be chosen carefully to gets the best trade-off between the switching induced jitter and the constant phase error results from the leakage, the current mismatch, and the systematic offset due to lower loop gain. A high I_{cp} needs to be chosen to have a fast locking but doesn't cause any stability issue or ringing in the loop response.

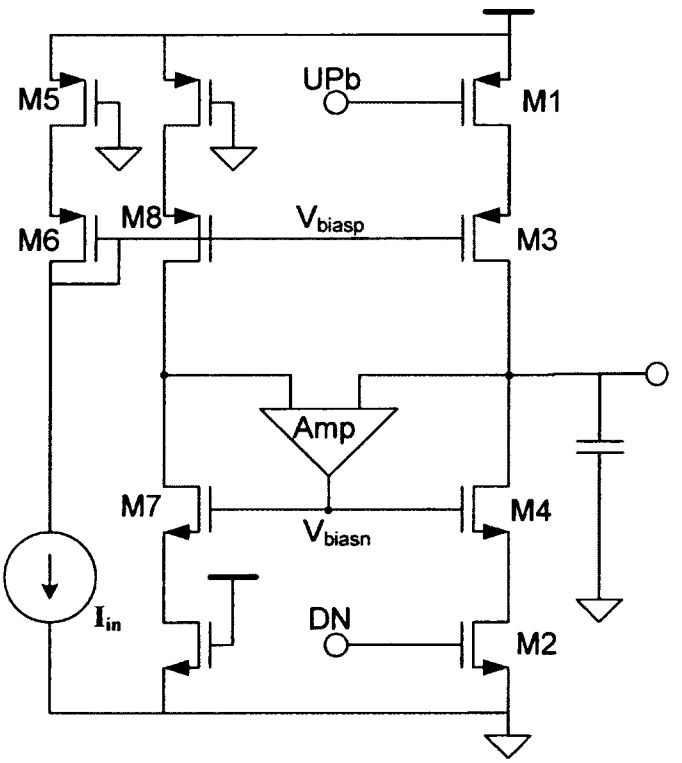


Figure 5.5.3: Active CP with switch at source

5.6 Phase interpolator design for Fractional-N DLL

The phase interpolator (PI) gets two or more phases of the clock and combines them to generate a clock with an arbitrary phase between the two input phases. The PI performance is usually measured by its resolution and linearity and expressed in terms of the step size, the DNL (Differential non-linearity) and the INL (Integral non-linearity). The phase step size is defined as the amount of the PI output clock phase change resulted from changing PI setting by one LSB (least significant bit). The DNL is defined as the difference of each step size with its ideal value. The ideal step size value is equivalent to the total phase difference between the two input clocks divided to the number of PI settings. The INL is defined as the total phase error associated to a specific PI setting.

Depending on the application the importance of these parameters may be different. For the fractional-N DLL the response time of the PI to the control signal also plays a crucial role. The PI step size directly depends on the synthesizer frequency resolution. For example a fractional-N DLL with frequency resolution of F_{ref} /100 requires a PI with minimum 100 steps. The response time of the PI to change of the control signal need to be shorter than the period of the reference clock (T_{ref}). The DNL is directly contributed to the timing error for small values of α (the fractional part of multiplication factor). The PI timing error can be as high as INL for large α values. INL can be assumed as higher limit for the PI induced timing error in this period synthesis architecture. There are different ways to implement PI in CMOS technology. Some of the famous PI topologies will describe in the following sections in two major single ended and differential categories.

5.6.1 Single ended phase interpolators

A single ended PI is generally made by combining two or more clock paths with different strengths. These clock paths are fed by different phases of the clock. Figure 5.6.1 shows a typical single ended PI made of parallel switched inverters. In this type of the PI, the strength of each path is determined by the number of the active (ON) inverters in that path. Usually the total number of the active inverters in each PI setting is constant and equal to the number of inverters of one path. If only the inverters of the first path are active, that phase would appear at the output. When one of the inverters of the first path turns off, one of the inverters of the second path becomes active and the output clock phase will be shifted slightly toward phase of the second input clock. This process can continue until the output phase is shifted completely to the second phase. To have a linear phase combination it is necessary to slow down the edge rate of the input clocks. The clock edge is slowed down by passing the clock through an RC circuit named pre-conditioner. The amount of the required pre-conditioning depends on the phase difference between two inputs. Figure 5.6.2 shows the phase/delay progress for the PI of Figure 5.6.1 with 16 inverters at two different pre-conditioning levels. More pre-conditioning provides more uniform phase stepping. The delay/phase progressing curves, step size and DNL are shown in Figures 5.6.3, 5.6.4, 5.6.5. Figure 5.6.6 shows another type of single ended PI built of the starving inverters. In this type of PI the strength of each path is set by the current of the starving inverter. Passing the input clocks through the pre-conditioner is still necessary to achieve a reasonable phase/delay linearity. The delay/phase progressing curves, step size and DNL for a starving inverter based PI are shown in Figures 5.6.7, 5.6.8, 5.6.9.

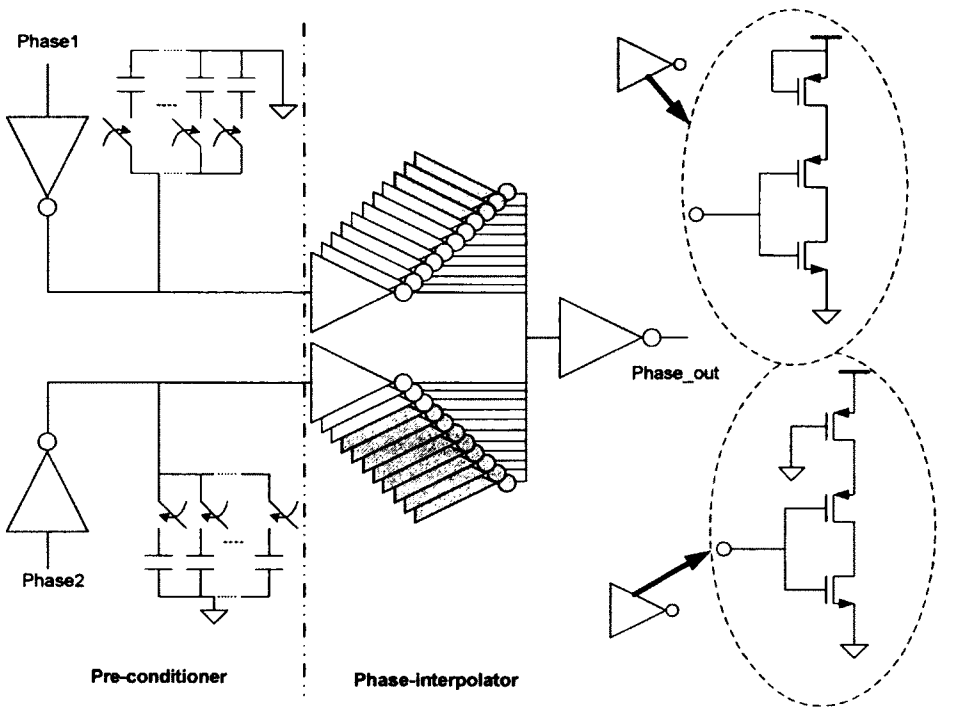


Figure 5.6.1: Single ended switched inverter PI

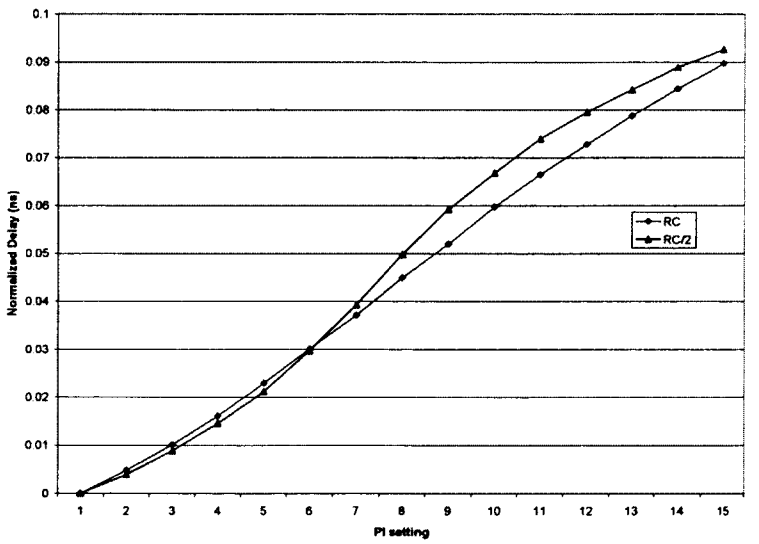


Figure 5.6.2: Phase/Delay progress of single ended switched inverter PI vs. number of active inverter in the first path at two different pre-conditioning levels

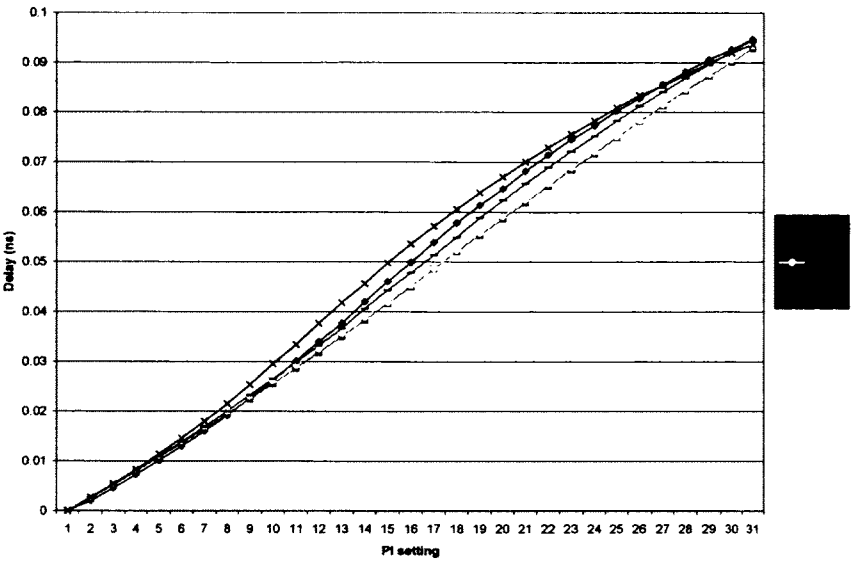


Figure 5.6.3: Phase/Delay progress of single ended switched inverter PI vs. number of active delay in the second path at different process corners

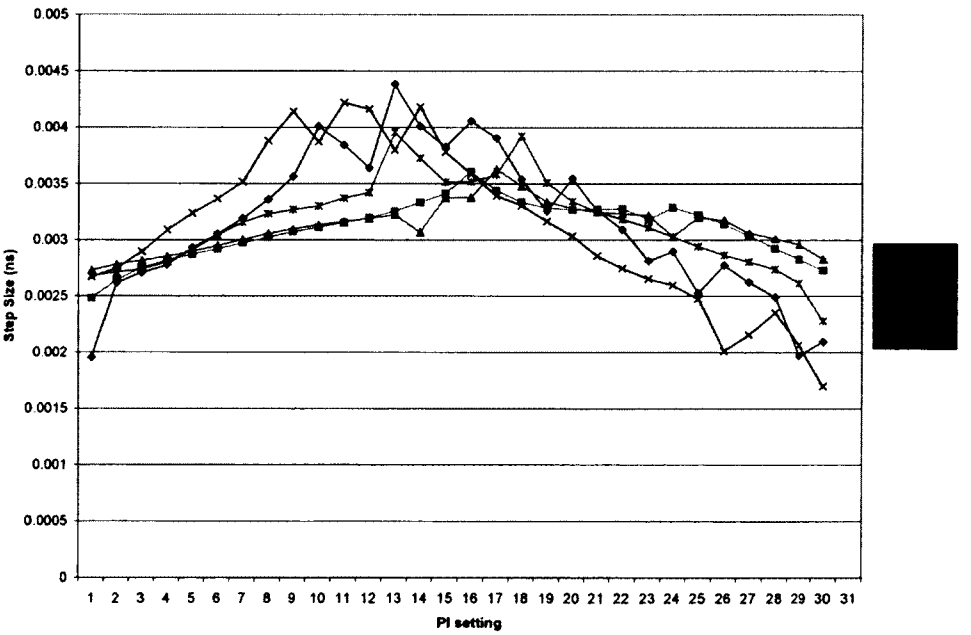


Figure 5.6.4: Step size for single ended switched inverter PI vs. number of active delay in the second path at different process corners

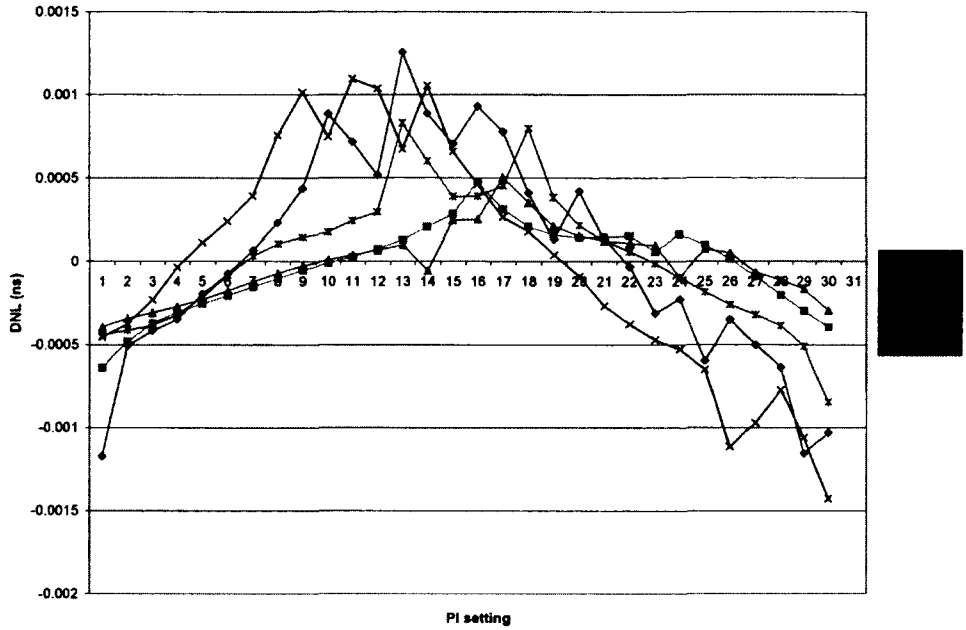


Figure 5.6.5: DNL for single ended switched inverter PI vs. number of active delay in the second path at different process corners

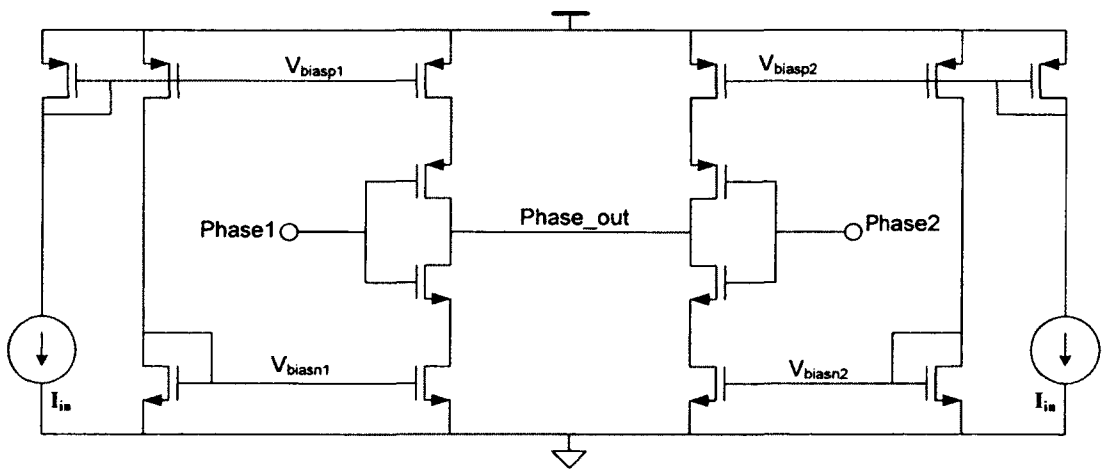


Figure 5.6.6: Single ended current control PI

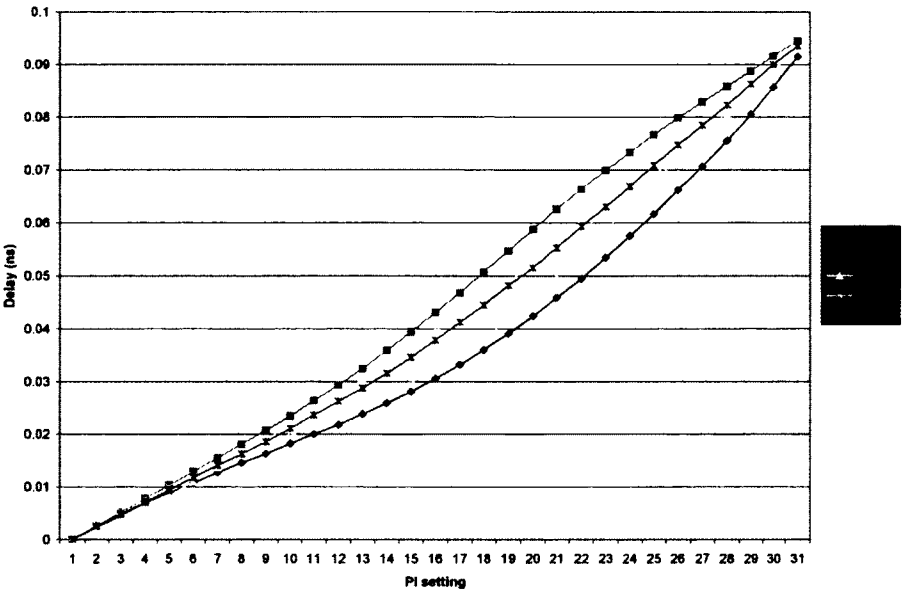


Figure 5.6.7: Phase/Delay progress of single ended switched current PI vs. PI current control DAC setting at different process corners

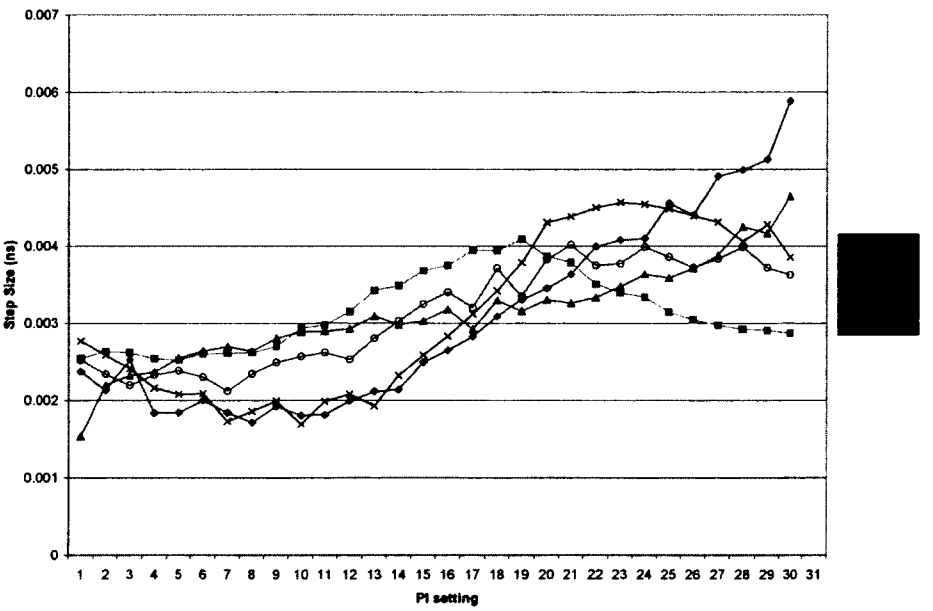


Figure 5.6.8: Step size of single ended switched current PI vs. PI current control DAC setting at different process corners

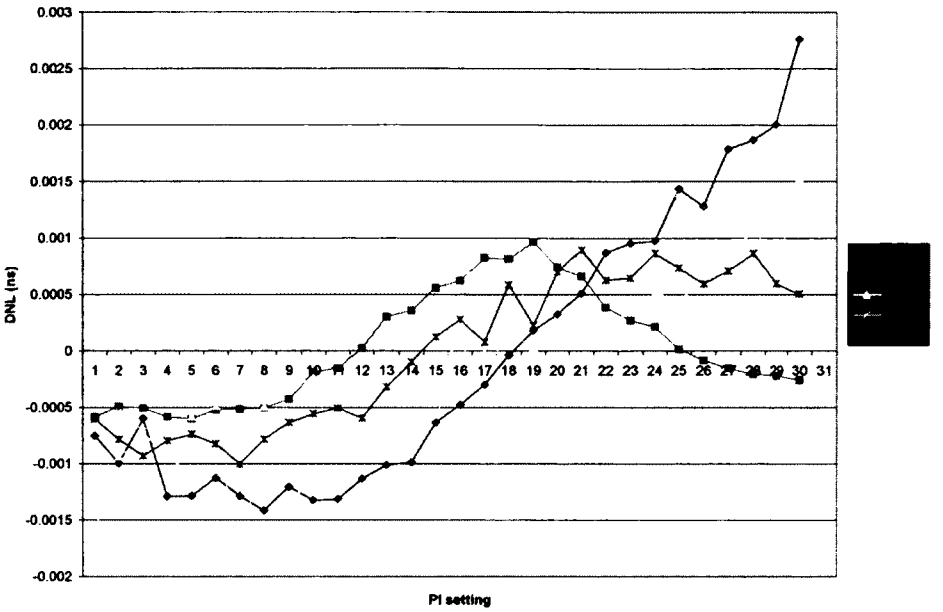


Figure 5.6.9: DNL of single ended switched current PI vs. PI current control DAC setting at different process corners

5.6.2 Differential phase interpolators

The schematic of a differential phase interpolator is shown in Figure 5.6.10. It has two main components: pre-conditioner and phase mixer. Two differential input clocks with some specific phase difference go through the pre-conditioner. The pre-conditioner slows down the edges of the clocks to get a better linear phase interpolation in the phase mixer part. To have a better linearity the edge rate is reduced to have a saw-toothed triangle shape waveform. The phase-mixer combines the two phases of the clock with the proper weight to generate a desired output clock phase. Current DACs are used to weight each clock in the phase mixer. A schematic of a typical 3bit/8stage current DAC is shown in Figure 5.6.11. The output current of the DAC, I_{out} depends on the number of the transistors which are connected to V_{bias}. The other transistors would be off by connecting

their gates to ground. The current switches can be also placed at the drain or the source of the current mirror devices as were explained in the section 5.5.2. The two current DACs of the phase mixer should always have a fixed number of ON current switches equal to the number of switches in one of the DACs. If all the switches of one DAC are ON, then the phase associated to that DAC would pass to the PI output. When one of the current switches turns off, one of the current switches in the second DAC turns ON and as a result the PI output phase moves slightly toward the other phase associated to the second DAC. This process will continue until all the current switches in the second DAC turn ON. At this point all the current switches in the first DAC are off and the clock phase associated with the second DAC would appear at the PI output. The phase (delay) progress of a differential PI and its associated step size and DNL curves are shown in Figures 5.6.12 and 5.6.13.

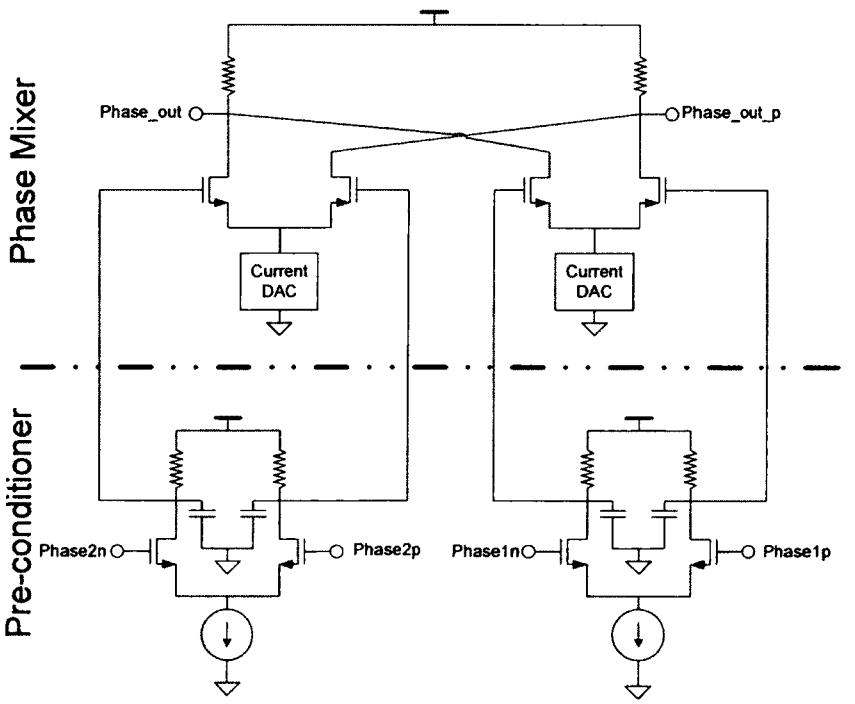


Figure 5.6.10: Differential PI contains two pre-conditioner and phase Mixer parts

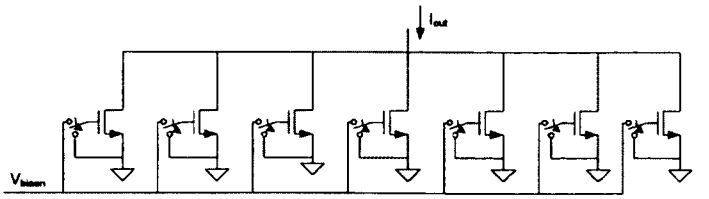


Figure 5.6.11: **A typical schematic of 3bits/8stages current DAC**

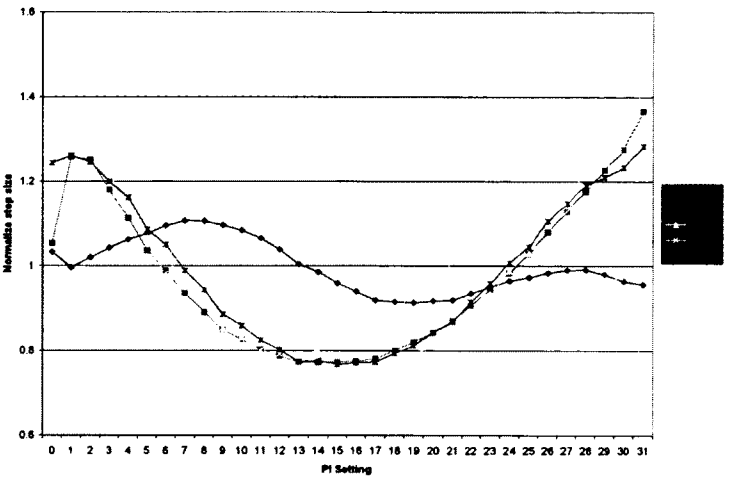


Figure 5.6.12: Normalize step size for a differential PI vs. its setting

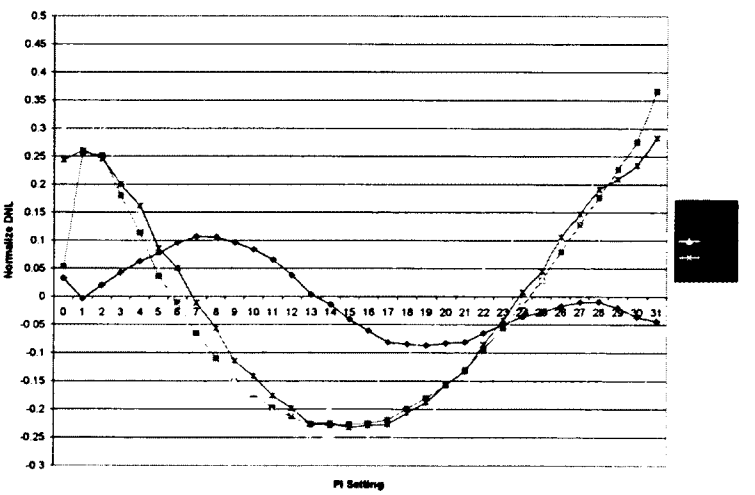


Figure 5.6.13: Normalize DNL size for a differential PI vs. its setting

5.7 Multi-Dimensional design decisions for DLL-based Fractional-N Synthesizer

The main building blocks of a fractional-N DLL-based synthesizer have been discussed in the previous sections of this chapter. Different topologies are presented for each block and their functionality, performance and trade-offs discussed in details. An important part of the design is to choose the subset of the topologies that makes the best implementation of the proposed architecture for a target application and in a given fabrication technology. These design decisions for deep sub-micron technologies (feature sizes smaller than 65m) and 5GHz target frequency are presented in this section. The performance of the resulting synthesizer will be presented in next chapter.

The delay cell is made of a combination of the current control delay element and the switched capacitor load as shown in Figure 5.3.9. As explained in section 5.3.2 this combination offers a good power supply sensitivity. A switch capacitor load acts as a course tuning knob to compensate for part of the process variation. It also helps to have a predetermined range of control current and thus eases up the stringent requirements of the charge pump circuit. As a serious drawback, switch capacitor delay elements induce duty cycle distortion which can not be addressed by transistor sizing for the entire process corner variation. This duty cycle distortion is a result of non-symmetric gate leakage current path impedance to power rail and ground. This non-symmetry would cause a difference in rising and falling edge delays at the switch capacitor node. In case of buffer delay cells, the delay difference between rising and falling edges causes a duty cycle distortion which is accumulated along the delay line. If a delay cell is constructed by two

inverted delay elements, each two consecutive delay elements will get both rising and falling edges and thus cancel the induced duty cycle distortion. In other words the delay cell (made of two inverted delay elements) doesn't induce any duty cycle distortion. Figure 5.7.1 shows the input and output of a 10 stage delay line constructed from buffered delay cells of Figures 5.7.2. As it can be observed, the delay line output has a considerable duty cycle distortion. Figure 5.7.3 shows the input and the output of a 10 stage delay line constructed from inverted delay cells (Figures 5.7.4). It can be observed that the delay line output has almost a perfect duty cycle. As it will be explained in this section, in order to get a better linearity and DNL in the PI, the delay line needs to be constructed from buffered delay cells. On the other hand duty cycle distortion would create intolerable glitches both at the delta sigma control MISO switch going to the lbz-PFD and the Global edge combiner final output and should be avoided at all cost. To address this problem, a delay cell is made of two buffered delay elements shown in Figure 5.7.5. The resultant delay line has a close to perfect duty cycle of inverted delay cells but provides buffered delayed outputs (Figure 5.7.6).

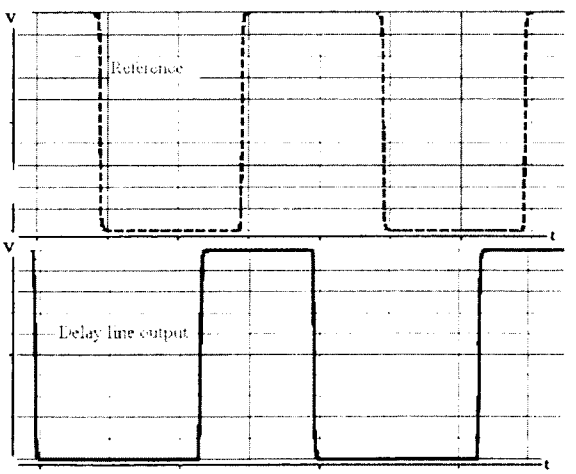


Figure 5.7.1: Input (reference) and output of a 10 stages long delay line constructed from buffered delay cell of Figure 5.7.2

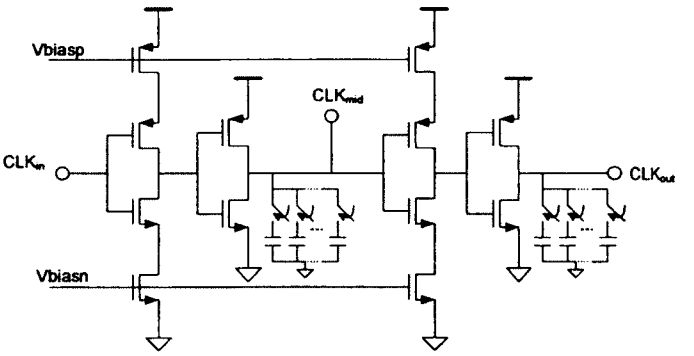


Figure 5.7.2: Typical two buffered delay cell

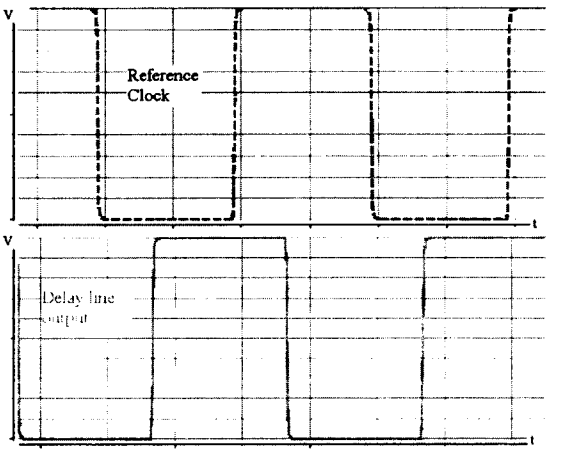


Figure 5.7.3: Input (reference) and output of a 10 stages long delay line constructed from invertered delay cell of Figure 5.7.4

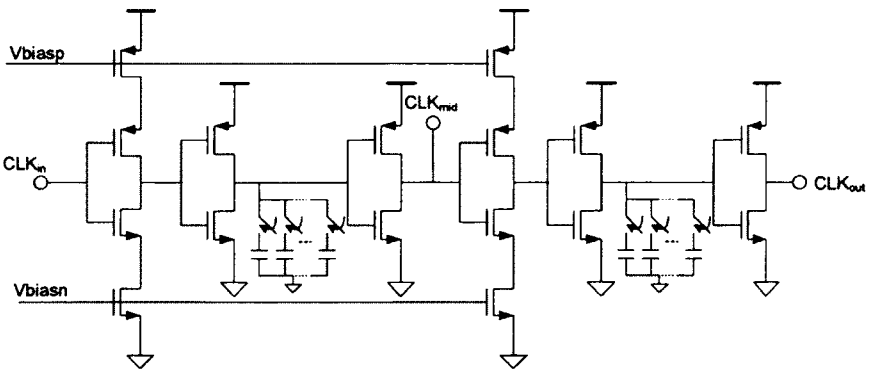


Figure 5.7.4: Typical two inverted delay cell

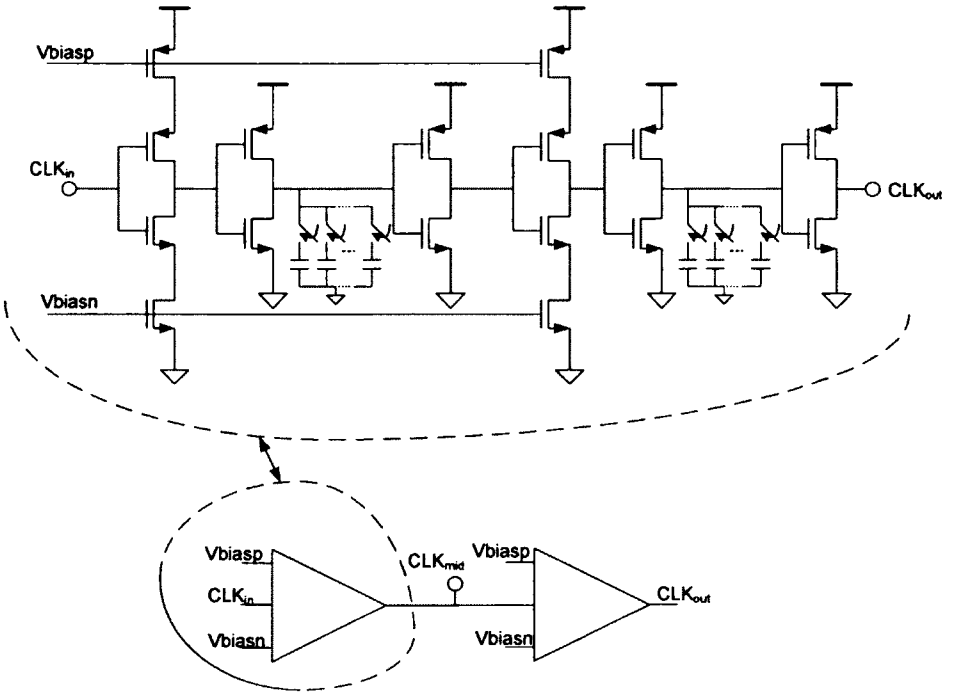


Figure 5.7.5: Proposed buffered delay cell with no duty cycle distortion

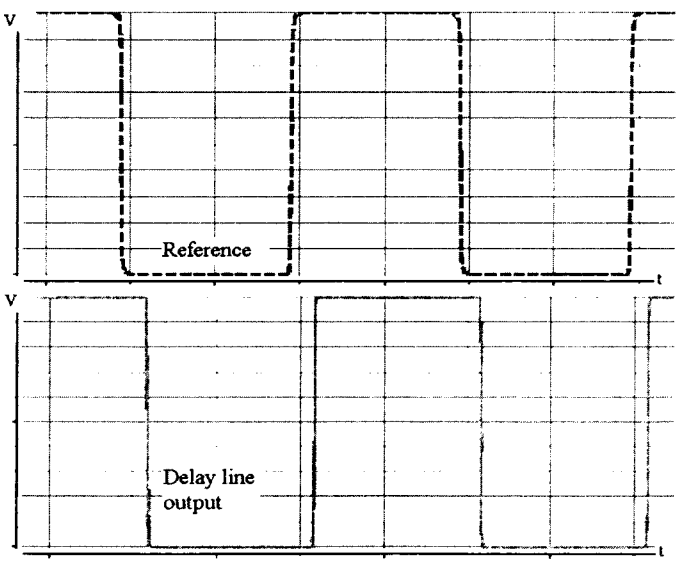


Figure 5.7.6: Input (reference) and output of a 10 stages long delay line constructed from buffered delay cell of Figure 5.7.5

A low blind zone phase frequency detector (lbz-PFD) described in section 5.4 is chosen as it offers a good integer locking and a better linearity for small phase differences between the feedback and the reference clocks. PFD linearity is extremely important in this topology. Especially for fractional numbers close to an integer it is important to detect and generate an error pulse linearly proportional to the small phase difference between the reference and the feedback clocks. This was the reason to choose the lbz-PFD from the PFDs which are explored in this chapter.

A charge pump in a deep sub micron technology is very challenging to design. This challenge even gets harder when the feature size is in the order of tens of nano-meters mainly for three reasons; First, good current mirrors are difficult to make due to the short channel effect and low output impedance of transistors. Second, transistor mismatch is more pronounced due to smaller feature size and, thus, matching currents are extremely difficult. Third, even in the steady state condition (here the fractional-N lock condition) charge pump has a very long ON (active) time. These reasons force the design toward using an active charge pump topologies, but for the first two mentioned reasons; making a high gain amplifier is also extremely difficult in these technologies. A lower supply voltage adds to the difficulties of designing a high gain amplifier. In this implementation a differential CP topology is chosen to alleviate the effect of the mismatch and provide a better immunity to power supply noise (compared to single ended counter parts). As discussed in the section 5.5.2, the differential charge pump topology addresses most of the mismatches. However, to reduce the mismatch effect further and optimize CP performance two active loops are added. The active loops adjust NMOS sink currents of both P and N sides of CP by comparing their output voltages to the output voltages of a

replica circuit. The schematic of the charge pump is shown in Figure 5.7.7. A separate amplifier is used for each side of the differential CP, as voltages Vcpn and Vcpp can be different and so as the bias condition and the impact of short channel effect on current source devices. To achieve necessary amplifier gain in deep submicron technology and its associated low supply voltage, a complementary folded cascode amplifier [68] has been designed. Folded cascode amplifiers [69,70,71] offer very good input voltage range and adequate gain at a low supply voltage despite the low device impedance of the nano-scale CMOS technology. The schematic of the folded cascode amplifier is shown in Figure 5.7.8. Figure 5.7.9 shows the AC gain of the designed amplifier versus frequency. Due to the output and input impedance of the amplifier and the fact that the loop is closed both from the CP and replica side the stability of the active loop needs an extra attention.

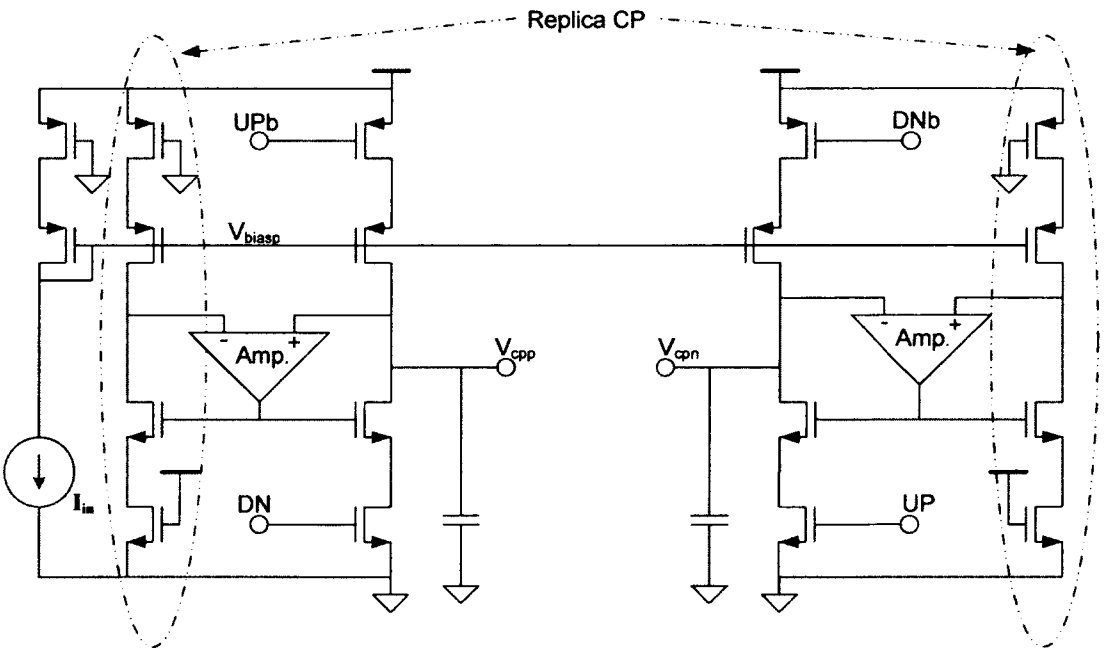


Figure 5.7.7: Active differential charge pump with two independent compensation loops and replica

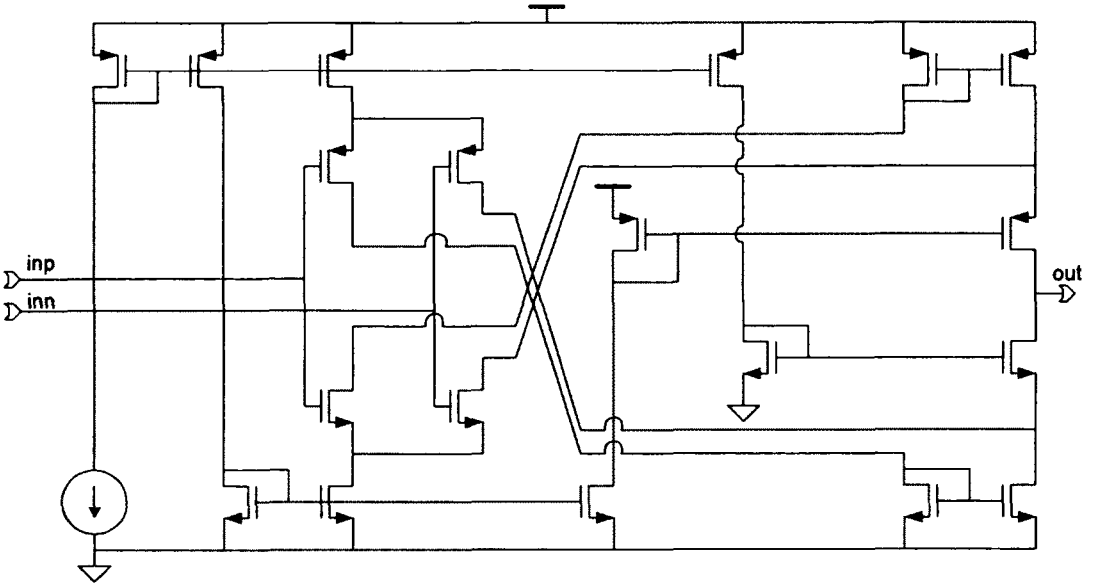


Figure 5.7.8: Folded Cascode Amplifier used for active CP compensation loop

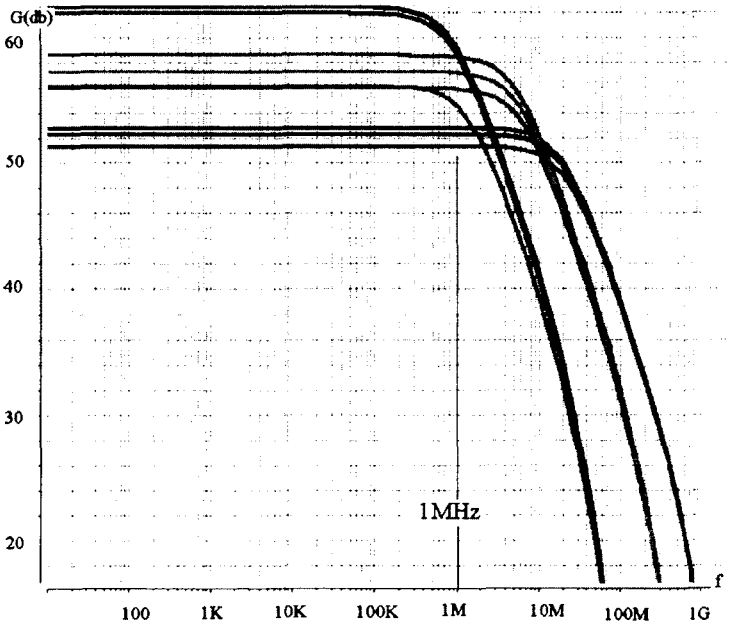


Figure 5.7.9: Folded Cascode Amplifier AC response for 9 main PVT corners

The dilemma is that the two symmetric nodes of the replica and the CP outputs have different capacitor values. This is equivalent to having two same order poles at different frequencies. It can also be considered as two feedback loops with one common pole at the

amplifier output. One option is to force the output of the amplifier (common pole) to be the dominant pole (of both loops). Note that the charge pump output capacitor would work against stability in this case. However, this solution may be still more appealing and more area efficient compared to adding another charge pump capacitor to the replica side. In this analog active loop design the bandwidth should trade with the gain whenever possible as the goal is to make a super damp and slow active loop with a high DC gain.

A differential voltage to current converter (VIC) after the loop filter (Master loop DLL filter) is used to convert the CP differential output voltages (Vcpn and Vcpp) to the control current for the delay line (Figure 5.7.10). The differential voltage to current converter (VIC) adds further common mode rejection to the overall CP+VIC topology and not only removes more PMOS/NMOS current mismatch but also provides a mean to have even more immunity to power supply noise. A coarse common mode correction can adjust the CP common mode voltage by forcing the CP current sources out of the saturation region. This correction can be performed on the fly during normal mode of operation without significant effect on the loop performance. The common mode circuitry sets both of the CP output voltages to mid rail at the power up initialization process.

The delay line is built of 10 buffer delay elements as shown in Figure 5.7.4. For a multiplication factor of five, each two delay elements make a delay cell. A 3 to 1 multiple input single output (MISO) switch connects delay cells 4, 5, 6 to the lbz-PFD. A similar MISO switch connects a reference clock to the other input of the lbz-PFD to equalize the delay. Both MISO switches have an enable signal. Switches need to be enabled well after the reference clock is provided to the delay line. Otherwise a long false UP or DOWN

pulse puts the lbz-PFD in the wrong state and the loop will never lock. Note that even before loop starts to work the delay line is working and the input clock passes to the final delay cell. The startup process is discussed in detail at the end of this section.

The single ended switched inverter PI is chosen for its simplicity, good linearity and fast response time. However, linearity would suffer when the input phases are more than 100ps apart. For a target output frequency of 5GHz, output phases of the delay line are 200ps apart. A simple solution is to make each delay cell from two identical buffered delay elements and use the intermediate output for interpolation. Figure 5.7.11 shows this implementation. Figure 5.7.12 and Figure 5.7.13 show the linearity curves for PI with two 200ps apart input phases and three 100ps apart input phases, respectively. The decoder for this implementation is similar to what is used for conventional 4 phase interpolators. It has a control bit to choose the segment (phases to interpolate) and some other bits (five in this design) to control the phase interpolation inside each segment. Decoder input and output are shown in Figure 5.7.14. The output of the PI goes to a period synthesis delay line, which is identical to the one in the master loop and also shares the same control voltages. The outputs of period synthesis delay line are connected to a MISO switch which selects the appropriate phases in each period. The MISO switch and PI are working together to make the period synthesis happen. Depending upon the synthesizer target multiplication factor, the PI and MISO switch may step forward or backward in the phase (delay) to generate a period larger or smaller than the reference period, respectively.

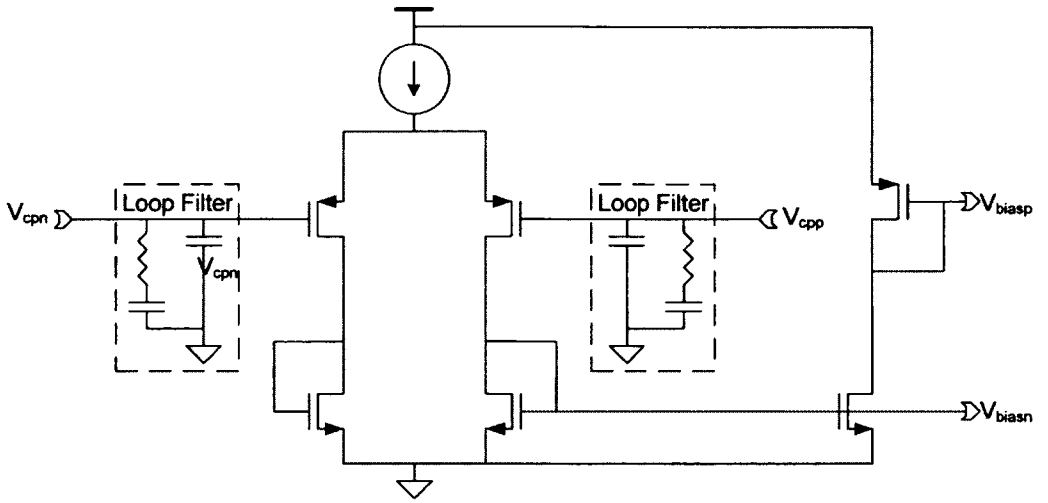


Figure 5.7.10: Voltage to current converter (V2IC) gets the CP voltages and generates control bias for delay cells

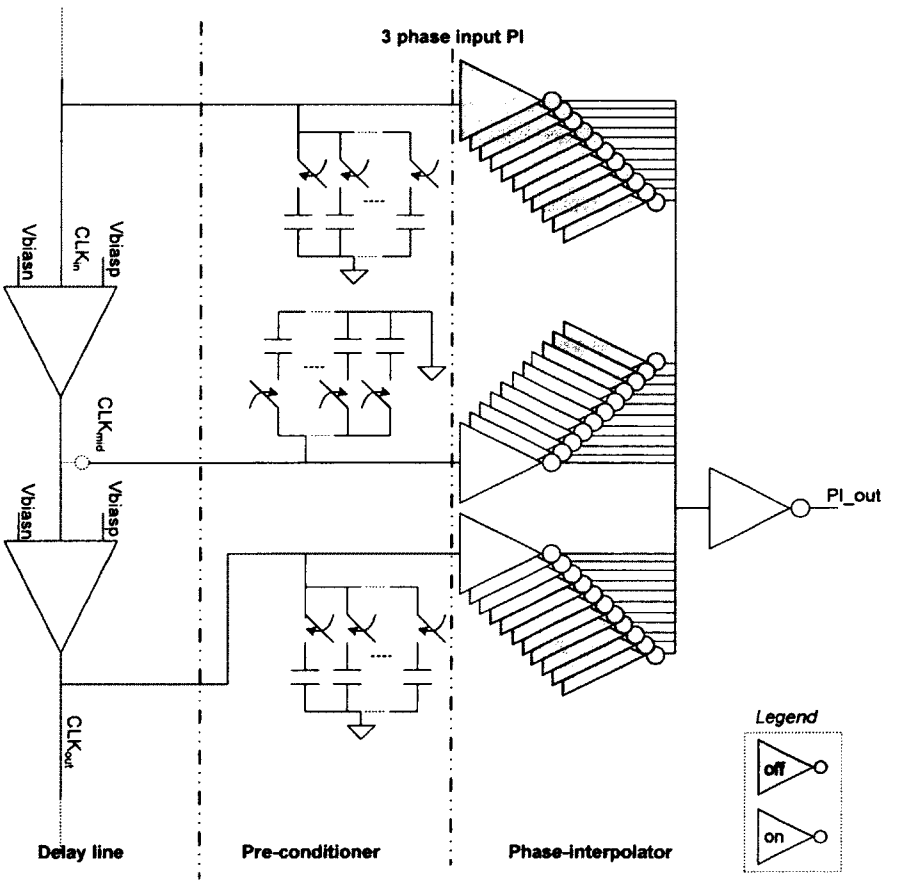


Figure 5.7.11: Implemented buffered delay line and 3 input PI

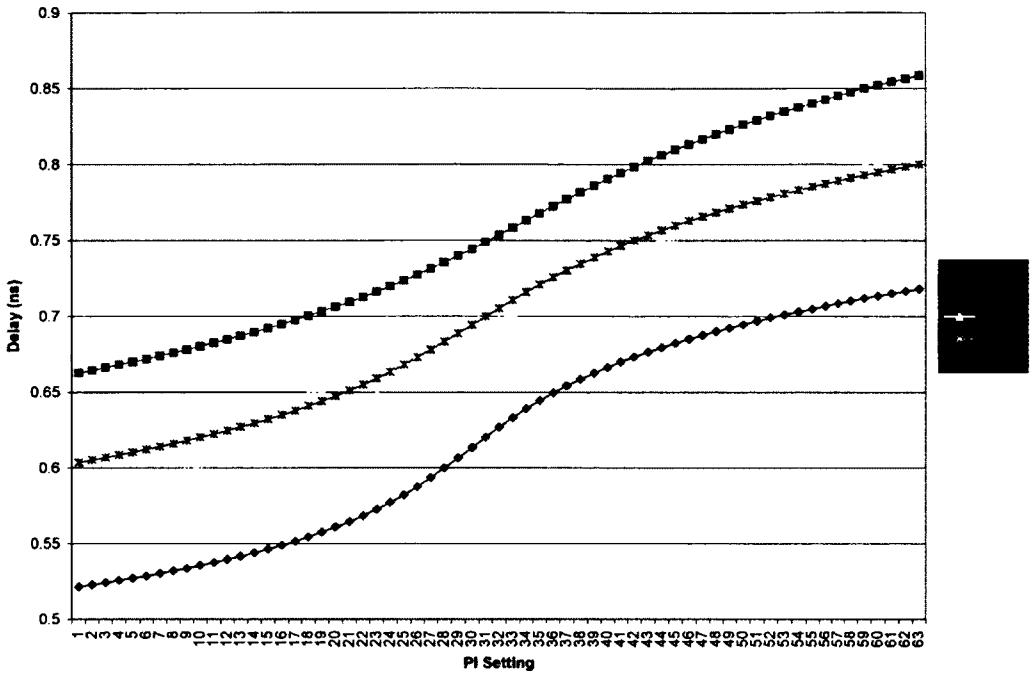


Figure 5.7.12: Phase (delay) curve for two input PI and 200ns delay difference of inputs

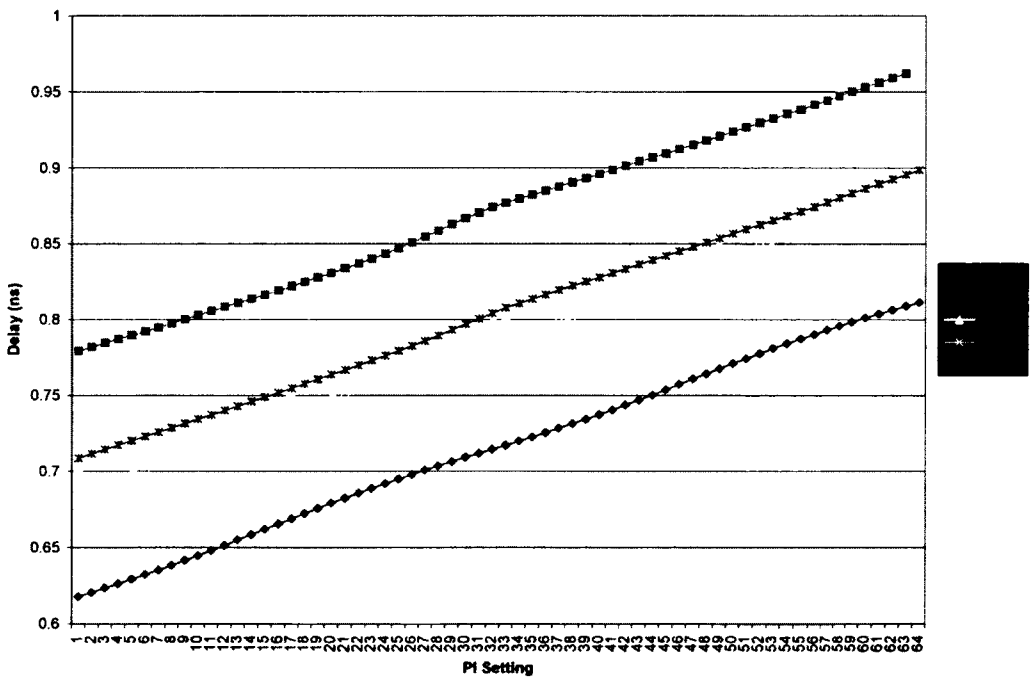


Figure 5.7.13: Phase (delay) curve for three input PI and 100ns delay between each two inputs (total 200ns delay)

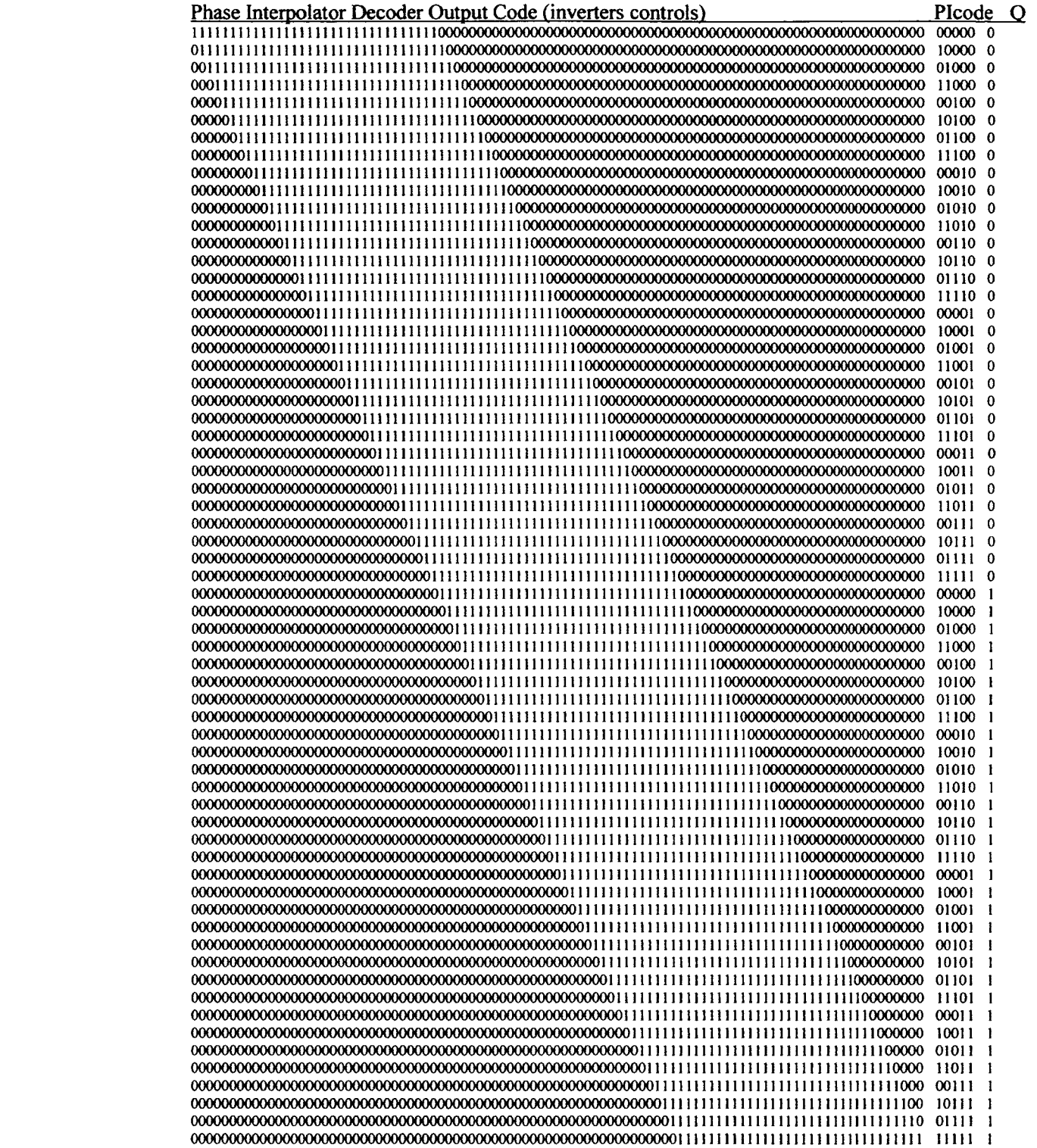


Figure 5.7.14: Three input PI decoder input and output; Q selects the segment and PIcode adjusts the output phase in each segment; each output bit controls one inverter

5.7.1 Switching schemes for period Synthesis

Two switching schemes for period synthesis explained in chapter four. The first scheme is less complex but results in a duty cycle distortion up to $\alpha.t_d'/T_{new}$, where α is the fractional part of the multiplication factor, t_d' is the delay of one delay cell (controlled by Master loop) and T_{new} is synthesized period (period of new generated reference clock). Duty cycle distortion of the new reference does not affect the final clock output if the Local edge combiner is used for frequency multiplication. When using Global edge combiner for frequency multiplication, duty cycle of new reference will induce timing error. This timing error appears as duty cycle distortion for small values of α and may cause cycle slipping when α has a value close to 1. Although a second switching scheme for period synthesis was introduced to resolve the duty cycle distortion problem for the Global edge combiner, the first scheme is chosen for this implementation mainly for its simplicity. By using the Local edge combiner the duty cycle resulting from this simpler scheme doesn't affect the performance of the synthesizer. Both schemes involve a careful timing to avoid a glitch when MISO switches phases. Switching must happen at the moment that both MISO switch outputs (initial and the one it will switch to) have the same state (either high or low).

Block diagrams of the digital control circuitry for backward and forward stepping are shown in Figures 5.7.15 and 5.7.16. The fractional part of the multiplication factor determines the input bits of the first accumulator which is sized equal to number of states in one segment of the PI. The number of the states in one segment of PI (or size of first accumulator) multiplied by the fractional part of multiplication factor (α) makes the input of the first accumulator. The output of the first accumulator is the PI code and its carry

out goes to a one bit accumulator for PI segment selection. The PI interpolates between phase 0 and phase 1 in the first segment (when Q is 0) and use phases 1 and 2 when the second segment is chosen (Q is 1). Both of these accumulators are clocked with the right phase of reference clock; i.e. falling edge of the third delay element of master loop delay line, to insure all PI input phases are grounded (not shown in the Figure 5.7.16 for simplicity and readability). When both the PI and segment accumulator carry outs are high, the MISO switch is adjusted to the next phase. An important and delicate timing issue here is that the MISO switch needs to be clocked with a new reference period to avoid glitching. This causes a cross clocking domain issue. Usually cross clocking issues like this are solved with a FIFO (first in first out) chain of registers but a simpler approach is used for this implementation. The period synthesis control circuitry shown in Figure 5.7.16 uses a Shift Register to control the period synthesis MISO switch. This shift register has a length equal to delay line (5 bits long for 5 stages delay line). The goal is to switch from N_{th} output of the MISO switch to $(N+1)_{th}$ output every time the one bit accumulator overflows. However, to avoid generating a glitch, switching must happen when both N_{th} and $(N+1)_{th}$ delay outputs have the same state (either high or low). This puts a stringent timing requirement on shift register and MISO switching time. Shift register control logic must distinguish between backward and forward switching schemes. Switching backward in phase is required to generate clock periods shorter than the reference period. Naturally in backward switching both PI code and MISO switch steps backward, i.e. PI code is decrements and MISO switches from N_{th} to $(N-1)_{th}$ delay line outputs.

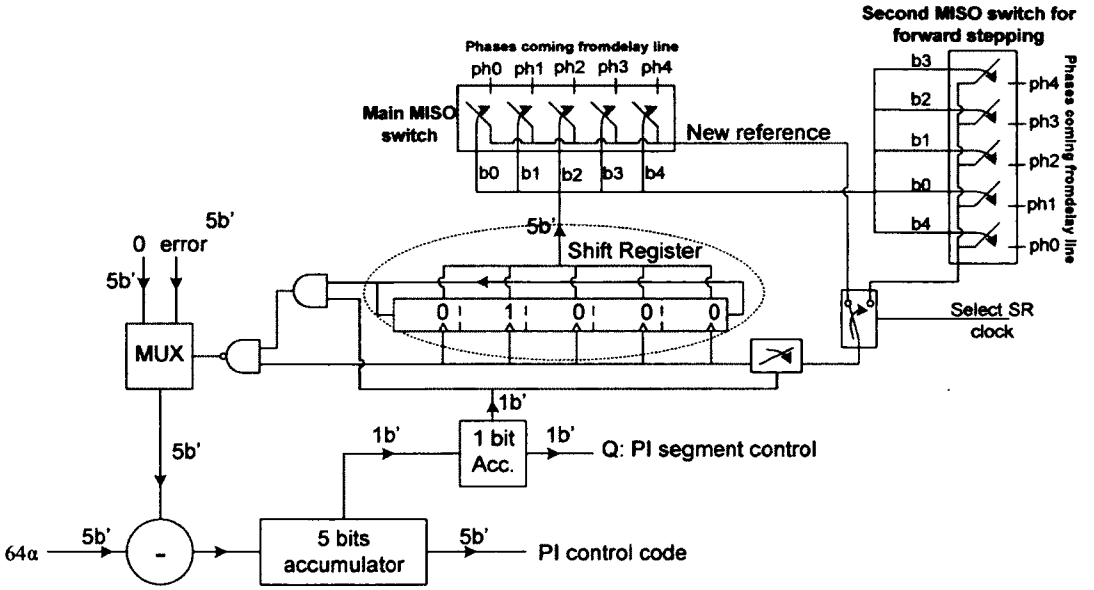


Figure 5.7.15: Switching scheme for backward stepping period synthesis

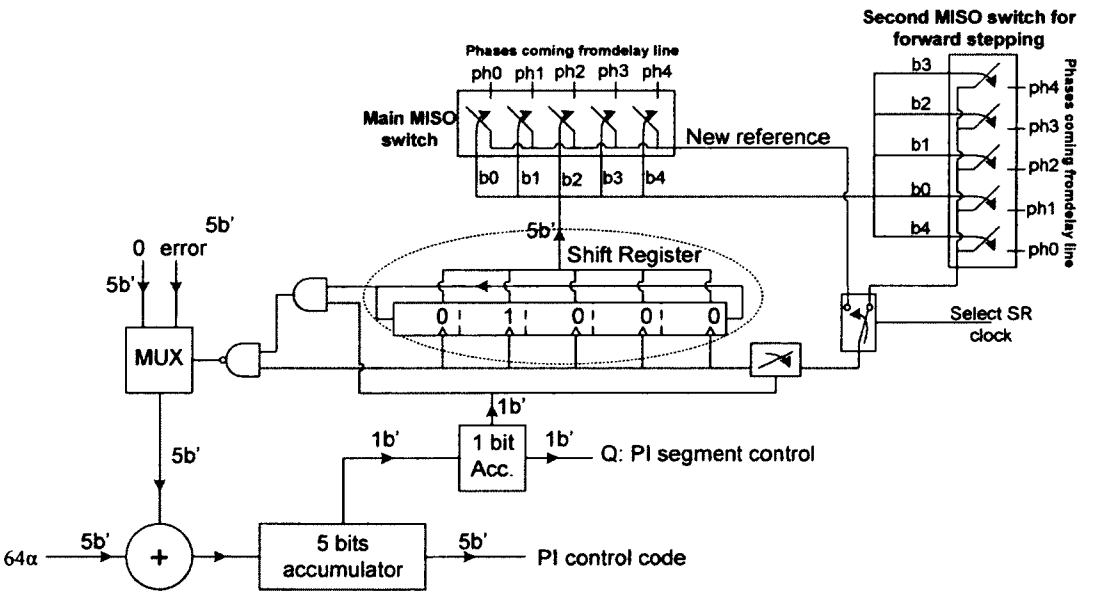


Figure 5.7.16: Switching scheme for implemented forward stepping period synthesis

Switching forward in phase is necessary to generate clocks with periods longer than the reference period. Table 5.7.1 gives a summary of the required new clock period and stepping style based on the value of the multiplication factor.

In forward stepping, the shift register is clocked by $(N+1)_{th}$ delay cell output. To access to right $(N+1)_{th}$ delay cell output another MISO switch is used with an incremental connection order as it shows in Figure 5.7.15. The backward stepping must use new reference clock to clock MISO switch. Switching between the backward and forward stepping is implemented by a switch which changes the source of the shift register clock. In forward switching, switching to last (fifth in this implementation) delay cell needs to delay by one clock cycle since $T_{new} > T_{ref}$,. This is implemented with an extra flip-flop and switch (not shown in Figure 5.7.15). In the event of PI accumulator overflow the timing aperture for the MISO to switch for both forward switching and backward switching is $W_{sw}=T/2-2*t_d'$ as shown in Figure 5.7.18. Switching window (W_{sw}) is t_d' smaller when a PI and a MISO switch are used together compare to other period synthesis schemes introduced in Chapter 4 which use only a MISO switch. For some values of T and t_d', the switching window (W_{sw}) can be too small compared to shift register and MISO switch delay. For example at 1GHz reference and 5 delay line stages, W_{sw} is only 100ps. In such cases a better choice would be to use the delay between adjacent delay cells as the time that both outputs have the same state. This yields an alternate switching window of $W_{sw}= t_d'$. In our example t_d'=200ps which is tow times larger than the original W_{sw}.

Table 5.7.1: Summary table of relationship of new synthesized period and reference period with multiplication factor and delay/phase stepping style

Fractional-N Synthesizer Multiplication Factor (f)	Period of new clock (T_{new})	Required Stepping
N<f<N+1	$T_{new}<T_{ref}$	Backward Stepping
N-1<f<N	$T_{new}>T_{ref}$	Forward Stepping

Naturally after both the 5 bits PI code accumulator and the 1 bit segment accumulator overflow, the MISO switch switches to the next phase, and both of the accumulators would have correct values. Only when the MISO switch jumps from the latest phase to the first phase, $N.t_d'$ delay change suddenly happens. This induces a timing error of $\alpha.t_d'$ = $N.t_d' - T_{ref}$ where N is the number of delay cells jump (equal to number of delay cells plus one to include the PI reset), t_d' is delay of one delay cell and T_{ref} is the period of the reference clock. Since the PI induces phase step of $\alpha.t_d'$ in each cycle, adding this error is equivalent to the PI taking two steps when the MISO switches back to first delay cell. Note that the step size depends on accumulator input and therefore on α, the fractional part of multiplication factor.

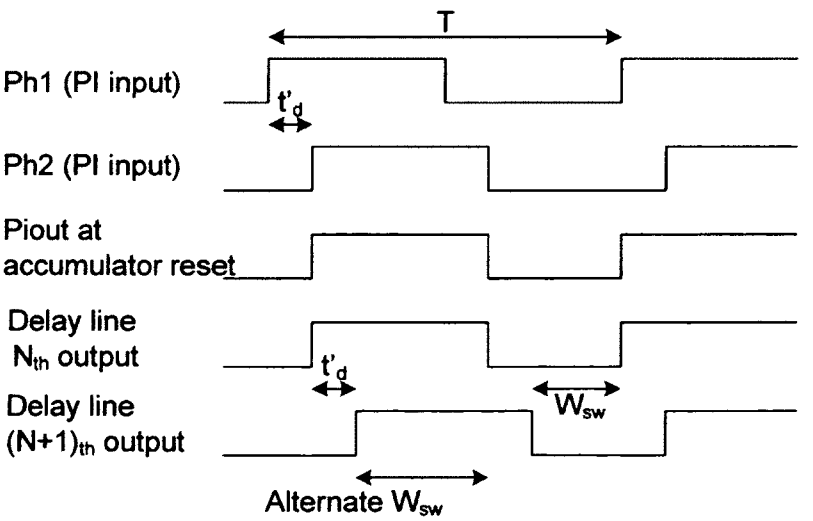


Figure 5.7.17: Implemented buffered delay line and 3 input PI

5.7.2 Startup calibration process

As shown in the results presented through Chapter 5, PVT corners affect the operational point and/or characteristic curves of the synthesizer building blocks. In some cases like the PI, the characteristic change is in within the acceptable range. In other extreme case, the delay of the DL (delay line) affected by process variation so adversely that DLL doesn't have enough range to compensate it. The startup calibration process, active control loops, extra circuitry and control knobs required to compensate for variations due to PVT changes are described in this sub-section.

5.7.2.1 Delay line PVT and mismatch calibration

The delay of one delay cell in this design can be controlled with two distinct mechanisms. One is the current provided to delay cell by the voltage to current converter (V2IC) which is controlled by the loop. The other is a switched capacitor controlled by a digital control system. The switch capacitor is used to calibrate the delay line at startup. This startup calibration has two parts. First part addresses the process variation and second part addresses the mismatch between delay cells. The startup calibration loop for the PVT calibration consists of a digital state machine, switch capacitor load and in-lock-range-detector. The in-lock-range-detector (explained in Chapter 2) is a digital circuitry which shows if DLL is in lock range. The charge pump is off for calibration and both of its outputs are charged to half supply voltage. In this situation the V2IC provides the nominal current to delay cells which centers delay cells delay in the middle of the control range. The digital state machine starts by switching off all the capacitors, which is equivalent to the setting for minimum delay. Then it starts to turn on the capacitors depending on the in-lock-range-detector output. Calibration is completed when the delay

line gets into the lock range. Note that this startup calibration only compensates for process variation. Any change in voltage and temperature during synthesizer operation is compensated by the loop. Second part of the calibration performed to remove/reduce delay mismatch of synthesizer delay cells. Mismatch calibration must be performed on all three delay lines in the synthesizer. Mismatch calibration loop consists of a MISO switch at the input of delay line, a second MISO switch at the output of delay line, a phase detector, a reference delay cell and a digital state machine as shown in Figure 5.7.18. Reference clock feeds to the reference delay cell and one selected delay cell inside the delay line through the input and output MISO switches. A bang-bang (binary) type phase detector shows the polarity of the phase difference. Here a D-flip-flop is used as bang-bang phase detector. Each delay cell has two banks of switch capacitors, global switch capacitors whose control are shared with all other delay cells in the synthesizer, and local switch capacitors which have individual controls. The global switch capacitors are 10 times larger than local switch capacitors in this design and used in first part of startup calibration for process compensation. Local/individual switch capacitors have smaller capacitor value and are used in second part of startup calibration for mismatch compensation. The calibration process starts when half of the local switch capacitors are on (caps are connected). If the D-ff output is constantly one/zero, then the state-machine increases/decreases the number of ON switches. The process continues until D-ff output switches state. If the D-ff output switches back and forth between 1 and 0, then two delays are completely matched and calibration is complete. To differentiate constant 1 or 0 output from changing output, D-ff output should be stored for a predetermined number of cycles. The smallest switch capacitor changes the delay of a delay cell around 2ps.

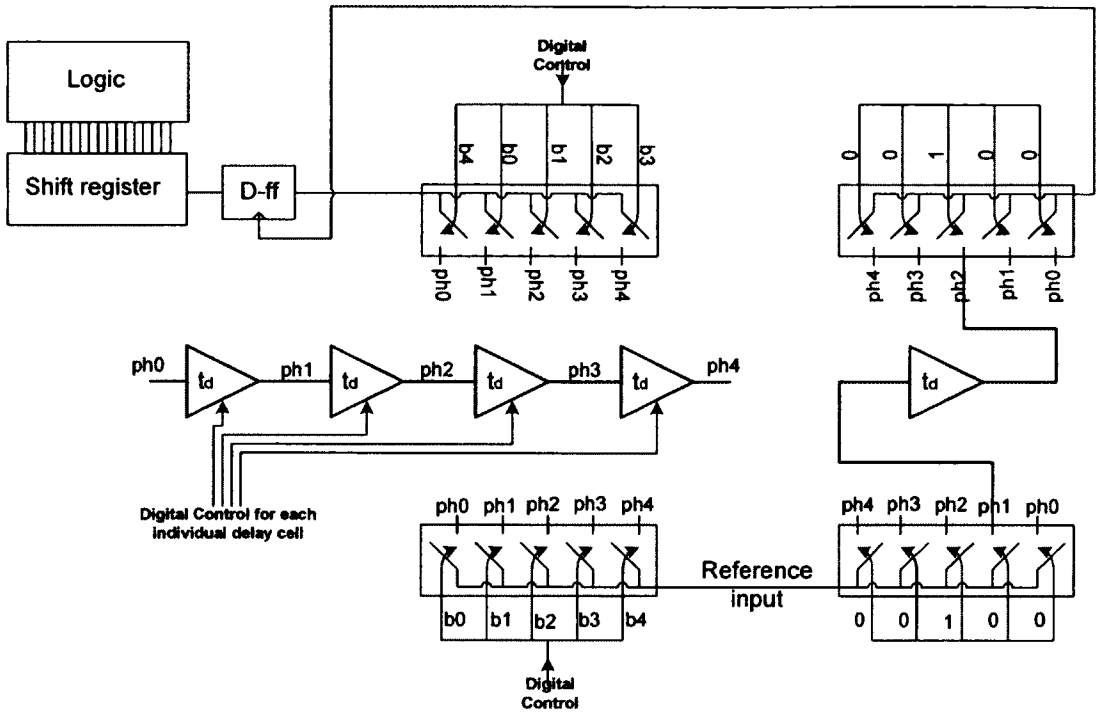


Figure 5.7.18: Simple schematic of delay line mismatch calibration scheme

Considering clock to data skew in D-ff and other sources of the mismatch, the delay can be calibrated to have under 3.5ps mismatch.

5.7.2.2 Charge pump calibration

The charge pump has two analog loops which compensate for both PVT variation and local mismatch variation. At startup both outputs are charged to half supply voltage before the charge pump starts to operate.

5.7.2.3 Phase detector at startup

Both inputs of the phase detector are driven by digitally controlled MISO switches. To avoid false long UP or DOWN signals at startup, both MISO switches are enabled when the DL output and reference clock are both high.

5.7.2.4 Shift register

The shift register must reset at startup in such a way that only one of the registers has a logical value of 1 and the rest have a logical value of 0.

5.8 Conclusion

The circuit implementation of the DLL-based fractional-N synthesizer involves many challenges such as finding the proper topology for each block, designing, and fine tuning the blocks for optimum performance. Chapter 5 explains the design procedure for the proposed DLL-based fractional-N architecture. Various topologies for each circuit block are investigated and their impact on the synthesizer performance is compared. Candidate topologies for each block and associated design challenges explained in more details. The simulation results presented to better describe the circuit behavior.

Both the single ended and differential topologies for the delay line and techniques of controlling delay have been investigated. Delay line behaviors which affect the synthesizer performance such as delay controllability, power supply sensitivity, and sensitivity to local device variation have been presented with associated circuit level simulations. A delay cell using both current and switched capacitor delay control techniques has been chosen as it shows the best power supply sensitivity while generating a rail to rail clock output.

Comparison of four well known topologies for PFD shows that lbz-PFD is a good candidate for the proposed synthesizer. Some issues similar to the blind zone problem appear in a ΔΣ-DLL when a fractional-N multiplication number close to an integer is

targeted. This half blind zone (as PFD would only fail to send signal when clock edge close to reference is sent to PFD) can be avoided with a lbz-PFD.

Investigation on three single ended CP topologies and associated techniques to reduce mismatch pushed the design toward a differential topology with two compensation loop. Compensation loops use a fully complementary folded-cascode amplifier to achieve the required gain in deep sub-micron technology. The CP works in association with the loop filter and voltage to current convertor to generate the delay line control signal. This combination has a low power supply sensitivity which is an essential factor to reduce total jitter.

Three topologies for phase interpolator (PI) have been investigated, two single ended and one differential. The single ended PI is used for better matching with the single ended delay line topology and to avoid single ended to differential conversion. The switched inverter single ended PI topology shows a better DNL/INL performance and has chosen for this implementation.

Many hard design decisions were made to match these building blocks together; many were specific to the target application. Design decisions, timing issues of the period synthesis, and the startup calibration procedure have also been covered in this chapter.

Chapter 6 Results

Simulation results for the implemented single loop DLL-based fractional-N frequency synthesizer are presented in this chapter and the main performance factors are measured. The synthesizer can generate an output frequency between 4 to 5GHz using a 1GHz input reference clock (fractional multiplication factor between 4 and 5). Output jitter depends on the fractional part of the multiplication factor and the power supply noise. The measured output jitter is less than 15ps across the range of multiplication factors with a typical power supply noise of 10mV. Jitter contribution of synthesizer sub-blocks is measured and compared in this chapter. The master loop, period synthesis and frequency multiplier are characterized in the first three sections. This chapter concludes with the synthesizer performance matrix including the jitter, power supply sensitivity (PSS), power consumption, locking time, frequency resolution and the output frequency range.

6.1 Master loop characterization

Figure 6.1.1 shows master loop of the proposed single loop DLL-based fractional-N synthesizer. A $\Delta\Sigma$ modulator controls which of the delay line outputs is used for comparison in the phase frequency detector (PFD). Charge pump (CP), loop filter (LF) and voltage to current converter (V2IC) provide the control signal for the delay line (DL). Delay cells also have a course digitally controlled tuning switch capacitor to compensate for the process corner variations. Figure 6.1.2 shows the delay of one delay cell versus time for a 1GHz (1ns period) input reference clock and multiplication factors of 5.000,

4.844 and 4.625. The synthesizer loop is first locked to an integer multiplication factor of 5, and then to the fractional-N multiplication factors of 4.844 and 4.625, respectively. The measured initial integer lock time is 250ns, with less than 100ps delay to converge to each fractional multiplication factor. Figures 6.1.3 and 6.1.4 are zoomed in on the steady state part of each fractional multiplication factor to show jitter due to $\Delta\sum$ switching. The CP output voltages (V_{cpp} and V_{cpn}) and the input voltage of the V2IC (loop filter output) are shown in Figures 6.1.5 and 6.1.6, respectively. It can be observed that the loop filter reduces the quantization noise of the $\Delta\sum$ modulator by almost an order of magnitude. A high loop bandwidth is used to achieve a faster locking time and a shorter simulation time. However, the loop response is damped and there is absolutely no overshoot. The Master loop delay line $\Delta\sum$ modulator induced jitter is between 2ps to 3ps for different fractional multiplication factors and voltage and temperature corners. Figure 6.1.7 shows the delay change trend for different target multiplication factors. The synthesizer always initialized by locking to the integer multiplication factor (five in this design) before choosing a fractional multiplication factor. The power supply sensitivity (PSS) of the Master loop has been tested by applying 85MHz sinusoidal noise with amplitude of 10mV. The frequency and amplitude of the power supply noise is chosen as an estimate for an average size package resonance frequency and the worse case regulator output noise in the current technologies [76,77]. The delay of one delay cell versus time for different fractional multiplication factors in the present of noise is shown in Figure 6.1.8. The $\Delta\sum$ modulator induced jitter is more severe than the power supply noise induced jitter in this design.

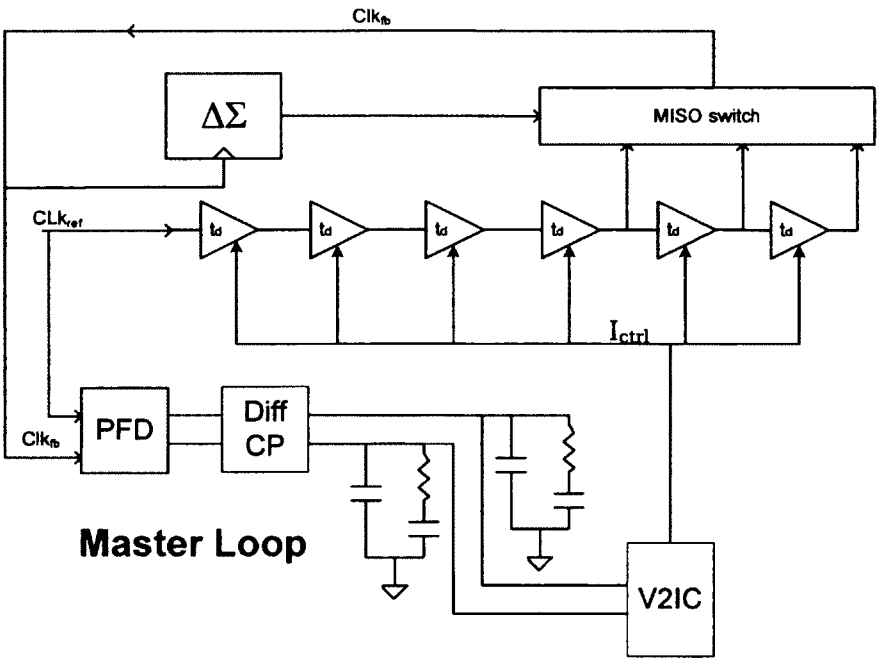


Figure 6.1.1: Schematic of Master loop

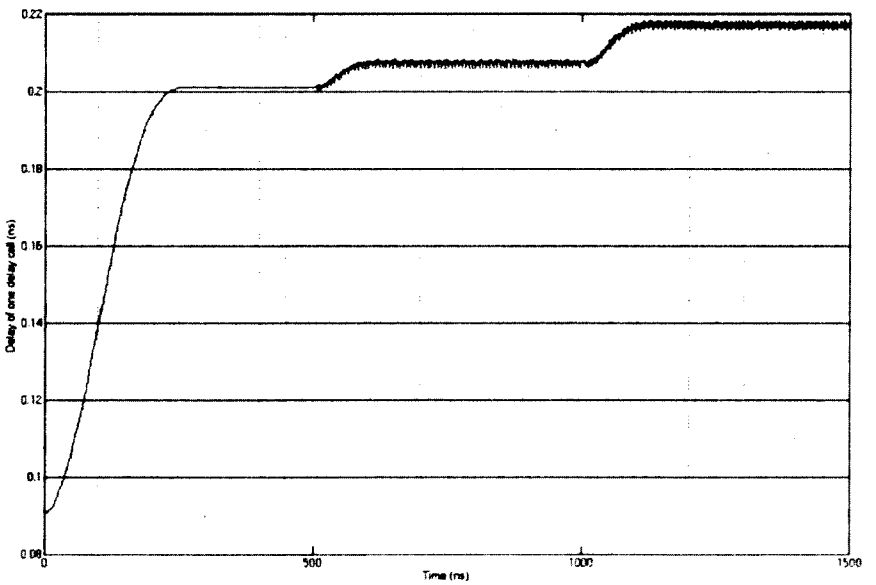


Figure 6.1.2: Delay of one delay cell inside Mater loop delay line versus time; loop locks with integer multiplication number 5 and then after each 500ns multiplication number changes to 4.844 and then 4.625, respectively.

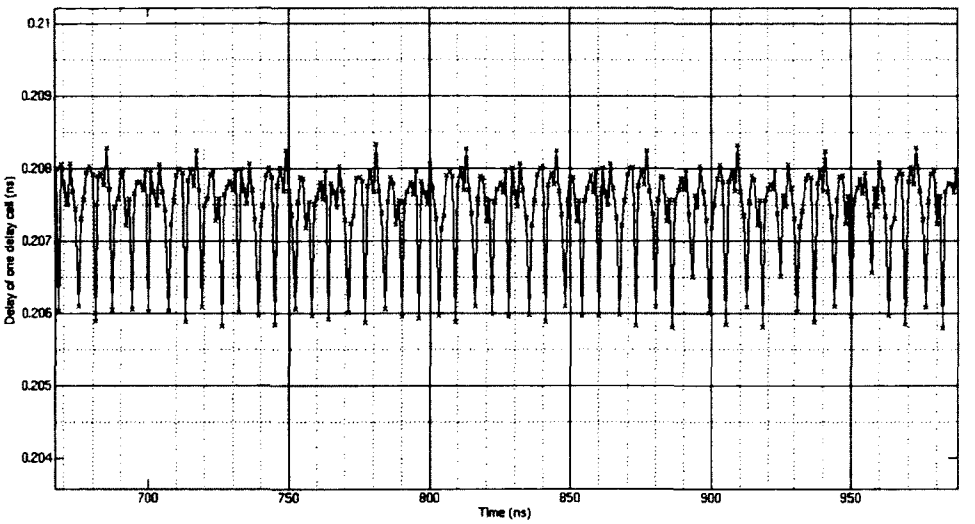


Figure 6.1.3: Delay of one delay cell versus time for multiplication factor of 4.844

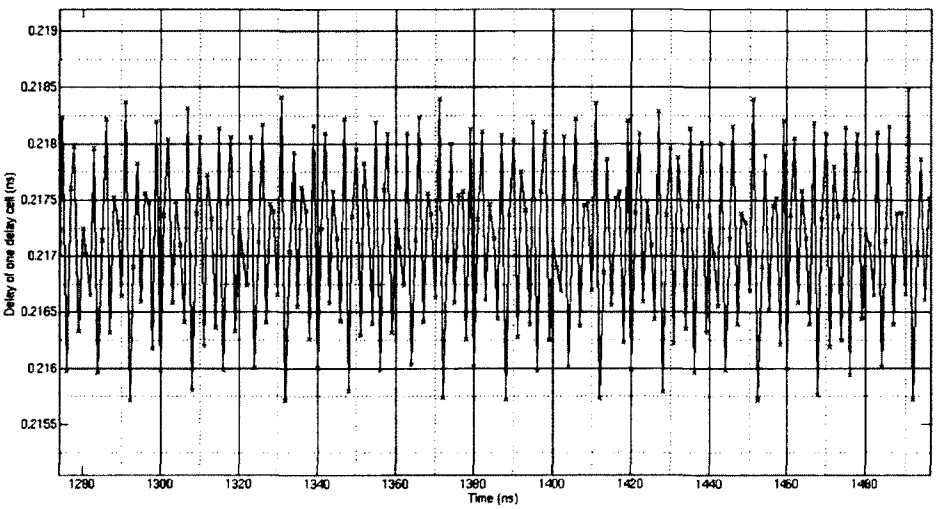


Figure 6.1.4: Delay of one delay cell versus time for multiplication factor of 4.625

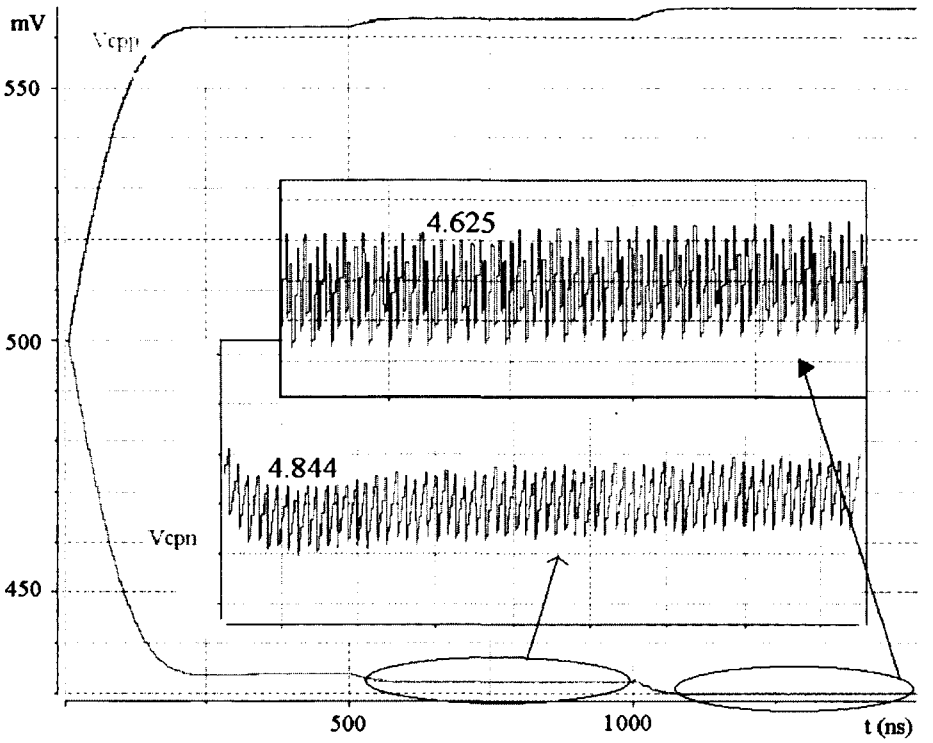


Figure 6.1.5: Charge pump output versus time when loop locks with integer multiplication factor of 5 and then after each 500ns multiplication factor changes to 4.844 and then 4.625, respectively.

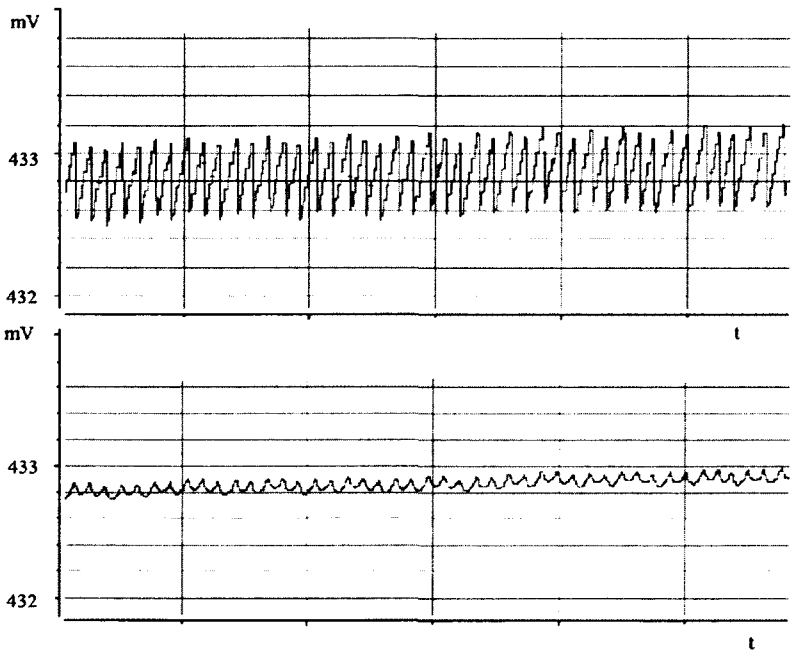


Figure 6.1.6: Charge pump output [Top waveform] and loop filter output (V2IC input) [bottom waveform] versus time for fractional multiplication factor 4.844

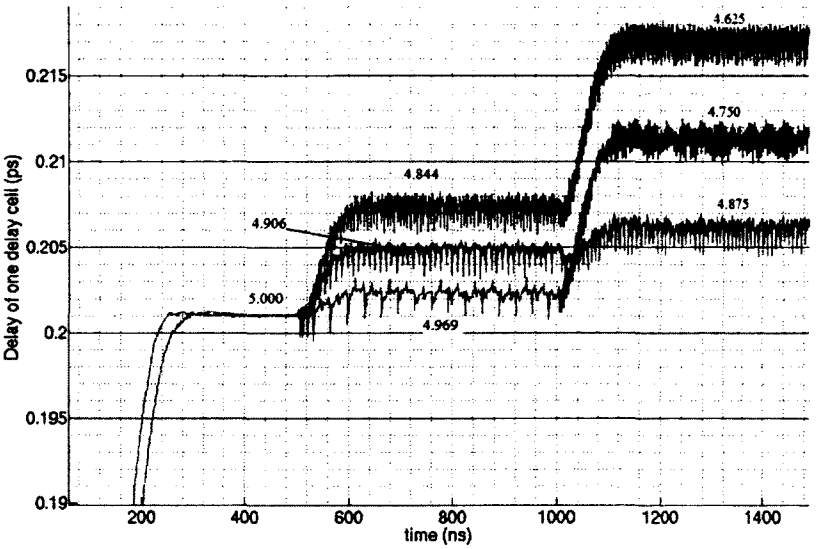


Figure 6.1.7: Trend of delay change for different values of multiplication factors

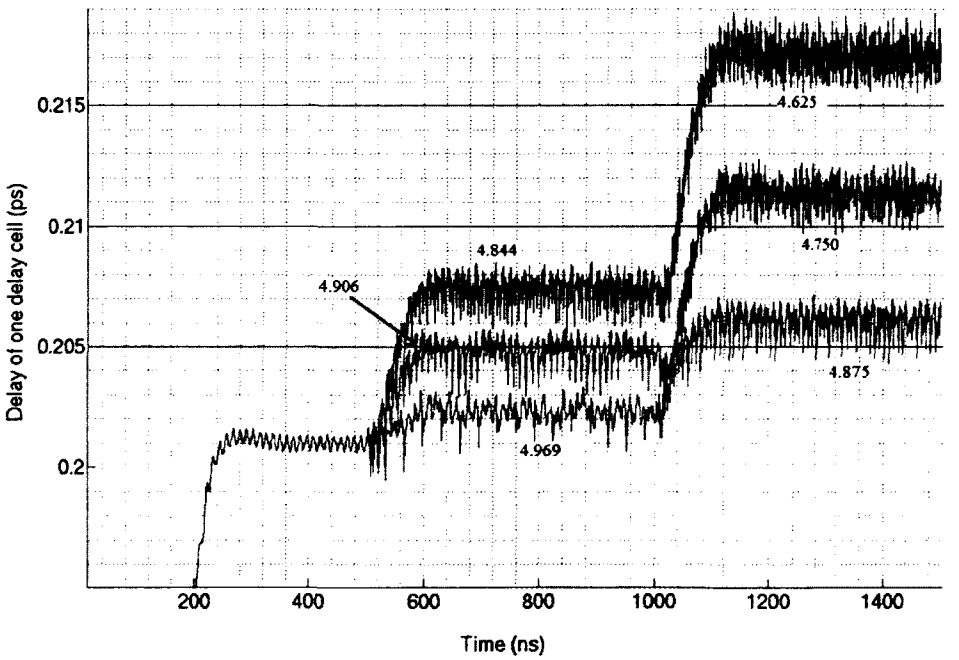


Figure 6.1.8: Delay of one delay cell inside Mater loop delay line versus time at present of power supply noise; first loop locks to integer multiplication number of 5 and then after each 500ns multiplication number changes to a fractional multiplication factor

6.2 Period Synthesis

Figure 6.2.1 shows a simplified schematic of the period synthesis circuit. The period synthesis technique is explained in sections 3.4.2 and 4.4. A phase interpolator is used to change the period of the reference input and a MISO switch and a DL are used to cover the span of one reference cycle. A phase interpolator (PI) provides reference period increment and/or decrement for fractional numbers around an integer which wasn't practical with any other techniques described in Chapter 4. As explained in Chapters 4 and 5, required change in reference period is equal to $\alpha.t'_d$, where α is the fractional part of the multiplication factor and t'_d is the delay of one delay cell when the same multiplication factor is enforced by the master loop. This equation is the basis of the implemented period synthesis circuit which interpolates between the input and output of a delay cell with the delay/phase difference of t'_d to generate the required change in the reference period. PI control bits have been retimed by the proper phase of the reference clock coming from master delay line (see section 5.7.1 for more detail). For small values of $|\alpha|$, PI delay error is on order of its differential nonlinearity (DNL) but for a large amount of α delay error can be as high as PI's integral nonlinearity (INL). Note that for each reported multiplication factor α is negative and $|\alpha|$ can be calculated as $|\alpha|=5-f$, where f is the fractional multiplication factor. The output period of PI versus time for different values of fractional multiplication factors are shown in Figure 6.2.2. PI accumulator overflow could cause a large phase error. However, the MISO switches to the next delay line upon overflow to prevent the phase error. The rate of PI overflow error depends on the value of $|\alpha|$, the fractional part of the multiplication factor.

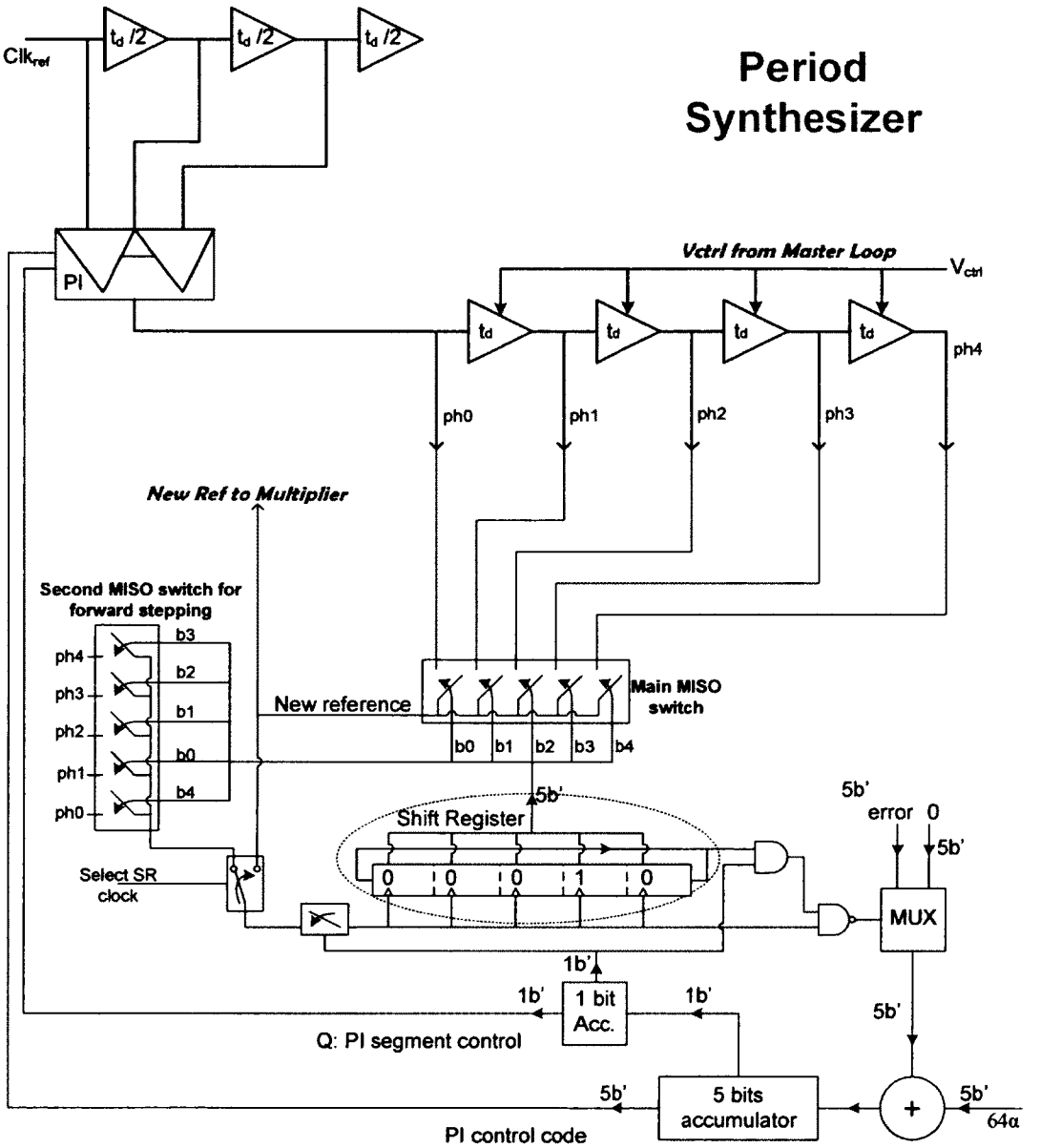
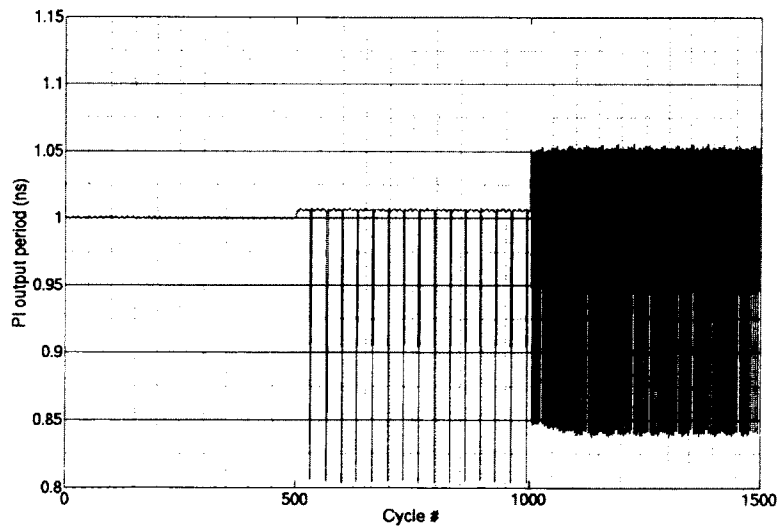
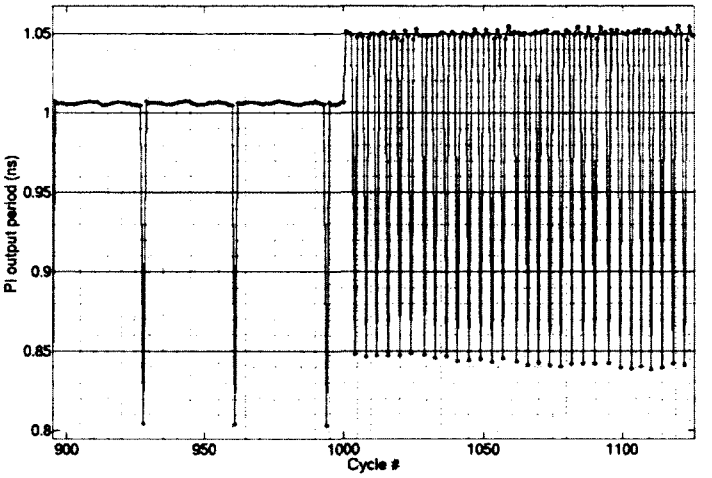


Figure 6.2.1: Schematic of Period Synthesis

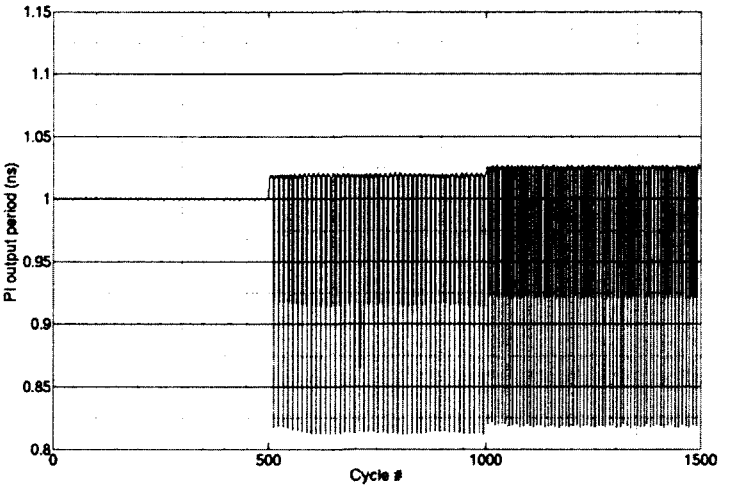
The output period of the period synthesis for different values of fractional multiplication factors are shown in Figure 6.2.3. The period variation is a result of the master loop's $\Delta\sum$ modulator quantization noise and the phase interpolator error. PI timing error is bounded by its DNL and INL depends on the absolute value of $|\alpha|$. For large values of $|\alpha|$, PI error is close to INL and is the dominant source of error. There are three methods to reduce the impact of PI INL error. First, inverters are weighted inside the PI to reduce its INL. Second, larger MISO switches are used to allow for extra delay line taps for the period synthesis (ten-in-one-out MISO and using all available phases of the delay line). A combination of forward and backward phase stepping in the PI and MISO switch reduces the required PI phase step by a quarter equivalents to maximum α of 0.25 in this implementation. Third, a second loop is implemented to control delay of the period synthesizer delay line (as it explained in Chapter 4) for large values of α.



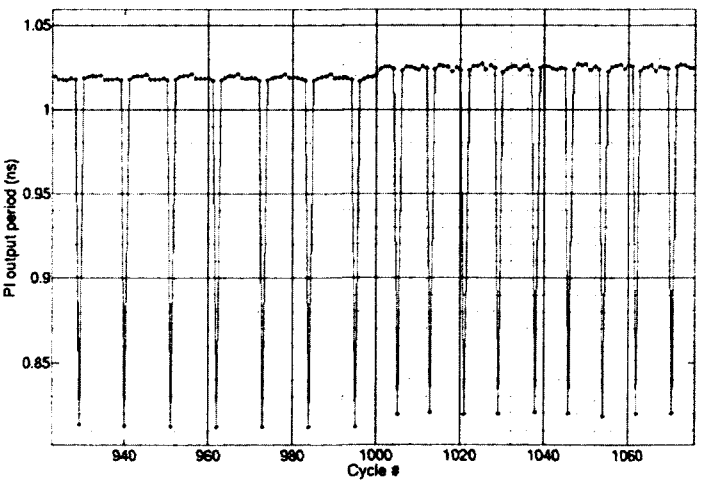
(a-up)


PI output period (ns)
Cycle #
1.05
1
0.95
0.9
0.85
0.8
900
950
1000
1050
1100

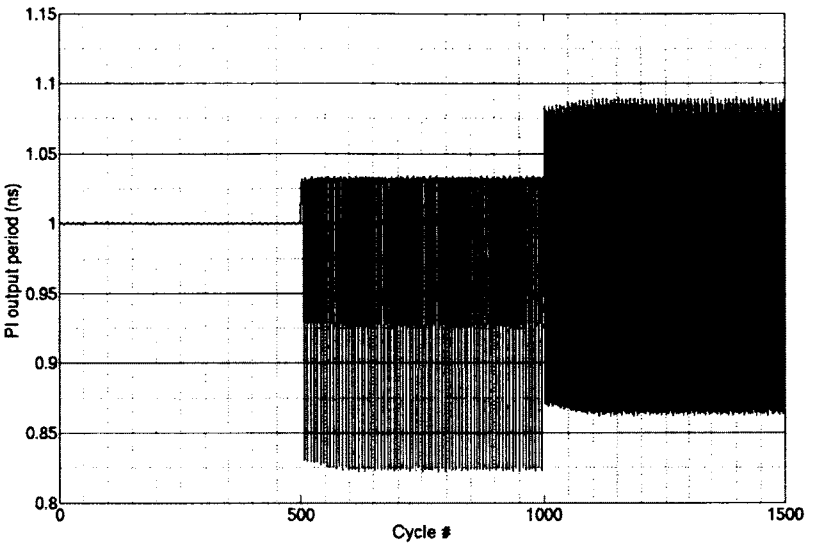
(a-dn)


PI output period (ns)
Cycle #
1.15
1.1
1.05
1
0.95
0.9
0.85
0.8
0
500
1000
1500

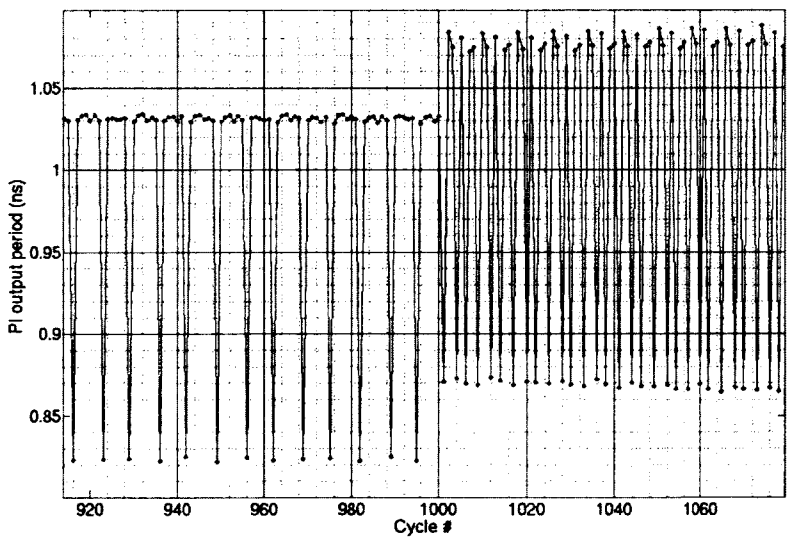
(b-up)


PI output period (ns)
Cycle #
1.05
1
0.95
0.9
0.85
940
960
980
1000
1020
1040
1060

(b-dn)

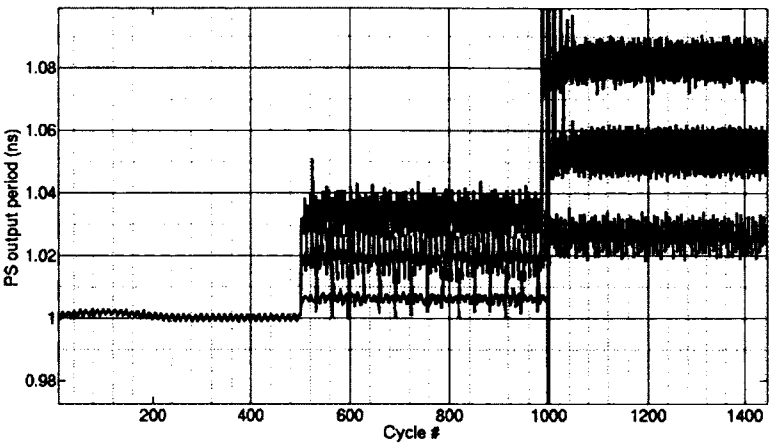


(c-up)

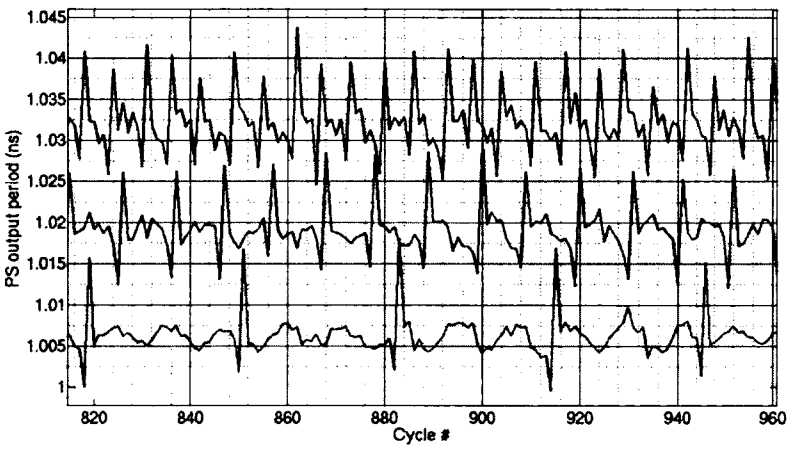


(c-dn)

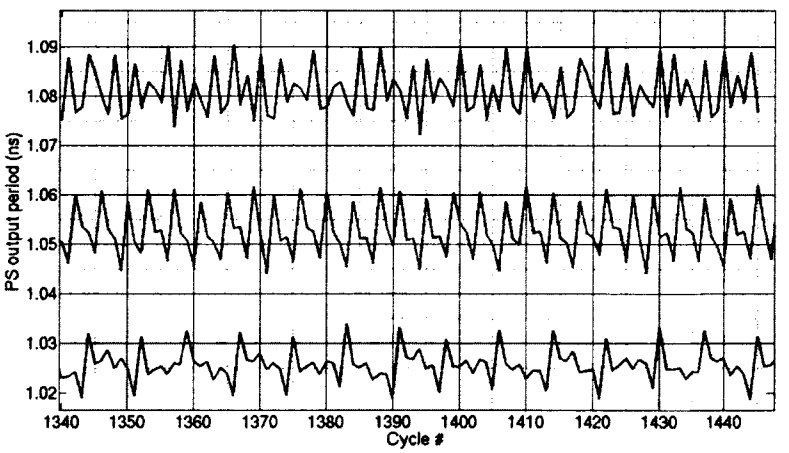
Figure 6.2.2: Output period of PI for fractional multiplication factors of a) 5, 4.969, 4.750 b) 5, 4.906, 4.875 c) 5, 4.844, 4.625 For full range (up figure) and zoom in to where fractional multiplication factor changes (down figure)



(a)



(b)



(c)

Figure 6.2.3: Period synthesis output a) for multiplication factor of 5, 4.969, 4.906, 4.875, 4.844, 4.750, and 4.625 b and c) zoom in at fractional multiplication factor parts for jitter comparison

6.3 Frequency Multiplication

Frequency multiplication is implemented by using a Local edge combiner. The Local edge combiner (explained in section 2.3.4.2) is chosen for its low sensitivity to input clock duty cycle distortion. The period synthesis introduces a systematic duty cycle distortion and thus using a Local edge combiner is unavoidable. The Local edge combiner circuit is shown in Figure 6.3.1. This Local edge combiner is designed to have maximum symmetry and speed. It combines two consecutive DL phases to generate one output clock cycle. The period versus the time for different multiplication factors are shown in Figure 6.3.2 and Figure 6.3.3. Every other delay line output are inverted since the delay cells are noninverting (Figure 6.3.4). As a result the output waveform wouldn't have a perfect duty cycle.

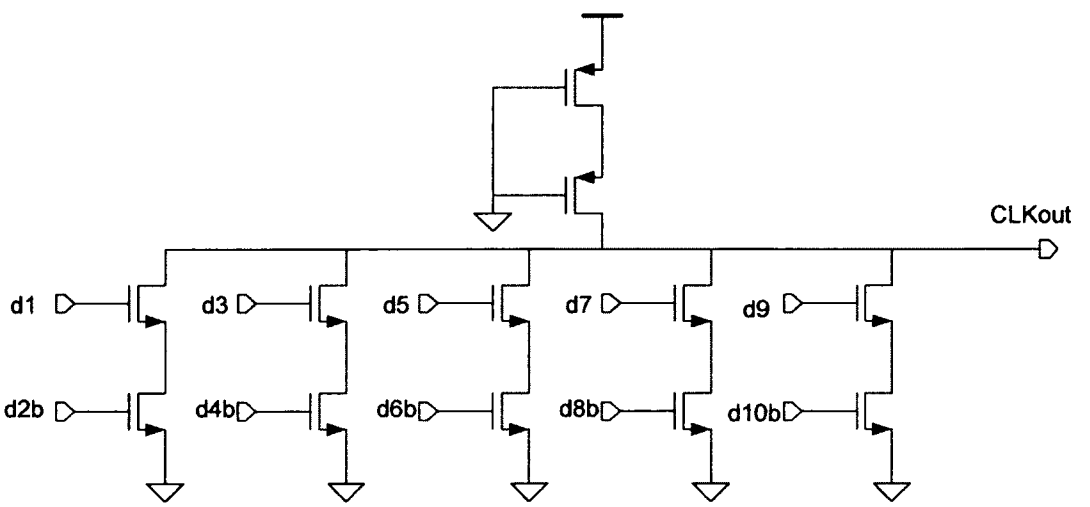
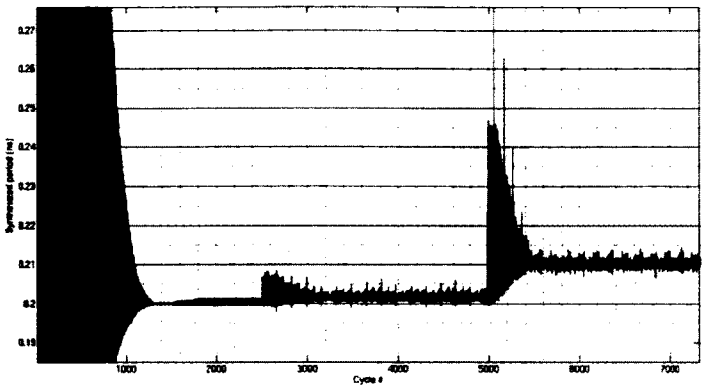
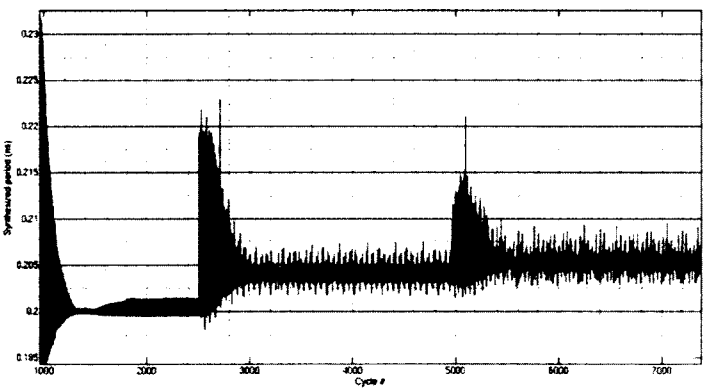


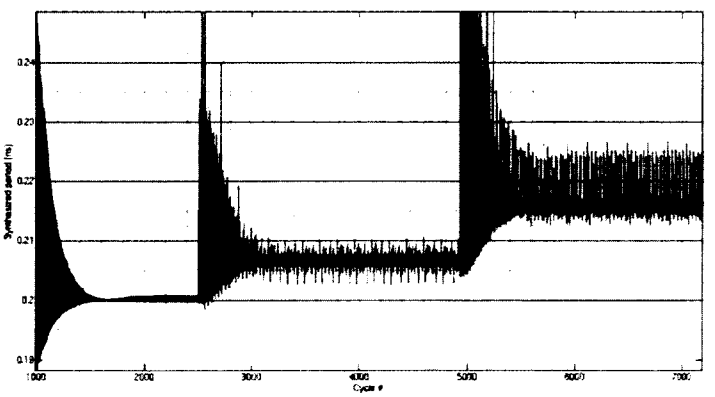
Figure 6.3.1: Schematic of Local edge combiner



(a)

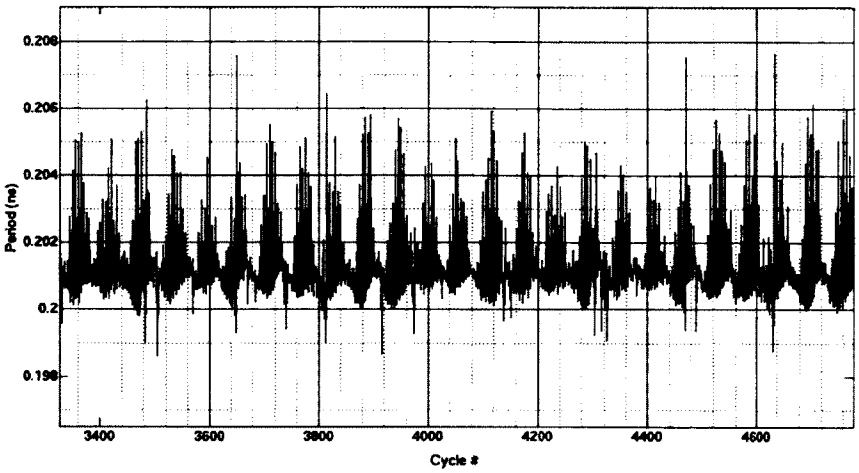


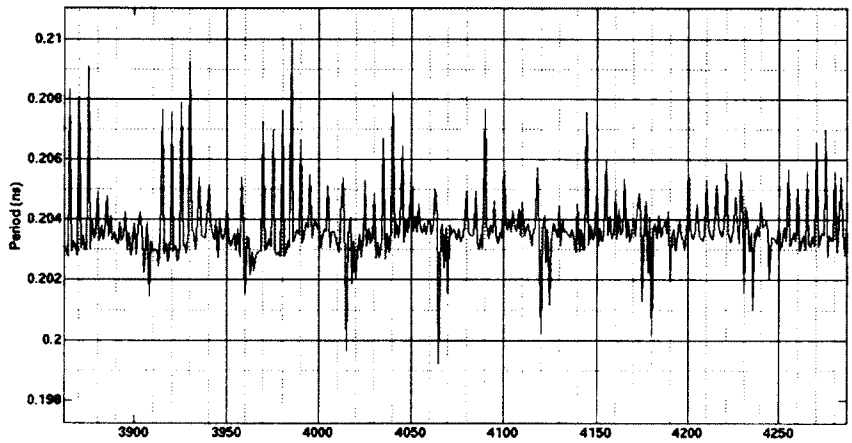
(b)

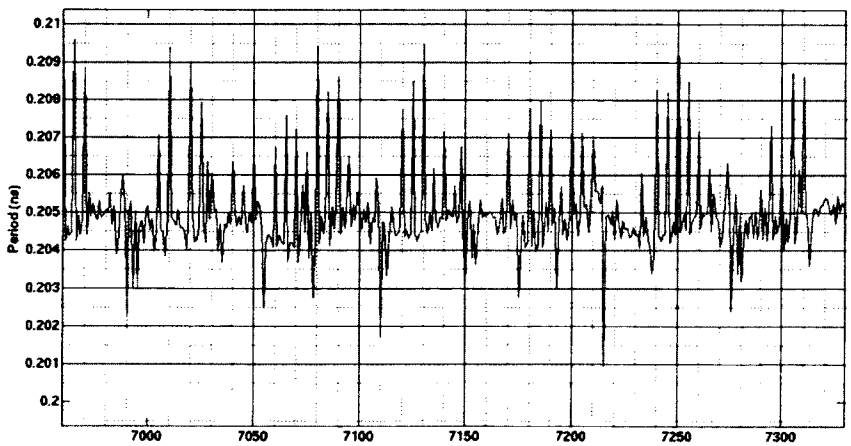


(c)

Figure 6.3.2: The period of the edge combiner output waveform versus time for different multiplication factors: a) 5, 4.969, 4.750 b) 5, 4.906, 4.875 c) 5, 4.844, 4.625


0.208
0.206
0.204
0.202
0.2
0.198
Period (ns)
3400
3600
3800
4000
4200
4400
4600
Cycle #


0.21
0.208
0.206
0.204
0.202
0.2
0.198
Period (ns)
3900
3950
4000
4050
4100
4150
4200
4250


0.21
0.209
0.208
0.207
0.206
0.205
0.204
0.203
0.202
0.201
0.2
Period (ns)
7000
7050
7100
7150
7200
7250
7300

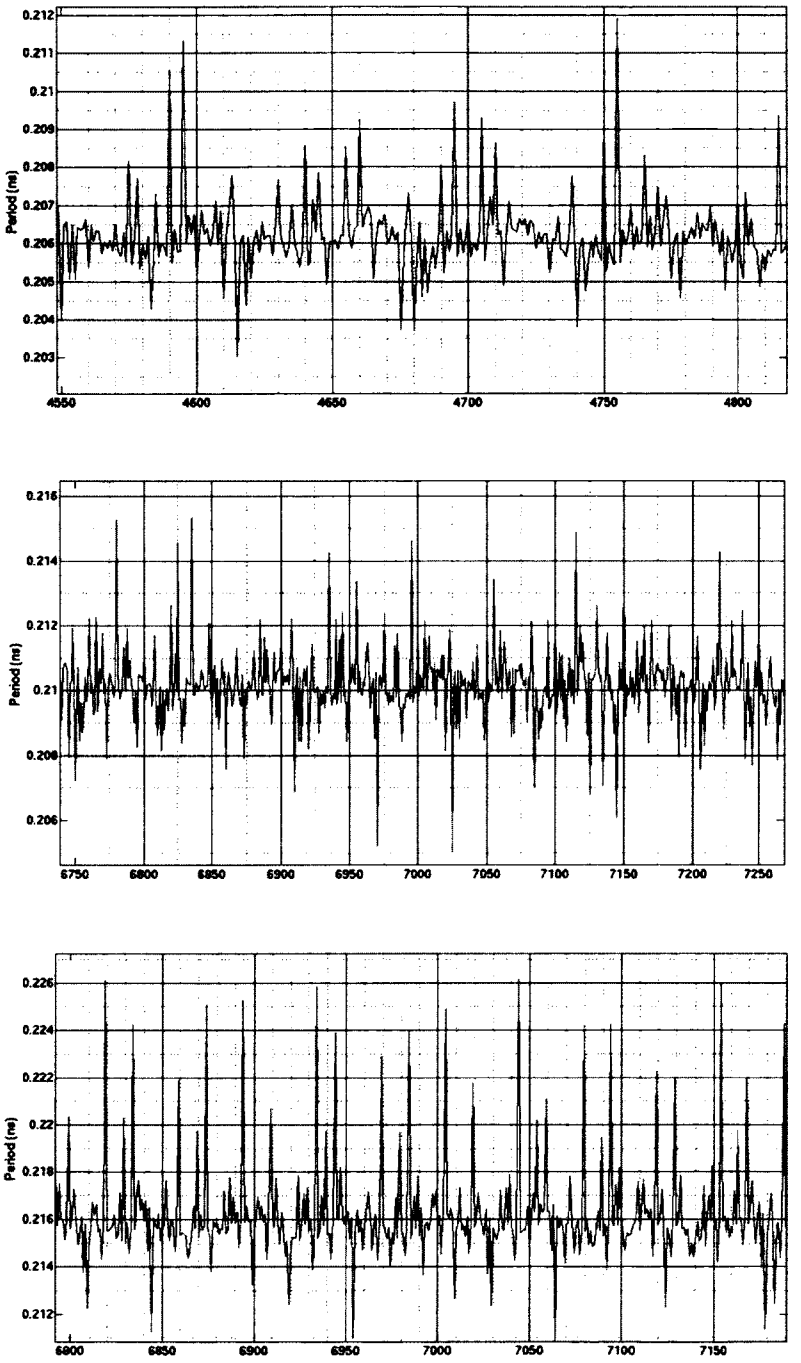


Figure 6.3.3: The period of the edge combiner output waveform versus time at fractional lock ; from up to down multiplication factors of 4.969, 4.906, 4.875, 4.844, 4.750, and 4.625

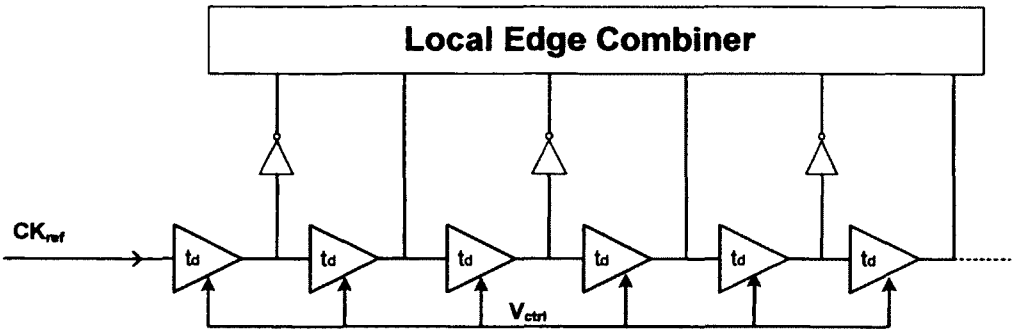
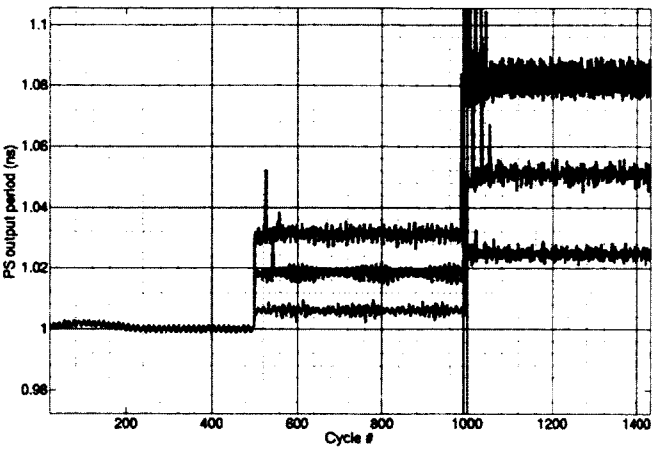


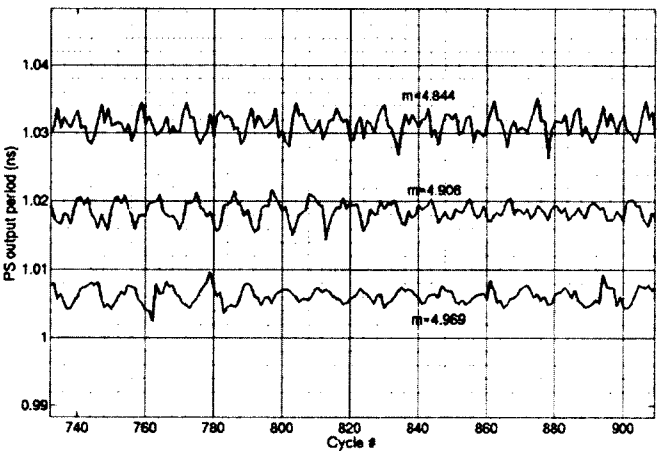
Figure 6.3.4: Schematic of delay line and its connection to Local edge combiner

6.4 Power supply sensitivity

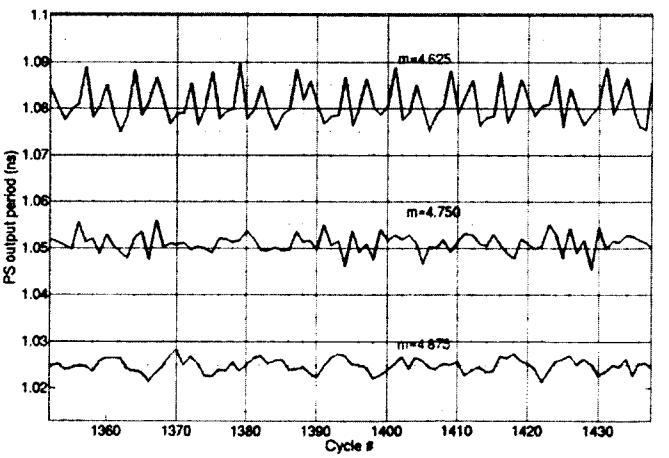
Power supply sensitivity of the synthesizer is tested with 85MHz applied sinusoidal noise with amplitude of 10mV. This frequency and amplitude are chosen as an estimate of average size package resonance frequency and the worse case regulator output noise in current technologies [76,77]. Figures 6.4.1 and 6.4.2 show the period of PS (period synthesizer) and frequency synthesizer outputs for different values of fractional multiplication factors. The jitter contribution of the power supply noise is in order of delta sigma quantization noise. Looking at two extreme scenario of using a good regulator and consider 10mV noise and using no regulator and get 50mV noise amplitude, the ratio of the power supply induced jitter to the $\Delta\Sigma$ modulator quantization induced jitter change from 0.5 and 2, respectively. Small values of α (the fractional part of multiplication factor) are considered for this comparison where the error induced by PI is in order of its DNL error. For higher values of α where the PI shows delay error close to its INL, the fractional-N multiplication factor induced a total extra jitter equivalent to 50mV power supply noise. The major source of the jitter is PI INL error in this case.



(a)

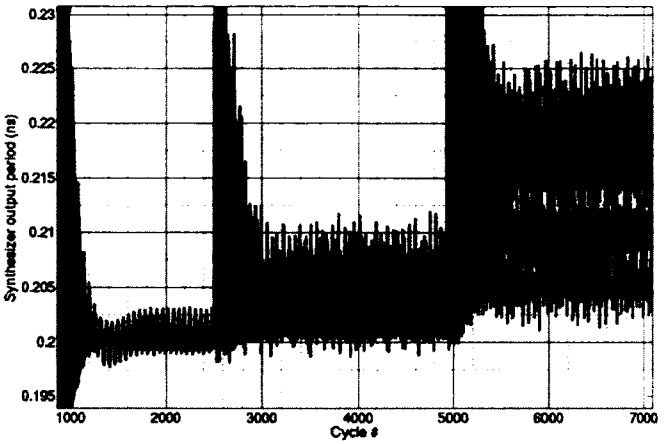


(b)

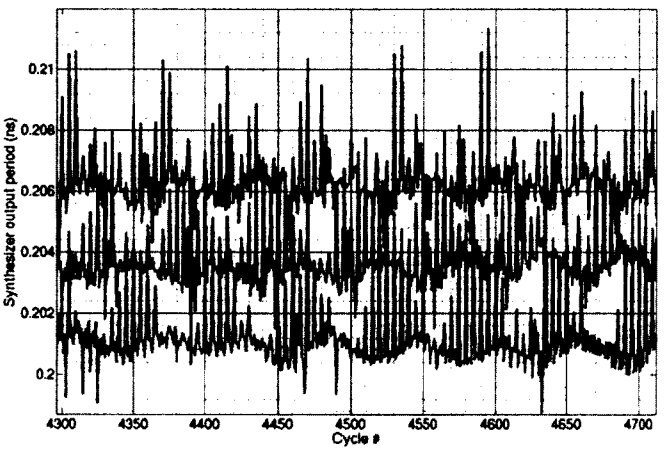


(c)

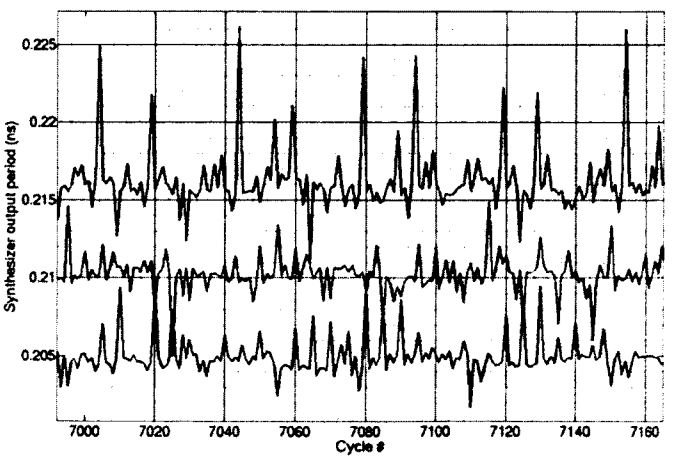
Figure 6.4.1: Period synthesizer output for each cycle in present of power supply noise; (a) For 5, 4.969, 4.906, 4.875, 4.844, 4.750, and 4.625 multiplication factors (b) Zoom in at 4.969, 4.906, 4.844 region (c) Zoom in at 4.875, 4.750, 4.625 region



(a)



(b)



(c)

igure 6.4.2: Frequency synthesizer output for each cycle in present of power supply noise; (a) For 5, 4.969, 4.906, 4.875, 4.844, 4.750, and 4.625 multiplication factors (b) Zoom in at 4.969, 4.906, 4.844 region (c) Zoom in at 4.875, 4.750, 4.625 region

6.5 Locking time

Figure 6.6.1 shows the loop filter output for high and low voltage and temperature at typical process corner. Locking time varies with the voltage and temperature. Two large capacitors in the loop have smaller variation compared to loop filter resistor. Compensating the resistor variation to less than 5% keeps locking time under 400ns. On-chip resistors are compensated to match an external reference resistor. A state machine and comparator are used to generate trim codes for on chip resistors to control their values and compensate resistance variation due to process. A globally generated trim code (for example to trim on chip termination in a high speed I/O link) can be scaled for the loop filter target value of 5KΩ

6.6 Switching time

Switching time is defined as time it takes for the synthesizer to switch from its current output frequency to the target frequency. Figure 6.7.1 and Figure 6.7.2 shows the output of the master loop filter when the multiplication factor changes from 5 to 4.844 and from 4.844 to 4.625 for selected voltage and temperature. This is equivalent to change the output frequency from 5GHz to 4.844GHz and 4.625GHz, respectively. Switching time changes with voltage and temperature and shows dependence on the magnitude of the frequency change. Switching time in general is smaller than the initial integer locking time and is in order of 200ns in this implementation.

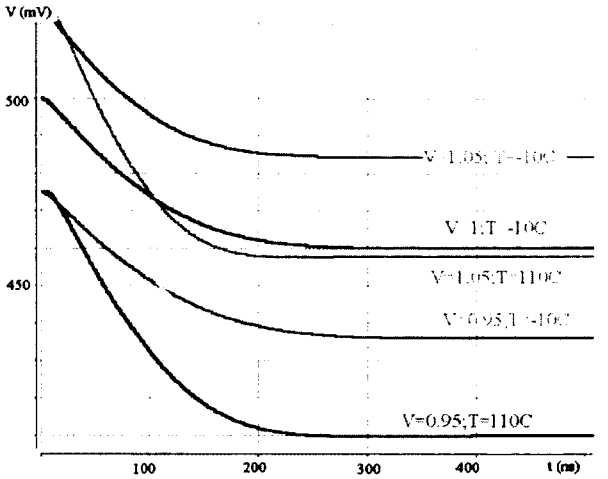


Figure 6.6.1: Master loop filter output versus time shows integer lock time of synthesizer

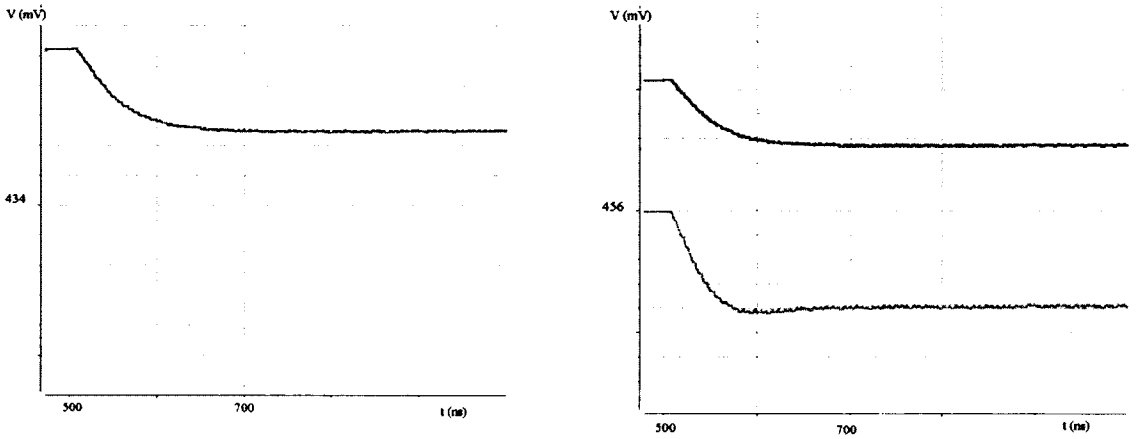


Figure 6.6.2: Switching time between 5GHz to 4.844GHz for selection of voltage and temperature at typical processing corner

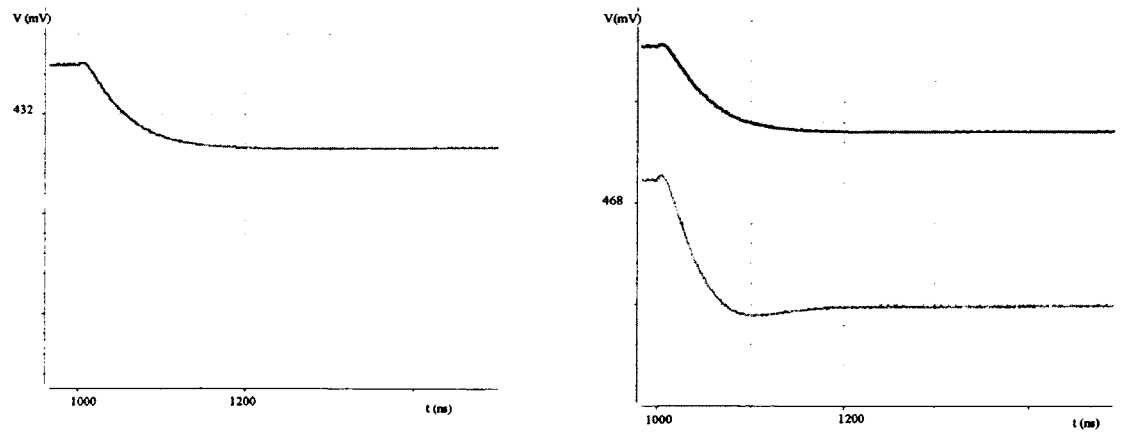


Figure 6.6.3: Switching time between 4.844GHz to 4.625GHz for selection of voltage and temperature at typical processing corner

6.7 Jitter in a DLL-based Fractional-N synthesizer

The two main sources of the jitter in a DLL-based fractional-N synthesizer are $\Delta\sum$ modulator quantization noise and power supply noise. Power supply noise directly affects the delay of both the master and slave delay lines. The loop detects the delay change inside the master loop delay line and corrects it within its bandwidth. The master loop does not detect delay change inside both of the slave delay lines (in period synthesis and frequency multiplication part) and consequently doesn't correct slave delay lines error. Therefore the best layout practice is to put all three delay lines close to each other in order to share one supply line. This not only helps mitigate local device variation but also increases the correlation between master and slave delay line power supply noise. A higher loop bandwidth reduces the effect of the power supply noise. If loop bandwidth is higher than the package resonance frequency (which contains most of the power supply noise power), then power supply induced noise can be reduced significantly. The estimate for a mid-size package resonance frequency of 85MHz is placed inside the loop bandwidth in this design. Therefore the power supply induced jitter reduces to around 1ps in the master loop. On the other hand $\Delta\sum$ modulator quantization noise appears in the reference path and thus a lower loop bandwidth would help to mitigate its impact (certainly with the penalty of longer locking time). The trade-off between the two primary sources of the jitter is a function of the loop bandwidth. In this implementation loop bandwidth is tuned such that the two sources have almost the same contribution to overall jitter. Figures 6.8.1, 6.8.2 and 6.8.3 show DLL-based fractional-N frequency synthesizer output jitter for different multiplication factors in the presence of power supply noise. Synthesizer doesn't show more sensitivity to other noise frequencies between 20MHz and 300MHz (typical package resonance frequencies for different size of packages).

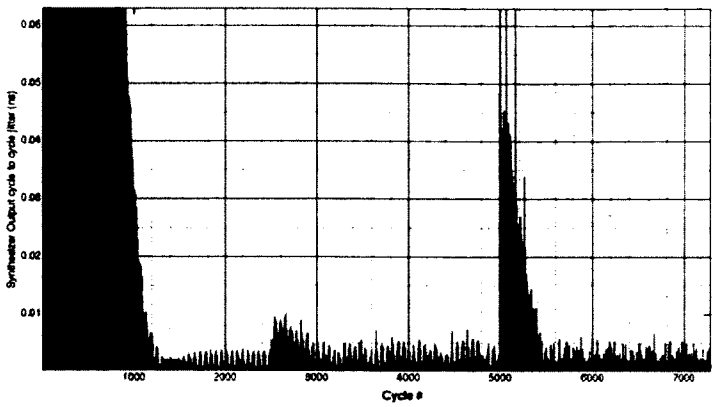


Figure 6.7.1: DLL-based Fractional-N frequency synthesizer output jitter for multiplication factors of 5, 4.969 and 4.750

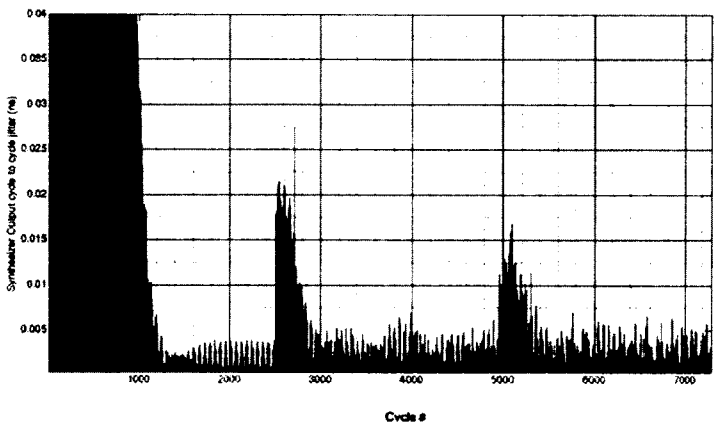


Figure 6.7.2: DLL-based Fractional-N frequency synthesizer output jitter for multiplication factors of 5, 4.906 and 4.875

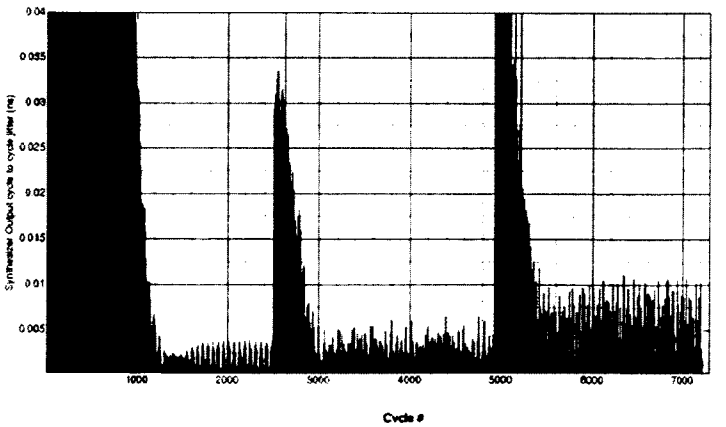


Figure 6.7.3: DLL-based Fractional-N frequency synthesizer output jitter for multiplication factors of 5, 4.844 and 4.625

The mismatch in the delay cells contributes to the total jitter in form of an extra deterministic jitter. Using the mismatch reduction methodology introduced in section 5.7.2.1 would reduce the mismatch to 2ps. This would result to estimated extra jitter up to 4ps (considering super impose of all mismatches) from what is reported in this section.

6.8 Layout considerations

A careful and well thought out layout is essential for implementing a high performance DLL-based fractional-N frequency synthesizer. This section presents the design specific layout considerations applied to this implementation.

- All paths from the master loop delay line to the MISO switch and PD have the same delay
- All paths from the period synthesis delay line to the MISO switch have the same delay
- MISO switches are symmetric and common centroid to minimize the delay mismatch
- All three delay lines drawn next to each other and share strong power supply connections. This helps to mitigate the mismatch and improve the power supply noise correlation between master and slaves delay lines
- Digital control circuitry separated from analog circuitry (loop components and delay line) within allowable distance and guard bands
- PI segments are interleaved and symmetric (partly common centroid where possible)
- Charge pump current path designed for low resistance and well shielded
- Voltage to current converter input path from loop filter, CP outputs and all the bias paths to delay lines are shielded

- Separate digital and analog power supply grids and pads to improves noise isolation and minimizes the impact of the digital switching noise on analog circuits

The layout of the synthesizer with all above considerations is prepared and fabricated in 0.18μm TSMC technology. Figure 6.9.1 shows the test chip photo of the implemented synthesizer. Total chip area is (1267×673) μm^2 which includes pads, metal fills, ESD protection diodes, digital and analog circuitry. The main synthesizer circuit occupies 15000 μm^2 of the chip area.

The fabricated synthesizer has been designed for lower output frequency suitable for fabrication in 0.18μm technology. A single ended input for reference signal is provided and a single ended output buffer supposed to provide synthesizer output to the pad. Unfortunately the output buffer is not strong enough to get a valuable waveform at the output.

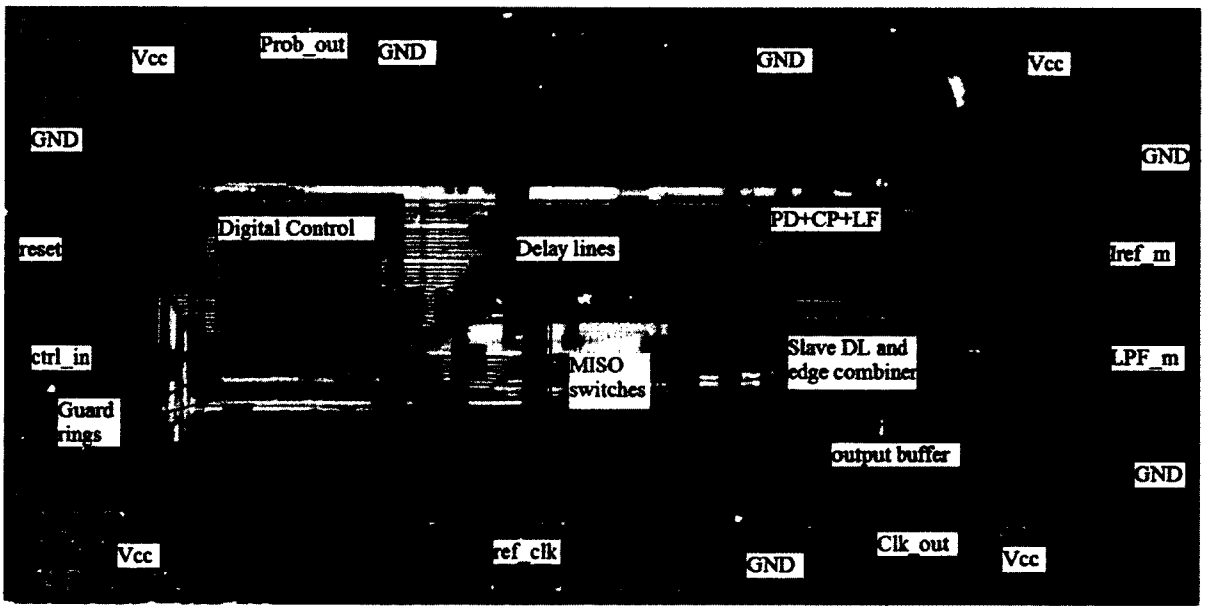


Figure 6.9.1: Test chip photo of DLL-based Fractional-N frequency synthesizer

6.9 Power consumption

The synthesizer power consumption in the typical process corner and different voltage and temperatures are shown in Table 6.5.1. The most power hungry block is the phase interpolator which consumes around 25% of the total power.

Table 6.9.1: Synthesizer power consumption in different voltages and temperatures

Process Corner	Temperature (C)	Voltage (V)	Power Consumption (mW)
Typical	110.0	0.95	7.76
Typical	-10.0	0.95	3.99
Typical	110.0	1.0	11.90
Typical	-10.0	1.0	6.00
Typical	110.0	1.05	17.43
Typical	-10.0	1.05	9.68

Chapter 7 Conclusions

Although delay locked loop (DLL) offers an outstanding low close-in phase noise due to non-accumulating behavior of the delay line, it is rarely used for the frequency synthesis due to its limited frequency resolution. This thesis presents the techniques and architectures that remove conventional DLL limitations and introduces the DLL-based fractional-N frequency synthesizer architecture [82]. The proposed DLL-based fractional-N frequency synthesizer is an alternative synthesizer class to widely used PLL-based fractional-N synthesizers. DLL-based fractional-N frequency synthesizer not only offers a competitive complexity performance trade-off, but also can be tuned to have superior performance in terms of close-in phase noise or frequency range and resolution. They tend to have more compact designs and be more compatible with the complementary metal oxide semiconductor (CMOS) technology scaling compared to the LC-VCO based phase locked loops (PLLs). They also offer much better phase noise compared to ring-oscillator based PLLs.

At a system level, DLL-based fractional-N frequency synthesizer benefits from two main techniques presented in this research: ΔΣ-DLL and Period Synthesis (PS). The ΔΣ-DLL use a ΔΣ modulator and a multiple input single output (MISO) switch to achieve fractional-N lock. Period synthesis technique is used to remove the resulting in-lock-error and its associated spurious tone from the output spectrum.

Among the four period synthesis techniques presented in this manuscript, variable length DLL PS [45] and the PI-based PS are original and ΔΣ-DLL-PS offers highest resolution. Using combinations of four period synthesis techniques, ΔΣ-DLL and MDLL, four distinct

architectures for the DLL-based fractional-N frequency synthesizer have been proposed in this dissertation. All four proposed architectures use ΔΣ-DLL to generate the desired fractional-N delay. Two of the architectures used two different period synthesis approaches to remove the timing error and its associated spurious tone from the output frequency spectrum. One used extra loops to detect the timing error and control the delay line of the period synthesis circuit. The other one used the main ΔΣ-DLL and a slave delay line to complete a phase interpolator (PI) based period synthesis. Both of these two architectures offer low close-in phase noise and spurious tone.

The two other architectures offer a lower complexity one with the price of high spurious tones and the other with higher close-in phase noise. The simplest one doesn't have any error correction schemes and can generate the desired fractional frequency accompany with spurious tones at f_{ref} away from the main output frequency. The second one combines ΔΣ-DLL and MDLL techniques to remove spurious tones but has a higher close-in phase noise as some part of the time the MDLL is in ring-oscillator mode.

The proposed single loop architecture with the PI-based period synthesis has been chosen for the practical implementation targeting high-speed-interconnect application. This implementation acts as a proof of concept and to showcase the capabilities and design challenges of a DLL-based fractional-N frequency synthesizer. Theoretically, this architecture can cover N-1 to N+1 multiplication factor but the period synthesis needs to take backward steps for N to N+1 multiplication factors. Only forward stepping is implemented to reduce the complexity of the synthesizer. Phase interpolator integral non-linearity (INL) is the major deterministic jitter contributor in this architecture. Therefore synthesizer shows a much better jitter performance for small values of α, the fractional part of multiplication factor. The INL jitter contribution can be

reduced to half by using a combination of backward and forward stepping in the PI and the period synthesis MISO switch. Many switching, digital timing and clock crossing issues have been resolved in order to let PI and MISO switch work together in a period synthesis scheme. In circuit level, different delay line topologies were investigated and the one that provides the best total power supply sensitivity of synthesizer was chosen. A fully compensated differential charge pump accompany with a differential voltage to current converter reduces the power supply noise impact on the loop dynamic and help reduces the output jitter. A low blind zone phase frequency detector is used to provide better linearity and reduces frequency error for multiplication factors close to an integer.

The implemented synthesizer can generate an output frequency with less than 15ps total jitter at frequency span of 1GHz, equivalent to $\Delta f/f_c$ of 20% while consuming ≈11.5mW power at nominal supply voltage of 1V.

7.1 Contributions

A thorough study of the fractional-N frequency synthesizers and delay locked loops led to discovery of main obstacles of making a DLL-based fractional-N synthesizer. Design parameters and their trade-offs were studied and the associated equations were derived and presented. Many well known fractional-N synthesis techniques modified for a DLL-based synthesis and novel techniques developed to address timing errors. The outcome of this part of the research was two novel period synthesis methods. The next step of the challenge was to find and match a sub-set of techniques to make a complete DLL-based fractional-N frequency synthesizer. Four novel architectures with different performance and complexity trade-offs were proposed for DLL-based fractional-N frequency synthesizer. The implementation of selected architecture

of many circuit level challenges from selecting the best topology for each building blocks to design and optimize every single block for the best overall synthesizer performance. Main contributions can be summarized as follow:

1. Introduced DLL-based fractional-N frequency synthesizer and its dynamic and equations
2. Proposed four novel architectures for DLL-based fractional-N frequency synthesizer
3. Proposed two novel period synthesis methods
4. Presented the equations and curves to quantify the dynamic behavior of fractional-N DLL
5. A thorough study of fractional-N synthesizers, $\Delta\Sigma$ modulators, delay locked loops
6. A thorough circuit level study of loop dynamics, delay lines topologies, charge pumps, phase frequency detectors, phase interpolators and their impact on DLL-based fractional-N frequency synthesizer performance
7. Designed a fully compensated differential charge pump and voltage to current converter as part of an original delay locked loop
8. Presented the related layout techniques for a high performance DLL-based fractional-N frequency synthesizer
9. Patented DLL-based fractional-N frequency synthesizer in United States [82]

7.2 Future research opportunities

Today, a simple search for "fractional-N synthesizer" would result to more than 100 journal papers, 300 conference papers and 13 book chapters. All were written about $\Delta\Sigma$-PLL based fractional-N synthesizers in the last 20 years, after it was introduced by Riley et. al. [19]. This research introduces a new class of fractional-N synthesizers, and shows the possibility and benefits of generating fractional-N multiplication factor using delay locked loop, and proposed

four architectures for DLL-based fractional-N frequency synthesizers. Certainly the outcome of this research does not match the enormous volume of literature resulting from two decades of hard work of hundreds smart researchers on PLL-based fractional-N synthesizers. This is just the beginning. Many aspects of the DLL-based synthesizers need to be investigated and new architectures would be developed to satisfy industry need of having more compact, lower power, higher quality frequency/clock synthesizer.

Handfuls of immediate future works are:

- Implement and optimize proposed multi-loop DLL-based fractional-N synthesizer
- Investigate semi/fully digital DLL-based fractional-N architectures and its implementation
- Investigate DLL-based fractional-N architecture with variable length delay line to provide very large frequency range and resolution
- Investigate method and techniques to reduce PI INL error
- Implement proposed ΔΣ-MDLL
- Research methods and techniques to reduce close-in phase noise in ΔΣ-MDLL
- Implement backward and mix backward and forward stepping in PI-based period synthesis to reduce the impact of PI INL error

References

[1] G. Allan, J. Knight, "Novel architecture for ultra low complexity mixed-signal DLL," *Analog VLSI Workshop*, Macau, China, Oct. 2004.

[2] R. B. Staszewski, D. Leipold, P. T. Balsara, "A first multigigahertz digitally controlled oscillator for wireless application," *IEEE Trans. on Microwave Theory and Techniques*, vol. 51, pp. 2154-2164, Nov. 2003.

[3] B. W. Garlepp, K.S. Donnelly, J. Kim, et al., "A portable digital DLL for high speed CMOS interface circuits," *IEEE J. Solid-State Circuits*, vol. 34, pp. 632-644, May 1999.

[4] S.Sidiropoulos, M. A. Horowitz, "A semi-digital dual delay-locked loop," *IEEE J. Solid-State Circuits*, vol. 32, pp. 1683-1692, Nov. 1997.

[5] J. G. Maneatis, "Low-jitter process-independent DLL and PLL based on self-biased techniques," *IEEE J. Solid-State Circuits*, vol. 31, pp. 1723-1732, Nov. 1996.

[6] H. H. Chang, J. W. Lin, S. I. Liu, "A fast locking and low jitter delay-locked loop using DHDL," *IEEE J. Solid-State Circuits*, vol. 38, pp. 343-346, Feb. 2003.

[7] R. B. Watson Jr., R. B. Iknaian, "Clock buffer chip with multiple target automatic skew compensation," *IEEE J. Solid-State Circuits*, vol. 30, pp. 1267-1276, Nov. 1995.

[8] C. H. Kim, et al., "A 64-Mbit 640-Mbyte/s bidirectional data strobed,double-data-rate SDRAM with a 40-mW DLL for a 256-Mbyte memory system," *IEEE J. Solid-State Circuits*, vol. 33, pp. 1703-1710, Nov. 1998.

[9] Y. Moon, J. Choi, K. Lee, D. K. Jeong, M. K. Kim, "An all-analog multiphase delay-locked loop using a replica delay line for wide-range operation and low-jitter performance," *IEEE J. Solid-State Circuits*, vol. 35, pp. 377-384, Mar. 2000.

[10] D. J. Foley, M. P. Flynn, "CMOS DLL-based 2-V 3.2-ps jitter 1-GHz clock synthesizer and tempreture-compensated tunable oscillator," *IEEE J. Solid-State Circuits*, vol. 36, pp. 417-423, Mar. 2001.

[11] H. Yahata, et al., "A 256-Mb double-data-rate SDRAM with a 10-mW analog DLL circuit," *Symp. VLSI Circuits Dig. Tech. Papers*, pp. 74-75, Jun. 2000.

[12] H. H. Chang, J. W. Lin, S. I. Liu, "A wide-range delay-locked loop with a fixed latency of one clock cycle," *IEEE J. Solid-State Circuits*, vol. 37, pp. 1021-1027, Aug. 2002.

[13] D. Byrd and C. Davis, "A fast locking scheme for PLL frequency synthesis," A*pplication Notes, National Semiconductor Corporation*, Jul. 1995.

[14] W. Djen, "SA8025 franctional-N synthesizer for 2GHz band application," *Application Notes, Phillips Semiconductor*, Sept. 1995.

[15] W. Rhee and A. Ali, "An on-chip phase compensation technique in fractional-N frequency synthesis," *Proc. ISCAS*, vol.1, Nov. 2001.

[16] M. H. Perrott, T.L. Tawkbury III, and C. G. Sodini, "A 27-mW CMOS fractional-N synthesizer using digital compensation for 2.5-Mb/s GFSK modulation," *IEEE J. Solid-State Circuits*, vol. 32, pp. 2048-2060, Dec. 1997.

[17] S. Willingham, M. Perrott, B. Setterberg, A. Grzegorek, and B. McFarland, "An integrated 2.5 GHz delta-sigma frequency synthesizer with 5 s settling and 2 Mb/s closed loop modulation," *ISSCC 2000 Dig. Tech. Papers*, pp. 200-201.

[18] V. Reinhardt and I. Shahriary, "Spurless fractional divider direct digital synthesizer and method," *US. Patent 4815018*, Mar. 21, 1989.

[19] T. A. Riley, M. Copeland, and T. Kwasniewski, "Delta-sigma modulation in fractional-N frequency synthesis," *IEEE J. Solid-State Circuits*, vol.28, pp. 553-559, May 1993.

[20] W. Rhee, B. Sup-Song, and A. Akbar Ali, "A 1.1GHz CMOS fractional-N frequency synthesizer with a 2-b third ordet delta-sigma modulator," *IEEE J. Solid-State Circuits*, vol.35, pp. 1453-1460, Oct. 2000.

[21] W. Rhee, B. Bisanti, and A. Ali, "An 18 mW 2.5GHz/900MHz BiCMOS dual frequency synthesizer with <10Hz carrier resolution," *IEEE J. Solid-State Circuits*, vol.37, pp. 515-520, Apr. 2002.

[22] T. A. Riley, N. M. Filiol, Quinghong Du, and Juha Kostamovaara, "Techniques for in-band phase noise reduction in delta-sigma synthesizers," *IEEE J. Circuits and Systems II: Analog and Digital Signal Processing*, vol.50, pp. 794-803, Nov. 2003.

[23] K. Kundert, J. White, and A. Sangiovanni-Vincentelli. *Steady-State Methods for Simulating Analog and Microwave Circuits.* Kluwer, Boston, 1990.

[24] M. H. Perrott, "Fast and accurate behavioral simulation of fractional-N frequency synthesizers and other PLL/DLL Circuits," *Proc. 39th Design Automation Conf.*, pp. 498-503, Jun. 2002.

[25] C. Park, O. Kim, and B. Kim, "A 1.8 GHz self-calibrated phased-locked loop with precise I/Q matching," *IEEE J. Solid-State Circuits*, vol.36, pp. 777-783, May 2001.

[26] K. Lee et al., "A single-chip 2.4GHz direct-conversion CMOS reciever for wireless local loop using multiphase reduced frequency conversion technique," *IEEE J. Solid-State Circuits*, vol.36, pp. 800-809, May 2001.

[27] C.Heng, B. Song, "A 1.8GHz CMOS fractional-N frequency synthesizer with randomized multiphase VCO," *IEEE J. Solid-State Circuits*, vol.38, pp. 848-854, Jun. 2003.

[28] B. Miller, R. J. Conley, "A multiple modulator fractional divider," *IEEE Trans. of Instrumentation and measurement*, vol.40, pp. 578-583, Jun. 1991.

[29] Y. Mutsuya, K.Uchimura, A. Iwata, T. Yoshitome, et al, "A 16-bits oversampling A-to-D conversion technology using triple integration noise shaping," *IEEE J. Solid-State Circuits*, vol.36, pp. 800-809, May 2001.

[30] L. Sun, T. Lepley, F. Nozahic, A. Bellisant, T. Kwasniewski, and B. Heim, "Reduced complexity, high performance digital delta-sigma modulator for fractional-N frequency synthesis," *IEEE proceedings of ISCAS*, vol. 2, pp. 152-155. May-Jun. 1999.

[31] G. R. Ritchie, "Higher order interpolation analog to digital converters," Ph.D. Dessertation, University of Pennsylvania, 1977.

[32] T. Musch, I. Rolfes, and B. Schiek, "A highly linear frequency ramp generator based on a fractional divider phase-locked-loop," *IEEE Trans. of Instrumentation and measurement*, vol.48, pp. 634-637, April 1999.

[33] A. Marques, V. Peluso, M. Steyaert, and W. Sansen, "Optimal parameters for modulators topologes," *IEEE J. Circuits and Systems II: Analog and Digital Signal Processing*, vol.45, pp. 1241-1332, Sept. 1998.

[34] K. Shu, E. Sanchez-Sinencio, et. al, " A comparative study of digital modulators for fractional-N synthesis," *Proc. ICECS*, vol. 3, pp. 1391-1394, Sept. 2001.

[35] B. De Muer, M. Steyaert, "A CMOS monolitic $\Delta\sum$-controlled fractional-N frequency synthesizer for DCS-1800," *IEEE J. Solid-State Circuits*, vol.37, pp. 835-844, Jul. 2002.

[36] B. De Muer, M. Steyaert, "On the analysis of fractional-N frequency synthesizers for high-spectral purity," *IEEE J. Circuits and Systems II: Analog and Digital Signal Processing*, vol.50, pp. 784-793, Nov. 2003.

[37] A. E. Hussein, M. I. Elmasry, "A fractional-N frequency synthesizer for wireless applications," *Proc. ISCAS*, vol.4, May 2002.

[38] G. Chien, P. R. Gray, "A900-MHz local oscillator using a DLL-based frequency multiplier technique for PCS applications," *IEEE J. Solid-State Circuits*, vol. 35, pp. 1996–1999, Dec. 2000.

[39] J. C. Rudell, J. J. Ou, T. Cho, G. Chien, F. Brianti, J. A. Weldon, and P.R. Gray, "A 1.9-GHz wide-band IF double conversion CMOS receiverfor cordless telephone applications," *IEEE J. Solid-State Circuits*, vol.32, pp. 2701–2088, Dec. 1997.

[40] C. Wang, Y. Tseng, H. She, R. Hu, " A1.2 GHz programmable DLL-based frequency multiplier for wireless applications," *IEEE J. VLSI Systems*, vol. 12, pp. 1377-1381, Dec. 2004.

[41] D.E. Calbaza, Y. Savaria, "A direct digital period synthesis circuit", *IEEE J. Solid-State Circuits, Vol. 37, pp. 1039-1045, August 2002.*

[42] H. Mair and L. Xiu, "An architecture of high-performance frequency and phase synthesis," *IEEE J. Solid-States Circuits*, vol. 35, pp. 835–846, June 2000.

[43] N. Weste and K. Eshraghian, *Principles of CMOS VLSI Design*, Addison-Wesley, 1993.

[44] G. Chien, "Low-noise local oscillator design techniques using a DLL-based frequency multiplier for wireless applications," Ph.D. dissertation, Electronics Research Lab, Univ. California, Berkeley, Jan. 2000.

[45] F. Zarkeshvari and T. A. Kwasniewski, "Analog DLL-based period synthesis circuit," *IEEE proceedings of ICCCAS*, vol. 2, pp. 1095-1098. May 2005.

[46] R. Farjad-Rad, and et al., "A low-power multiplying DLL for low-jitter multigigahertz clock generation in highly integrated digital chips," *IEEE J. Solid-State Circuits*, vol.37, pp. 1804-1811, Dec. 2002.

[47] P. Torkzadeh, A. Tajalli, M. Atarodi, "A wide tunning range, 1GHz-2.5GHz DLL-based fractional frequency synthesizer," *Proc. ISCAS*, pp. 5031-5034, May 2005.

[48] J. M. Kizer, Lau C. Benedict, "Locked loop with dual rail regulation" *US patent 6911853, June 28, 2005.*

[49] R. Liu, S. Dong, X. Hong, D. Long, J. Gu, "Two-dimensional common-centroid stack generation algorithms for analog VLSI," *Proc. ICASIC,* vol. 1, pp.128-131, 2003.

[50] D. Long, X. Hong, S. Dong, "Optimal two-dimension common centroid layout generation for MOS transistors unit-circuit," *Proc. ISCAS,* vol. 3, pp. 579-585, 2007.

[51] P. Drennan, C. C. McAndrew, "Understanding MOSFET mismatch for analog design," *IEEE J. Solid State Circuits,* vol. 38, pp. 450-455, March 2003.

[52] D. Kim, and et al., "CMOS Mixed-Signal circuit process variation sensitivity characterization for yield improvement," *Proc. CICC,* pp. 365-368, 2006.

[53] F. M. Gardner, *Phaselock Techniques*, 3rd ed. New York: Wiley Interscience, 2005.

[54] B. Razavi, *Monolitic phase-locked loops and Clock Recovery Circuits : Theory and Design*, IEEE press.

[55] R. E. Best, *Phased-locked loops: Design, Simulation, and applications*, 4th ed. New York, US: McGraw-Hill, 1999.

[56] N. H. E. Weste, K. Eshraghian, *Principle of CMOS VLSI Design,* 2nd ed., Reading MA: Addison Wesely, 1993.

[57] V. Gupta, G. A. Rincon-Mora, "Achieving less than 2% 3-σ mismatch with minimum channel-length CMOS Devices," *IEEE J. Circuits and Systems II: Express Briefs,* vol. 54, issue 3, pp. 232-236, 2007.

[58] P. Larsson, "Skew safety and logic flexibility in a true single phase clocked system," *Proc. ISCAS,* vol. 2, pp. 941-944, 1995.

[59] H. O. Johansson, "A simple precharged CMOS phase frequency detector," *IEEE J. Solid State Circuits,* vol. 33, pp. 295-299, Feb. 1998.

[60] W. Chen, M. E. Inerowicz, B. jung, "Phase frequency detector with minimal blind zone for fast frequency acquisition," *IEEE J. Circuit and Systems II: Express Briefs,* vol. 57, pp. 936-940, Dec. 2010.

[61] A. Waizman, "A delay line loop for frequency synthesis of de-skewed clock," *ISSCC Digest of Technical Papers,* 1994.

[62] V. Kaenel, et al., "A 320MHz 1.5mW at 1.35V CMOS PLL for microprocessor clock generation," *ISSCC Digest of Technical Papers,* 1996.

[63] J. Maneatis, "Low-jitter and process-independent DLL and PLL based on self-biased techniques," *ISSCC Digest of Technical Papers,* 1996.

[64] W. Rhee, "Design of high-performance CMOS charge pumps in phase-locked loops," *Proc. ISCAS,* vol. 2, pp. 545-548, 1999.

[65] W. H. Lee, J. D. Cho, S. D. Lee, " A high speed low power phase-frequency detector and charge pump," *Proc. ASPDAC,* vol. 1, pp. 269-272, 1999.

[66] P. E. Su, S. Pamarti, "Mismatch shaping techniques to linearize charge pump errors in fractional-N PLLs," *IEEE J. Circuit and Systems I,* vol. 57, pp. 1221-1230, June 2010.

[67] S. Cheng, et al., "Design and analysis of an ultrahigh-speed glitch-free fully differential charge pump with minimum output current variation and accurate matching," *IEEE J. Circuit and Systems II: Express Briefs,* vol. 53, pp. 843-847, Sept. 2006.

[68] F. Roewer, U. Kleine, "A novel class of complementary folded-cascode opamps for low voltage," *IEEE J. Solid State Circuits,* vol. 37, pp. 1080-1083, Aug. 2002.

[69] R. S. Assaad, J. Silva-Martinez, "The recycling folded cascode: A general enhancement of the folded cascode amplifier," *IEEE J. Solid State Circuits,* vol. 44, pp. 2535-2542, Sept. 2009.

[70] G. A. Fahmy, et al., "Indirect compensation technique based two-stage recycling folded cascode amplifier for reconfigurable multi-mode sigma-delta ADC," *Proc. EDSSC,* pp. 978-980, 2010.

[71] B. Lipka, et al., "Design of a complementary folded cascode operational amplifier," *Proc. SOCCON,* pp. 111-114, 2009.

[72] M. A. Ferriss, M. P. Flynn, "A 14mW fractional-N PLL modulator with a digital phase detector and frequency switching scheme," *IEEE J. Solid State Circuits,* vol. 43, pp. 2464-2471, Nov. 2008.

[73] P. Y. Wang, et al., "A digital intensive fractional-N PLL and all-digital self-calibration scheme," *IEEE J. Solid State Circuits,* vol. 44, pp. 2182-2192, Aug. 2009.

[74] X. Yu, et al., "A $\Delta\sum$ fractional-N synthesizer with customized noise shaping for WCDMA/HSDPA applications," *IEEE J. Solid State Circuits,* vol. 43, pp. 2193-2201, Aug. 2009.

[75] M. Zanuso, et al. "A wideband 3.6 GHz digital $\Delta\sum$ fractional-N PLL with phase interpolation divider and digital spur cancellation," *IEEE J. Solid State Circuits,* vol. 46, pp. 627-638, March 2011.

[76] A. P. Patel, A. Rincon-Mora, "High power-supply-rejection (PSR) current-mode low-dropout (LDO) regulator," *IEEE J. Circuit and Systems II: Express Briefs,* vol. 57, pp. 868-873, Nov. 2010.

[77] M. Ho, K. N. Leung, K. L. Mak, "A low-power fast-transient 90-nm low-dropout regulator with multiple small-gain stages," *IEEE J. Solid State Circuits,* vol. 45, pp. 2466-2475, Nov. 2010.

[78] F. Zarkeshvari, P. Noel, T. A. Kawasniewski, "PLL-Based Fractional-N Frequency Synthesizers", *International Workshop of System on Chip (IWSOC)*, Banff, Canada, July 2005.

[79] B. Fitzgibbon, M. P. Kennedy, F. Maloberti, "A nested digital delta-sigma modulator architecture for fractional-N frequency synthesis," *Proc. PRIME,* pp. 1-4, 2010.

[80] B. Fitzgibbon, M. P. Kennedy, F. Maloberti, "Hardware reduction in digital delta-sigma modulators via bus-splitting and error masking – Part I: constant input," *IEEE J. Circuit and Systems I: Regular Papers,* vol.58 pp. 2137-2148, Sept. 2011.

[81] V. R. Gonzales-Diaz, et. al, "Efficient dithering in MASH sigma-delta modulators for fractional frequency synthesizer," *IEEE J. Circuit and Systems I: Regular Papers,* vol. 57 pp. 2394-2403, Sept. 2010.

[82] Tad Kwasniewski, Farhad Zarkeshvari, "Fractional-N Delay-Locked Loops." *Patent 7940098. 05*, February 5th, 2010.