THE AMERICAN UNIVERSITY IN CAIRO

School of Sciences and Engineering

MODELING THE PHASE STEP RESPONSE OF DIGITAL BANG-BANG PLLS

A Thesis Submitted to

Department of Electronics

In partial fulfillment of the requirements for

the degree of Master of Science

By

MOATAZ A. ABDELFATTAH

Under the supervision of Prof. YEHEA ISMAIL and Dr. MAGED GHONEIMA

June 2012

Cairo, Egypt

The American University in Cairo

School of Sciences and Engineering (SSE)

MODELING THE PHSAE STEP RESPONSE OF DIGITAL BANG-BANG PLLS

A Thesis Submitted by
Moataz A. Abdelfattah

Submitted to the Department of Electronics
June 2012

In partial fulfillment of the requirements for
The degree of Master of Science

has been approved by

[Name of supervisor] _____
Thesis Supervisor
Affiliation:
Date _____

[Name of first reader] _____
Thesis first Reader
Affiliation:
Date _____

[Name of second reader] _____
Thesis Second Reader
Affiliation:
Date _____

Dr. _____
Department Chair
Date _____

To my father

ACKNOWLEDGMENTS

The work in this thesis would not have been possible without the help of many people.

First, I would like to thank my advisor, *Prof. Yehea Ismail*, for his continuous support and help to me even long before this work started. He has always been very keen to make me and my research colleagues engage with the industry to add a practical dimension to our work. I learned a lot from him both technically and on the personal level.

I would like to thank my co-advisor, *Dr. Maged Ghoneima*, for the insightful discussions and directions he has given me throughout the project. I also would like to thank him a lot for teaching me how to organize my thoughts and express them effectively.

I owe a great deal to my mentors at Intel MER program, *Nasser Kurd* and *Mohammed Abdel-moneum*. They have given me all the resources I needed to learn as much as possible. They have always dedicated some of their precious time to discuss with me and make sure I am on the right track. Thanks to them, I acquired the knowledge upon which this work is based. Many thanks as well to my colleagues at Intel for their support and the nice team spirit: *Lotfy, Abdelsalam and Saeed*.

Specially, I would like to express my gratitude and thankfulness to the people who were there for me all the time, and helped me pass through the darkest times. Thanks to my fiancé, my parents and my brothers.

I would also like to thank my friends whom I have counted on many times and they had never failed me. Thanks *Ahmed, Omar, Mahfouz, Karim and Ramy.*

Finally, I thank my examiners Prof. Serag Habib, Prof. Ali Darwish, and Dr. Amr Helmy for the effort and time they gave to study and examine my thesis.

ABSTRACT

OF THE THESIS OF

Moataz Abdelsamiea Mahmoud Abdelfattah for Master of Science

Major: Electronics Engineering

The American University in Cairo

Title: Modeling the Phase Step Response of Digital Bang-Bang PLLs

Supervisor: Prof. Yehea Ismail Co-Supervisor: Dr. Maged Ghoneima

Bang-Bang Phase Locked Loops (BB-PLLs) are a class of phase locked loops that incorporate binary phase detectors. BB-PLLs offer a low power implementation of PLLs at the cost of nonlinear loop dynamics. Since low power design is currently one of the most significant research areas in the field of VLSI, BB-PLLs have recently gained an increasing interest in the VLSI research communities. Moreover, BB-PLLs can be easily implemented using digital circuits, and thus, enable seamless scaling across technology nodes.

The nonlinearity of the BB-PLLs, however, results in a number of problems associated with the design of BB-PLLs. One of the main problems of BB-PLLs is the difficulty in modeling the nonlinear system, which leads to the lack of a well-defined design methodology. Another problem of BB-PLLs is that the response of the system depends on the Phase Error Magnitude (PEM), and thus, the system does not have a constant Bandwidth (BW). These problems make the BB-PLLs undesirable for many applications.

The main focus of this work is to develop a generic modeling methodology that can be applied to any Digital BB-PLL to predict the response of the BB-PLL prior to starting the circuit design. The benefit of this modeling methodology is to define a design methodology for BB-PLLs, and to facilitate the design process. In order to verify the proposed model, a conventional Digital BB-PLL is implemented on the circuit level, and

compared to the model. Moreover, the insights gained by the model are used to propose techniques that can be used to enhance the system linearity.

Verified by AMS simulations, the model is proved to be successful in predicting the system response. Furthermore, the proposed techniques are compared to the conventional system. Simulations demonstrated that the impact of the proposed techniques is about 35% enhancement in the system linearity, and 55% reduction in the settling time of the phase step response.

PUBLICATIONS FROM THIS WORK

1. Abdelfattah M., Lotfy A., Abdel-moneum M., Kurd N., Ghoneima M., Taylor G., Ismail Y. "*Modeling the Response of Bang-Bang Digital PLLs to Phase Error Perturbations*", IEEE in proceedings of CICC 2012.

2. Abdelfattah M., Lotfy A., Abdelsalam M., Abdel-moneum M., Kurd N., Ghoneima M., Taylor G., Ismail Y. "*A Novel DLF Architecture for Digital Bang-Bang PLLs*" IEEE in proceedings of SOCC 2012.

3. Abdelfattah M., Lotfy A., Abdel-moneum M., Kurd N., Ghoneima M., Taylor G., Ismail Y. "*Modeling the Phase Step Response of Bang-Bang Digital PLLs to Phase Error Perturbations*", submitted to TCAS I.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABREVIATIONS

- DPLL Digital Phase Locked Loops
- SPLL Software Phase Locked Loops
- ADPLL All-Digital Phase Locked Loops
- CP-PLL Charge Pump Phase Locked Loops
- TDC-PLL Time-to-Digital based Phase Locked Loops
- BB-PLL Bang-Bang Phase Locked Loops
- CDR Clock and Data Recovery
- BB-PD Bang-Bang Phase Detector
- PEM Phase Error Magnitude
- DCO Digitally Controlled Oscillator
- VCO Voltage Controlled Oscillator
- DLF Digital Loop Filter
- PI Proportional-Integral
- FCW Frequency Control Word
- FC Frequency Comparator
- BSA Binary Search Algorithm

CHAPTER 1 : INTRODUCTION

Phase Locked Loops (PLLs) are essential circuits for all synchronous electronic systems. Almost every person relies in his daily life on a device that uses PLLs. To name a few: cell phones, laptops, PCs, tablets, luxurious cars, and many other. The PLL is the main circuit used to generate the clock signal for any synchronous electronic system.

A class of PLLs called “Bang-Bang PLLs” (BB-PLLs) offers low power implementation but suffer from nonlinear response. This nonlinearity makes the modeling of the system difficult. In addition, the nonlinearity of the system yields an unreliable response. In order to leverage the low power offered by BB-PLLs, the problems of nonlinearity need to be solved.

This thesis addresses the problems of BB-PLLs. The work starts by introducing a preliminary background on PLLs and their types then concentrates on the problems of the BB-PLLs, and how the literature dealt with these problems. Before new solutions are proposed, an in-depth description of the nonlinear behavior of the BB-PLLs is discussed. Once a good understanding of BB-PLLs is acquired, a modeling methodology for Digital BB-PLLs is proposed. The modeling methodology is applied to a conventional All-Digital BB-PLL. Meanwhile, the conventional All-Digital BB-PLL is implemented at the circuit level so that the proposed model can be verified. Finally, the system insights gained by the proposed model lead to proposing new circuit techniques to enhance the linearity of the system response.

In this introduction, a brief background of PLLs and their applications will be presented. Then, the motivation of this work and the problem definition, along with the methodology and the scope of work are described. Finally, the previous work presented in the literature that is related to the problem addressed in this work is reviewed.

1.1. BACKGROUND OF PLL SYSTEMS

The PLL is a negative feedback control system that is used to align the phase and frequency of its output signal (*φout, Fout*) to those of the input signal (*φref, Fref*) as illustrated by Figure 1-1**Error! Reference source not found.**. The common applications of PLLs are frequency synthesis and timing synchronization.

Figure 1-1. Top level of the PLL

Frequency synthesis circuits are often used in microprocessors, and in wireless transceivers. In microprocessors, PLLs are used to generate the high frequency clock signal which allows the processor to execute instructions at high speed. In wireless transceivers, PLLs are used to generate the Local Oscillator signal, which represent the carrier, to up-convert the base-band signal to the RF range in the transmitter, or to down-convert the RF signal to the base-band in the receiver. For example, the architecture of an RF transmitter is shown in Figure 1-2.

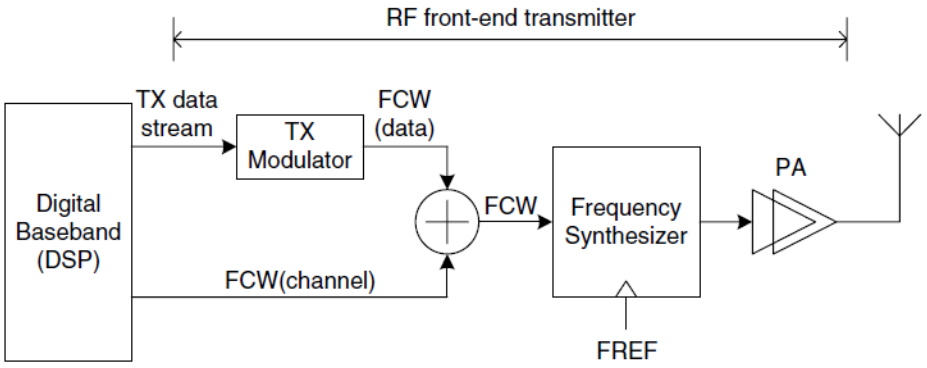

Figure 1-2. Block Diagram of the RF transmitter showing the use of PLLs in the application as frequency synthesizers [1]

Timing synchronization circuits are used in wireless receivers to synchronize the clock at the receiver with that at the transmitter. Moreover, timing synchronization is used in Clock and Data Recovery (CDR) circuits. In a CDR application, the PLL is used to recover the clock from a stream of a binary random signal with a high data rate. CDR circuits are mostly used with high speed serial links in the IOs module of microchips.

The history of PLLs dates back to 1932 when it was first described by the French Henri de Bellescize to be used in Local Oscillators in coherent communication [2]. The research continued to develop the PLLs until nowadays. One of the important milestones in the PLL history is when F.M Gardner presented a class of PLLs called the Charge Pump PLLs (CP-PLLs) in 1980 [3]. Ever since CP-PLLs were introduced, most of the applications using PLLs incorporate CP-PLLs because they offer fast lock and tracking. Moreover, CP-PLLs have well-defined design steps, and can be controlled to suit different applications.

1.1.1 Basic Concept of PLL Operation

The main objective of PLLs is to produce a high frequency clock with low "phase noise". In general, any oscillator can produce a clock signal that oscillates at a certain frequency with a non-ideality factor represented by the phase noise. An ideal oscillator would produce a single tone signal. A practical oscillator produces the desired frequency with the highest power, but other frequencies around the ideal one can have low power as shown in Figure 1-3. As more of the total power is concentrated in the desired frequency, the phase noise parameter decreases, and vice versa. It is known that phase noise of oscillators increase as the desired frequency increase.

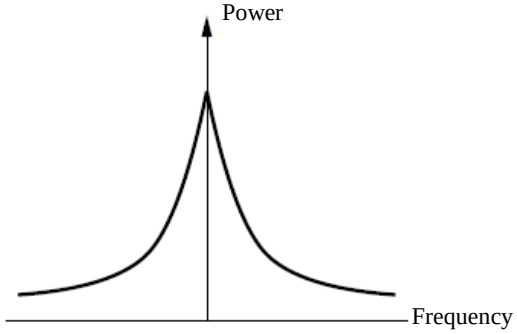

Figure 1-3. Phase Noise in Oscillators

The concept of PLLs is to produce a high frequency signal and calibrate its phase noise using a pure low frequency signal (reference). The negative feedback measures the error between the high frequency output and the low frequency reference. The error information is used to tune the high frequency signal yielding a low phase noise output. Thus, the output of the PLL is a high frequency signal with a low phase noise.

Note that the phase noise parameter is a representation of the oscillator non-ideality in the frequency domain. The time domain parameter that represents this non-ideality is the "jitter". The clock jitter is the range of uncertainty in the timing of the clock edge. Figure 1-4 illustrates jitter in clock signals.

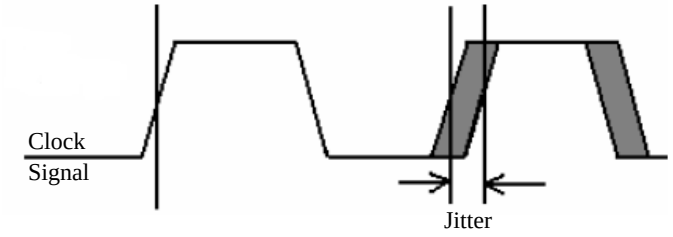

Figure 1-4. Jitter in clock signal

The generic block diagram of PLLs is shown in Figure 1-5. The basic PLL system consists of a controlled oscillator, a frequency divider, a phase detector and a loop filter. The high frequency signal produced by the oscillator is fed back to the phase detector through a frequency divider. The frequency divider down-converts the high frequency signal into the range of the reference frequency. The feedback signal is compared to the reference signal through the phase detector. The phase detector detects the error between the two signals, and passes it to the loop filter. The error information is rectified by the loop filter, and produces a control signal. This control signal is used to tune the oscillator frequency in order to eliminate the error between the feedback signal and the reference signal. When the error is eliminated, the PLL is locked. In other words, the frequency and phase of the output signal are aligned with the reference clock signal.

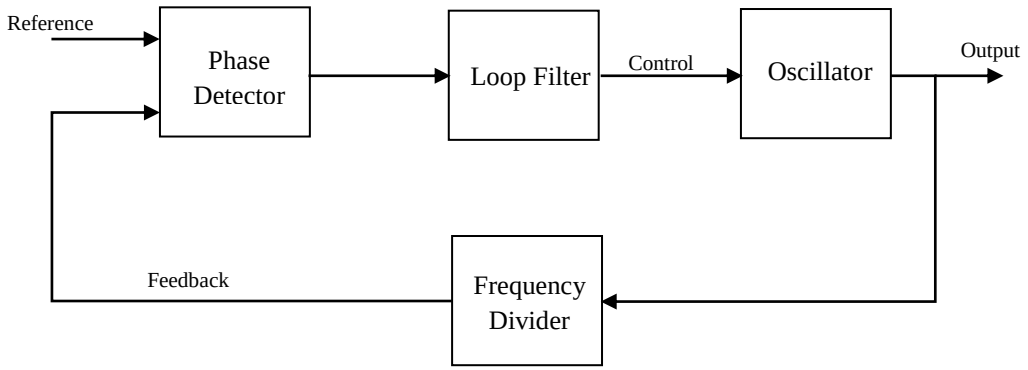

Figure 1-5. General block diagram of PLL

1.1.2 Types of PLLs

Phase Locked Loops can be categorized into three main categories: analog PLLs, Digital PLLs (DPLLs), and Software PLLs (SPLLs). The analog PLL is the traditional PLL used since PLLs were invented to date. The Digital PLLs are similar to analog PLLs in concept with some or all of the building blocks implemented in the digital domain. Software PLLs also have the same concept of operation of general PLLs, but the whole system is implemented in software using a digital processor usually a DSP. The SPLLs are more configurable than other types of PLLs, thus, are more suitable for Software Defined Radios. However, similar to any comparison between software and hardware systems, the software implementation of PLLs compromises the performance of the PLL for the sake of the system flexibility [2].

Analog PLLs are the traditional PLLs used in most of the applications until recently. As technology advances and the scaling of the feature size continues, the design of analog PLLs is becoming a challenge that is getting harder with time. This is because of the difficulty in realizing analog components, and the reduced operating ranges due to lower supply voltages for new technologies. More difficult design indicates a slower time-to-market plus additional design effort (man power). Furthermore, the difficulty in realizing analog components makes it more difficult to integrate the analog PLL on-chip. Consequently, using analog PLLs adds cost to the system. However, analog PLLs achieve the best performance among other PLLs until now. Thus, critical applications with tough specs may prefer performance over cost.

Digital PLLs, on the other hand, offer a number of advantages over analog PLLs as shown in Figure 1-6. The key advantage that makes DPLLs an inevitable choice for future systems is scalability. Digital PLLs are mostly implemented using standard cells. Thus, the design is portable across technologies saving time and design effort. Moreover, standard cell implementations are well suited for on-chip integration.

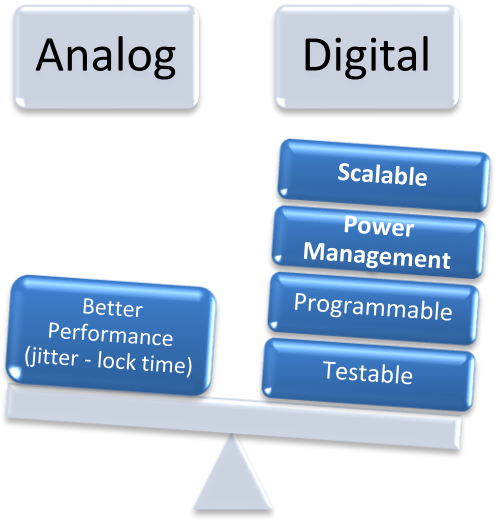

Figure 1-6. Comparison between analog and digital PLLs in terms of the advantages of each PLL type

Another important advantage of DPLLs is power management. Digital implementation of PLLs opens the door for using power management techniques, since digital implementation is more suitable for power gating. Hence, when the PLL tries to lock on the same frequency again, the lock time is significantly small. This way, the entire PLL can be power gated because it can lock on its target frequency very fast, while the system operation is not affected. Analog PLLs, on the other hand, cannot be power gated because they take a relatively large time to acquire lock, and the system operation is often affected by this delay.

Furthermore, similar to other digital systems, DPLLs have the advantages of easier testability and programmability. DPLLs are easy to test using Design For Testability (DFT) techniques. DPLLs are also easier to program, such that certain parameters can be programmed to change the characteristics of the design. For example,

the loop filter parameters control the loop dynamics, and are directly related to the system response in terms of bandwidth, stability, jitter, and lock time.

On the other hand, the performance of DPLLs is poor relative to analog PLLs. However, due to the necessity of using scalable PLLs, on-going research strives to enhance the performance of DPLLs. Another disadvantage of DPLLs is the difficulty of the design at the first time. This is because DPLLs are digital implementations of a system that is analog by nature. Therefore, digital PLLs (DPLLs) have gained a lot of interest.

Based on the phase detector used in the DPLL, two categories of DPLLs exist: Time-to-Digital Converter based PLL (TDC-PLL) and Bang-Bang PLLs (BB-PLLs). TDC-PLLs represent the digitized version of the traditional CP-PLL. TDC-PLLs use linear phase detectors, as shown in Figure 1-7. Linear phase detectors measure the phase error between the feedback, and the reference signals then use it to control the output frequency. TDC-PLLs have good linearity, and hence good performance, yet are power hungry. TDC-PLLs are power hungry because the TDC circuit is a complex circuit similar to the Analog to Digital Converter. Figure 1-8 shows the architecture of the TDC.

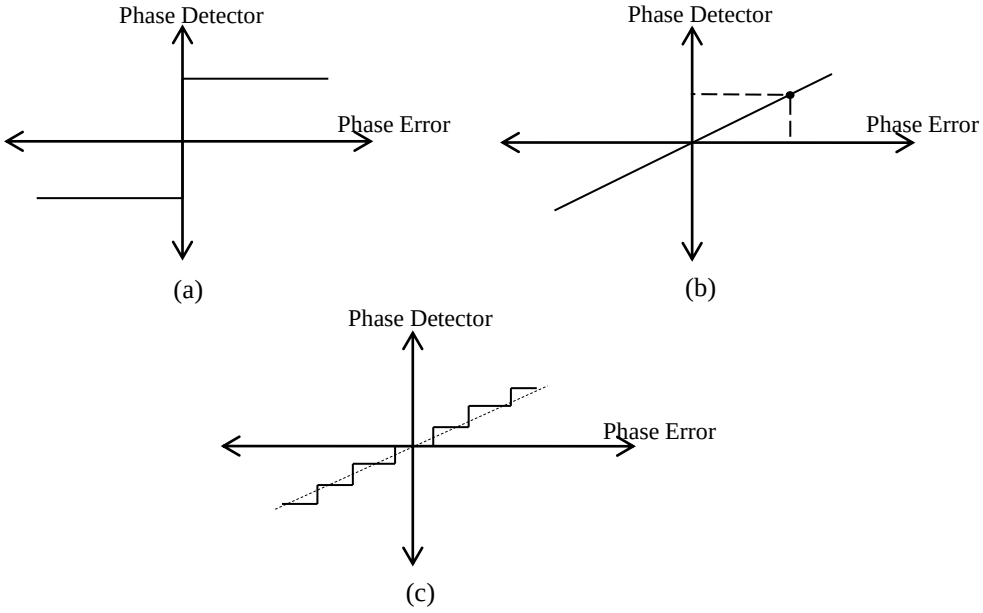

Figure 1-7. Transfer function of different phase detector types: (a) Binary Phase Detector. (b) Linear Analog Phase Detector. (c) TDC Phase Detector

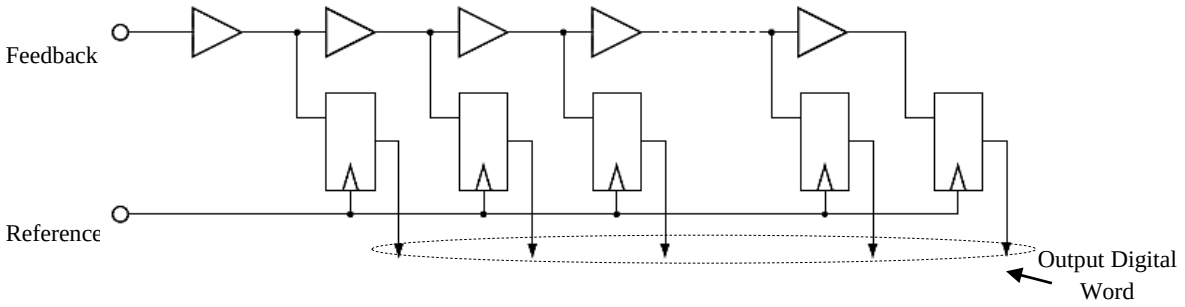

Figure 1-8. Block diagram of the TDC [4]

The architecture of the TDC shown in Figure 1-8 is one of the most common architectures of TDCs, and is called “Delay Line based TDC” (DL-TDC). The operation of the DL-TDC shown in Figure 1-8 depends on using delay cells to get delayed versions of the feedback signal. Each delayed version of the feedback signal is taped, and sampled by the reference signal. The output of the comparisons forms a digital word that represents the magnitude of the phase error. The resolution of the TDC is limited by the delay of a single delay cell.

It is also worth mentioning that in order to achieve good performance, the TDC circuit is implemented in analog domain. However, because TDC-PLLs offer the best performance in Digital PLLs, they are often used in high performance wireless applications [5] – [7]. The limit on TDC-PLL performance is the quantization step (resolution) of the TDC. The trend in the state-of-art research in TDC-PLLs is to develop techniques to enhance the TDC resolution [8].

The second category of DPLLs is the Bang-Bang PLLs (BB-PLLs). BB-PLLs incorporate binary phase detectors which decrease the power consumption since the implementation of the binary phase detector is simply a single flip-flop. However, binary phase detectors cause nonlinear loop dynamics. The transfer function of the BB-PLL is shown in Figure 1-7. The nonlinear dynamics of BB-PLLs result in an unreliable response, and the lack of a well-defined design methodology. These problems make the BB-PLLs undesirable for many applications. BB-PLLs are widely used in CDR applications. For instance the work in [9] – [11] uses BB-PLLs for CDR. The reason why

BB-PLLs are used in CDR application is that BB-PLLs support high data rates, thus can handle the high data rate at the input of the CDR [12]. Hence, the nonlinearity of BB-PLLs usually restricts their domain of applications only to CDR circuits.

1.2. MOTIVATION AND PROBLEM DEFINITION

1.2.1. Motivation

Low power design is currently one of the most significant research areas in the field of VLSI. Technology trends to increase the frequency of operation have given that significance to low power designs. Moreover, the rapidly emerging System-on-Chip (SOC) technology integrates a large number of circuits on a single die requiring low power circuits to be used.

Consequently, a great deal of interest has been given lately to BB-PLLs. Although BB-PLLs suffer from a number of problems that restrict their use to a narrow domain of applications, the low power implementation of PLLs offered by BB-PLLs have gained them that interest. Until the last decade, all the research work that addressed BB-PLLs was targeting CDR applications. Since BB-PLLs are already suited for CDR, then the research efforts did not attempt to solve the nonlinearity problems of BB-PLLs, rather attempted to optimize BB-PLLs for CDR applications.

Only recently in the 2000's, a number of publications that target solving the nonlinearity problems of BB-PLLs started to appear. The main objective of such publications is to enhance the performance of BB-PLLs to be used in different applications other than CDR. For example, the work in [13], [14] suggested using BB-PLLs as frequency synthesizers for microprocessor applications. Moreover, although wireless applications require very high performance synthesizers that use linear phase detectors, the work in [15], [16] suggested some modifications in the BB-PLL design to be used for high performance wireless applications. It was found by [15], [16] that although BB-PLLs demonstrated worse jitter performance than TDC-PLLs, BB-PLLs achieved a better power-to-jitter ratio. Furthermore, very recently in 2012, a BB-PLL to be used for mobile SoC was presented [17]. Table 1-1 compares between two published results of a BB-PLL and a TDC-PLL [17].

Table1.1. Comparison between published results of a BB-PLL and a TDC-PLL [17]

	BB-PLL	**TDC-PLL**
Technology	65nm	65nm
Supply Voltage	1.1-1.3V	1.8V
Freq. Range	0.6 - 0.8 GHz	1.4 - 2.2GHz
Power	2.9 mW @ 0.8GHz	6.3 mW @1.8GHz
Area	$0.027mm^2$	$0.11mm^2$

The above discussion introduced the motivation behind the work presented in this thesis. The motivation driving this research study is the rising importance of BB-PLLs for different applications because of the low power offered, given the fact that BB-PLLs suffer from a lot of problems because of their nonlinearity. Therefore, the objective of this work is to contribute in developing and enhancing BB-PLLs in order to facilitate their use in various applications.

1.2.2. Problem Definition

Due to the nonlinearity of the BB-PLL, there are a number of problems associated with using BB-PLLs. One of the main problems that make BB-PLLs undesirable is the difficulty in modeling. Since the BB-PLL is a non-linear system, the well-developed mathematical techniques for linear systems, such as transfer function and Laplace transform, cannot be used to model the system. Hence, BB-PLLs do not have a well-defined design methodology, and are usually avoided by designers for that reason. The absence of a well-defined design methodology makes the design cycle longer, and imposes additional design effort to ensure that the design is working properly.

Another important problem associated with BB-PLLs is that the response of the BB-PLL is unreliable, since it depends on the magnitude of the input phase error. For instance, the bandwidth of the BB-PLL system is not a constant bandwidth, but a variable bandwidth that depend on the magnitude of the phase error between the reference and the feedback signals. This is another reason why BB-PLLs can easily be avoided for many applications even if the performance of BB-PLLs is acceptable for these applications.

The main focus of this work is to develop a generic modeling methodology that can be applied to any digital BB-PLL to predict the response of the BB-PLL prior to starting the circuit design. The benefit of this modeling methodology is to define a design methodology for BB-PLLs, and to facilitate the design process. Furthermore, in this work, two techniques are proposed in order to enhance the system linearity, and reduce the dependence of the system response on the phase error magnitude. As a result, the response of the BB-PLL becomes more reliable.

1.3. METHODOLOGY AND SCOPE OF WORK

The methodology of the research is described as follows: At first, deep understanding of the BB-PLL system operation is acquired. Each building block is studied, and then the overall system operation is studied through comparison with conventional linear CP-PLLs in order to learn about the differences. Based on the comprehensive insight of the BB-PLL system and its building blocks, the time-domain behavior of the individual blocks is mathematically modeled. The response of the system is represented by a set of linear and quadratic equations that are easily implemented in Matlab. In order to verify the correctness of the derived equations, an All-Digital BB-PLL system is implemented on the circuit level and simulated using Cadence AMS tools.

Furthermore, the insights gained by the proposed model were used to suggest two circuit techniques that would help enhancing the BB-PLL system linearity. The two techniques are implemented in this work as well. The effects of the proposed techniques on the linearity of the system are demonstrated by comparison with the conventional All-Digital BB-PLL system. Moreover, the enhanced linearity due to the proposed techniques is illustrated through theoretical equations as well.

Note that the proposed modeling methodology can be used to model only the phase step response of the BB-PLL system in the locked state. In other words, the proposed model is mainly used to predict the response of the system when the locked state is disturbed by a phase step error. However, modeling the phase step response is enough to provide a well-defined methodology for the BB-PLL system. This is justified because the phase step response of BB-PLLs captures all the significant characteristics of

the BB-PLL response such as the system stability, and Bandwidth (BW). Detailed explanation regarding this point is provided in Chapter 2.

1.4. LITERATURE REVIEW ON THE PROBLEMS OF BB-PLL SYSTEMS

The problems of BB-PLLs addressed in this work are the modeling of BB-PLLs and the dependence of the system response on the Phase Error Magnitude (PEM). The second problem is sometimes referred to as the Bandwidth problem, since the BW of the system is variable based on the PEM. In this section, the work in the literature that addresses these two problems is reviewed. Since the interest in BB-PLLs has aroused recently, the work found in the literature that target the problems of BB-PLLs is limited.

1.4.1. Literature Review on Modeling Problem

In the literature, there has been only one modeling methodology presented by Razavi in [18]. Any other work in the literature related to the modeling problem is only considered a revised version of the model in [18]. Almost all publications that implement BB-PLLs use this model such as [19], [20]. In addition, a rigorous nonlinear BB-PLL analysis is provided in [24].

The model presented in [18] is based on linearizing the binary phase detector that introduces the nonlinearity in the system. The concept of linearizing the phase detector relies on two factors: the metastability state in the output of the phase detector, and the random distribution of the input jitter. The first factor of metastability suggests that the transfer function of the binary phase detector is not ideally sharp as in Figure 1-7, rather has a linear range as in Figure 1-9. The smoothed transfer function suggested in Figure 1-9 is due to the metastability phenomenon. The metastability in the binary phase detector happens at the moment when the edges of the two input signals to the phase detectors are separated by a very small period of time that is less than the rising/falling time of the signal. At this moment, the output of the phase detector is neither a zero nor one, but a value in between. Hence, if the phase error between the two input signals is very small, the binary phase detector is considered a linear block and the whole system can be modeled using the Laplace transform method.

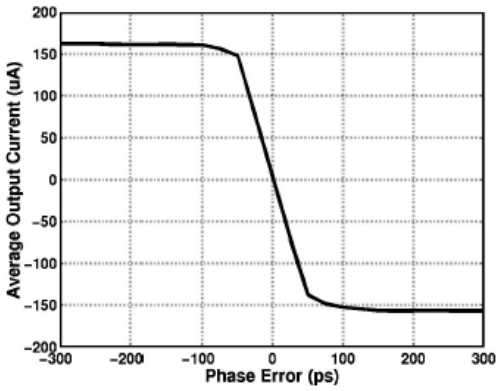

Figure 1-9. The transfer function of the binary phase detector is smoothed by the effect of the metastability [18]

Moreover, there is a second factor that helps smoothing the transfer function of the binary phase detector at small phase errors. The second factor is that the input jitter in the reference signal causes the edge of the reference clock to take a normal distribution in time rather than to come at a deterministic time as in Figure 1-10. This simply adds to the probability that the edge of the reference clock coincide with the edge of the feedback signal, and thus the output of the phase detector is not binary. Thus, this input jitter factor increases the range of phase errors at which the phase detector can be modeled as a linear block.

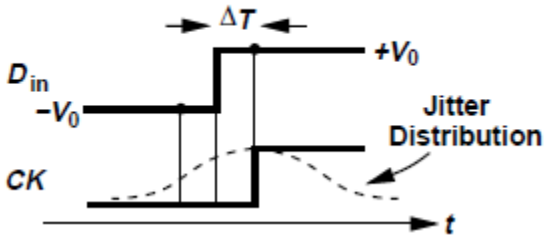

Figure 1-10. The random distribution of the input jitter affects the timing of the reference clock edge [18]

The foregoing discussion of the model in [18] shows that the BB-PLL can be modeled with linear modeling techniques as long as the phase error is small. In [18] the model is also extended to the case where the phase error is large. However, this extension to large phase errors was derived specifically to one case of input jitter waveform that is common in the CDR applications.

Therefore, the model in [18] is good to be used to model small phase errors only, or small and large phase errors for CDR applications. This model is not valid to be used for a general case of phase errors in applications different than CDR. Moreover, the model in [18] is derived for an analog PLL, and cannot be directly applied to digital BB-PLLs.

Other major contributions in the literature to the modeling problem of BB-PLLs are introduced by Da Dalt [21], [22] and Samori [23]. Both contributions are only enhancements of the model in [18]. In [21] the linear gain of the phase detector in the model in [18] is modified such that more practical factors are taken into consideration, and an expression is given for that gain. In [22] the limits of the linearized model are investigated. Finally, [23] extends the use of the linearized model, in the case of small phase errors only, to the application of frequency synthesis.

In conclusion, to date, no generic modeling methodology that can model the response of the BB-PLL system to small and larger phase errors has been introduced in the literature. In this work, the proposed methodology can be used to model any Digital BB-PLL application. The proposed modeling methodology can predict the BW and stability of the system for any given phase error magnitude.

Unlike the work in literature, the proposed methodology does not depend on linearizing the BB-PD, rather, the time domain behavior of the system blocks is modeled. The proposed model gives insight to system stability, and settling using few calculations. Such insight reduces the design cycle by providing design recommendations for the loop parameters, and guarantees better reliability for BB-PLL systems.

1.4.2. Literature Review on the Bandwidth Problem

Since the response of the BB-PLL depends on the phase error, the BB-PLL system does not have a fixed BW [18]. For example, if two phase errors of magnitudes A1 and A2 were introduced while the BB-PLL was in lock, the loop will respond with a bandwidth BW1 and BW2 respectively. Unlike BB-PLLs, linear PLLs have constant BW irrespective of the magnitude of the phase error.

A number of solutions have been proposed in the literature to enhance the linearity of the BB-PLL system, and in turn, the BW variability. For instance, the work in [16] used an infinite impulse response filter that averages the decision of the binary phase detector over time. A small phase error will cause rapid changing binary decision, and thus will give a small average. The loop filter coefficients are then adjusted accordingly. The problem with this solution is that large phase error magnitudes produce a fixed binary decision, since the one of the two signals stays leading/lagging the other for a while. Thus, the average of the binary decision in this case would not represent the phase error, since any large phase error will cause a constant binary decision. Hence, this approach enhances the system linearity only for the case of small phase errors, but not for large phase error magnitudes.

Another approach proposed in [24], and used in [15], is based on the mathematics of linear systems theory. The approach suggested linearizing the BB-PLL by auto-correlating the binary decision of the phase detector to a random generated signal over a number of samples to estimate the impulse response of the loop, and then adjust the loop filter coefficients accordingly as shown in Figure 1-11. The results obtained from using this approach were not illustrated in either reference. This approach may be able to enhance the linearity of the system to a good extent. However, it requires a large number of samples in order to be accurate. Thus, this approach increases the computation effort considerably (i.e. more power), and adds noise to the system through injecting a random generated signal into the loop.

The techniques proposed in this work are simple, and do not add much complexity to the system design. Moreover, unlike the work in the literature, the proposed techniques take into consideration the case of large phase errors. Furthermore, the proposed techniques not only enhance the system linearity, but also reduce the lock time of the BB-PLL systems.

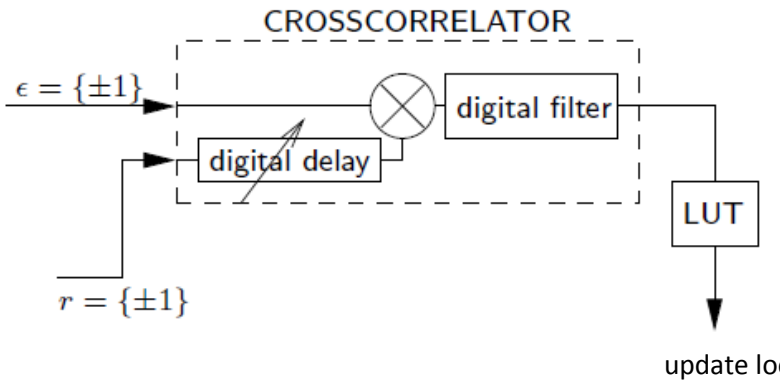

Figure 1-11. Cross correlation of the binary phase detector output (ε) with a random generated signal (r) to adjust the loop parameters through a lookup table (LUT) [24]

To sum up, Digital PLLs can have a linear implementation using a TDC, or a nonlinear implementation using a Bang-Bang Phase Detector. A Bang-Bang implementation of DPLLs reduces power consumption at the cost of nonlinear response. There is not much work in the literature that addresses the problems of BB-PLLs specially the modeling problem. This work focuses on developing a modeling methodology for BB-PLLs and facilitating the design process.

1.5. Thesis Outline

Chapter 2 gives an in-depth understanding of the nonlinear operation of BB-PLLs. Each building block of the BB-PLL system is described in details. Then, the effects of the nonlinear loop dynamics on the overall system performance are illustrated. At the end of chapter 2, the relation between the phase step response of the BB-PLL, and the nonlinearities of the system are shown.

Based on the details given in Chapter 2, a modeling methodology of the BB-PLL system is proposed in Chapter 3. The modeling methodology is based on representing the time-domain behavior of the BB-PLL system by mathematical equations. Then, the proposed methodology is used to derive a model for the conventional BB-ADPLL system.

To verify the proposed model in Chapter 3, the conventional BB-ADPLL system is implemented on the circuit level in Chapter 4. The details of circuit implementation of each building block are given first. Then, the performance of the implemented PLL is

obtained by AMS simulations. Finally, the phase step response obtained by AMS simulations is compared to that predicted by the model.

In Chapter 5, two proposed techniques to enhance the BB-ADPLL system performance are presented. Both techniques are implemented on the circuit level, and compared to the conventional BB-ADPLL system presented in Chapter 4.

Finally, Chapter 6 concludes the work in this thesis. Then, Future directions of this research are presented.

CHAPTER 2 : BB-PLL SYSTEM OPERATION

In this Chapter the building blocks of the generic BB-PLL system will be described in details. Then, the closed loop operation of the BB-PLL system is explained while emphasizing the non-linear behavior of the BB-PLL system. After that, the effects of the nonlinearity on the performance of the BB-PLL system are illustrated. Finally, the benefits of studying the phase step response of the BB-PLL system, and the insight provided by this methodology are described.

2.1. BASIC BUILDING BLOCKS OF THE BB-PLL SYSTEM

Similar to the generic PLL shown in Figure 1-5, a BB-PLL system consists mainly of a phase detector, a loop filter, a controlled oscillator, and a frequency divider. However, a BB-PLL uses a specific type of phase detectors called "Bang-Bang Phase Detector" (BB-PD). BB-PLLs can be implemented using analog design, or digital design. Thus, there are Bang-Bang Analog Phase Locked Loops, Bang-Bang Digital Phase Locked Loops (BB-DPLLs), and Bang-Bang All Digital Phase Locked Loops (BB-ADPLLs).

2.1.1. Bang-Bang Phase Detector

The Bang-Bang Phase Detector (BB-PD) is a binary phase detector. It takes two signals as input, and gives a binary decision as an output. The binary decision determines which signal leads/lags the other. In concept, the BB-PD is simply a flip-flop, as shown in Figure 2-1, with one of the two input signals as the data (D), and the other as the clock (CK).

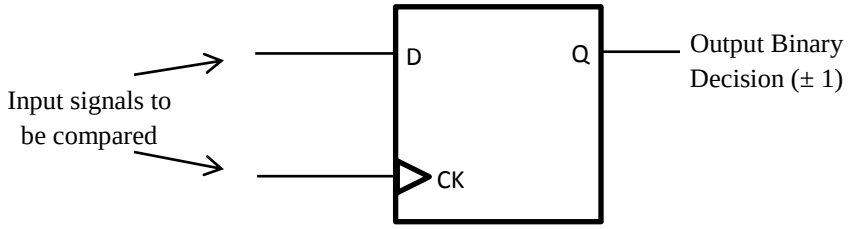

Figure 2-1. Basic structure of the BB-PD: single flip-flop

The transfer function of the BB-PD is similar to that of a hard-limiter shown in Figure 1-7-(a). In other words, the BB-PD quantizes only the direction of the phase error between the two input signals, and neglects the Phase Error Magnitude (PEM) information. Thus, the BB-PD exhibits a non-linear transfer function, and hence, introduces hard nonlinearity to the BB-PLL system. In fact, the BB-PD is the main source of nonlinearity in BB-PLLs.

On the other hand, a linear phase detector has a linear transfer function as shown in Figure 1-7-(b). Thus, a linear phase detector quantifies both the direction and the magnitude of the phase error between the input signals of the phase detector. Unlike BB-PLL, a PLL that uses a linear phase detector avoids hard nonlinearity. Hard nonlinearities cause a considerable degradation in PLL performance, and may limit the use of the PLL to certain application which do not require high performance. The effects of nonlinearities on the PLL performance will be detailed in section 2.3 of this Chapter. Examples of PLLs that use linear phase detectors are the Charge Pump Phase Locked Loop (CP-PLL), and the Time-to-Digital Converter based Digital Phase locked Loop (TDC-DPLL).

One common problem in BB-PDs is the possibility of metastability at small or zero phase error case. Since in the case of small phase errors the clock signal edge is aligned with the data signal edge, the clock samples an undefined value that may result in a functional problem on the system level. A possible solution to overcome the metastability problem is to delay the output of the BB-PD before it is sampled by another circuit. Thus, the BB-PD output voltage would have more time to resolve the metastability state and settle at a certain voltage.

The first BB-PD was proposed by Alexander [25] for the CDR application. Despite the hard non-linearity introduced by the BB-PD circuit, it was shown that the BB-PD is suitable for the CDR application more than other linear phase detectors. The reason behind that is that CDR circuits process very high speed random binary data signals. Hence, it is only required to sample the data and cope with high data rate. On the other hand, linear phase detectors reduce the overall speed of the system, and limit the system's ability to process high data rate, because they require more processing than

nonlinear phase detectors. Moreover, the complexity of linear phase detectors circuitry introduces an undesirable propagation delay between the clock and data signals, which misleads the overall CDR loop action [12].

Consequently, until recently BB-PDs were only used for CDR applications. However, due to the increasing demand for low power circuits, different researchers started to study the possibility of using BB-PDs in frequency synthesis and other PLL applications.

2.1.2. Loop Filter

After the phase detector compares the two input signals, and gives a decision that represents the phase error, this information is passed to the loop filter. Traditionally, in analog PLLs, a loop filter has two jobs. The first one is to filter the high frequency noise and the ripples that are imposed on the phase detector output. The second job is to adjust the dynamics of the loop. For instance, the loop filter determines the order and type of the loop. Figure 2-2 shows the frequency domain model of the traditional analog PLL. It can be concluded from the figure that different choices of the loop filter transfer function lead to different overall loop characteristics, which in turn affects the system stability and performance.

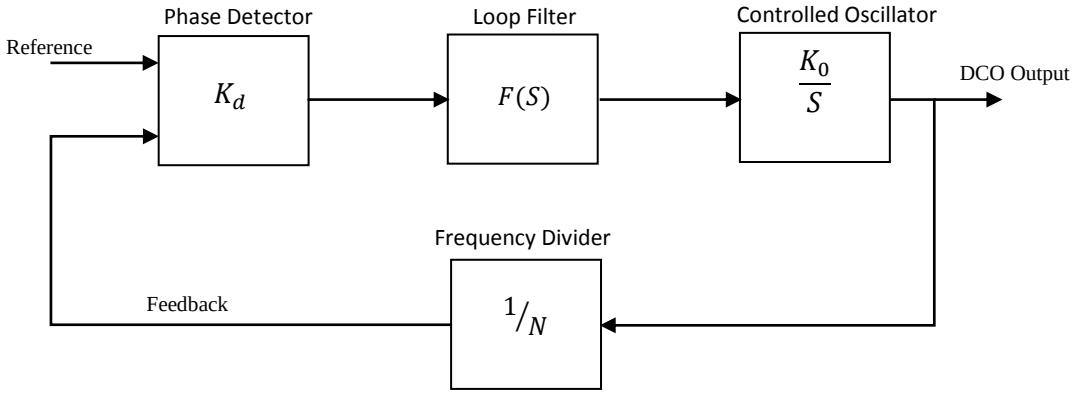

Figure 2-2. Frequency domain model of the conventional analog PLL. K_d is the linear gain of the phase detector, K_0 is the gain of the controlled oscillator, and N is the frequency division ratio. $F(s)$ is the transfer function of the loop filter

The typical PLL used in most of the microprocessor and RF applications, as concluded from the literature [26], [27], is a second order, and type two PLL.

Figure **2-3** shows the analog implementation of the conventional loop filter used to realize the typical PLL that is widely used and accepted for different applications.

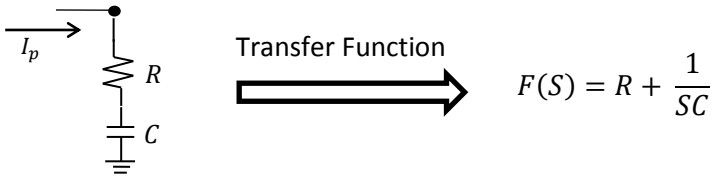

Figure 2-3. Analog implementation of the conventional loop filter. Ip is the control current, R and C represent the loop filter parameters

For digital PLLs the loop filter is implemented in the digital domain, usually using standard cell libraries, and the output of the Digital Loop Filter (DLF) is a digital word instead of a voltage in the case of analog loop filter. The methodology used in designing DPLLs and their DLFs is to use an analogy between the digital discrete time DPLLs and their analog continuous time counterparts as proposed in [28]. The DPLL is first designed as an analog system using the traditional transfer function analysis, and then the desired analog transfer functions are mapped into the discrete time domain using the Bi-Linear transformation method.

In order to implement a DPLL that has similar characteristics as the typical analog PLL in Figure 2-3 , the transfer function of the conventional filter is converted into the Z-domain by means of a Bi-Linear transformation, which gives the following transfer function:

$$F(z) = \beta + \alpha \frac{1}{1-Z^{-1}} \qquad (1.1)$$

The block diagram implementation of (1.1) is represented in Figure 2-4. This representation is easily implemented using an HDL, and then synthesized into standard cells. From a control theory point of view, the conventional DLF in Figure 2-4 is a proportional-integral (PI) loop filter with β as the proportional coefficient, and α as the integral coefficient.

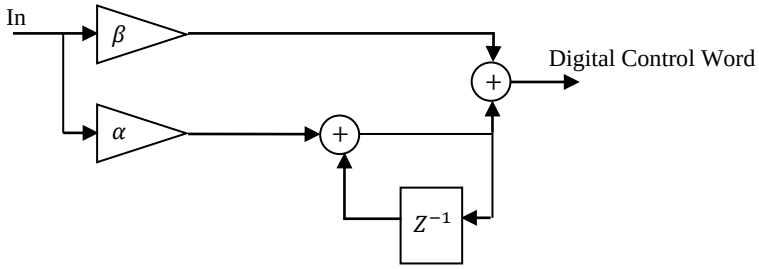

Figure 2-4. Digital Implementation of the conventional loop filter

For BB-PLLs a transfer function analysis cannot be used due to the nonlinearity of the system. However, the suitable DLF architecture for BB-PLLs was derived by Walker in [12], and was found to be a PI DLF similar to the conventional DLF in Figure 2-4. The effect of the DLF parameters on the loop stability and system performance is rigorously analyzed in [12] and [29]. In brief, it was shown that:

- As the ratio (β/α) decreases, the system stability decreases, and the ratio should be maintained larger than 1, such that the effect of the proportional path on the phase of the DCO signal dominates that of the integral path.
- A larger β yields worse jitter performance.
- As α increases, the lock time of the system decreases, meanwhile, the system response becomes more oscillatory (stability degrades).

Finally, in the digital implementation of loop filters, a delay element is usually inserted after the output of the DLF as shown in Figure 2-5. This delay element represents the update rate of the loop. This indicates that the output of the DLF updates the DCO every *D* reference cycles not every reference cycle. Thus, the DLF uses the "update clock", which is *D* times slower than the reference clock. The update rate *D* is inserted in the loop for stability reasons as analyzed in [29].

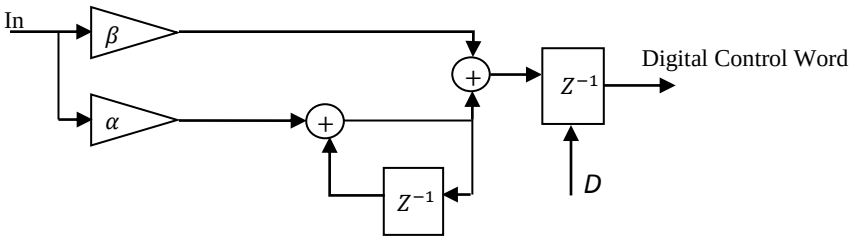

Figure 2-5. DLF implementation with the update rate D

2.1.3. Controlled Oscillator

The decision given by the phase detector is adjusted by the loop filter to be passed to the controlled oscillator in order to control the output frequency of the oscillator to lock to a given target frequency. There are two types of controlled oscillators that are widely used in PLL applications: LC Oscillators, and Ring Oscillators (RO). Figure 2-6 (a) and (b) show the general architecture of LC Oscillators and ROs, respectively. Generally, LC oscillators have better phase noise performance than ROs. On the other hand, ROs are easier to implement and integrate on-chip, while LC oscillators need huge inductors and are not easy to integrate.

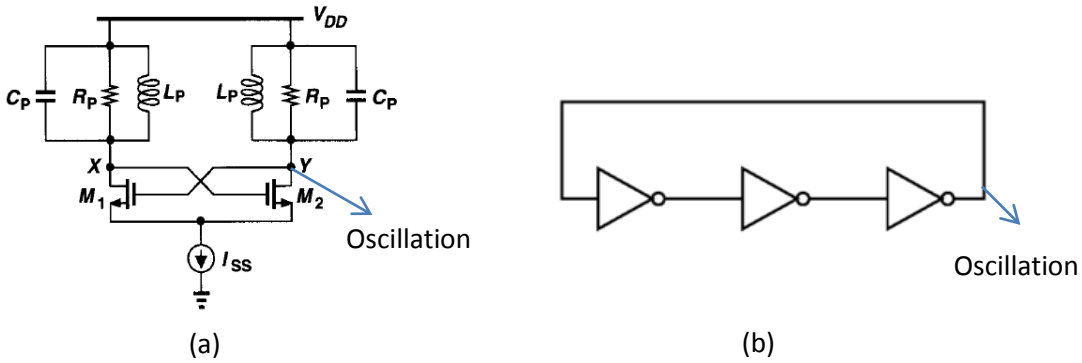

Figure 2-6. Basic architectures of (a) LC oscillator and (b) Ring Oscillator

For an analog PLL, the controlled oscillator is a Voltage Controlled Oscillator (VCO). A VCO is an oscillator whose output frequency changes corresponding to the control voltage provided by the loop filter. An example VCO is shown in Figure 2-7. In Figure 2-7, the control voltage (V_{tune}) is used to change the value of the variable capacitor, and the output frequency of the oscillator is given by $\omega_{out} = 1/\sqrt{LC}$. Hence, the frequency of the oscillator is controllable by the voltage signal (V_{tune}). On the other hand, for digital PLLs the output of the loop filter, which represents the control quantity of the controlled oscillator, is a digital word. Hence, the controlled oscillator used for digital PLLs is a Digitally Controlled Oscillator (DCO).

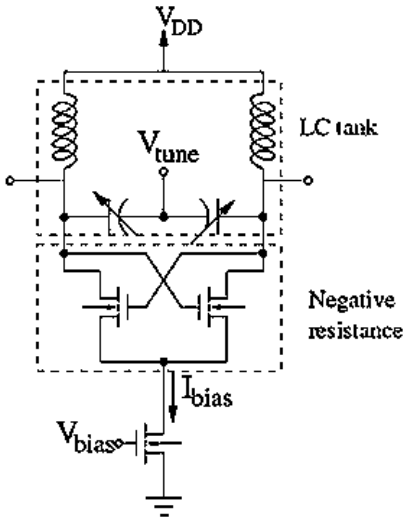

Figure 2-7. Example of an LC VCO

There are two ways to implement a digitally controlled oscillator. One is to implement a typical VCO and precede it by a Digital to Analog Converter (DAC). In this case the system is called a Digital PLL (DPLL). Another way to implement the DCO is to build a fully digital oscillator that is controlled directly by the digital word provided by the DLF. In this case the system is rather called an All-Digital PLL (ADPLL). From this point on, the abbreviation (DCO) will only refer to the fully digital implementation used in ADPLLs. Figure 2-8 shows an example RO DCO. In Figure 2-8 an input digital word varies the size of the inverters in the RO inverter chain, and thus controls the frequency of the oscillator output, since the oscillator frequency is given by $\omega_{out} = 1/Nt_d$, where N is the number of delay cells used, and t_d is the delay of one cell.

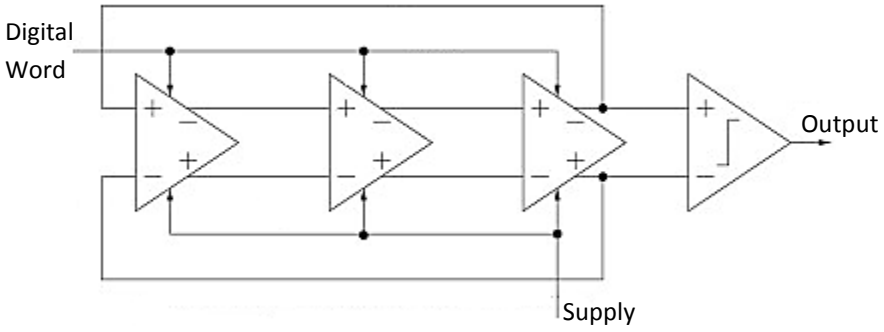

Figure 2-8. Example of a Ring Oscillator DCO

The most important specifications of a DCO are: phase noise and minimum frequency step (frequency resolution). Different applications have different restrictions on these specs. For instance, RF applications require very low phase noise and a very small frequency resolution in the range of 10 Hz. Microprocessor applications, on the other hand, are more relaxed. Microprocessor applications may tolerate larger amount of phase noise and frequency resolution in the range of 1 MHz.

2.1.4. Frequency Divider

The frequency divider is a circuit that takes a high frequency signal as an input, and gives a lower frequency version of the input signal as an output. The purpose of the frequency divider circuit in a PLL system is to convert the high frequency signal produced by the oscillator into a lower frequency similar to that of the reference signal so that the two signals can be compared and the phase error can be determined.

There are two types of frequency dividers: integer-N and fractional-N dividers. Integer-N PLLs can synthesize frequencies that are only multiples of the reference frequency. For example, if the reference frequency is 100 MHz, then the PLL can synthesize 1 GHz, 1.1GHz, 1.2GHz… etc., but the PLL cannot synthesize a 1.15 GHz or other frequencies that are not multiples of 100 MHz. On the other hand, a fractional-N divider can synthesize fractions of frequencies. The integer-N divider consists of a counter. Each bit of the counter represents a divided version of the input signal. The fractional-N divider shown in Figure 2-9 is composed of two integer-N dividers that divide the frequency of the input signal by N, and by N+1. Passing the input signals by the two integer-N dividers with different ratios yield a fractional-N divider. For example, if the input signal is divided by N one time and by N+1 another time (selection ratio = 0.5), then the overall division ratio is (N+N+1)/2 which equals to N+0.5. Thus, using different combinations of the two integer-N dividers yield a fractional-N divider.

Generally speaking, fractional-N dividers are used in RF applications more often since RF applications usually require more frequency resolution. However, microprocessor applications have more relaxed specifications on the frequency resolutions, and hence microprocessor applications often use integer-N dividers.

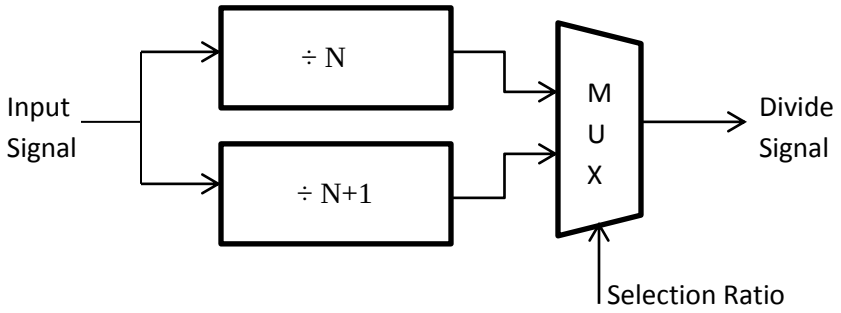

Figure 2-9. Block diagram of a dual modulus fractional-N divider

For BB-PLLs both types of dividers can be used. However, fractional-N dividers impose difficulty to BB-PLL system design, and reduces the performance of BB-PLLs. This is because BB-PLLs already suffer from hard non-linearity, and the traditional fractional-N divider, shown in Figure 2-9, introduces periodic jumps in frequency in order to get a fraction on average. These periodic jumps are not favorable by the BB-PD, and give rise to spurious tones in the spectrum of the output signal. Hence, BB-PLLs are usually used with integer-N dividers. However, BB-PLLs can still be used with fractional-N dividers if the divider itself is designed such that it does not introduce hard jumps in the frequency, or if some compensation technique is used to compensate for the jumps introduced by the traditional fractional-N divider. As mentioned earlier, there are some attempts in the literature, in [15] and [16], to use a fractional-N divider with BB-PLLs by modifying the architecture of the fractional-N divider.

2.2. BB-PLL CLOSED LOOP OPERATION

Although the analysis in this work applies to both BB-DPLLs and BB-ADPLLs, more focus is given to BB-ADPLLS. Hence, the illustrations and derivations throughout this work will refer to BB-ADPLLs.

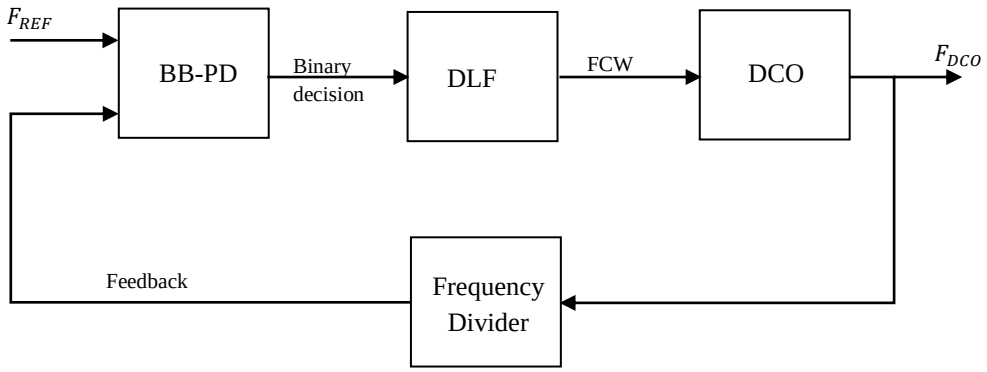

Figure 2-10. General block diagram of BB-ADPLL

Figure 2-10 shows a general block diagram of the BB-ADPLL system. The purpose of the system is to produce a high frequency signal (F_{DCO}) that has a pre-defined (target) frequency, and has the same phase as the reference signal (F_{REF}). The target frequency is defined by the division ratio *N*, such that $F_{DCO} = N*F_{REF}$. The operation of the BB-PLL proceeds as follows: A Digitally Controlled Oscillator (DCO) generates a high frequency signal that is then divided by *N* to produce the feedback signal. The feedback signal is then compared to the reference signal via the Bang-Bang Phase Detector (BB-PD). The BB-PD quantizes the direction of the phase error between the reference and the feedback signal, and feeds the decision to a Proportional-Integral Digital Loop Filter (PI-DLF). The DLF then produces a Frequency Control Word (FCW) that is used to tune the DCO frequency in order to lock onto the target frequency.

The loop passes by three stages in order to acquire lock: frequency acquisition, phase acquisition, and tracking. In the frequency acquisition stage, the loop tunes the DCO to produce a frequency that is close to the target frequency but not exact. This step can be done using the typical architecture in Figure 2-10, or a frequency acquisition loop can be used to aid the frequency acquisition process, then the loop switches to the phase acquisition mode, and uses the DLF as shown in Figure 2-11.

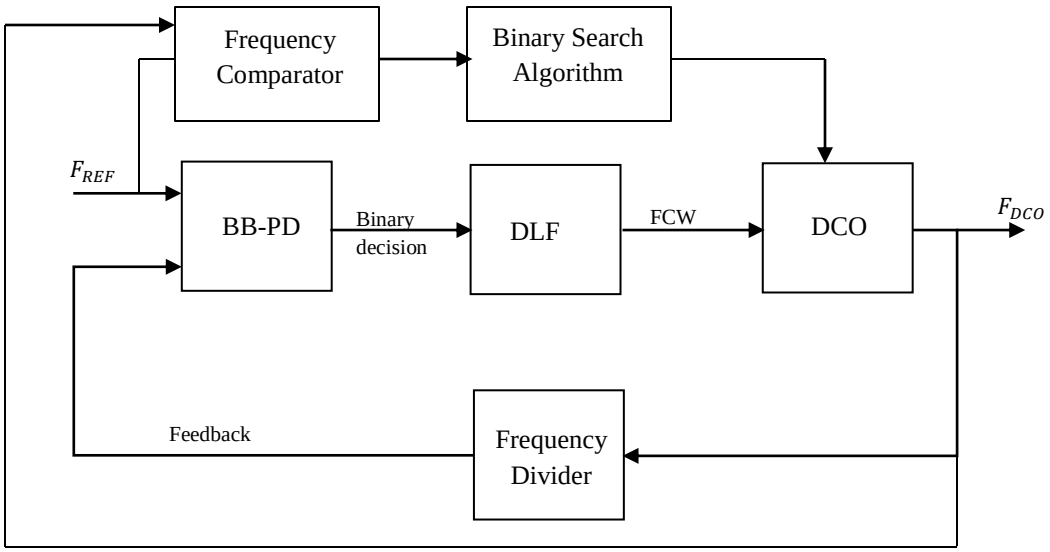

Figure 2-11. BB-ADPLL with a frequency acquisition loop

A frequency acquisition loop, shown in Figure 2-11, replaces the BB-PD by a frequency comparator and the DLF by a binary search algorithm. The frequency comparator consists of a counter and a comparator. The frequency comparator counts the number of DCO cycles contained in one feedback cycle then compares it to N. Thus, it can be determined if the current DCO frequency is faster or slower than the desired target frequency. The fast/slow decision produced by the frequency comparator is then processed by the binary search algorithm, which forces the frequency of the DCO to approach the target frequency fast.

Once the frequency acquisition stage is finished, the system goes to the phase acquisition stage, where the frequency acquisition loop is isolated and the system operates through the original loop. In the phase acquisition stage, if the direction of the phase error detected by the BB-PD is "ahead", the DCO frequency decreases by the value of the proportional value β through the DLF. Then, if the phase error direction is not changed (remains as "ahead"), the DLF decreases the DCO frequency by the value of the integral coefficient α, then by 2 α, and so on. The DLF keeps decreasing the frequency of the DCO until the polarity of the BB-PD decision flips, say it becomes "behind", then the DLF increases the DCO frequency by β and repeats the operation.

From the above description of the phase acquisition process, it can be noticed, that if the values of β and α are not chosen carefully, the loop may become unstable and

never acquires lock. Thus, it is clear that the phase acquisition stage is mainly shaped by the DLF parameters. This confirms the job of the loop filter in shaping the loop dynamics as mentioned earlier.

Finally, when the system acquires lock it goes to the third stage which is tracking (maintaining lock). In this stage, in normal conditions, the loop keeps oscillating around the desired frequency by a value $\beta*K0$, where $K0$ is the frequency gain of the DCO. This agrees with the literature in that the value of β directly controls the jitter performance of the BB-PLL system. However, the tracking stage is not as simple as it may sound, since the lock stage is often interrupted by phase error perturbations that may arise due to voltage variations, temperature variations, and random noise. These interruptions represent a threat to the locked state for any PLL in general, and for BB-PLLs specially. This is because BB-PLLs are not favorable to sudden jumps in the phase or the frequency, since the BB-PD does not propagate the PEM information to the loop, and hence, the loop action is blind to some extent, and would not easily sense the large jump in phase/frequency. Thus, phase error perturbations, especially large ones, may cause the BB-PLLs to be pulled out of lock, or may affect the performance badly. In this work, modeling the response of the BB-PLLs in the locked state to phase error perturbations is the main objective.

2.3. EFFECT OF NONLINEARITY ON BB-PLL OPERATION

To further define the problem addressed in this work, the effects of nonlinearities on the BB-PLL system are thoroughly investigated. Based on problems caused due to these nonlinearities, the rest of the thesis will propose solutions that facilitate the BB-PLL design process and enhance the overall system performance.

Generally, feedback loops depend on measuring the error, and feeding the error information back to the loop so that the loop can learn about how far its output is from the target output. Consequently, the feedback loop acts based on the error information to correct its output. The BB-PLL is a feedback loop, however, it measures part of the error information, which is the error direction, and leaves another part, which is the error

magnitude. Hence, the feedback operation does not work properly as it does for PLLs with linear phase detectors such as CP-PLLs and TDC-PLLs.

2.3.1. Increased Lock Time

The BB-PD quantizes only the direction of the phase error, and does not capture the Phase Error Magnitude (PEM) information. As a result of this nonlinear behavior the loop action taken to adjust the DCO frequency is fixed and irrespective of the error magnitude. Thus, the phase acquisition stage takes a longer time for BB-PLLs because the loop takes a fixed action (β, α) in response to large phase errors. One way to decrease the lock time is to increase the fixed action taken by the loop. However, this will result in increased jitter in lock stage, because the loop takes a large fixed action in response to a small phase error (in-lock phase error).

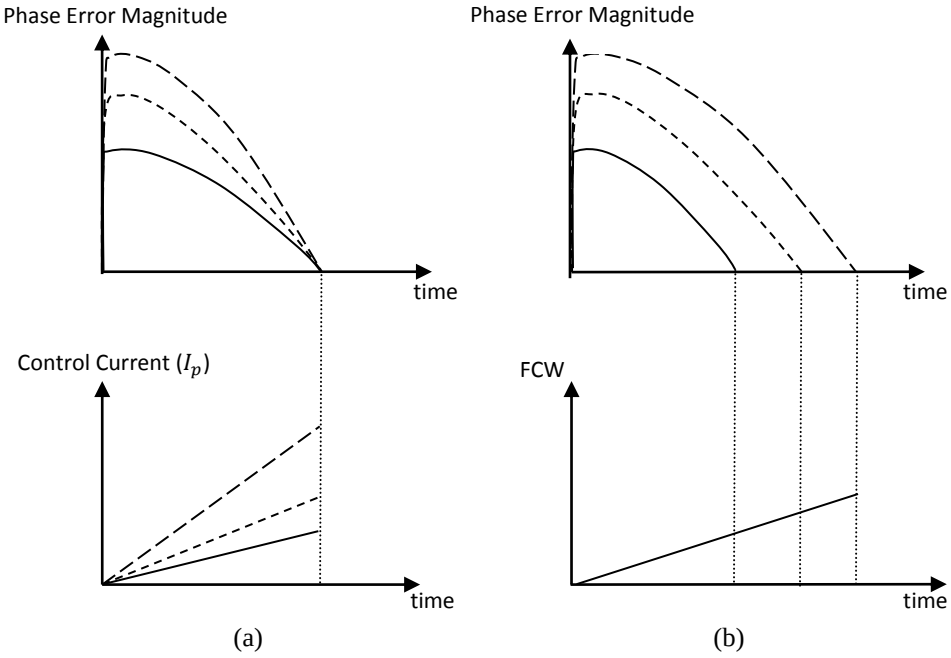

Figure 2-12. The response of the control current/word to different phase error magnitudes (a) CP-PLL, (b) BB-PLL

Figure 2-12 shows a comparison between CP-PLL and BB-PLL in terms of the action taken by both systems in response to different phase error magnitudes.

Figure 2-12-(a) shows that the CPPLL changes its control quantity, which is the charge pump current (Ip), proportional to the phase error magnitude. For a large phase error magnitude the charge pump current slope is increased to change the DCO frequency in large steps proportional to the large phase error magnitude. For a small phase error magnitude the slope of the charge pump current is relaxed. On the other hand,

Figure 2-12-(b) shows the response of the control quantity of the BB-PLL, which is the Frequency Control Word (FCW), to different phase error magnitude. The slope of the FCW is fixed irrespective of the PEM.

2.3.2. Undefined Bandwidth

The closed loop Bandwidth (BW) of a PLL is defined as: the speed of the system to track change in the input phase. In other words, the BW of the PLL is the maximum frequency allowed for the change in the input phase in order for the PLL to maintain lock. For a CP-PLL, the speed of the system in tracking different phase errors is constant, as shown in Figure 2-12. This leads to a constant well-defined BW. For BB-PLLs, however, they speed of the system response at any time depends on the phase error magnitude at that time. For BB-PLLs the BW is dependent on the PEM. Hence, the BW of BB-PLLs has no well-defined value.

Many applications have a strict requirement on the BW. In this case, the BB-PLL cannot be used for such applications due to the undefined BW problem. For other applications, such as CDR and microprocessors, the BW requirement may be more relaxed, and the BB-PLL may be suitable. However, even in applications with relaxed BW requirements, a variable BW may cause performance issues. Hence, the BW problem of the BB-PLL can restrict its domain of applications.

As illustrated in Chapter 1, many solutions to this problem have been proposed in the literature, usually by introducing BW regulation circuits and algorithms into the loop. Although no solution fixes the problem completely, some enhancements in the system BW make the BB-PLL acceptable for some applications. However, for applications with strict requirements on the BW, such as RF applications, the BB-PLL still cannot be used.

In this work, as well, a novel methodology that enhances the BW of the BB-PLL is proposed in Chapter 5.

2.3.3. Unreliable Response

The fact that the response of the BB-PLL is dependent on the PEM which is often a random variable, and sometime time varying, make the BB-PLL unreliable to many applications. For example, applications that expect large phase error perturbations in their normal operations may be pulled out of lock if using a BB-PLL. Another example is the microprocessor applications. Although the requirement on the BW may not be very strict, a microprocessor sometimes uses cascaded PLLs. In this case, cascading two PLLs with different BW are not favorable, since the linear superposition of their outputs is no longer guaranteed.

2.3.4. Difficulty in Modeling

A major drawback in BB-PLL system design that stems from the nonlinearity of the system is the difficulty in system modeling. Difficulty in modeling leads to difficulty in the design process of the BB-PLL, and may lead to not choosing BB-PLL for a certain application even if the specs are most suited with the BB-PLL features. The lack of a system model in the design cycle means a longer design cycle and unexpected behavior of the system since it is not modeled. This drawback adds to the unreliability of the BB-PLL.

The design procedure used with PLLs that use linear phase detectors is the same as that used with any traditional linear system. The transfer function analysis is used. Each building block of the system is modeled by the Laplace transform of its time domain response, and then a transfer function is obtained for the whole system. As shown in Figure 2-13, the transfer function is expressed one time in terms of the system response parameters (damping ratio ζ, natural frequency ω_n), and another time in terms of the PLL system parameters (phase detector gain K_d, loop filter parameters (β, α), oscillator gain K_0). The first set of parameters, system response parameters, is determined based on the desired specs such as BW and stability. Accordingly, by equating the two expressions of

the transfer functions, the values of the system parameters that would give the desired response are chosen.

$$H(s) = \frac{K_d \,.K_0 \,.\omega_{LPF}}{S^2 + \omega_{LPF} \,.S + K_d \,.K_0 \,.\omega_{LPF}}$$

TF in terms of design parameters

$$H(s) = \frac{{\omega_n}^2}{S^2 + 2\zeta\omega_n \,.S + {\omega_n}^2}$$

TF in terms of system response

Figure 2-13. Design methodology for linear PLLs

For a nonlinear system such as the BB-PLL, the nonlinear block BB-PD prevents this traditional methodology in design. Since the BB-PD is a nonlinear block, it has no transfer function. In turn, the whole BB-PLL system cannot be represented by a transfer function. Transfer function in linear systems represents the link between system parameters and specs. Moreover, the transfer function of a system allows the system to be modeled mathematically or on Matlab, and hence, quick results and good expectations of the system can be obtained before the design is even started.

In brief, the nonlinearity of the BB-PLL prevents using traditional conventional methodologies to model the system. In order to overcome the modeling problem of BB-PLLs Razavi suggested a modeling methodology for the specific case of analog BB-PLLs used in CDR applications in [18] (details were given in Chapter 1). In this work, a generic modeling methodology that can be applied for any BB-PLL is proposed, and applied on a BB-ADPLL.

2.4. PHASE STEP RESPONSE ANALYSIS

The phase step response of the BB-PLL system is a very powerful tool that can be used to analyze the system response and capture most of its characteristics. The phase step response can be used to estimate the system BW, and demonstrate the effects of

nonlinearity previously described. Moreover, the phase step response models the stage of tracking when disturbed by an error due to an external source such as voltage variations. Thus, by obtaining the phase step response of the BB-PLL to different phase error perturbations, the behavior of the system in a noisy environment can be predicted.

In order to estimate the BW of the system through the phase step response, the settling time is first obtained from the phase step response. The settling time is the time taken by the BB-PLL to recover from a phase step error. Since the settling time obtained by the phase step response indicates the speed of the loop in responding to change in input phase, then the BW as defined in part 2.3.2 can be represented as the reciprocal of the settling time. Figure 2-12-(a) aids this suggestion since the settling time for different PEMs in the case of the CP-PLL is constant, which translates into a constant BW. Moreover, in Figure 2-12-(b) the settling time for different PEMs in the case of the BB-PLL is dependent on the PEM, which is the same as the BW.

Furthermore, the phase step response of the BB-PLL reflects the nonlinearity effects of the system. For example, the phase step response of the system to a large PEM can show how stable the system is in the case of large PEMs, and whether the BB-PLL would be pulled out of lock due to the large PEM or not. Hence, this work is focused on modeling the step response of the BB-PLL in order to capture the nonlinearities in the system and design accordingly.

2.5. Conclusion

The building blocks of the BB-DPLL are similar to those of other DPLLs except for the BB-PD. The binary PD incorporated by the BB-PLL results in nonlinear loop dynamics. This nonlinearity of the system results in: difficulty in modeling the system, increased lock time, undefined BW and unreliable response. Finally, it was noticed that the phase step response of the BB-PLL capture all these effects of the nonlinearity on the system response. Thus, the phase step response is to be modeled in the Chapter 3.

CHAPTER 3 : SYSTEM MODELING

The nonlinearity of the BB-PLL system caused by the BB-PD makes the modeling of the system a daunting task as mentioned earlier in Chapter 2. The disability to model the BB-PLL system makes the design process of the system very difficult, since the designer starts building the circuit with no prediction of what output he will get, whether the system will be stable or not, and other important performance characteristics. Accordingly, the designer will have to run exhaustive simulations to verify the system functionality. The problem gets even worse given that DPLLs are mixed signal systems that require Analog-Mixed Signal (AMS) simulations. AMS simulations may take tens of hours to run depending on the system complexity.

On the other hand, having a model for the system before starting the design, saves a lot of time and effort, and guarantees functionality across different conditions, since the designer can run as many model simulations as he needs, because a model simulation takes only few seconds. Furthermore, having a system model gives good insight to the system dynamics, which allow the designer to tweak the system parameters according to the required specs.

Conventional modeling methodologies such as transfer function cannot be used due to the nonlinearity of the BB-PD. In this Chapter, a novel modeling methodology is proposed. The proposed modeling methodology is based on modeling the time-domain behavior of the individual blocks of the BB-PLL system, then predicting the system performance given the system parameters. Analogous to transfer function methodology used with linear systems, the proposed methodology introduces a link between system response parameters and system design parameters. This gives the designer the ability to tweak the design parameters through the proposed model in order to achieve the desired specs.

Thus, using the proposed modeling methodology, the designer can identify critical system response parameters such as BW and stability, and link them to the design parameters. Furthermore, the proposed modeling methodology can be used to define a

figure of merit (*SL*) that describes the system linearity. Using this figure of merit (*SL*), the linearity of the system can be assessed and compared to linearity of other systems.

In this Chapter, the response parameters to be predicted by the model, which are BW and stability, are defined. Then, the proposed modeling methodology is described. Later, the proposed modeling methodology is used to derive a model for the conventional BB-ADPLL system. Finally, the benefits gained by the model along with the system linearity figure of merit (*SL*) are illustrated.

3.1. NOTATIONS

Since this Chapter will contain a lot of math, it is better to start with defining the different parameters that will be used throughout the Chapter in order to make the logic easier to follow.

Figure 3-1 shows the phase step response of a BB-ADPLL. In Figure 3-1 the system is in lock, and then a phase step of magnitude (E_{p0}) is introduced at time (t_0). The first zero crossing on the phase error curve Figure 3-1 after the step is at time (t_1). The number of update cycles between t_0 and t_1 is m_1, i.e., the phase step is brought to zero after m1 update cycles ($m_1 D$ reference cycles). Similarly, the number of update cycles between any two consecutive zero crossings i-1, i is represented by m_i.

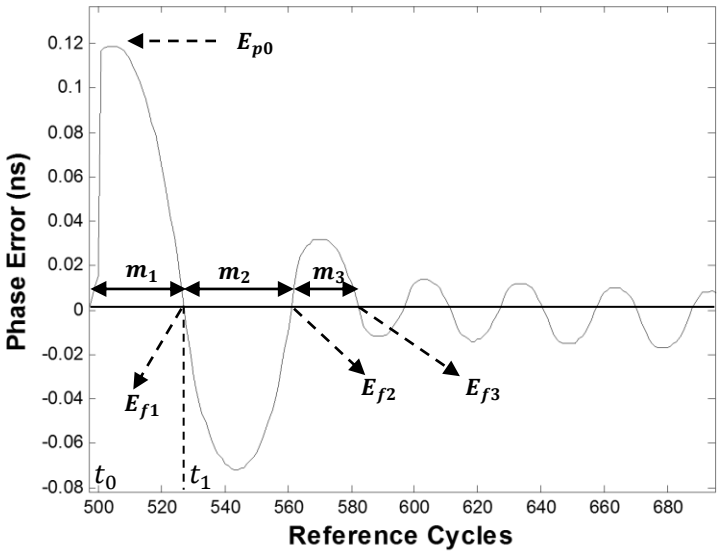

Figure 3-1. Phase step response of BB-ADPLL

Another important parameter to be used is the frequency error at the zero crossing points in Figure 3-1. The frequency error at the zero crossing i in Figure 3-1 will represented by E_{fi}. Figure 3-2 illustrates what is meant by the frequency error at the zero crossing points. Since the frequency error is the differentiation of the phase error, the frequency error at the zero phase error points has a maximum value as illustrated by Figure 3-2. The lower part of Figure 3-2 depicts the period of the feedback in the locked state oscillating around 10ns with a small value. This oscillation of the period has maxima at the zero phase error points on the upper part of Figure 3-2. Hence, there is a maximum frequency error at the zero crossings of Figure 3-25. This frequency error is represented by the parameter E_{fi}.

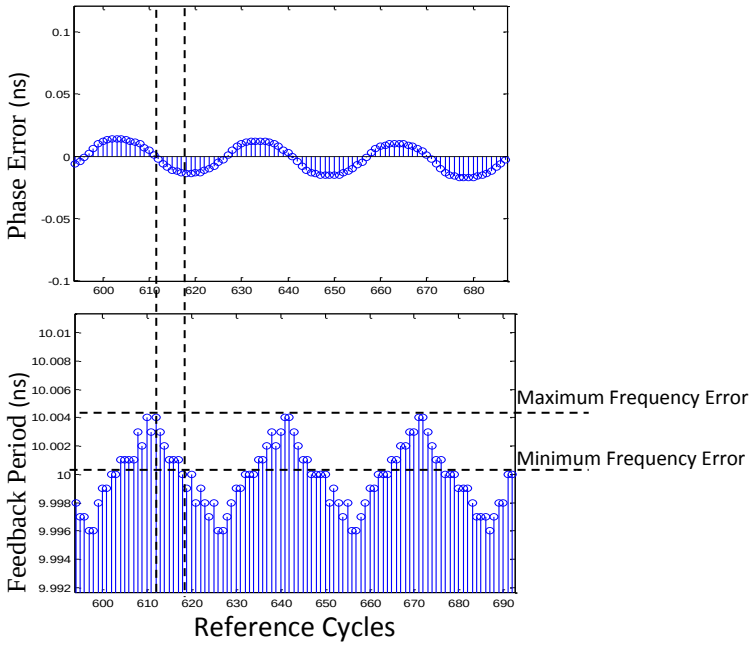

Figure 3-2. Maximum frequency error at zero phase error and vice versa

Finally, the change in the phase of the feedback signal due to one bit (LSB) change in the frequency control word (FCW) is denoted as $\Delta\emptyset$. Where, one step of the FCW (one bit change) causes the frequency of the DCO to change by Δf. This change in the frequency reflects on a change in the period of the DCO Δt, where Δt is given by: $\Delta t = \frac{\Delta f}{f(f+\Delta f)}$. Hence, the period of the feedback signal is changed by $N\ \Delta t$, where N is the frequency division ratio. A change in the period of the feedback is equivalent to a

change in the phase of the feedback signal such that $\Delta\emptyset = \frac{N\Delta t * 2\pi}{T}$, and T is the period of the feedback signal. Thus, this change in the phase of the feedback signal due to one step of the FCW is represented by the parameter $\Delta\emptyset$.

3.2. RESPONSE PARAMETRS: BANDWIDTH AND STABILITY

The phase step response varies depending on the PEM. The BB-PLL system responds differently to different PEMs in terms of BW and stability. Hence, to model the BB-PLL system, the BW and stability parameters need to be predicted for every possible PEM. The best way to capture the BW and stability of the system for different PEMs is to model the phase step response of the system. This way, the model takes the magnitude of the step as an input, which represents the PEM, and indicates the BW and stability of the system in response to that PEM.

In order to develop a modeling methodology that is able to predict the system response parameters, BW and stability, both parameters need to be quantified by a given expression. In the following sections, suggested expressions that quantify the BW and stability are derived.

3.2.1. BW Expression

The BW of the PLL can be well indicated by the settling time of the phase step response of the PLL as illustrated from the BW definition provided in Chapter 2. Hence, predicting the settling time is a good indication of the BW. For BB-PLLs the BW is dependent on the PEM, and so is the settling time. Hence, the objective of the model is to predict the settling time of the phase step response to different PEMs.

Form Figure 3-1 the settling time can be represented by the following expression:

$$T_{settling} = D\ (m_1 + m_2 + \cdots + m_n)\ Tclk \qquad (3.1)$$

where, $T_{settling}$ is the settling time, $Tclk$ is the period of the reference clock, and *n* is the zero crossing where the system is considered back in lock. *n* is chosen such that E_{fn} is smaller than a predefined value E_{limit} . The value of E_{limit} is the maximum frequency

error allowed in the locked state. Hence, predicting the values of m_1, m_2, ... m_n determine the settling time for the given PEM, and in turn, the BW.

3.2.2. Stability Limit

In linear PLLs the stability of the system can be determined by investigating the phase margin of the system. Unfortunately, the phase margin is obtained from the transfer function of the system, which is not available for the BB-PLL system, as previously mentioned. Hence, the stability limits for BB-PLLs need to be defined.

Since the model presented in this work is concerned with a time-domain analysis, a time-domain limit is defined for stability. The system is considered stable as long as the following relation holds:

$$E_{fi} < E_{limit} \quad \forall\, i > n \tag{3.2}$$

Relation (3.2) ensures that the maximum frequency error (frequency error at zero crossings) in the locked state stays smaller than the allowed limit for acceptable performance.

Figure 3-3 depicts possible trajectories of the frequency error at the zero crossings. Curve 1 is for an unstable system, i.e., a system that was pulled out of lock due to the phase step error, since the frequency error does not get below the defined limit (E_{limit}). Curve 2 also represents an unstable system, since the frequency error gets below the limit but does not stay at this state. Finally, Curve 3 represents a stable system.

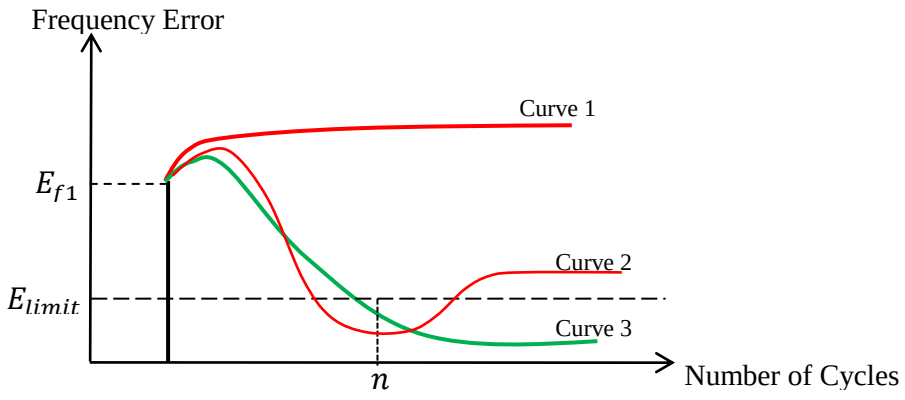

Figure 3-3. Trajectories of the frequency error in response to a phase step

Thus, given the values of E_{f1}, E_{f2},.. E_{fq}, where q is an arbitrary number larger than n, the stability of the system in response to the given PEM can be anticipated. Meanwhile, the values of E_{fi}, for $i > n$, are related to the jitter performance of the system.

Note that the term stability used in this work is referring to maintaining lock in response to phase step error, and not the ability of the system to acquire lock in the phase acquisition stage. Another important note is that $T_{settling}$ in (3.1) will only have a meaningful value if the condition in (3.2) is satisfied, otherwise, the system does not settle and $T_{settling}$ will assume a value of ∞.

3.3. MODELING METHODOLOGY

Based on the foregoing discussion in this Chapter, it is evident that the objective of the model is to predict the phase step response of the BB-PLL when the lock state is disturbed by a phase step of a certain magnitude. It has also been shown that to estimate the system response (BW and stability) the values of m_i, E_{fi} in Figure 3-1 need to be known. Thus, the proposed model will estimate the values of E_{fi} and $\mathrm{m_i}$ for a given PEM (E_{p0}) at a given frequency, and hence, determines system stability and settling time.

The PEM is a statistical variable whose distribution varies from one application to another. Accordingly, the impact of PEM should be statistically analyzed. One of the advantages of the proposed design methodology is that it can be plugged into a statistical design flow. However, since the scope of this work is to present the methodology itself, the simplistic case of a deterministic phase step error is considered. Yet, this work can be further extended to the case of a statistical time varying phase error, which will be considered in future work.

With the aid of Figure 3-1 the modeling methodology is presented as follows: the introduced phase step (E_{p0}) will cause a large phase error between the reference and the feedback signals. When this phase error is passed by the BB-PD a decision specifying the direction of the phase error is produced. The DLF then processes this decision and

produces a FCW to the DCO accordingly. The DCO will modify its frequency in order for the phase error to be eliminated. Because the phase error caused by the step is large, the change in the DCO frequency due to the loop action will not be enough to align the phase of the reference and feedback signals quickly. Thus, the direction detected by the BB-PD will stay constant for a while.

When the loop action compensates the phase step, and brings the phase error to zero, the direction of the BB-PD decision is flipped. Thus, the first change of polarity in the decision of the BB-PD denotes the first zero crossing point on the phase error curve in Figure 3-1. At this point the phase of the feedback signal ($\emptyset_1$) that has been accumulated over the time ($t_1 - t_0$) due to the loop action to compensate the step is equal to the phase step itself ($\emptyset_{e1}$). In the general case of a time varying phase error, the phase component $\emptyset_{e1}$ should be a function of time.

The phase component $\emptyset_1$ is composed of the DLF action over m1 update cycles, and the accumulation of the initial frequency error (E_{f0}) over $m_1 D$ reference cycles. E_{f0} represents the frequency error that exists at the moment the phase step is introduced. Although the PLL is initially in lock, the frequency error is never perfectly zero, rather, the frequency error oscillates around the zero within a small range. The oscillation of the frequency error around the zero in the locked state represents the jitter phenomenon in the output clock signal.

The DLF action contributing to the phase component $\emptyset_1$ can be calculated given the DLF architecture and the phase step of the DCO. For example, for the conventional DLF in Figure 2-5, the proportional path modifies the FCW by β steps over the time $(t_1 - t_0)$. Thus, the proportional path contribution to the phase component $\emptyset_1$ can be calculated by multiplying the number of FCW steps taken by the effect of one step on the phase of the feedback signal.

Thus, given the DLF architecture, the DCO phase step, and the PEM, both $\emptyset_1$ and $\emptyset_{e1}$ can be calculated in terms of m_1. Equating both phase components $\emptyset_1$ and $\emptyset_{e1}$ will give the value of m_1.

Since the PLL was initially locked before the phase step was introduced, then the initial frequency error is small, and thus, the frequency error at the first zero crossing point is mainly equal to the difference between the frequency at the first zero crossing point and the frequency at the initial time (t_0). This difference in frequency is due to the deterministic FCW steps taken due to the loop action over m1 while the loop worked to eliminate the phase error caused by the phase step. Thus, given the DLF architecture and m_1, the value of E_{f1} can be calculated.

Due to the relatively large frequency error E_{f1}, the DCO output frequency is no longer tightly locked to the target frequency. This directly reflects on the phase error, where the frequency error causes a phase error that increase over time due to the difference in frequency between the feedback and the reference signals. The loop slowly detects the phase error and starts to act through the DLF to eliminate the phase error reaching the second zero crossing point after m_2 update cycles on the phase error curve in Figure 3-1.

The concept of equal phase components at the first zero crossing that was used to determine m_1, E_{f1}, still holds, and can be used to get m_2, and E_{f2}. Between the first and second zero crossing points there is two phase components opposite in direction, and cancel at the second zero crossing point. The first phase component ($\emptyset_{e2}$) is composed of the accumulation of the frequency error at the first zero crossing point (E_{f1}) over $m_2 D$ reference cycles. The phase component $\emptyset_{e2}$ will be referred to as the error component. The second phase component ($\emptyset_2$) is the loop action component. Similar to the phase component $\emptyset_1$ at the first zero crossing point, the phase component $\emptyset_2$ is composed of the DLF action over m_2 update cycles multiplied by the DCO phase step.

Thus, given the DLF architecture and the DCO phase step, the loop phase component $\emptyset_2$ can be calculated, and given the frequency error at the first zero crossing point (E_{f1}), the error phase component $\emptyset_{e2}$ can be calculated. By equating the two phase components $\emptyset_2$ and $\emptyset_{e2}$, the number of update cycles at the second zero crossing point (m_2) can be calculated. Consequently, the frequency error at the second zero crossing

point (E_{f2}) can be calculated as the difference between E_{f1} and the FCW steps taken due to the DLF action over m_2 .

Following the exact same logic in determining the values of m_2 , and E_{f2}, the value of all m_i , and E_{fi} can be calculated for a given PEM, given DLF parameters, and a given DCO phase step.

Thus, the settling speed and the stability of the system in response to any PEM can be predicted in terms of the loop filter parameters and the oscillator gain. Hence, the design could be adjusted to give the desired response to a certain range of PEMs depending on the application. To illustrate the proposed modeling methodology, a complete example is presented next.

The advantage of the proposed methodology is that it is generic for any digital BB-PLL architecture. It can be used with the conventional DLF, or any other architecture. Also, it can be used for the simplistic case of phase step or for the more general case of random time varying phase variation.

3.4. MODEL DERIVATION

3.4.1 Mathematical Derivation

In this section, the proposed modeling methodology is used to derive a model for the conventional BB-ADPLL in Figure 2-10. The inputs to the model are the magnitude of the phase step error (PEM), the DLF parameters (α, β), the update rate (D), and the DCO phase step ($\Delta\emptyset$). The expected outputs of the model are the values of m_i , E_{fi} for all $i < q$. The outcomes of the model are to use the values of m_i , E_{fi} in predicting the BW and stability of the system for the given PEM.

As discussed in the previous section, the phase component $\emptyset_{e1}$, at the first zero crossing, is caused by the step error (E_{p0}) as well as the initial frequency error (E_{f0}) that exists at the moment the step error is introduced. Unlike the static phase error (E_{p0}), the frequency error (E_{f0}) causes phase accumulation over m_1 update cycles (Dm_1 reference cycles). Thus, $\emptyset_{e1}$ is given by:

$$\emptyset_{e1} = E_{p0} + E_{f0}\, Dm_1\, \Delta\emptyset \tag{3.3}$$

where $\Delta\emptyset$ is the effect of one DCO step on the phase of the feedback signal, i.e., $\Delta\emptyset$ represents the DCO time domain gain multiplied by the frequency division ratio (N) as illustrated above in section 3.1. Note that E_{f0} is a small value dithering around zero since the PLL was originally locked. For worst case analysis, the maximum value of E_{f0} is used in this model. Determining the value of maximum E_{f0} depends mainly on the acquisition scheme used and can be estimated by the designer. For the system in consideration, the value of the proportional coefficient β is a good estimate of the maximum E_{f0}.

The second phase component $\emptyset_1$, which is caused by the DLF action during m_1 update cycles is the superposition of the proportional path contribution and the integral path contribution to the phase. The proportional path updates the FCW by β steps over the m_1 update cycles. The direct effect of β steps of the FCW on the phase is $\beta\Delta\emptyset$. Moreover, this direct effect on the phase is accumulated over the time t_1, i.e., over Dm_1 reference cycles. Thus, the contribution of the proportional path on the phase component $\emptyset_1$ is

$$\emptyset_{1,due\ to\ proportional} = \beta\, Dm_1\, \Delta\emptyset \tag{3.4}$$

Modeling the effect of the integral path on the phase $\emptyset_1$ is a bit more complicated. The integral path adds a number of α steps to the FCW every update cycle (every D reference cycles). First, the effect of α FCW steps will accumulate over Dm_1 reference cycles. Then, after D reference cycles another α FCW steps are added, and will accumulate over $D(m_1 - 1)$ steps, and so on. This effect of the integral path on the phase can be represented by the following equation:

$$\emptyset_{1,due\ to\ integral} = \alpha\, Dm_1\, \Delta\emptyset + \alpha\, D(m_1 - 1)\, \Delta\emptyset + \cdots + \alpha\, D\Delta\emptyset \tag{3.5}$$

Equation (3.5) can be simplified in a closed form as follows:

$$\emptyset_{1,due\ to\ integral} = \alpha\, D\, \frac{m_1(m_1+1)}{2}\, \Delta\emptyset \tag{3.6}$$

By combining (3.5) and (3.6) the phase component $\emptyset_1$ can be expressed by:

$$\emptyset_1 = \alpha\, D\, \frac{m_1(m_1+1)}{2}\, \Delta\emptyset + \beta\, D m_1\, \Delta\emptyset \tag{3.7}$$

By equating both phase components $\emptyset_1, \emptyset_{e1}$ the value of m_1 can be calculated. Note that $\emptyset_1, \emptyset_{e1}$ represent the summation of phase over a large number of DCO cycles. Hence, the effect of phase noise is averaged and can be neglected in this analysis.

The number of FCW steps (Z_i) between crossing point $i-1$ and crossing point i, is due to both proportional and integral coefficients and is given by:

$$Z_i = \alpha\, m_i + \beta \tag{3.8}$$

The frequency error at the first zero crossing point is the difference between the number of FCW steps and the assumed initial frequency error:

$$E_{f1} = Z_1 \Delta\emptyset - E_{f0} \tag{3.9}$$

Similarly, at any zero crossing point there are two phase components equal to each other, one due to frequency error at the previous zero crossing point ($\emptyset_{ei}$), and one due to DLF action ($\emptyset_i$). A general form for $\emptyset_{ei}$, for all $i > 1$, is:

$$\emptyset_{ei} = D\, m_i\, E_{fi-1} \tag{3.10}$$

The term ($D\, m_i\, E_{fi-1}$) represents the phase accumulated over Dm_i reference cycles due to the frequency error at the previous zero crossing (E_{fi-1}).

Similar to equation (3.4), a general form for $\emptyset_i$, for all $i > 1$, is:

$$\emptyset_i = \alpha\, D\, \frac{m_i(m_i+1)}{2}\, \Delta\emptyset + D m_i\, \beta\, \Delta\emptyset \tag{3.11}$$

The value of m_i can be determined by equating (3.7), (3.8). Consequently, a general form for the frequency error (E_{fi}), for all $i > 1$, is given by:

$$E_{fi} = Z_i\, \Delta\emptyset - E_{fi-1} \tag{3.12}$$

Equation (3.12) is derived similar to equation (3.9). The first term of the right hand side (RHS) represents the total FCW steps between point $i-1$ and point i, including the effect of the proportional action (β) at the previous zero crossing point $i-1$. The second term of the RHS is the previous frequency error E_{fi-1} that is to be subtracted from the total FCW steps to give the current frequency error.

From the derived system of equations all values of m_i and E_{fi} can be found, and hence the stability and settling of the system can be determined for a given frequency and PEM according to (3.1) and (3.2). Note that the model assumed the zero crossing points to be multiples of the update rate D for simplicity, which might cause slight error in the calculations. However, the model is more concerned with trend rather than accuracy.

For convenience, equations (3.1 – 3.12) that represent the proposed model can be easily combined in Matlab to provide the system response for a given range of frequencies and range of PEMs at once.

3.3.2 System Linearity Figure of Merit

In order to represent system linearity, the settling time in response to different PEMs is estimated. The dependence of the settling time on the PEM reflects the nonlinearity of the system. Accordingly, a figure of merit (*SL*) is defined to quantify the nonlinearity of the system.

Based on equations (3.7), The phase of the feedback signal ($\emptyset_{fb}$) at any time t can be calculated given that the number of cycles Dm_i can be substituted by the time t. Thus, the phase is given as a function of time by:

$$\emptyset_{fb}(t) = \left(\beta t + \frac{\alpha}{D}\frac{t\,(t+1)}{2}\right)\Delta\emptyset \tag{3.13}$$

At the time (t_1) the output phase $\emptyset_{fb}(t_1)$ equals the step error E_{p0}. This time (t_1) coincides with the first zero crossing on the phase error curve in Figure 3-1, where the phase error equals to zero. Thus, substituting in (3.13), the value of t_1 is given by:

$$t_1^2 + \left(\frac{2D\beta}{\alpha} - D\right)t_1 - \frac{2D}{\alpha}\frac{E_{p0}}{\Delta\emptyset} = 0 \tag{3.14}$$

Equation (3.14) is in good agreement with the expression derived in [12] that estimates the phase step response of BB-PLL system up to the first zero crossing point (t_1). From equation (3.14), the value of t_1 can be calculated for a given phase step magnitude (E_{p0}). Getting different values of t_1 for different phase step magnitudes is a good indication of the system linearity. A more linear system is one that shows less dependence in the value of t_1 on the value of E_{p0}, which translates into a constant BW, as in the case of CP-PLL illustrated in Figure 2-12.

In order to quantify the improvement in system linearity due to the proposed DLF, a figure of merit (SL) that indicates system linearity is defined as follows:

$$SL = \frac{\Delta t_1}{\Delta(phase\ error\ magnitude)} \tag{3.15}$$

The smaller the value of SL, the more linear the system is. For a perfectly linear system $SL = 0$, indicating a perfectly constant BW. Equation (3.14) is implemented in Matlab, and values of SL for different phase error magnitudes are obtained for both systems. The results are presented in Chapter 6.

3.5. Conclusion

The proposed modeling methodology can predict the phase step response of any BB-DPLL. The proposed modeling methodology is based on representing the time domain behavior of the system by mathematical equations. These equations were used to model a BB-ADPLL system, and predict its phase step response. In light of the system mathematical representation, a figure of merit (*SL*) that quantifies the linearity of the system was defined. In Chapter 4, the same BB-ADPLL system is implemented on the circuit level. The phase step response of the system will be obtained from AMS simulations, and compared to the predictions of the model in this Chapter.

CHAPTER 4 : BB-ADPLL SYSTEM IMPLEMENTATOIN

In order to verify the model derived in Chapter 3, an All-Digital Bang-Bang PLL is implemented on the circuit level in this Chapter. First, the implementation of each building block is described, along with the whole system operation. Then, the implemented system is simulated (AMS simulations), and the performance of the system is illustrated. Finally, the response obtained by AMS simulations is compared with the response predicted from the model in Chapter 3.

4.1. SYSTEM IMPLEMENTATION

Since the purpose of implementing the system is to verify the model, the conventional BB-ADPLL architecture in Figure 2-10 is used in this work. The BB-ADPLL implemented here targets clock synthesis for microprocessor applications. Hence, the target range of frequency is in the range of 1GHz – 4GHz.

Figure 4-1 shows the detailed block diagram of the implemented system. Figure 4-19 shows the pin diagram of the system. Except for the DCO, all other building blocks of the system are designed using an RTL flow, i.e., a Verilog system representation to be synthesized into standard cells. The DCO is a custom designed ring oscillator. In practice, the integer-N frequency divider should be custom designed as well. This is because the frequency divider has the high frequency DCO signal as an input. Standard cells cannot handle such high frequency (up to 5 GHz). The RTL code, however, can handle the high frequency signals. Since the main purpose of implementing the system is to verify the model, the system was only implemented in RTL, but was not synthesized. Further verification by taking the system into the physical design stage will be considered in future work.

The BB-ADPLL in Figure 4-1 uses a frequency acquisition loop to aid the frequency acquisition stage as illustrated in Chapter 2. The BB-ADPLL system starts its operation due to a reset signal, or due to a change in the value of the division ratio N, which indicates a change in the target frequency. When the system starts operation, it first

starts in the frequency acquisition loop shown in Figure 4-1. The DCO produces a high frequency signal that is directly fed to the Frequency Comparator (FC). The output of the frequency comparator is processed by the Binary Search Algorithm (BSA) block. The action taken by the BSA block is first decoded by a thermometer decoder, and then fed to the DCO to tune the frequency in order to approach the target frequency.

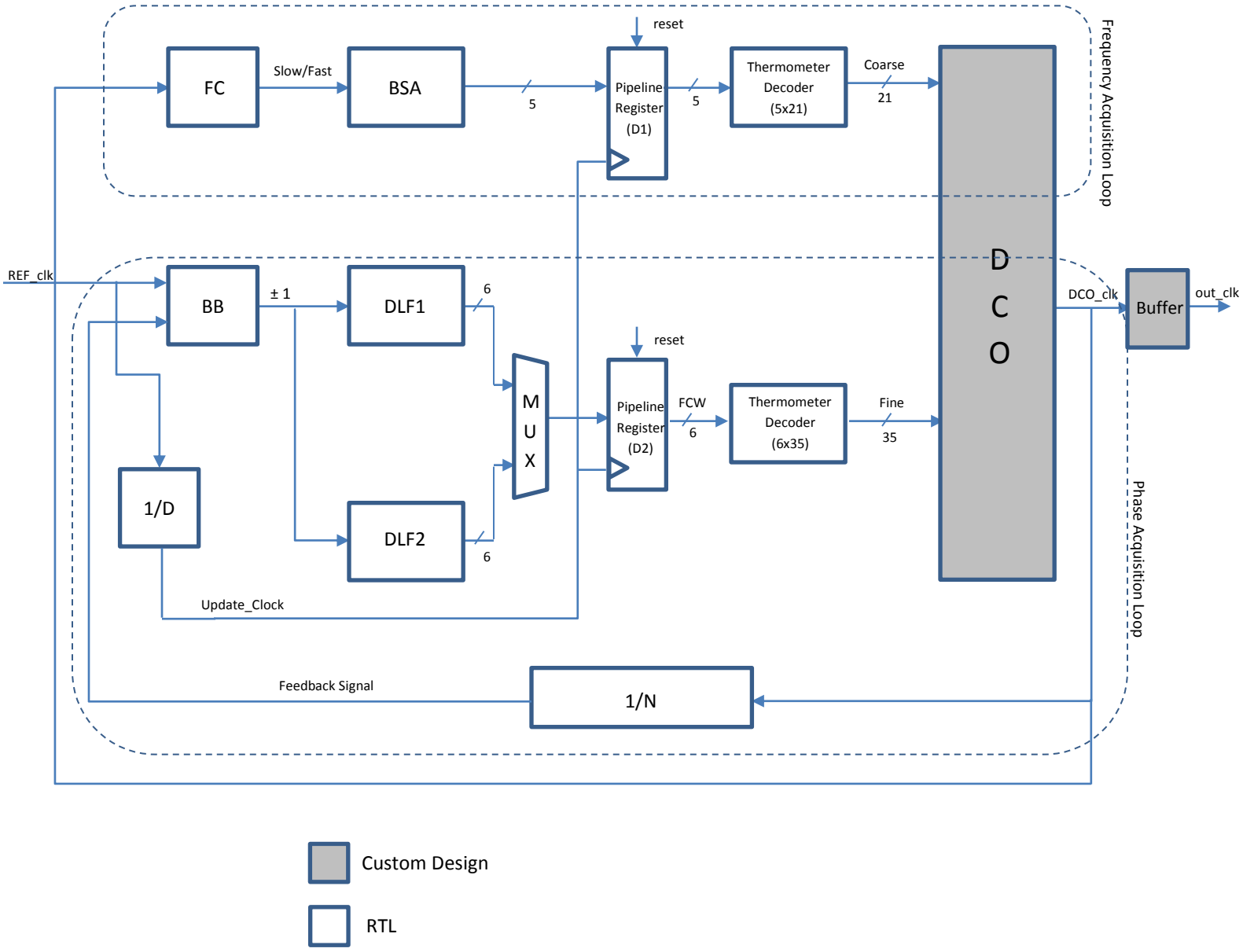

Figure 4-1. Architecture of the implemented BB-ADPLL

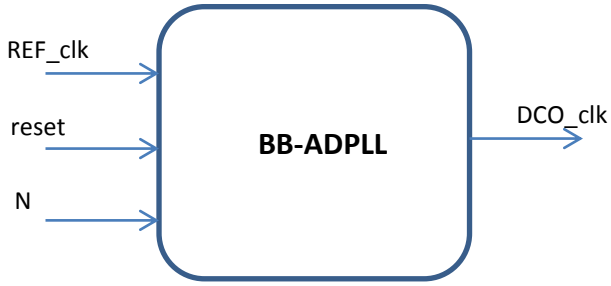

Figure 4-2. Pin diagram of the BB-ADPLL

Once the frequency acquisition stage is finished, the loop operates through the phase acquisition loop shown in Figure 4-1. The DCO frequency is divided by the division ratio N to produce the feedback signal. The error signal detected by the phase detector is processed by the DLF then decoded by a thermometer decoder to control the DCO. Finally, the output of the DCO is buffered in order to get a sharp square wave (out_clk). The details of the circuit implementation of each building block are described in this Chapter.

For increased tracking resolution and enhanced performance, the phase acquisition loop operation is divided into two similar sub-loops. The first sub-loop (the one using DLF1 in Figure 4-1) is used directly after the frequency acquisition loop is done, and lasts for a short period of time. Then, the second sub-loop (using DLF2 in Figure 4-1) starts its operation and stays active until the PLL is reset. The first sub-loop uses the loop filter DLF1, and controls certain bits in the DCO called “fine” bits. The second sub-loop uses the loop filter DLF2. The output of DLF2 is multiplexed with that of DLF1 to control the “fine” bits. DLF1 and DLF2 are the exact same filter architecture but with different parameters values. The reason behind this is that DLF1 has the values of the parameters (β, α) chosen such that the lock time spec is minimized, while the parameters of the DLF2 has its parameters chosen to minimize the jitter. Thus, the architecture in Figure 4-1 optimizes both the lock time and the jitter.

Note that the DCO is controlled through a pipeline register. The clock of the pipeline registers are controlled by the value of the update rate D. The update rate D takes different values from one stage to another.

4.1.1. Digitally Controlled Oscillator (DCO)

The DCO implemented in this system is a seven stage ring oscillator shown in Figure 4-3. A digitally controlled ring oscillator consists of a chain of an odd number of inverters. The size of each inverter can be controlled through an input digital word. The output frequency of a ring oscillator is: $F_{dco} = 1/(N*t_d)$, where N is the number of inverters in the chain, and t_d is the delay of one inverter. As the control word increases, the delay of each inverter decreases, i.e., the output frequency of the oscillator increases. Thus, the minimum output frequency is achieved when the FCW equals to zero. The delay of one inverter (t_d) used in this design is in the range of 100ps. Thus, to achieve a minimum frequency in the range of 1GHz, seven stage ring oscillator is used.

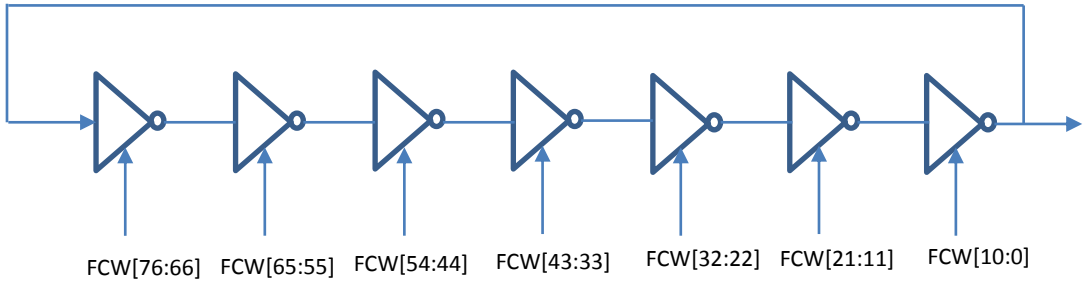

Figure 4-3. architecture of the DCO

Figure 4-4 shows one cell of the conventional digitally controlled ring oscillator used in the literature. The DCO cell in Figure 4-4 consists of a main inverter, a pair of transistors for the fine control, and a pair of transistors for the coarse control. The control transistors (fine or coarse) are connected to a number of transistors in parallel. The coarse transistors have relatively smaller size, and in turn, a relatively higher resistance. Hence, the frequency step due to the addition/subtraction of one coarse transistor is large. This is why the coarse transistors are used in frequency acquisition stage, because the loop needs to take large steps in frequency to jump fast to the range of the target frequency. The fine transistors, on the other hand, are of relatively larger sizes. Thus, the fine step is a small frequency step, and is used in the phase acquisition and tracking stages.

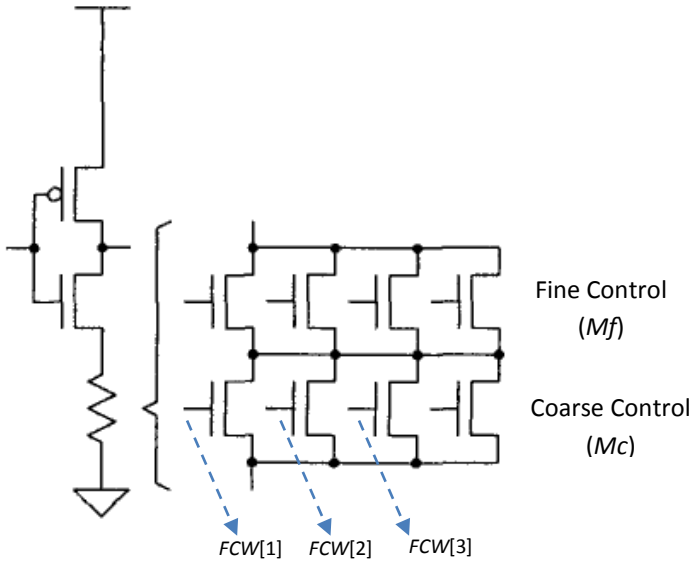

Figure 4-4. Conventional RO DCO [30]

The gates of the parallel transistors are connected to bits of the FCW. When all bits of the FCW are zeros, all the parallel transistors are off, and the DCO operates at the minimum frequency. When one bit of the FCW is high, say *FCW*[1] in Figure 4-4, the equivalent resistance of the coarse transistor *Mc* is decreased. Consequently, the delay of the cell is decreased. Thus, as the FCW increases, the delay of the cell decreases and the output frequency of the DCO is increased. Note that Figure 4-4 shows only the lower part of the DCO, where a similar coarse and fine set of PMOS transistors actually exist in series with the PMOS.

A similar architecture to the conventional DCO in Figure 4-4 is used here with a small modification as shown in Figure 4-5. The modification is to add one more control transistor with its parallel transistors for better frequency control. However, the disadvantage of this solution is the long stack of transistors that may require large voltage headroom. For the setup used in this work, the supply voltage is 1.05 V. The proposed DCO architecture is verified to operate properly for a variation of +/- 20% of the supply voltage. However, the concern is that the proposed solution may not be scalable for more advanced technologies with lower supply voltage.

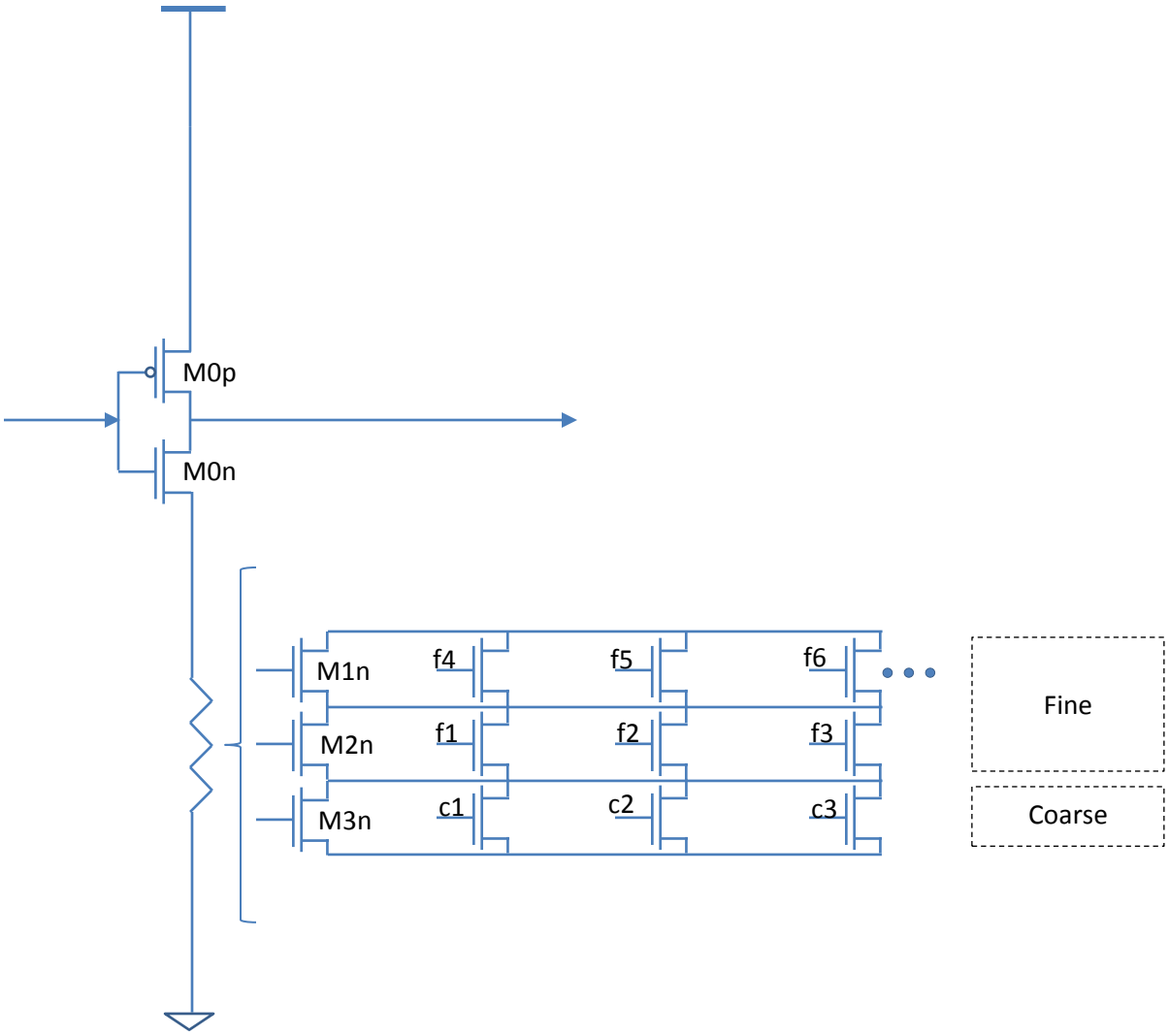

Figure 4-5. Architecture of the proposed DCO

The DCO architecture in Figure 4-5 is composed of the main inverter (M0n, M0p), main fine tuning transistors (M1n, M1p, M2n, M2p), and main coarse tuning transistors (M3n, M3p). Note that the PMOS transistors are not shown in the figure. Each of the main control transistors (fine, coarse) is connected in parallel to a number of control transistors that are controlled by the FCW bits. The main coarse transistor is connected to 3 parallel transistors. The main fine tuning transistors (M1, M2) are connected to 8 parallel transistors. Thus, there is 11 tuning transistors per DCO single cell. For the seven stages DCO used, there is a total of 77 tuning transistors. Note that the control bits are connected directly to the gates of the NMOS transistors, and through an

inverter to the PMOS transistors (not shown in figure). Thus, when one control bit goes high, it opens both the NMOS and PMOS transistor connected to it.

All the coarse control bits across the seven cells of the DCO (21 bits) form the "coarse code". Similarly, the "fine code" is composed of fine control 35 bits, where some of the fine tuning transistors shares the control bits. Thus, the number of the control bits is less than the number of the tuning transistors.

The main challenge in designing the DCO in Figure 4-5 is the sizing of the transistors. In order to get the desired frequency range and frequency step, the sizing of the transistors is chosen as shown in Table 4.1. Intuitively, the size of the main fine tuning transistor (M1) is the smallest such that an addition of a parallel transistor causes a small step in frequency. Similarly, the size of the main coarse transistor is the largest, and the size of the main fine transistor (M2) is in between. Note that in Table 4.1 the transistors with two subscripts, such as m31, represent the sizing of the parallel transistors.

Table 4.1 Sizing of the transistors of the DCO

m0= 35 * 0.5 um	m1= 50 * 0.5 um m11= 6 * 0.5 um m12= 6 * 0.5 um m13= 6 * 0.5 um m14= 6 * 0.5 um m15= 6 * 0.5 um	m2= 15 * 0.5 um m21= 4 * 0.5 um m22= 5 * 0.5 um m23= 6 * 0.5 um	m3= 2 * 0.325 um m31= 3 * 0. 325 um m32= 4 * 0. 325 um m33= 7 * 0. 325 um

The sizes of the parallel transistors are chosen to increase progressively. The reason behind this is that as parallel transistors are added, the frequency is increased. At higher frequencies the same change in the control transistor results in a larger frequency step. In order to compensate this effect, the transistors are sized progressively.

A note worth mentioning is that although any increase in the size of the main control transistors (M1, M2, M3) increases the output frequency, any increase in the size of the main inverter transistors (M0) decreases the output frequency. This is because any increase in the size of the control transistors only decreases the resistance of one cell.

However, an increase in the size of the main inverter transistors decreases the resistance of one cell and in the same time increases the capacitance of the next cell.

Finally, note that the main coarse transistor of the seventh stage has a bit larger size compared to other stages. This is because the seventh stage is the one responsible for the output clock. Faster transistors will result in a sharper slope in the transitions of the output clock. The coarse transistor specially is oversized, no other transistors, because it is the smallest transistor, and has more control. In other words, the coarse transistor can be thought of as the bottle neck in the cell for the current passing. Thus, increasing its size (decreasing its resistance) allows more current to pass, and hence, sharper transitions. The size used for M3 at the seventh stage is 4*0.325 um (not in Table 4.1)

Figure 4-6 and Figure 4-7 show the DCO output waveform at a certain control code and different supply voltages. In order to demonstrate the relation between the control codes and the frequency, the following curves are obtained. Figure 4-8 shows the impact of coarse tuning on the output frequency in case of ±20% supply variations. Figure 4-9 shows the fine control in the tracking stage (through DLF2) against frequency for a coarse code = 8 (decimal conversion of 35 binary bits). Table 4.2 shows the coarse step, and the fine step of the DCO. The fine step of the DCO is considered the DCO resolution.

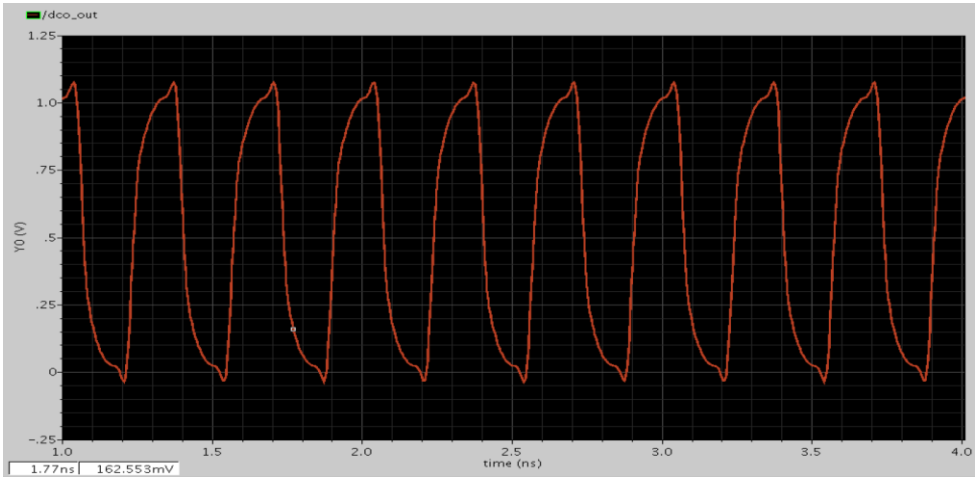

Figure 4-6. DCO output waveform at coarse code = 8, fine code = 16, supply=1.05V, output frequency is 3GHz

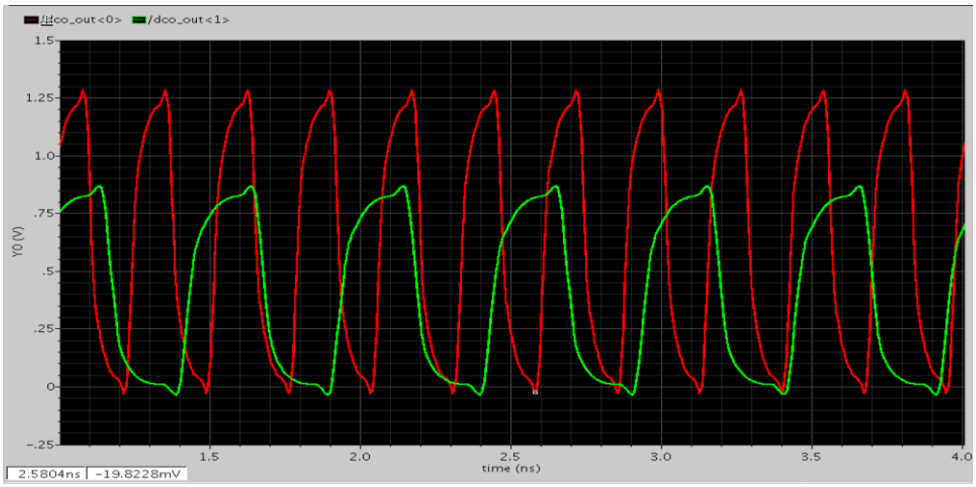

Figure 4-7. DCO output waveform at same control code, supply ±20%, output frequency is 3.7 GHz at 1.26 V supply, and 2 GHz at 0.84 V supply

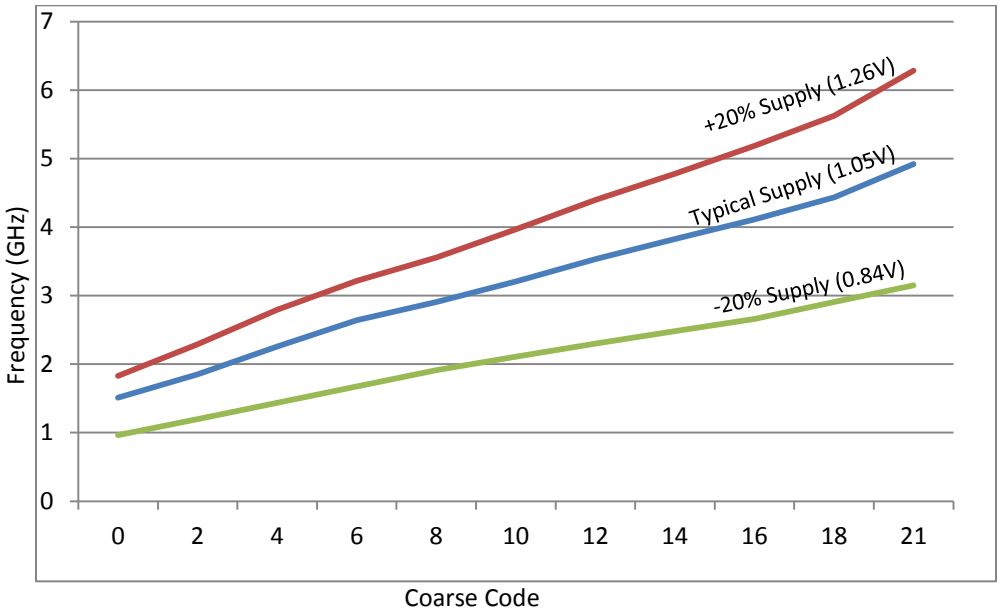

Figure 4-8. DCO frequency versus the range of the coarse code

In Figure 4-8 the range of the DCO frequency for the typical supply is 1.5GHz – 5GHz. Note that the fine code is fixed to their middle range. This means that the actual frequency range is a bit larger. The actual frequency range of the DCO is (1.429 – 5.263) GHz.

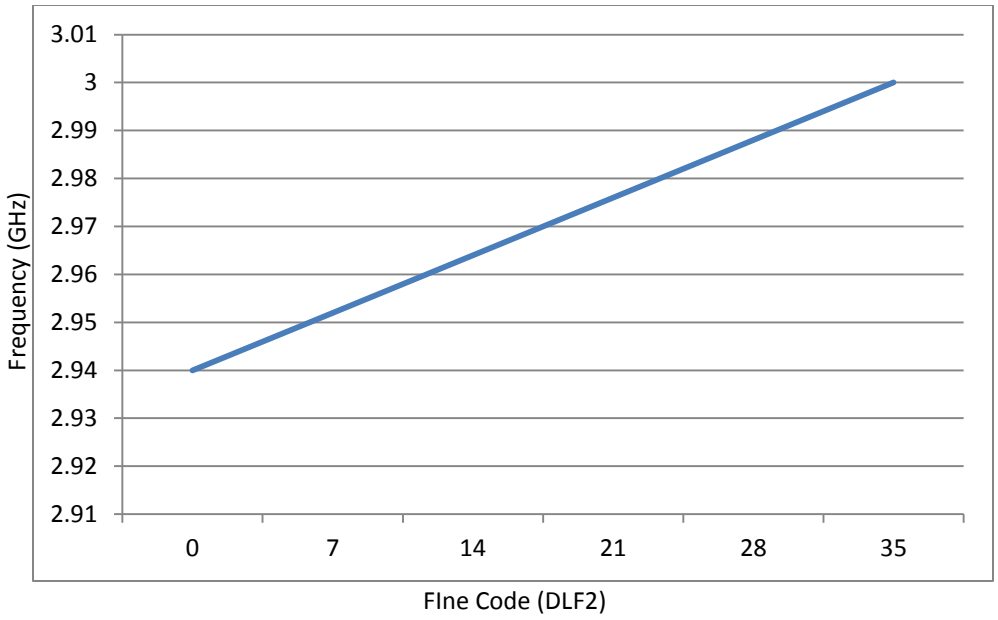

Figure 4-9. DCO frequency versus the range of the fine code using M1 at coarse = 8

Table 4.2 Frequency steps of the DCO

Frequency (GHz)	Coarse Step (MHz)	Fine Step (MHz)
1.5 (Min)	171.5	0.7
3 (Average)	149.5	1.45
4.9 (Max)	243	2.5
Average ➡	**162.3**	**1.55**

4.1.2. Frequency Comparator

The purpose of this block is to compare the output frequency to the target frequency, and gives a binary signal indicating whether the output frequency is slower/faster than the target frequency. Figure 4-10 describes the architecture of this block. The output of the DCO is passed by a counter that is reset at the rising edge of the reference clock. In case the DCO frequency is equal to the target frequency, it is expected

that the output of the counter is the division ratio N. Thus, the output of the counter is compared through a comparator with the value of N. In case the count is smaller than N, the frequency comparator gives a low signal indicating “slow”, otherwise, it gives a high signal indicating “fast”.

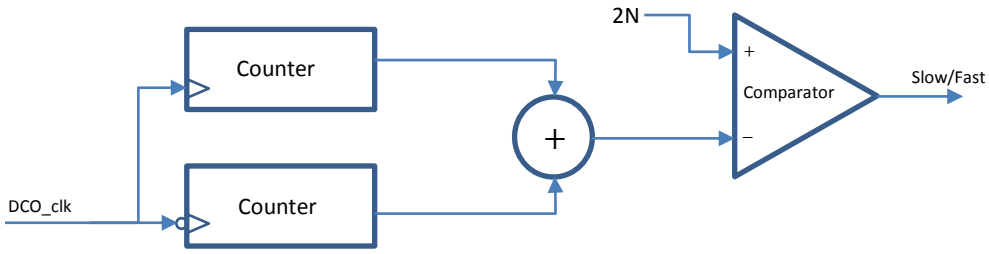

Figure 4-10. Architecture of the Frequency Comparator

Of course the maximum error resolution of the architecture described above is in the DCO cycle, i.e., if the difference between the DCO frequency and the target frequency is smaller than one DCO cycle, the frequency comparator cannot detect it. Note that in Figure 4-10 there are two counters. One counts the positive edges of the DCO signal, and the other counts the negative edges of the DCO signal. Thus, the resolution of the frequency comparator is doubled to half of a DCO cycle. Since two counters are used, the output of the counters is summed and compared to $2*N$ as in Figure 4-10.

Note that in practice, if the architecture in Figure 4-10 is to be used as is, then it has to be custom designed, since the standard cells will not be able to handle the high frequency of the DCO. However, the frequency comparator circuit can be built using standard cells if the DCO signal is divided by a small ratio, say 4, to down convert the high frequency of the DCO to a reasonable frequency for the standard cells. This solution, of course, comes at the cost of lower error resolution of the frequency comparator.

4.1.3. Binary Search Algorithm (BSA)

The BSA in the frequency acquisition loop, as well as the DLF in the phase loop, modify the FCW that controls the frequency of the DCO. For the DCO architecture used in this implementation the FCW consists of 77 bits, which is composed of concatenating

the {coarse code, fine code} respectively. The amount of change in the FCW imposed by the BSA or by the DLF is called the FCW gain. If the change in the FCW affects only the coarse bits, then the amount of change in the coarse bits is called the coarse gain.

The BSA affects only the coarse bits by giving values of the coarse gain. Based on the output of the frequency comparator, the BSA uses the logic flow shown in Figure 4-11 to determine the value of the coarse gain, which in turn determines the coarse code that will be passed to the DCO.

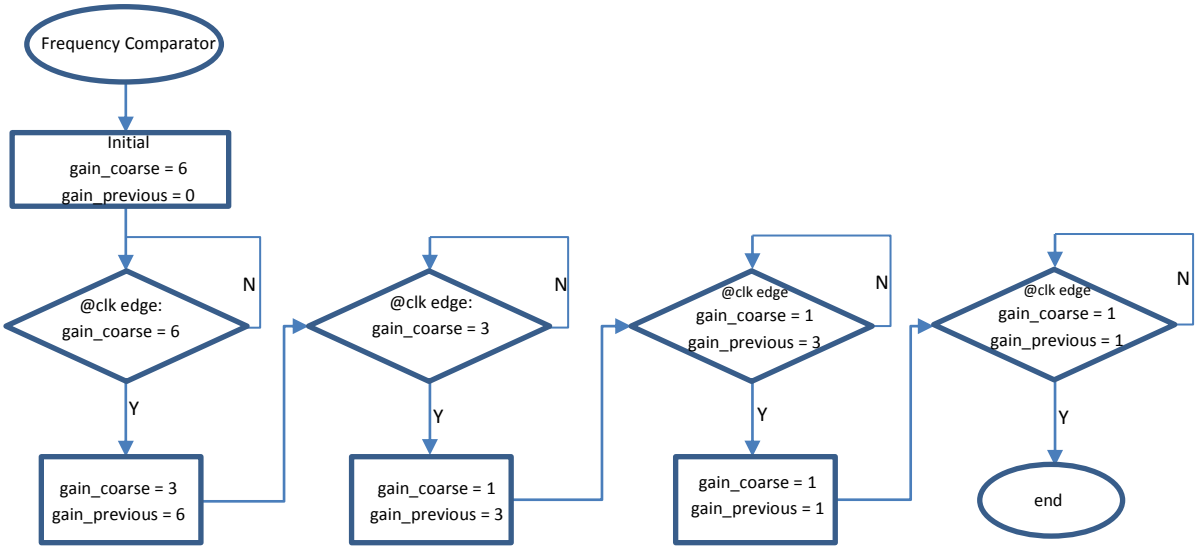

Figure 4-11. Flow Chart of the BSA logic

Figure 4-12 illustrates the operation of the BSA. The coarse code starts initially at the middle of its range at the code 11 (decimal value of the code). The maximum decimal value of the code is 21, and the minimum value is 0. Based on the slow/fast decision of the frequency comparator, the coarse code is increased/decreased in a binary manner. The gain added/subtracted from the code is first 5, then 3, then 1, then 1 again, and finally is zero indicating the end of the frequency acquisition stage.

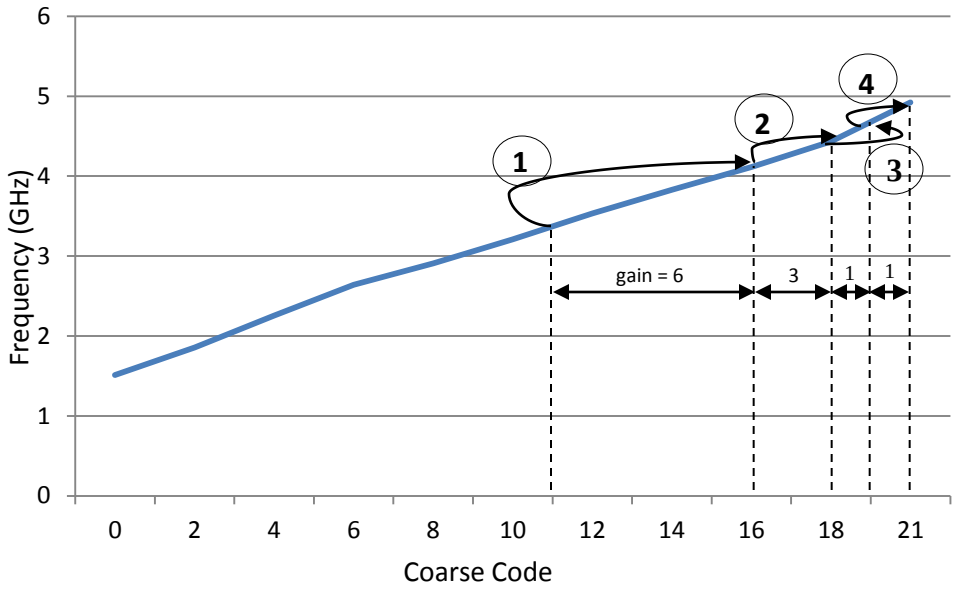

Figure 4-12. Operation of the BSA illustrated on the coarse code curve – illustrating example of the operation at target code 21

For example, let µs consider the scenario where the target frequency is closest to the maximum frequency in the DCO range. Thus, the target coarse code is 21. The coarse code starts at 11, and the BSA gets a slow signal. First, the code is increased by 5 to be 16, and still gets slow signal. Then, the code is increased by 3, then 1, and then 1 again, and the code is 21. At this time, the coarse gain becomes zero, which signals the loop to move from using the frequency acquisition loop to using the phase loop.

4.1.4. Frequency Divider

Once the system is operating through the phase loop the DCO signal is first passed by the integer-N frequency divider. Figure 4-13 shows the architecture used for the frequency divider. The frequency divider consists of a counter that counts the positive edges of the DCO signal. The count is compared with the division ration N. When the count is equal to N, the counter is reset. The output of the counter is the divided feedback signal, and is also used to reset the counter as shown in Figure 4-13. An example wave form of the divider output is shown in Figure 4-14.

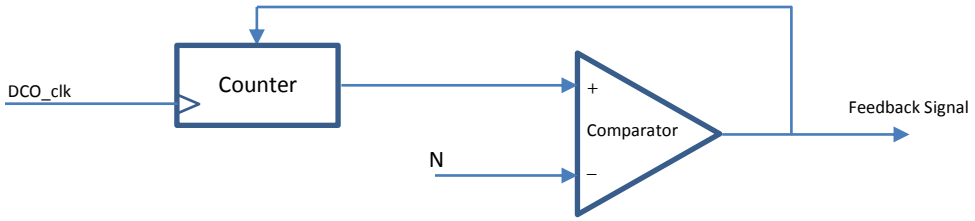

Figure 4-13. Architecture of the Frequency Divider

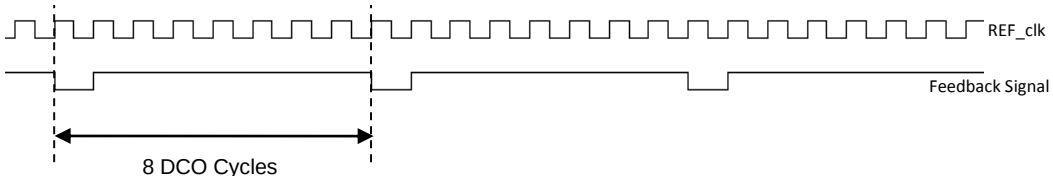

Figure 4-14. An example waveform of the divided signal (feedback signal) with N=8

From Figure 4-14, the duty cycle of the divided signal is not 50%. However, the proper operation is only concerned with the correct positive edge position. This is because the BB-PD compares the rising edges only of the feedback signal and the reference signal.

4.1.5. Bang-Bang Phase Detector (BB-PD)

The simple theoretical BB-PD is just a D flip-flop with one of the two signals to be compared as the clock input, and the other as the D input. In practice, this simple architecture causes the well-known issue of bang-bang PLLs, cycle slips. For example, if the D signals frequency is much higher than the clock signal frequency, then the sampling clock will slip some cycles of the data signals rendering a wrong decision. Consequently, in practice, some additional circuitry is added to the single flip-flop in order to minimize the effect of the cycle slips.

In this design, the use of a frequency acquisition loop ensures that the feedback signal, and the reference signal are of the same frequency, and that the error between the two signals is in the range of half DCO cycle (as illustrated in section 4.2). Hence, it is justified that the simple single flip-flop architecture can be used for this design. Although the phase acquisition stage can change the frequency of the feedback signal causing cycle

slips to happen, the frequency of cycle slipping would be low, and would not affect the operation seriously.

Thus, the single flip-flop architecture in Figure 2-1 is used. The feedback signal is used as the clock of the flip-flop, and the reference signal is used as the data. Figure 4-15 is a snapshot of the BB-PD operation. In Figure 4-15, the feedback signal samples the REF_clock at the rising edge, and the sampled data represent the inversion of the BB-PD output. In the locked state, the output of the BB-PD changes polarity more rapidly since the DCO frequency oscillates around the target frequency with a small error. Therefore, the output of the BB_PD is often used as a lock indicator.

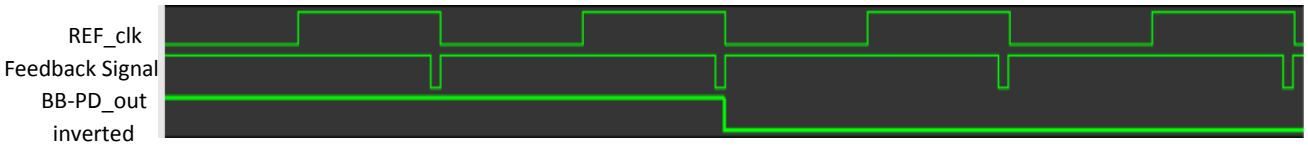

Figure 4-15. BB-PD output waveform

4.1.6. Digital Loop Filter (DLF)

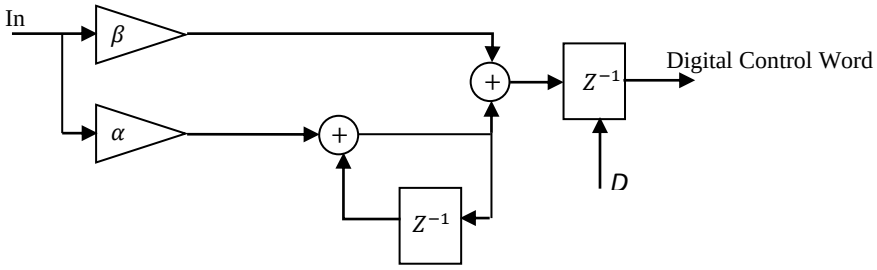

Figure 4-16. Architecture of the DLF

The conventional DLF architecture shown in Figure 2-5, and repeated here in Figure 4-16, is used in this system. The input to the DLF is a binary signal (1/-1). The proportional coefficient β affects the FCW only at the polarity change of the input binary decision. The integral coefficient α affects the FCW every update cycle (every D reference cycles). The following example illustrates the operation of the DLF: Initially, the FCW has an initial value *FCW0*. Assume that the binary input is 1, then the output of the DLF is: $FCW0 + \beta + \alpha$. After one update cycle, the output becomes: FCW0 + β + 2α.

The output keeps increasing every update cycle in this form: FCW0 + β + α, *FCW0* + β + 2α, *FCW0* + β + 3α... until the binary input is changed to -1. At this moment assume that the output of the DLF had reached a value *FCWx*. Then, the output starts decreasing every update cycle in this form: *FCWx* - β - α, *FCWx* - β - 2α, *FCWx* - β - 3α ...until the polarity change again, and so on.

In lock the FCW oscillates within a small range around the target FCW that achieves the target frequency. The minimum range of oscillation for the FCW in lock is bound by the value of β. Thus, the value of β is directly related to the minimum achievable jitter performance for the output clock.

Since DLF1 in Figure 4-1 is responsible for the lock time, the value of β is chosen to be large.. On the other hand, since DLF2 is responsible for the jitter performance, the value of β is chosen to be small. Yet, the ratio of β/α must be kept larger than 1 for stability. This is because a small value of β results in a better jitter performance, as discussed in Chapter 2.

4.1.7. Thermometer Decoder

Since the coarse code can take decimal values from 0 to 21, then it can be represented by 5 bits. Similarly, the fine code is represented by 6 bits. Thus, the FCW that is produced from the DLF is 11 bits. In order to decode these 11 bits into the 77 bits that control the DCO, thermometer decoders are used as in Figure 4-1.

A thermometer decoder decodes the input binary code such that the number of ones in the decoded output increases as long as the decimal value of the input code increases. Table 4.3 shows the truth table of an example 3x6 thermometer decoder. Note that the number of bits in the decoded output of the thermometer decoder does not have to be the power of 2 of the number of bits in the input code. However, the number of output bits must be less than the power of 2 of the number of bits in the input code.

The architecture of the 5x21 thermometer decoder is shown in Figure 4-17. The decimal value of the input code is used as the number of times to left shift a predefined vector of ones. The output of the shifter is then inverted. Thus, the resulting vector will

have a number of ones in the least significant bits that is equal to the decimal value of the input code.

Table 4.3. Truth table of a 3x6 Thermometer Decoder

Input Code (3 bits)			Decimal Equivalent	Decoded Output (7 bits)					
0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0	0	1
0	1	0	2	0	0	0	0	1	1
0	1	1	3	0	0	0	1	1	1
1	0	0	4	0	0	1	1	1	1
1	0	1	5	0	1	1	1	1	1
1	1	0	6	1	1	1	1	1	1

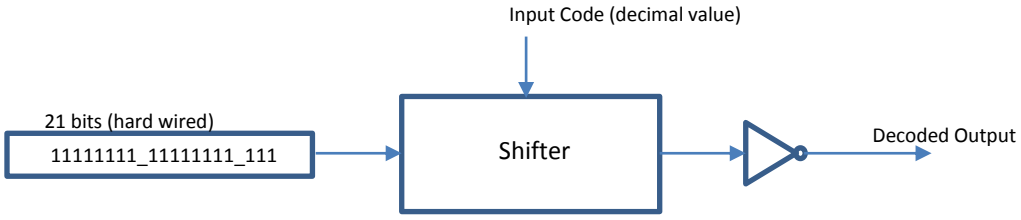

Figure 4-17. Architecture of 5x21 Thermometer Decoder

4.1.8. Notes on Closed Loop Operation

In Figure 4-1, the system operation through the frequency acquisition loop and the first phase acquisition loop (DLF1), is only a transient operation that takes no longer than 3 µs. During this stage, most of the system operation is in the second phase acquisition loop (DLF2). In the frequency acquisition loop, the coarse code of the FCW is determined. Then, in the first phase acquisition loop (DFL1), the fine code of the FCW is determined. During this stage, a multiplexer is used to pass the output of DLF1 to the DCO. The second phase acquisition loop (DLF2) determines the fine code of the FCW (from DLF2) at which the system is locked. After that, the tracking process is achieved

through DLF2 loop. In the tracking stage, the fine code (of DLF2) is adjusted continuously to maintain lock.

In order to verify the model in Chapter 3, step phase error is applied to the implemented system in the locked state, i.e., affecting the fine code of DLF2. The results of the implemented system along with the simulations for model verification are presented in Chapter 6.

4.2. SIMULATION RESULTS

The results of the AMS simulations of the implemented BB-ADPLL in Chapter 4 are illustrated showing the output waveforms and the system performance in terms of frequency range, clock jitter and lock time.

Figure 4-18 shows the output waveform of the BB-ADPLL at 3GHz frequency. The lower part of Figure 4-18 is the output frequency over time showing a steady output frequency at 3GHz. Figure 4-19 shows the lock operation of the BB-ADPLL. The curve in Figure 4-19 is the output of the phase detector. In Figure 4-19, a negative reset signal is first set to zero to initialize the system, and then rises to one. The division ratio N is set to 30 in order to get an output frequency of 3GHz, as the reference clock used is 100MHz. The system starts with the frequency acquisition until the coarse gain reduces to one. Note that after the frequency acquisition is done, neither the coarse_gain value nor the slow/fast signal affects the operation. After the frequency acquisition the loop goes to phase acquisition with DLF1, and then to phase acquisition with DLF2, and finally the loop is locked and tracking using DLF2.

In lock, the phase of the feedback signal is aligned with the phase of the reference signal, and hence, the output of the phase detector changes polarity rapidly indicating that the loop is locked. Note that in the stage of the frequency acquisition, the output of the phase detector is not relevant, since the phase detector is not active in this stage. In the phase acquisition stage, the output of the phase detector changes slowly, because the reference and the feedback are not aligned in phase, and the loop takes time to eliminate the phase difference. Finally, Figure 4-20 shows the output of the BB-PD when the locked state is disturbed by a phase step.

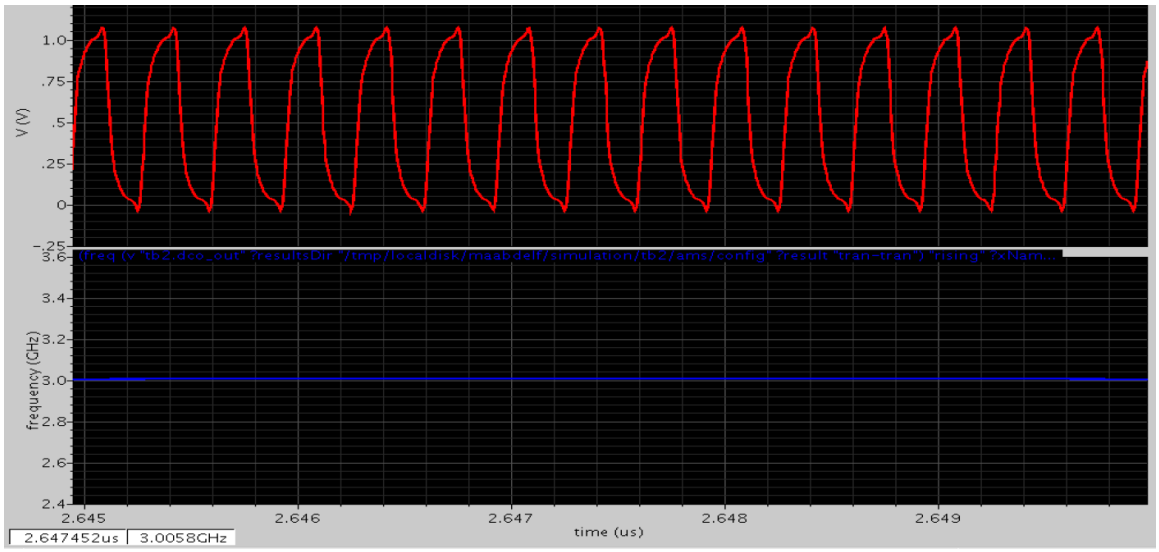

Figure 4-18. BB-ADPLL output waveform at 3 GHz frequency (AMS simulation)

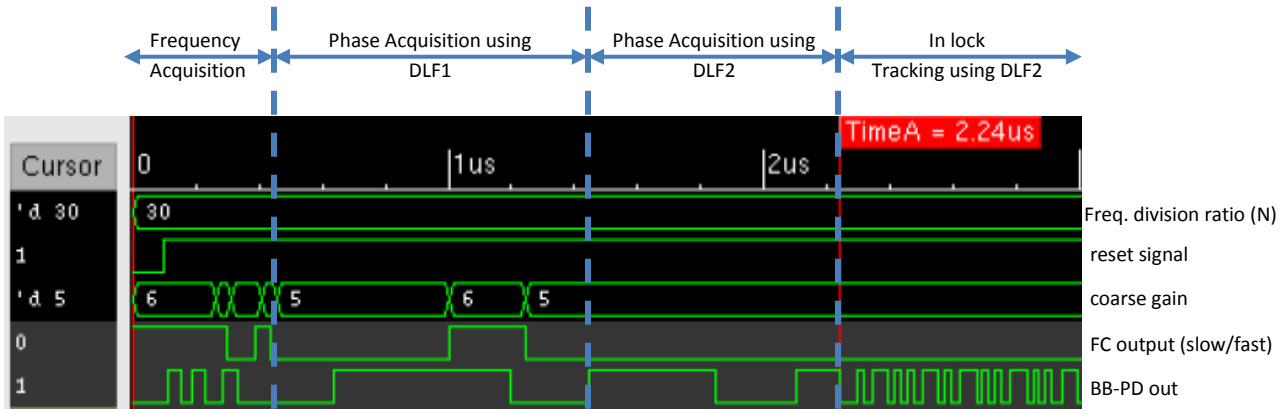

Figure 4-19. BB-ADPLL lock operation from AMS simulation

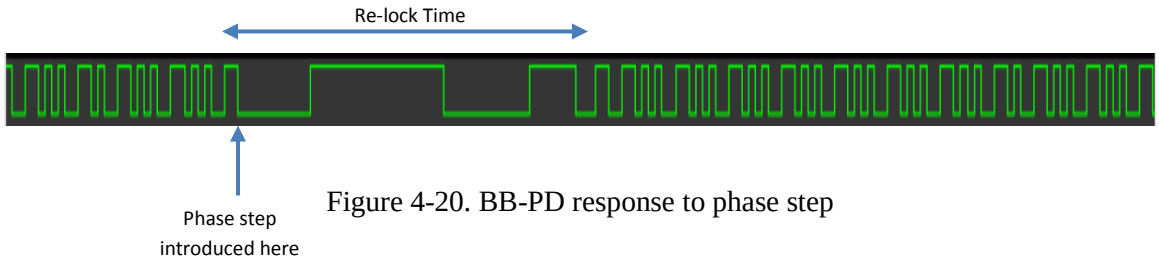

Figure 4-20. BB-PD response to phase step

Table 4.4 shows the jitter and lock time performance of the implemented BB-ADPLL at different operating frequencies of the DCO. Table 4.4 shows two

measurements of jitter: RMS jitter and Peak-to-Peak jitter (PP). The RMS jitter is calculated by comparing the output period of each DCO cycle to the ideal period of the target frequency, and then taking the root mean square of the differences. The PP jitter is the maximum deviation of the DCO cycle period from the ideal period. Note that all simulations are done in typical conditions and with typical supply voltage (1.05V).

Table 4.4. BB-ADPLL performance parameters

Frequency	Lock Time (µs)	RMS Jitter (ps)	Peak-to-Peak Jitter (ps)
2 GHz	2.5	1.13	4.2
3 GHz	2.2	0.88	3.6
5 GHz	1.6	0.7	3.4

4.3. MODEL VERIFICATION

In order to verify the model derived in Chapter 3, the phase step response is modeled for different PEMs at 3GHz and compared to the phase step response of the BB-ADPLL from simulations. The model predicts the phase response by calculating certain points on the phase error curve (m_i, E_{fi}). Figure 4-21 depicts the values of (m_i, E_{fi}) both calculated by the model, and obtained from AMS simulations for a phase error magnitude (PEM) 600ps. Any point (i) in Figure 4-21 will have the following coordinates ((m_1+ m_2+ … m_i), E_{fi}), where i is the order of the point in its curve. For example, the point surrounded by a circle in the figure is the second point in its curve, thus will have the coordinates (m_1+ m_2, E_{f2}). In a sense Figure 4-21 represents the frequency error trajectory over time similar to Figure 3-3, where the x-axis is time, and the y-axis is the frequency error E_{fi}. If the model perfectly predicts the system response, the points of the two curves would have the same coordinates.

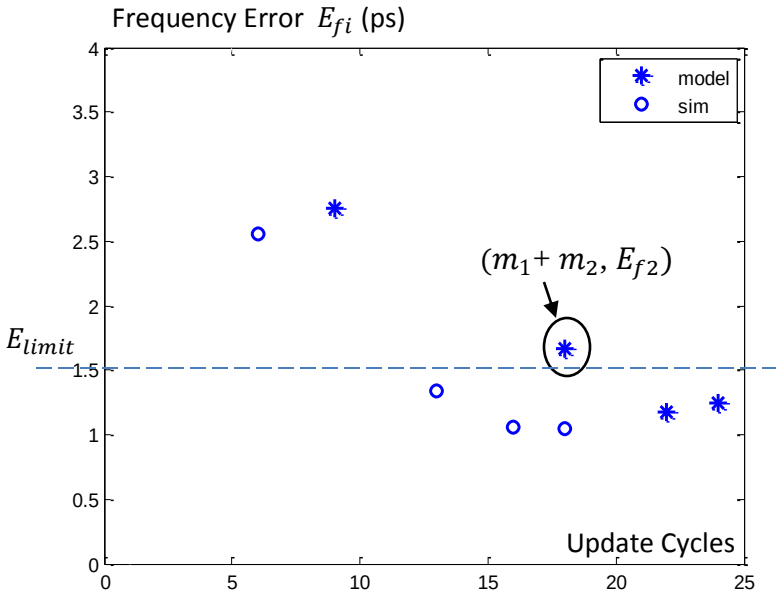

Figure 4-21. Model Vs. Simulations of phase step response for PEM 600ps

The design parameters used are $\alpha = 2,\ \beta = 4,\ D = 2,\ \Delta\emptyset = 1.5\,ps$. Figure 4-22, Figure 4-23, and Figure 4-24 are the same as Figure 4-21 but for different PEMs: 1200ps, 2300ps, and 5000ps respectively. On each figure, the limit for an acceptable frequency error (E_{limit}) is endicated. When the frequency error E_{fi} gets below E_{limit}, the system is considered stable, as previously discussed in Chapter 3. The value of E_{limit} is chosen for this design to be 1.5ps.

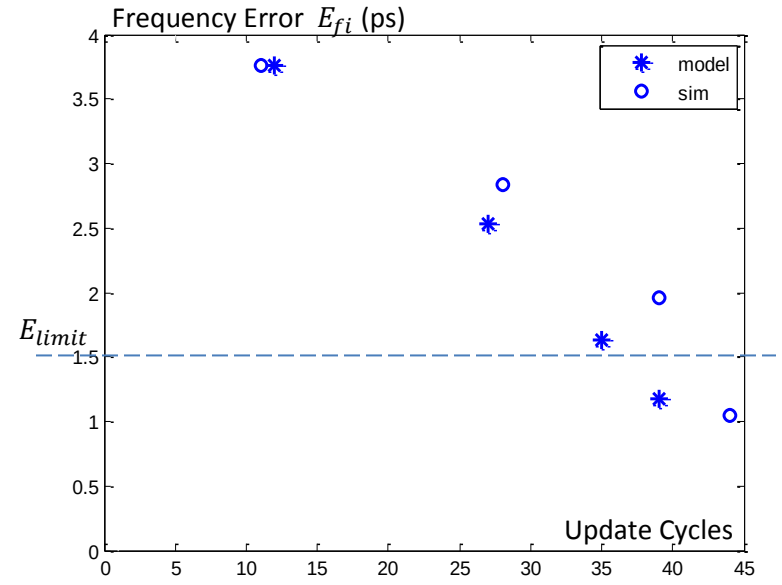

Figure 4-22. Model Vs. Simulations of phase step response for PEM 1200ps

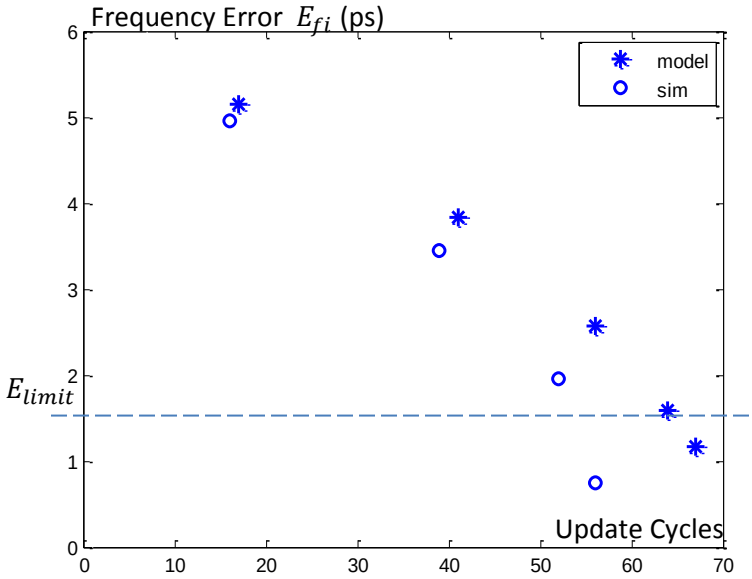

Figure 4-23. Model Vs. Simulations of phase step response for PEM 2400ps

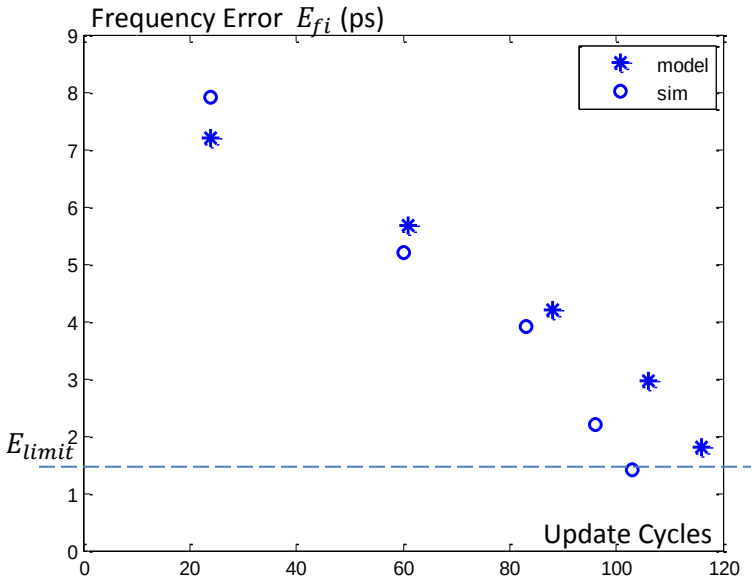

Figure 4-24. Model Vs. Simulations of phase step response for PEM 5000ps

The values of (m_i, E_{fi}) estimated by the model are used to calculate more meaningful system parameters such as the settling time ($T_{settling}$) , the system linearity figure of merit (*SL*), and the system stability. System stability is simply indicated by

substituting the values of E_{fi} in equation (3.1). From the previous figures the system was predicted to be stable by the model and verified by the simulation curves. Note that in order to make sure of the system stability then more points need to be calculated by the model to ensure that the frequency error will stay below E_{limit}, as illustrated previously in Figure 3-3.

Table 4.5 compares the settling time obtained by the simulation to that calculated from the model by equation (3.2). Table 4.5 compares the figure of merit *SL* obtained by the simulation to that calculated from the model by equations (3.14), (3.15). For convenience, the equations used are repeated here:

$$T_{settling} = D\,(m_1 + m_2 + \cdots + m_n)\,Tclk \tag{4.1}$$

$$t_1^2 + \left(\frac{2D\beta}{\alpha} - D\right) t_1 - \frac{2D}{\alpha}\frac{E_{p0}}{\Delta\emptyset} = 0 \tag{4.2}$$

$$SL = \frac{\Delta t_1}{\Delta(phase\ error\ magnitude)} \tag{4.3}$$

Table 4.5. Comparison of *SL* calculated by model and obtained by simulations

PEM (ps)	t_1 (model)	t_1 (sim.)	SL (model)	SL (sim.)
600	180 ns	165 ns		
1200	246 ns	236 ns	110	118
2400	340 ns	327 ns	78.3	75.8
5000	482 ns	480 ns	45.6	58.8

Table 4.6. Comparison of $T_{settling}$ calculated by model and obtained by simulations

PEM (ps)	$T_{settling}$ (model)	$T_{settling}$ (sim.)
600	480 ns	360 ns
1200	780 ns	880 ns
2400	1280 ns	1120 ns
5000	2320 ns	2190 ns

Table 4.5 and Table 4.6 show good agreement between the simulation and the model predictions. In Table 4.5, the value of the *SL* in the second row is calculated by subtracting t_1 in the first row from t_1 in the second row, and then dividing by the difference in the PEM. From Table 4.5, it can be noticed that the value of the SL decreases as the PEM increases, i.e., the system linearity is enhanced as the PEM increases. A reasonable explanation for that can be concluded from equation (4.2), where the dependence of the first zero time (t_1) on the PEM is through a square root function. Since the roots of (4.3) include E_{p0} under the square root, then the dependence of t_1 on the PEM decreases as PEM increase. This insight about BB-PLLs operation is gained thanks to the model proposed in this work. Such insight is considered a contribution by itself, since the literature is very short on operation details of BB-PLLs.

The trend of the settling time in Table 4.6 agrees with the expected behavior of BB-PLLs. The settling time varies depending on the PEM yielding a variable BW, as previously discussed.

Thus, the above figures (Figure 4-21, Figure 4-22, Figure 4-23, Figure 4-24) and tables (Table 4.5, Table 4.6) have demonstrated the success of the mathematical model presented in Chapter 3 in predicting the phase step response of the system, and in turn, to provide good insight and understanding of digital BB-PLLs. Note that the difference in values between model and simulations is referred to the assumptions made during the derivation of the model, such as assuming a maximum value of E_{f0}, along with neglecting the phase noise of the oscillator. However, this resulted in a much more simplified analysis with no big degradation in accuracy.

4.4. Conclusion

The implemented BB-ADPLL system is composed of a custom designed part (DCO), and an RTL part (the rest of the system). The circuit implementation details of each building block were described in this Chapter. AMS Closed loop simulations were performed to verify the functionality of the system, as well as to measure the system performance in terms of jitter and lock time. Finally, the phase step response of the implemented BB-ADPLL system was obtained by AMS simulations, and compared to

that predicted by the model in Chapter 3. The comparisons showed good agreement between the model and the simulations. Hence, the model is a useful tool that can be used to facilitate and speed up the design process of BB-DPLLs.

CHAPTER 5 : PROPOSED TECHNIQUES TO ENHANCE BB-ADPLL SYSTEM PERFORMANCE

The problems encountered by BB-PLLs due to the nonlinearity of the system have been stated in Chapter 2 as: the increased lock time, the undefined BW, and the difficulty in modeling. In previous Chapters, a modeling methodology for BB-DPLLs have been introduced, and applied to the conventional BB-ADPLL system. Thus, the modeling problem is solved.

In this Chapter, two modifications to the conventional BB-ADPLL are proposed in order to decrease the severeness of the first two problems, mentioned above. Both of the proposed techniques in this Chapter are inspired from the insight of the system gained by the model proposed in Chapter 3. Finally, the impact of the proposed techniques on the system performance is measured form simulations and compared to the conventional system implemented in Chapter 4.

5.1. PROPOSED DLF ARCHITECTURE TO ENHANCE LINEARITY

A novel DLF architecture is proposed. The objective of the proposed DLF is to enhance the system linearity. The outcomes expected from enhancing the system linearity are better response reliability, and a less variable BW, i.e., the dependence of the BW on the PEM is reduced. Unlike other approaches used in literature, the technique used in this work is simple, and takes the case of large phase error magnitudes into consideration. The idea of the proposed DLF is developed in the following section through a CP-PLL analogy.

5.1.1. Idea Development through CP-PLL Analogy

The main difference, in concept, between a CP-PLL and a BB-PLL is the nonlinear phase detector of the BB-PLL. A CP-PLL has a linear response as it uses a phase frequency detector (PFD) along with a charge pump (CP) to quantify both the direction, and the magnitude of the phase error. Thus, a CP-PLL changes the slope of its control quantity, which is the charge pump current (I_p), proportional to the existing phase

error. Hence, a CP-PLL maintains the same loop BW irrespective of the phase error magnitude. BB-PLLs, on the other hand, ignore the phase error magnitude information yielding a response that is dependent on the phase error magnitude.

Based upon the previous discussion of CP-PLL behavior, a solution is proposed to enhance the linearity of BB-PLLs. If the relative magnitude of the phase error can be estimated through observing BB-PD decision pattern over time, then the loop filter coefficients can be adjusted accordingly, and hence the linearity of the loop response is enhanced. Figure 5-1 shows the block diagram of a general CP-PLL (a), compared to a BB-PLL system with the proposed technique (b). In Figure 5-1-(b), the proposed technique introduces a counter that counts the number of repeated decisions of the BB-PD. The output of the counter is proportional to the relative magnitude of the phase error. The output of the counter is then used to adjust the coefficients of the DLF. Thus, the counter mimics the role of a charge pump in a CP-PLL.

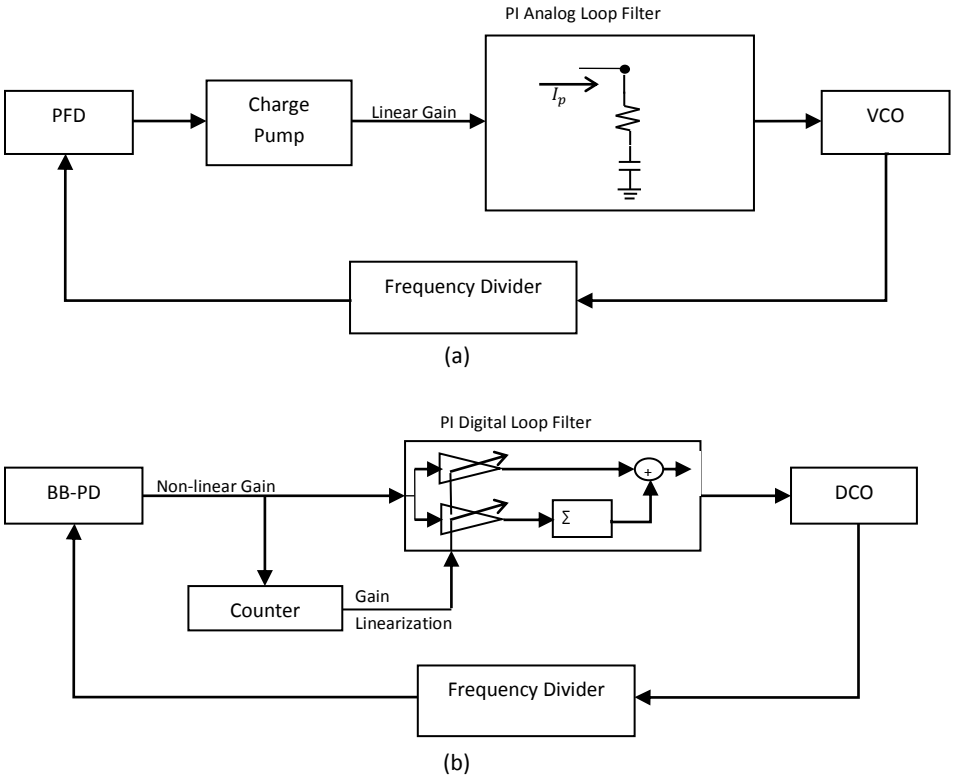

Figure 5-1. Block diagram of (a) CP-PLL, (b) proposed BB-PLL

Figure 5-2 illustrates the effect of the proposed technique on the system response in comparison with the response of a CP-PLL. Figure 5-2-(a), similar to Figure 2-12-(a), shows that the CP-PLL increases the slope of the charge pump current (I_p) to adapt to a higher phase error magnitude, and thus, maintains a constant BW. Analogous to the behavior of the CP-PLL, the proposed technique changes the slope of the FCW in the BB-PLL to relatively adapt to different phase error magnitudes as shown in Figure 5-2-(b). The slope of the FCW is increased when the counter output reaches the value P_1. The FCW slope increases again when the counter output reaches the value P_2, and so on. Therefore, the equivalent slope of the FCW increases as the magnitude of the phase error increases, and vice versa. Thus, the proposed technique leads to a more linearized response and a relatively constant BW compared to a conventional DLF that maintains a constant slope of FCW irrespective of the magnitude of the phase error.

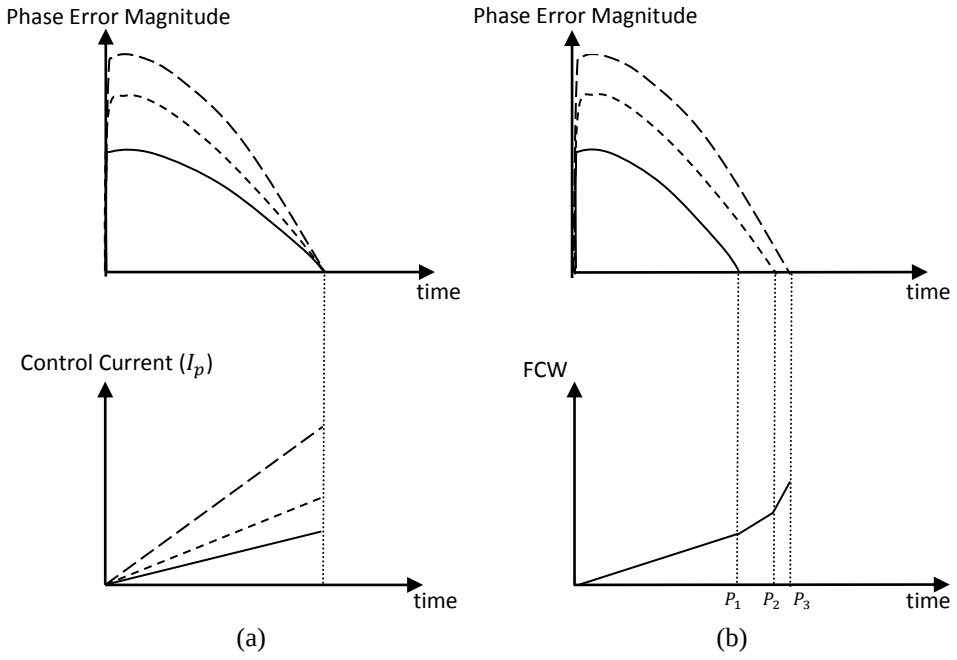

Figure 5-2. Change in slope of (a) the charge pump current ($\boldsymbol{I_p}$) and (b) the FCW in response to different PEMs

5.1.2. Proposed DLF Architecture

To implement the linearization technique presented in the previous section, a novel DLF is proposed. Figure 5-3 shows the architecture of the proposed DLF. Similar

to the conventional DLF in Figure 4-16, the proposed DLF consists of a proportional path and an integral path. However, the coefficients of the DLF are varied proportional to the magnitude of the phase error. The BB-ADPLL implemented in this Chapter, acquires lock by means of binary search, and then maintains lock by using the proposed DLF.

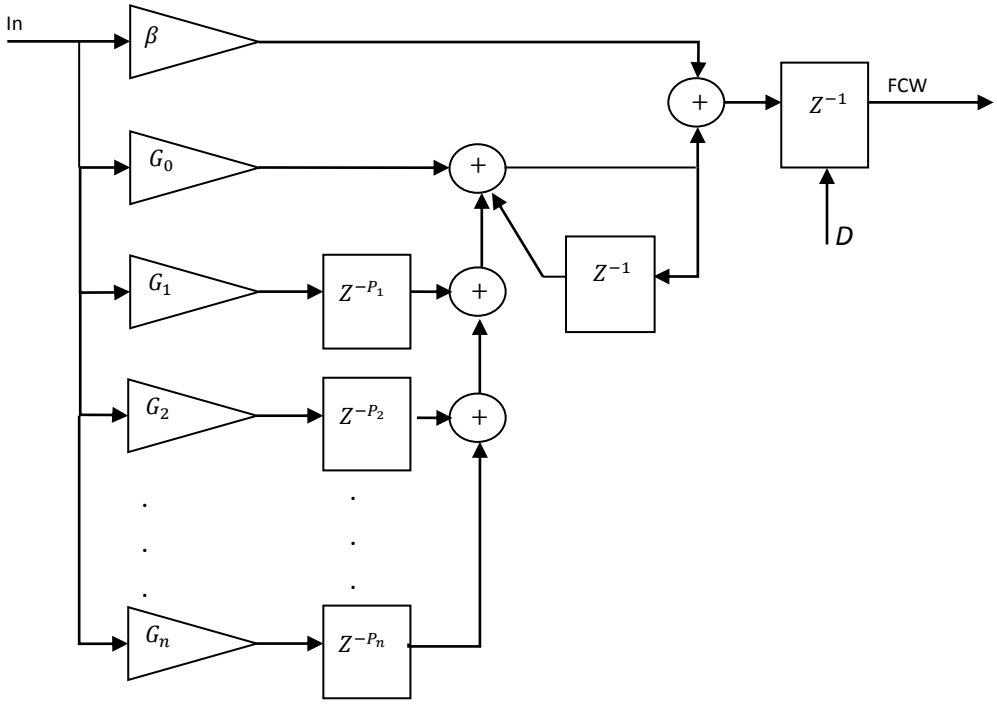

Figure 5-3. Proposed DLF architecture

In a conventional proportional-integral DLF, the integral coefficient is fixed to a certain value irrespective of the phase error magnitude. The proposed DLF, on the other hand, uses a fixed integral coefficient (G_0) for a period of time P_1. If the direction of the BB-PD decision is not changed in less than P_1 cycles, this indicates that the current phase error has a relatively large magnitude. Accordingly, the DLF increases the slope of the FCW by increasing the integral coefficient by a value G_1. Similarly, the integral coefficient is increased after P_n cycles by a value of G_n, where n is the number of tabs used, and is an arbitrary number set by the designer depending on the application. The value of n represents the sensitivity of the loop to different phase error magnitudes. For example, if $n = 1$, the loop quantizes the phase error magnitude into small and large magnitudes only. On the other hand, if $n = 10$, the loop quantizes the phase error magnitudes into 11 categories. A larger value of n results in a more linear response at the

cost of more design complication, and more power consumption. Therefore, the value of n should be optimized, depending on the target application.

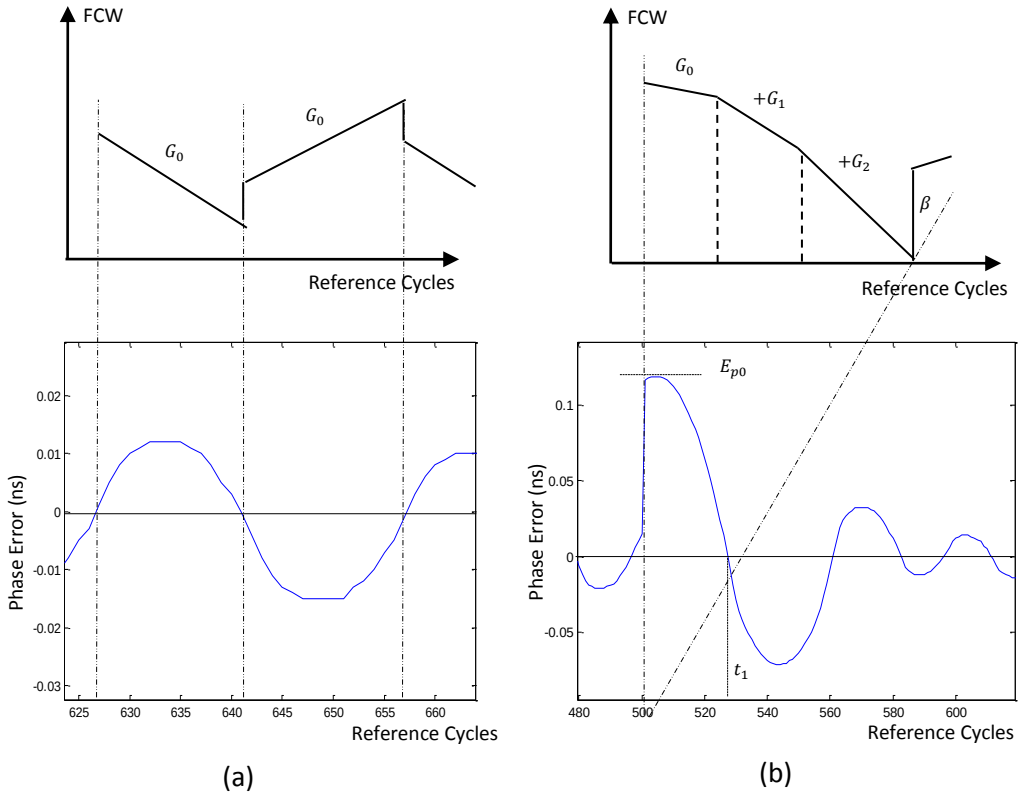

Figure 5-4. The phase error response of the proposed BB-ADPLL are shown in the lower part, and the corresponding control of the loop (FCW) is shown in upper part. (a) Undisturbed locked state, (b) Locked state disturbed by phase step error E_{p0}

The operation of the proposed DLF is illustrated by Figure 5-4. Figure 5-4-(a) shows the phase error response of the proposed system in the locked state, and the corresponding waveform of the FCW. In the undisturbed locked state of Figure 5-4-(a), the slope of the FCW is small to obtain a good jitter performance. In this case, the proposed DLF approaches the behavior of the conventional DLF, and exhibits fixed proportional and integral coefficients.

Figure 5-4-(b) shows the phase error response and the corresponding waveform of the FCW when the locked state is disturbed by a phase step error of a large magnitude. After a period of time P_1, the DLF senses that the phase error is large, and starts to increase the slope of the FCW. Then, at P_2, the DLF further increases the slope in order to adapt to the large phase error

5.1.3. Effect on System Linearity

The theoretical analysis of the effect of the proposed DLF on the system linearity is similar to that in section 3.4.2 in Chapter 3. Recall equation (3.14):

$$t_1^2 + \left(\frac{2D\beta}{\alpha} - D\right) t_1 - \frac{2D}{\alpha}\frac{E_{p0}}{\Delta\emptyset} = 0 \tag{5.1}$$

For the proposed DLF, equation (5.1) holds. However, the value of the integral coefficient is varied according to the relative value of E_{p0}, as illustrated earlier in this section. Hence, the dependence of t_1 on E_{p0} is less for the proposed system than that of a conventional DLF, and the linearity is enhanced. Thus, equation (5.1) needs to be modified to take the changes of the proposed system into account.

The advantage of the proposed modeling methodology is evident here, since the same methodology can be applied to the proposed system in order to develop a system model.

Recall from Chapter 3 that the two phase components at the first zero crossing ($\emptyset_1$, $\emptyset_{e1}$) are equated in order to get the time (t_1), which is used to calculate the system linearity figure of merit *SL*. For the proposed system the error phase component $\emptyset_{e1}$ is the same as in equation (3.3):

$$\emptyset_{e1} = E_{p0} + E_{f0}\, D m_1\, \Delta\emptyset \tag{5.2}$$

The loop phase component ($\emptyset_1$) is changed according to the architecture proposed in this Chapter in Figure 5-3. For a DLF with two taps (n=2), equation (3. 7) evolves to be:

$$\emptyset_1 = \beta\, D m_1\, \Delta\emptyset \; + \begin{cases} G_0 D\; a(m_1), & m_1 \le P_1 \\ G_0\, D\; a(m_1) + G_1 D\; a(m_1 - P_1), & P_1 < m_1 \le P_2 \\ G_0\, D\; a(m_1) + G_1\, D\; a(m_1 - P_1) \; + \; G_2\, D\; a(m_1 - P_2), & P_2 < m_1 \end{cases} \tag{5.3}$$

,where *a(x)* is:

$$a(x) = \frac{x(x+1)}{2} \tag{5.4}$$

In equation (5.3), the first term of the RHS represents the proportional contribution, and is similar to that in equation (3.7) of the conventional system. The second term of the RHS represents the integral contribution of the proposed DLF to the phase component $\emptyset_1$. This integral contribution varies as the value of m_1 is higher. Thus, there is three possibilities for a system with two taps (*n=2*). The first possibility is that m_1 is smaller than the counter value P_1. In this case, the system behaves exactly as the conventional system. The second possibility is that the value of m_1 is between P_1, and P_2. In this case, the accumulation of phase due to the integral coefficient is caused by the factor (G_0) over m_1 cycles, in addition to (G_1) over ($m_1 - P_1$) cycles. Finally, the third possibility happens when the value of m_1 is larger than P_2. In this case, the integral contribution is the same as the previous case in addition to phase accumulation due to (G_2) over ($m_1 - P_2$) cycles.

Since the value of m_1 is initially unknown, then it cannot be decided which branch in equation (5.3) to be used. The methodology used to solve this equation is to first assume that m_1 is smaller than P_1, and solve using the first branch. If the result achieves the assumption then the assumption was right, else assume that m_1 is between P_1, and P_2, and solve using the second branch, and so on.

Finally, equating (5.2) and (5.3) results in a quadratic equation that can be solved to get the values of t_1. Table 5.1 shows the values of t_1 and SL, for both the proposed and the conventional DLF. Both systems have the same DLF proportional constant ($\beta = 6$). The value of the integral coefficient in the conventional DLF is ($\alpha = 2$). The values of the proposed DLF parameters are: $n = 2,\ P_1 = 7, P_2 = 14,\ G_0 = 1,\ G_1 = 2, G_2 = 2$. Note that $\mathrm{n} = 2$ indicates the use of two taps in the architecture proposed in Figure 5-3.

Table 5.1 shows that the proposed system has enhanced the system linearity by 9% for small PEMs, and about 35% for larger PEMs.

Table 5.1. Theoretical analysis of the System Linearity for Both the Proposed DLF, and a Conventional DLF

PEM (ps)	t_1 (conv.)	t_1 (prop.)	SL (conv.)	SL (prop.)	% enhancement
600	180 ns	160 ns			
1200	246 ns	220 ns	110	100	9%
2400	340 ns	280 ns	78	50	36%
5000	482 ns	360 ns	45	30.7	33.3%

5.2. PROPOSED TECHNIQUE TO REDUCE RE-LOCK TIME

In this section, a technique that helps reduce the re-lock time of the BB-ADPLL is proposed. The "re-lock time" is the settling time that the PLL takes to regain lock after a phase step is introduced. The proposed technique is based on the understanding of the dynamics of the frequency error, and phase error of the loop.

To develop the idea, the case of phase step error disturbing the locked state as in Figure 3-1 is considered. Since the system is initially locked before the phase step, then the frequency error at the moment the step is introduced is a small acceptable error. Due to the phase step error, the loop takes action by changing its frequency in order to compensate the phase step error, resulting in E_{f1} in Figure 3-1.

Consequently, the system state at the first zero crossing point on the phase error curve can be described as: zero phase error, and E_{f1} frequency error. This means that the phase is right at that point, but the FCW is away from the target by E_{f1}. Hence, if the value of E_{f1} can be estimated and added/subtracted from the FCW at the first zero crossing point, then the system state would immediately change to zero phase error, and zero frequency error. Thus, the re-lock time will significantly drop.

On the circuit level, it is easy to add circuitry in order to estimate E_{f1}. The value of E_{f1}, as illustrated in Chapter 3, is a function of the number of update cycles before the first zero crossings (m_1), and can be estimated by:

$$E_{f1} - E_{f0} = (\alpha\, m_1 + \beta)\Delta\emptyset \tag{5.5}$$

The values of the DLF parameters α, β are predefined. The value of m_1 can simply be estimated by a counter that resets at zero crossings, i.e., counts the cycles between zero crossings. Finally, the value of $\Delta\emptyset$ is not required for the estimation of E_{f1}, because only the value of E_{f1} normalized to $\Delta\emptyset$ is of interest. This is because the objective of estimating E_{f1} is to subtract its value from the FCW. The FCW is independent of the phase step $\Delta\emptyset$.

Note that the value estimated by equation (5.5) is $(E_{f1} - E_{f0})$ not E_{f1}. Thus, practically, when this value is subtracted from the FCW at the first zero crossing point, the frequency error drops to E_{f0} not zero. This is not a problem since the value of E_{f0} is small and is acceptable in the locked state.

In order to examine the benefits of the two proposed techniques in this Chapter on enhancing system linearity and lock time, both techniques are implemented on the circuit level (RTL). Each technique is implemented separately, and then each technique is compared to the BB-ADPLL system described in Chapter 4.

5.3. IMPACT OF THE PROPOSED TECHNIQUES COMPARED TO CONVENTIONAL SYSTEM

The techniques proposed in Chapter 5 are implemented on the circuit level and simulated by AMS tools. The enhancements introduced by these techniques are captured and compared to the conventional BB-ADPLL in Chapter 4.

5.3.1. Proposed DLF for Enhanced Linearity

The proposed DLF is implemented and compared to the conventional BB-ADPLL. Different values of PEM are applied to both systems to calculate the *SL* figure of merit by equations (4.2) and (4.3). Table 5.2 shows the comparison between the two systems indicating a significant enhancement in system linearity (35% average). The values of t_1 and *SL* for the proposed system in Table 5.2 are in good agreement with the predicted values by the model in Table 5.1. The values of the DLF parameters of both systems are the as these used in Table 5.1. Both systems have the same DLF proportional constant ($\beta = 6$). The value of the integral coefficient in the conventional DLF is ($\alpha =$

2). The values of the proposed DLF parameters are: $n = 2$, $P_1 = 7, P_2 = 14$, $G_0 = 1$, $G_1 = 2, G_2 = 2$.

Table 5.2. comparison between *SL* of the conventional and proposed systems. Data is from simulations.

PEM (ps)	t_1 (conv.)	t_1 (prop.)	*SL* (conv.)	*SL* (prop.)	% enhancement
600	165 ns	154 ns			
1200	236 ns	212 ns	118	96.7	18%
2400	327 ns	270 ns	75	48.3	35.6%
5000	480 ns	359 ns	58	34.23	40.9%

Figure 5-5 depicts the values of t_1 for both the proposed and the conventional system as given in Table 5.2. Figure 5-5 shows that dependence of t_1 , and hence the BW, on the PEM is reduced for the proposed system.

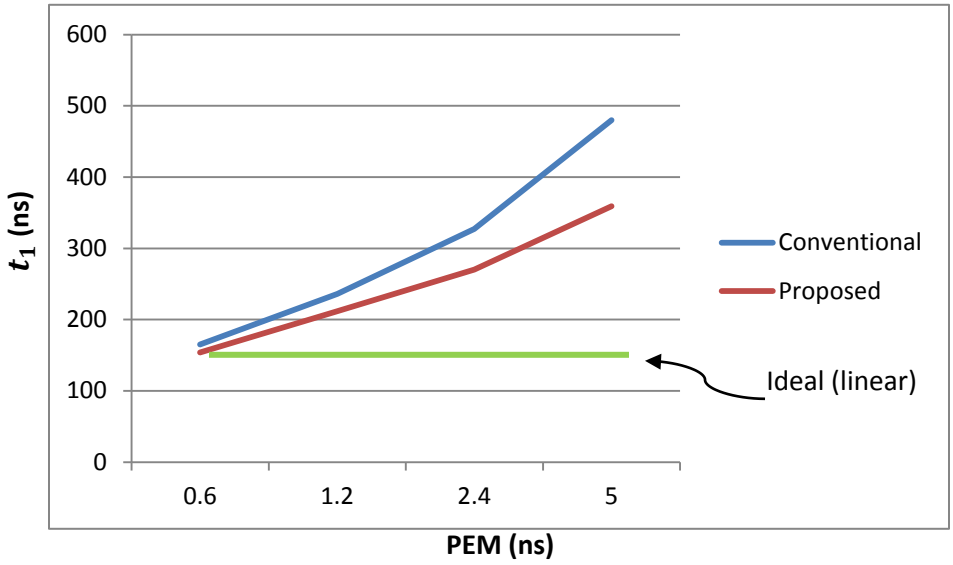

Figure 5-5. Values of t_1 for both conventional and proposed system. t_1 values for the proposed system are more close to be constant indicating that the BW of the proposed system is more close to e constant

5.3.2. Proposed Technique for Re-lock Time Reduction

Similarly, the technique proposed in Chapter 5 to reduce the re-lock time is implemented, and different PEMs were applied to the BB-ADPLL with the proposed technique, and the conventional BB-ADPLL. The lock time is obtained in both cases and

compared in Table 5.3. Significant reduction in lock time is shown in the case of the proposed technique. Note that the BB-ADPLL with the proposed technique here does not use the proposed DLF in Chapter 5. In other words, the proposed techniques in Chapter 5 are experimented here separately.

Table 5.3. Comparison of $T_{settling}$ obtained from AMS simulations once for the conventional BB-ADPLL, and the other for the proposed BB-ADPLL

PEM (ps)	$T_{settling}$ **(conventional)**	$T_{settling}$ **(proposed)**	**Percentage Reduction**
600	480 ns	215 ns	55.2 %
1200	880 ns	406 ns	53.8 %
2400	1120 ns	457 ns	59.1 %

Table 5.3 shows around 55% reduction in the re-lock time due to the proposed technique. The implementation of the proposed technique is very simple; the trick is in understanding the system operation. Again, thanks to the proposed model, a good understanding of the system is acquired allowing for innovations such as the techniques proposed to enhance the re-lock time and the system linearity.

5.4. Conclusion

The insights gained by the model presented in Chapter 3, suggested innovative techniques that helped enhancing the linearity of BB-PLLs. The first proposed technique suggested modifying the architecture of the DLF such that the coefficients of the DLF become adaptive to the relative magnitude of the phase error. The second proposed technique was based on implementing a circuit that can estimate the frequency error and update the DCO accordingly yielding a fast lock process. Both techniques were implemented on the circuit level and compared to the conventional BB-ADPLL implemented in Chapter 4. Comparisons showed around 35% enhancement in the system linearity due to the first proposed technique, and around 55% enhancement in the re-lock time due to the second proposed technique.

CHAPTER 6 : CONCLUSION AND FUTURE WORK

7.1. CONCLUSION

BB-PLLs represent a class of PLLs that incorporate binary phase detectors. Using binary phase detectors has the advantage of simple implementation and low power consumption, which is why BB-PLLs have gained a lot of interest recently. However, binary phase detectors results in nonlinear loop dynamics, which cause a lot of problems to the design of BB-PLLs. The main problems are difficulty of modeling leading to lengthy design process, and the dependence of the system response on the magnitude of the phase error yielding a non-constant BW.

This work presented a modeling methodology for Digital BB-PLLs. The modeling methodology presented is generic and can be applied to any Digital BB-PLL regardless of the application. The proposed modeling methodology can predict the phase step response of the system in the locked state. The information gained by predicting the phase step response indicates the system BW and stability. Moreover, the predicted response is in terms of the system design parameters. Thus, the proposed modeling methodology can be used to further facilitate the design process. The proposed modeling methodology is based on modeling the time domain behavior of the system building blocks by mathematical equations, and then these equations are easily plugged into Matlab.

In order to apply the concept described above, the proposed modeling methodology was used to derive a system model for the conventional BB-ADPLL system. Meanwhile, the conventional BB-ADPLL system was implemented on the circuit level and simulated by Cadence AMS tools. The predicted response by the system model was compared to the response obtained by AMS simulations for the sake of model verification. The results demonstrated good agreement between the modeled response and the simulated one.

Finally, the insights gained by the system model were used to propose two different circuit techniques that would help enhance the system linearity. The proposed

techniques were first analyzed theoretically then implemented on the circuit level. In order to measure the impact of the proposed techniques on the system linearity, the simulation results of the proposed techniques were compared to the results of the conventional BB-ADPLL. Comparisons demonstrated 35% enhancement in system linearity, and 55% reduction in the settling time of the phase step response.

7.2. FUTURE WORK

To further develop the work presented in this thesis future research needs to be carried out. First of all, the model used a number of assumptions that resulted in some inaccuracy in the model predictions. These assumptions need to be replaced with actual conditions for increased prediction accuracy. Moreover, the model did not take into consideration the effects of the physical design. This may cause the accuracy of the model to degrade when compared to post-layout simulations of the system. Hence, further modifications in the model are the main goal of this research in the future.

Another crucial point that needs to be investigated in the future work is the pattern of the input phase errors. The work in this thesis dealt with the case of the phase step error. However, the phase error could have a statistical distribution depending on the application. Thus, the input to the model can be a statistical variable rather than a fixed step. Furthermore, the case of time-varying phase errors is to be considered in the next version of the proposed model.

Finally, the proposed techniques in Chapter 5 can be combined for improved system linearity. Moreover, the second proposed technique only predicted the frequency error at the first zero crossing point, and updated the loop accordingly. In future, research this technique can be further developed by estimating the frequency error at every zero crossing point for a boost in the lock time reduction.

BIBLIOGRAPHY

[1] B. Staszewski, All Digital Frequency Synthesizers in Deep Submicron CMOS. New York: John Wiley and Sons, 2006.

[2] R. E. Best, Phase-Locked loops: Theory, Design and Applications. McGraw-Hill, 2003.

[3] F. Gardner, "Charge-Pump Phase-Lock Loops," IEEE Trans. Communications, Reg. Papers, vol. 28, no. 21, pp. 1849–1858, 1980.

[4] S. Henzler, Time-to-Digital Converters. Spriger Series in Advanced Microelectronics, 2010.

[5] R. B. Staszewski, J. L. Wallberg, and S. Rezeq et al., "All-digital PLL and transmitter for mobile phones," IEEE J. Solid-State Circuits, vol. 40, no. 12, pp. 2469–2482, Dec. 2005.

[6] B. Staszewski et al., "All Digital Phase Domain TX Frequency Synthesizer for Bluetooth Radios in 0.13um CMOS," in Int. Solid-State Circutis Conference Dig. Tech. Papers, 2004.

[7] Effendrik, P., Wenlong Jiang, van de Gevel, M., Verwaal, F., Staszewski, R.B. "Time-to-Digital Converter (TDC) for WiMAX ADPLL in 40-nm CMOS," European Conference on Circuit Theory and Design Dig. Tech. Papers, pp. 365-368, 2011.

[8] R. B. Staszewski, "State-of-the-Art and Future Directions of High-Performance All-Digital Frequency Synthesis in Nanometer CMOS ," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 58, no. 7, pp. 1490–1510, 2011.

[9] Jri Lee; Razavi, B., "A 40-Gb/s clock and data recovery circuit in 0.18-µm CMOS technology," IEEE J. Solid-State Circuits, vol. 38, no. 12, pp. 2181 - 2190, 2002.

[10] Che-Fu Liang; Sy-Chyuan Hwu; Shen-Iuan Liu, "A 10Gbps Burst-Mode CDR Circuit in 0.18µm CMOS," Custom Integrated Circuits Dig. Tech. Papers, pp. 599 - 602, 2006.

[11] Rylyakov, A.; Tierno, J.; Ainspan, H.; Plouchart, J.-O.; Bulzacchelli, J.; Deniz, Z.T.; Friedman, D., "Bang-bang digital PLLs at 11 and 20GHz with sub-200fs integrated jitter for high-speed serial communication applications," in Int. Solid-State Circutis Conference Dig. Tech. Papers, pp. 94–95,95a 2009.

[12] R. C.Walker, “Designing bang-bang PLLs for clock and data recovery in serial data transmission systems,” in Phase-Locking in High-Performance Systems, B. Razavi, Ed.. New York: IEEE Press, 2003, pp. 34–45.

[13] J.A. Tierno, A.V. Rylyakov, and D. Friedman, “A Wide Power Supply Range, Wide Tuning Range, All Static CMOS All Digital PLL in 65 nm SOI”, IEEE J. Solid -State Circuits, vol. 43, no. 1, pp. 42-51, Jan., 2008.

[14] Ching-Che Chung; Chen-Yi Lee, “An all-digital phase-locked loop for high-speed clock generation,” IEEE J. Solid-State Circuits, vol. 38, no. 2, pp. 347–351, Feb. 2003.

[15] D. Tasca, M. Zanuso, G. Marzin, S. Levantino, C. Samori, A.L. Lacaita, “A 2.9-to-4.0GHz Fractional-N Digital PLL with Bang-Bang Phase Detector and 560fsrms Integrated Jitter at 4.5mW Power,” IEEE J. Solid-State Circuits, vol. 46, no. 12, pp. 2745–2758, Feb., 2011.

[16] D. Kim, H. Song, T. Kim, S. Kim, and D. Jeong, “A 1.35 GHz all-digital fractional-N PLL with adaptive loop gain controller and fractional divider,” IEEE J. Solid-State Circuits, vol. 45, no. 11, pp. 2300–2311, Nov. 2010.

[17] N. August, H. Lee, M. Vandepas, R. Parker, “A TDC-less ADPLL with 200-to-3200MHz range and 3mW power dissipation for mobile SoC clocking in 22nm CMOS,” Int. Solid-State Circutis Conference Dig. Tech. Papers, pp. 246-248, Feb., 2012.

[18] J. Lee, K. S. Kundert, and B. Razavi, “Analysis and modeling of bangbang clock and data recovery circuits,” IEEE J. Solid-State Circuits, vol. 39, no. 9, pp. 1571–1580, Sep. 2004.

[19] Tertinek, S.; Gleeson, J.P.; Feely, O., “ Binary Phase Detector Gain in Bang-Bang Phase-Locked Loops With DCO Jitter ,” IEEE Trans. Circuits Syst. II, Reg. Papers, vol. 57, no. 12, pp. 941–945, 2010.

[20] Chih-Fan Liao; Shen-luan Liu, “A 40 Gb/s CMOS Serial-Link Receiver With Adaptive Equalization and Clock/Data Recovery,” IEEE J. Solid-State Circuits, vol. 43, no. 11, pp. 2492–2502, 2008.

[21] N. D. Dalt, “Markov Chains-Based Derivation of the Phase Detector Gain in Bang-Bang PLLs,” IEEE Trans. Circuits Syst. II, Reg. Papers, vol. 53, no. 11, pp. 1195–1199, 2006.

[22] N. D. Dalt, “Linearized Analysis of a Digital Bang-Bang PLL and Its Validity Limits Applied to Jitter Transfer and Jitter Generation,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 55, no. 11, pp. 3663–3675, Dec. 2008.

[23] M. Zanuso, D. Tasca, S. Levantino, A. Donadel, C. Samori, A.L Lacaita, “Noise Analysis and Minimization in Bang-Bang Digital PLLs ,” IEEE Trans. Circuits Syst. II, Reg. Papers, vol. 56, no. 11, pp. 835-839, 2009.

[24] Da Dalt, “Theory and Implementation of Digital Bang-Bang Frequency Synthesizers for High Speed Serial Data Communications” Ph.D. dissertation, Rheinisch-Westfälischen Technischen Hochschule Aachen, Aachen, Germany, 2007.

[25] J. D. H. Alexander, “Clock Recovery from Random Binary Data,” Electronics Letters, vol. 11, pp. 541-542, Oct. 1975.

[26] B. Razavi, Design of Analog CMOS Integrated Circuits. McGraw-Hill, 2003.

[27] F. M. Gardner, Phaselock Techniques. New York: John Wiley and Sons, 1979.

[28] V. Kratyuk et al., “A design procedure for all-digital phase-locked loops based on a charge-pump phase-locked-loop analogy,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 54, no. 3, pp. 247–251, Mar. 2007.

[29] N. D. Dalt, “A design-oriented study of the nonlinear dynamics of digital bang-bang PLLs,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 52, no. 1, pp. 21–31, Jan. 2005.

[30] Saint-Laurent, M.; Muyshondt, G.P., “A digitally controlled oscillator constructed using adjustable resistors,” Southwest Symposium on Mixed-Signal Design Dig. Tech. Papers, pp. 80-82, 2001.