A DELAY-LOCKED LOOP FOR MULTIPLE CLOCK PHASES/DELAYS GENERATION

A Dissertation
Presented to
The Academic Faculty

by

Cheng Jia

In Partial Fulfillment
of the Requirements for the Degree of
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology
December, 2005

A DELAY-LOCKED LOOP FOR MULTIPLE CLOCK PHASES/DELAYS GENERATION

Approved by:

Dr. Linda Milor, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Russell Callen
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Abhijit Chatterjee
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. John Cressler
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. David Keezer
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Alfred Andrew
School of Electrical and Computer Engineering
Georgia Institute of Technology

Date Approved: August 22, 2005

ACKNOWLEDGEMENTS

I would like to express my gratitude and appreciation to everyone who has helped to make this thesis possible. I wish to thank my advisor, Professor Linda Milor, for her guidance, help, encouragement, and support during my study at the Georgia Institute of Technology. I also wish to extend my thanks to my committee members, Professor Russell Callen, Professor Abhijit Chatterjee, Professor John Cressler, Professor David Keezer, and Professor Alfred Andrew for their time and valuable inputs. Finally, I owe gratitude to all of my friends and colleagues.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

This thesis presents our work in the design of a Delay-Locked Loop (DLL) for the generation of multiple clock phases/delays. A novel DLL design is proposed with several new techniques to help achieve wide lock range, short locking time, and reduced jitter. The DLL can be used for a variety of applications which require precise time intervals or phase shifts.

CHAPTER 1

INTRODUCTION

Delay-locked loop (DLL) is a critical circuit component widely used in many timing applications. In this thesis, we present a novel DLL design which can be used for a variety of applications. Specifically, we have built a DLL which is able to generate multiple clock phases/delays with low jitter, short locking time, and wide lock range. To achieve this design goal, several techniques and algorithms are used in our design. In this chapter, we will introduce the motivation for our research, formally state the problem we are facing, and present the organization of this thesis.

1.1 Motivation

The research presented in this thesis is originally motivated by the need of precise delay generation in the testing of digital integrated circuits. Digital testing often requires digital input signals which have a precisely-controlled timing relationship (delay) to a reference clock signal [1]. The job of delay generation is usually done by an external tester. Thus, the generation of accurate and stable delays, especially in the testing of timing-sensitive specifications, such as the I/O setup and hold time, often demands high end test equipment. In other words, higher frequency signals or more advanced techniques must be used by the tester in order to produce the desired delays. This approach inevitably brings up the cost of the test equipment as well as the overall test cost. As a result, some tests, such as the test of setup and hold time, are often conducted only on a few selected I/O pins in practice [2]. Naturally, there is a desire to find other approaches which are more cost-effective to generate these delays, thus reducing the dependence on expensive test equipment.

The International Technology Roadmap for Semiconductors states the future directions and challenges of the semiconductor industry. According to the 2004 update of the Roadmap [3], test cost per unit and test equipment capital cost considerations continue to dominate manufacturing test methodology decisions. The cost of high performance automatic test equipment (ATE) is continuously rising. In [3], the following ATE cost model is suggested

$$Tester\ Cost = b + \sum (m \times x)_n\ , \tag{1-1}$$

where b is the base cost of a test system with zero pins, m is the incremental cost per pin, and x is the number of pins. Table 1-1 shows some typical values of b, m, and x [3].

Table 1-1 ATE cost parameters

Tester Segments	**b (base cost in K$)**	**M (cost per pin in $)**	**x (pin count)**
ASIC/MPU	250-400	2700-6000	512
Mixed-signal	250-350	3000-18000	128-192
DFT Tester	100-350	150-650	512-2500
Low-end Microcontroller	200-350	1200-2500	256-1024
Commodity Memory	200+	800-1000	1024
RF	200+	50000	32

As can be seen from Table 1-1, the cost of ATE for ASIC, MPU, or Mixed-signal chips is very high. However, if we are able to conduct the test by using DFT or BIST techniques, the required DFT tester is much cheaper. For example, in a typical memory devices with 32 I/O pins, the test cost using a high end tester can be computed according to (1-1) as

$$Cost\ of\ Test\ (High\ End\ Tester) = 250k + 2700 \times 32\ , \tag{1-2}$$

while the test cost using low end DFT tester can be computed as

$$Cost\ of\ Test\ (DFT\ Tester) = 100k + 150 \times 32\,. \tag{1-3}$$

resulting in a 68.85% reduction in the cost of test. Nevertheless, these testers have low precision in timing accuracy, where even high end ATEs have limited timing accuracy (resolution). According to the data in [3], current overall ATE accuracy is around 120ps, which is not enough for precise timing control in many applications. A carefully designed BIST circuit is able to achieve at least comparable accuracy, and this accuracy will keep improving as the technology scales.

BIST is a methodology in which some circuit blocks are built on-chip dedicated for the test purpose [4]. BIST involves some on-chip circuitry and sometimes a low-cost off-chip tester to provide the necessary source signals. As the rising cost of test equipment has been driving the industry to look for BIST solutions which are less expensive, many BIST circuits have been developed to test chip memories or logic cores, such as memory BIST (MBIST) and logic BIST (LBIST). However, in order to fully benefit from the use of BIST by moving to low cost testers, a complete BIST solution is desired which includes both the BIST for the chip core and the BIST for the chip interface, including the test of chip I/O setup time and hold time.

Both phase-locked loops (PLLs) and DLLs are extensively used in many timing circuits. When either a DLL or a PLL can be used, a DLL is preferred in many cases because of its better stability, less jitter accumulation, and faster locking time compared to a PLL. Therefore, in our research, we have identified that the best circuit to generate stable time delays on chip is a DLL. In other words, a DLL-based BIST circuit can be made to serve the purpose of creating the required delays for digital interface testing. A DLL is widely used as a timing circuit in many systems for the purpose of clock generation, signal synchronization, and others. For example, a DLL is able to provide multiple clock signals which are separated from each other by a well-controlled phase shift (delay). When appropriate logic, such as edge combining is used, a new clock signal which is of a different frequency can be generated by the DLL. Such an application of a DLL has been reported in [5] for personal communication services (PCS). Another application of a DLL is for the purpose of clock deskewing in synchronous data transfer among communication chips. Reducing clock skew has become increasingly important with larger die size and higher clock frequency. A DLL for this application requires fast lock time and excellent phase alignment between the reference signal and the corrected output signal. One example of a DLL serving as a clock deskew buffer was reported in [6]. In addition, a DLL can also be used for signal synchronization, for example, between a CPU and a co-processor so that they can share the same data bus. This application was reported in [7].

Due to the wide range of applications of DLLs, the motivation of our research has evolved to designing a high performance DLL which can be used for a variety of applications, including delay generation for testing purposes as well as for other timing applications, such as signal synchronization. Fortunately, many of the above applications share similar requirements for the DLL, such as a short locking time, low jitter, and a wide locking range. These common requirements have become our design goals.

1.2 Problem statement

The primary objective of the proposed research is to design a high performance DLL that could serve a variety of purposes.

In this research, we need to investigate and evaluate various existing DLL design techniques. A DLL typically includes three basic blocks: a phase detector (PD), a charge pump (CP), and a voltage-controlled delay line (VCDL). A simplified DLL block diagram is shown in the following figure.

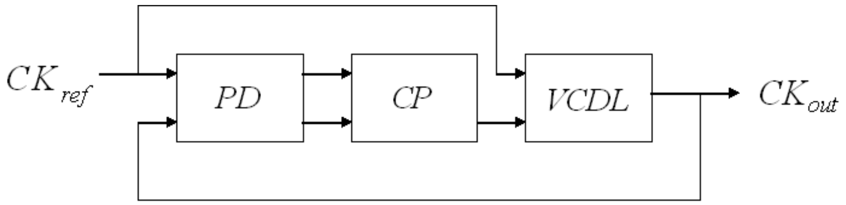


Figure 1-1 Simplified block diagram of a DLL

For a DLL, locking time, lock range and jitter performance are the most important metrics. Locking time refers to the time interval a DLL takes to achieve a stable locking state from an initial state. Generally, locking time is related to the speed of the PD, the magnitude of the charging or discharging current in the CP, and the overall delay loop bandwidth. Lock range refers to the maximum and minimum delays of the VCDL, which set the range in which the delay of the VCDL can be varied. A DLL is able to achieve lock only in this range. Lock range directly affects the operating frequency range of a DLL. Phase noise, or time jitter, is the random variations of the period or phase of a clock signal [8]. With the constantly rising data rate or clock frequency, the clock period becomes increasingly smaller, and so does the tolerance to the amount of time jitter. Therefore, a great deal of design effort should be aimed to improve the jitter performance of a DLL.

In this research, we need to analyze the pros and cons of the previous work, and propose new methods and algorithms to improve a DLL's locking time, locking range, and jitter performance. By adopting these methods, we need to design a new DLL that can achieve our design goals. Since a DLL's dynamic response is the theoretic basis for a DLL's stability, we also need to study the DLL's dynamic response in the *s*-domain.

1.3 Thesis contributions

The major contributions of this thesis include:

(1) A new approach which uses a DLL to generate precise time intervals (or phase shifts) on chip for applications such as BIST;

(2) A novel circuit structure which performs the functionality of both the phase detector and charge pump;

(3) A novel start-control circuit to avoid locking failure or false locking to harmonics;

(4) A new algorithm to increase the locking range of a DLL;

(5) A novel test circuit to indicate if a phase locking has been achieved in a DLL.

A performance comparison between the proposed DLL and some other recently published DLLs is shown in Table 1-2.

Table 1-2 DLL performance comparisons

	JSSC99 [9]	ISSCC00 [10]	ISSCC01 [11]	VLSI02 [12]	VLSI03 [13]	[14]	This work
Process	0.4-um CMOS	0.15-um CMOS	0.17-um CMOS	0.13-um CMOS	0.13-um CMOS	0.18-um CMOS	0.18-um CMOS
Timing resolution	40ps	10ps	X	14ps	X	X	80ps
Max operation frequency	667MHz	1GHz	250MHz	500MHz	500MHz	550MHz	700MHz
Min operation frequency	250MHz	X	100MHz	66MHz	66MHz	350MHz	160MHz
Jitter	250ps	128ps/29ps (quiet)	640ps/200ps (quiet)	X	25ps (quiet)	33ps	25ps
Lock time	2.9us	X	X	>100Cycles	150Cycles	X	11Cycles

X: not mentioned.

1.4 Thesis organization

This thesis is organized into eight chapters. Chapter 1 introduces the motivation of our research, outlines the problems addressed in this thesis, and states the contributions we have made. In Chapter 2, we present the background of this research and review the related previous work. Chapter 3 discusses the general approach of the research. Chapter 4 describes the proposed DLL design in detail. In Chapter 5, the layout of the prototype chip is introduced. Chapter 6 discusses the DLL's performance evaluation. In Chapter 7, we proposes a BIST circuit to test the DLL. Finally, a summary is given in Chapter 8.

CHAPTER 2

PREVIOUS WORK

2.1 DLL overview

DLLs are widely employed in microprocessors, memory, and communication ICs in order to reduce on chip clock buffering delays and improve I/O timing margins. They can also be used to generate multiple clock signals on chip for applications such as for BIST circuits. The essential function of a DLL is to achieve phase alignment between the input clock and the output clock from the final stage of the VCDL. After the phase alignment is achieved, each VCDL delay stage is able to provide a stable clock signal which is phase shifted from the input clock. However, the rising clock speeds and integration levels of digital circuits have made the phase alignment task increasingly difficult [15]. For example, power supply and substrate noise resulting from the switching of digital circuits affects the operation of DLL and leads to output clock jitter. Thus, a detailed study of prior work in DLL design and analysis is needed for us to achieve the design goals of wide lock range, low jitter, and fast locking.

2.1.1 DLL operation principle

A DLL is essentially a nonlinear negative feedback system. However, it is a common practice to characterize a DLL by linear analysis. Although linear analysis is not able to produce a very accurate result, it can still serve as a reasonable first-order approximation and can lead to some useful insights into a DLL's operation.

In a DLL, the input clock signal propagates through the VCDL and develops phase shift (or time delay) at every delay stage of the VCDL. The phase shift of each delay stage is controlled by the voltage of a loop filter. The output is taken from one of the delay stages. The phase of the output signal is compared with the phase of the input clock in the PD. The phase error information generated by the PD (usually in the form of a voltage or a current) is then transferred to the CP. The CP uses the phase error information to adjust the voltage of the loop filter and thus to change the delay of the VCDL. Owing to such a negative feedback mechanism, the phase error is gradually reduced until it finally becomes zero. At that time, the delay of the whole VCDL line becomes equal to one clock period, and the voltage of the loop filter is stabilized, which indicates that a locked state has been established.

2.1.2 DLL closed loop dynamics

The DLL's loop dynamics has been extensively studied before [16]-[17]. There are two types of DLLs with different architectures. In the first type, the input reference clock CK_{ref} is compared with a delayed version of itself in the PD [17]. This DLL structure is often used for frequency synthesis, clock generation, and signal synchronization.

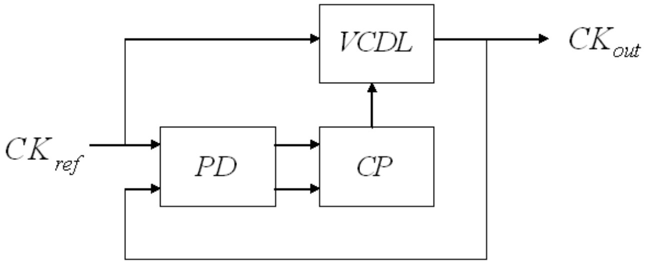


Figure 2-1 The first type of DLL

There is another type of DLL in which the reference signal CK_{ref} is compared with a delayed version of another uncorrelated signal CK_{src} [17], as shown in Figure 2-2.

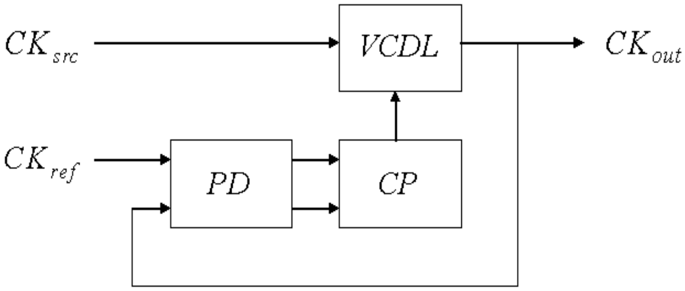


Figure 2-2 The second type of DLL

This type of DLL requires two input signals (CK_{ref} and CK_{src}) and it is often used in some clock recovery circuits.

Since it is the first type of DLL that will be used in the proposed BIST circuit, our focus will be placed on this type of DLL. The DLL's loop dynamics can be analyzed with a continuous time (*s*-domain) approximation where the sampling nature of the PD is ignored. This approximation is valid as long as the DLL's lower limit of the bandwidth is about a decade or more below the operating frequency [16]. Figure 2-3 shows the DLL block diagram in terms of the input and output phase [16].

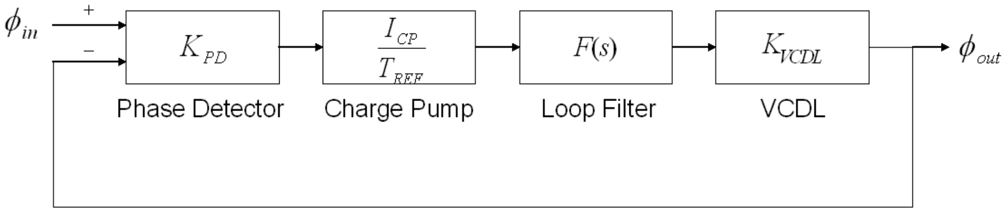


Figure 2-3 *S*-domain model in terms of phase

The parameters in the above figure include the phase detector gain K_{PD} (second/rad), the charge pump current I_{CP} (A), the loop filter transfer function $F(s)$, and the VCDL gain K_{VCDL} (rad/V) which is proportional to the number of delay cells. The input and output phases are denoted by ϕ_{in} and ϕ_{out}, respectively. The period of the input reference clock is T_{REF}.

In most cases, the loop filter consists of only a single capacitor. Thus, the transfer function $F(s)$ can be written as

$$F(s) = \frac{1}{sC_{loop}}. \tag{2-1}$$

Hence, in the steady locked state, the close-loop behavior of a DLL can be characterized by a first-order transfer function

$$\frac{\phi_{out}}{\phi_{in}} = \frac{1}{1 + s/\omega_N}. \tag{2-2}$$

The parameter ω_N represents the close-loop bandwidth (3-dB frequency)

$$\omega_N = \frac{K_{PD}K_{VCDL}I_{CP}}{2\pi C_{loop}}\omega_{REF}\,. \tag{2-3}$$

where ω_{REF} is the frequency of the input reference frequency. It can be seen from Equations (2-2) and (2-3) that a conventional DLL is a single-pole system and therefore is unconditionally stable. Equation (2-3) also shows that the loop bandwidth ω_N can track the operation frequency ω_{REF} as long as the term $K_{PD}K_{VCDL}I_{CP}/C_{loop}$ is kept constant. The effect of such tracking is two-folded. On one hand, the tracking is advantageous because as the operation frequency ω_{REF} increases, the loop bandwidth ω_N also increases which results in a faster acquisition speed. On the other hand, such tracking may not be desirable since a narrower loop bandwidth can help attenuate more high-frequency input noise, since the input jitter would be attenuated at 20dB/decade outside the loop bandwidth. In [17], the following design guideline was suggested:

$$\frac{\omega_N}{\omega_{REF}} = \frac{K_{PD}K_{VCDL}I_{CP}}{2\pi C_{loop}} \le \frac{1}{10} \tag{2-4}$$

In some cases, the loop filter contains an extra pole ω_p [18], which will make the DLL a second-order system.

The problem with the above analysis is that this diagram does not take into account the fact that the output phase ϕ_{out} is one clock period delayed compared to the input phase ϕ_{in}.

2.1.3 DLL jitter analysis

Jitter is one of the most important performance metrics in the DLL design. Jitter refers to the random variations in the period of the output clock signal, and it characterizes the uncertainty of the output clock in the time domain. Assume that t_n is the time point when the nth minus-to-plus zero crossing happens. The nth clock cycle period can then be calculated as $T_n = t_{n+1} - t_n$. Ideally, the clock signal has a clock period of $\overline{T}$. The difference ΔT_n between T_n and $\overline{T}$ ($\Delta T_n = T_n - \overline{T}$) is an indicator of jitter [8].

There are three definitions of jitter in literature [8]. The first one, which is called absolute jitter or long term jitter, is defined as

$$\Delta T_{abs}(N) = \sum_{n=1}^{N} \Delta T_n \tag{2-5}$$

Absolute jitter $\Delta T_{abs}(N)$ represents the accumulated jitter in the first N clock cycles. The second definition of jitter, called cycle jitter, is the long term RMS value of ΔT_n. The definition of cycle jitter is

$$\Delta T_c = \lim_{N \to \infty} \sqrt{\frac{1}{N} \sum_{n=1}^{N} \Delta T_n^2} \tag{2-6}$$

Cycle jitter represents the long term average effect of clock cycle fluctuation. The third definition of jitter is called cycle-to-cycle jitter, which is defined as the RMS difference between two consecutive clock cycles

$$\Delta T_{cc} = \lim_{N \to \infty} \sqrt{\frac{1}{N} \sum_{n=1}^{N} (T_{n+1} - T_n)^2} \tag{2-7}$$

All of the above three parameters can be used to describe the jitter characteristics. In general, absolute jitter is used to describe a PLL as there is significant jitter accumulation in a PLL. For other timing circuits, including DLLs, the other two parameters are usually used [8].

Typically, jitter includes deterministic jitter as well as random jitter [15]. Deterministic jitter is also referred to as systematic jitter and is generally caused by duty cycle distortion and device mismatch. In contrast, random jitter, which is also known as nonsystematic jitter, is mainly due to some random noise sources such as thermal noise, substrate noise, and power supply noise.

The primary random noise sources in a DLL consist of the input clock noise and the noise originating from the VCDL buffer stages. The input clock noise is the noise associated with the input clock signal, while the VCDL noise refers to the noise originating from the MOSFET devices of each buffer stage as well as the substrate and power supply noise coupled to the VCDL. All these noise sources cause jitter in the DLL output signal. The precision and resolution requirements of the BIST application impose a strict limit on the allowable jitter in the output signal. As a result, minimizing random noise is one of the major considerations in VCDL design.

The noise sources for each MOSFET device include channel thermal noise, channel flicker noise, and the noise caused by the gate leakage current [19]. For long-channel MOSFET devices operating in saturation, the channel thermal noise can be represented by a noise current generator connected between the source and the drain with a spectral density of [20]

$$\overline{i_{n,t}^{2}} = 4kT\gamma g_m . \qquad (2\text{-}8)$$

In (2-8), k is the Boltzmann constant (1.38×10^{-23} J/K). The coefficient g_m is the transconductance of the MOSFET and γ is a device-dependent parameter. The channel flicker noise and the noise caused by the gate leakage current are ignored here to simplify the analysis.

To help the design process, the noise analysis of the VCDL buffer stage that will be used in the proposed DLL is introduced here. The topology of the VCDL buffer stage and its noise sources are shown in Figure 2-4.

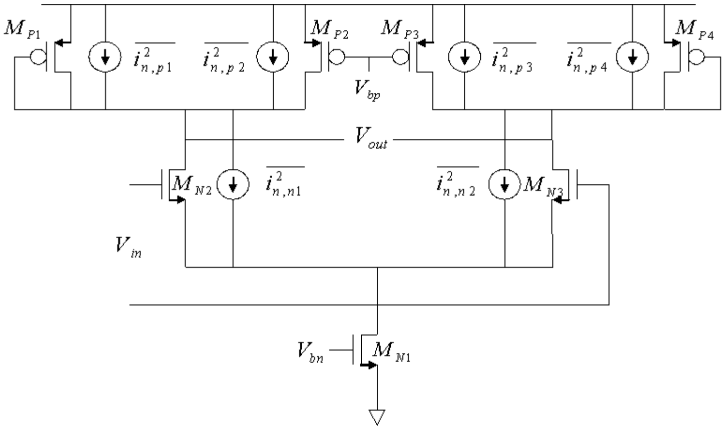


Figure 2-4 VCDL buffer stage and the noise sources

Each noise current source in Figure 2-4 includes both a thermal noise component and a channel flicker noise component. The noise from the tail current source MN1 is not present here since it does not contribute significantly to the overall noise [19]. To calculate the *total* output noise voltage, the two inputs are shorted together and all the noise power densities are converted to the output nodes and then summed up, which is possible since all the noise sources in the circuit are uncorrelated.

The noise current for the two input transistors can be converted to the output nodes as

$$\overline{v^2_{n,out,i}} = \overline{i^2_{n,i}} r_L^2 \quad (i = 1,2), \tag{2-9}$$

where r_L is the PMOS load resistance at one of the output nodes. Similarly, the noise current for each of the PMOS load transistors can be converted to the output nodes as

$$\overline{v^2_{n,out,j}} = \overline{i^2_{n,j}} (r_L // r_{O,N2})^2 \quad (j = 1,2,3,4), \tag{2-10}$$

where $r_{O,N2}$ (or $r_{O,N3}$) is the output resistance of transistor MN2 or MN3. By summing up all the noise power spectral density at the output nodes, the total output noise power spectral density becomes

$$\overline{v^2_{n,out,total}} = 8kT\gamma g_{m,N2} r_L^2 + 16kT\gamma g_{m,P1} (r_L // r_{O,N2})^2 . \tag{2-11}$$

Dividing the total output noise power spectral density by the square of the differential gain $g_{m,N2}^2(r_L // r_{O,N2})^2$ gives the *total* input-referred noise power spectral density

$$\overline{v_{n,in,total}^2} = \frac{8kT\gamma r_L^2}{g_{m,N2}(r_L // r_{O,N2})^2} + \frac{16kT\gamma g_{m,P1}}{g_{m,N2}^2}. \qquad (2\text{-}12)$$

Equation (2-12) can be modified by including the channel flicker noise as

$$\overline{v_{n,in,total}^2} = \frac{8kT\gamma r_L^2}{g_{m,N2}(r_L // r_{O,N2})^2} + \frac{16kT\gamma g_{m,P1}}{g_{m,N2}^2} + \frac{2K_N}{C_{ox}W_{N2}L_{N2}}\frac{1}{f} + \frac{8K_P}{C_{ox}W_{P1}L_{P1}}\frac{1}{f}\frac{1}{g_{m,N2}^2(r_L // r_{O,N2})^2},$$

(2-13)

where K is a process-dependent constant. From Equations (2-12) and (2-13), it can be seen that in order to minimize device noise and flicker noise, the transconductance of the NMOS device $g_{m,N2}$ (or $g_{m,N3}$) must be increased and the transconductance of the PMOS device $g_{m,P1}$ (or $g_{m,P2}$) must be decreased. In addition, larger device area $(W \times L)$ also helps to minimize flicker noise.

Power supply and substrate noise is another noise source of the VCDL, which is especially noticeable in the environment of mixed-signal ICs. The effect of power supply noise can be divided into two components: static power supply noise and dynamic power supply noise [19]. Static power supply noise describes the variations of the stage delay due to changes in the DC values of the power supply voltages, while dynamic power supply noise refers to the temporary fluctuations of the stage delay due to transient change of the supply voltages [19].

One of the most popular VCDL stage configurations is a source-coupled NMOS differential pair plus a NMOS tail current source. The delay of such a stage is proportional to the product of the load resistance and the drain capacitance. If the load is implemented by MOS devices, the resistance of such MOS loads is further dependent upon the tail current. Therefore, to minimize static power supply noise, the tail current source must be made less susceptible to static changes in DC power supply voltage. Several techniques have been proposed for this purpose, such as a cascode current source, supply filtering, and dynamical biasing [19].

To minimize dynamic power supply noise, the loads must provide a differential-mode resistance which is independent of the common-mode power supply noise. A linear resistor is the best candidate for this purpose. However since resistors are expensive in CMOS technology, some special configurations of MOS loads are often used.

Substrate noise represents the substrate voltage variations caused by the coupling of switching activities of neighboring digital circuits, since all the NMOS devices share the same substrate in an N-well process. Variations of the substrate voltage affect the operation of NMOS transistors through altering the threshold voltage by

$$V_{th,N} = V_{th0} + \gamma\left(\sqrt{|2\phi_F + V_{SB}|} - \sqrt{|2\phi_F|}\right), \tag{2-14}$$

where γ is the body effect coefficient and ϕ_F is a process-dependent parameter. PMOS devices are located in local cells and thus are less subject to substrate noise. Therefore, in order to minimize the effects of substrate noise, the rule of thumb is that the external supplied control voltage of the VCDL buffer stage must be referenced to the positive power supply and must be applied only to PMOS devices [16].

2.1.4 Different DLL architectures and comparison

A conventional DLL is typically composed of three components: a PD, a CP, and a VCDL. Conventional DLLs can be categorized as either analog or digital. Generally speaking, a digital DLL has the advantage of being more robust, enabling better process portability, requiring lower supply voltage, and being simpler design. On the other hand, an analog DLL has the advantages of higher power and substrate noise rejection, smaller area and power consumption, and finer phase resolution. The following sections will discuss both digital DLLs and analog DLLs, with the focus on analog DLLs. After that, some other mixed-type DLL architectures will also be introduced.

2.2 DLL design

2.2.1 Digital DLLs

The following figure shows a block diagram for a common digital DLL [21].

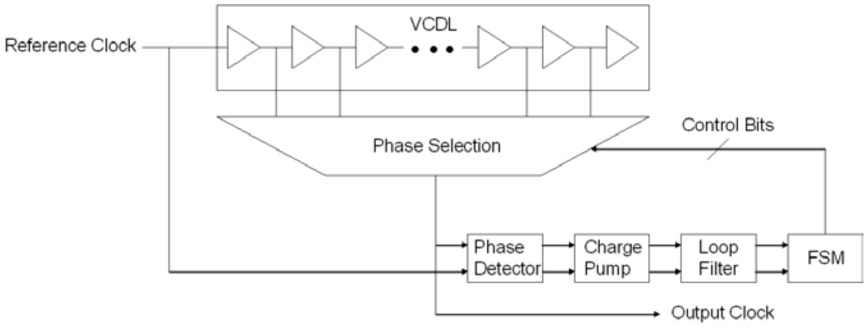


Figure 2-5 Block diagram of a typical digital DLL

As can be seen from the diagram, digital DLLs are characterized by the use of a digital VCDL (using inverters as the delay cells) and a finite state machine (FSM) to select the output signal. Sometimes the PD in a digital DLL is also made from digital elements such as logic gates and flip-flops. Employing more digital parts in the circuit increases the first-run success rate and also improves the portability across different processes. Moreover, digital DLLs are more adaptable to lower supply voltages than analog DLLs. The reason is that for analog DLLs, the voltage headroom problem will occur as supply voltage drops, whereas digital DLLs can still function correctly as long as the supply voltage is sufficient to maintain proper noise margins. The third advantage of digital DLLs is from the perspective of power consumption. When the supply voltage drops, the power consumption of digital circuits will decrease proportional to V^2, while the power consumption of analog circuits only goes down roughly proportional to V [22].

In digital DLLs, since the delay of a single delay element determines the phase resolution of the DLL, a delay element that can provide minimum delay, such as a basic inverter is generally preferred. Because each inverter introduces a 180° phase shift, the delay line is tapped at every other inverter output to provide adjacent phases. Thus, the phase resolution becomes the delay of two basic inverters. A digital DLL must select an output signal whose phase most closely matches the phase of the input signal. Since the desired output is *not* necessarily from the last tap, a FSM which generates digital control bits is usually used for the output selection.

Conventional digital DLLs suffer from several drawbacks [22]. Firstly, the phase resolution of digital DLLs using basic inverter delay lines is limited to two inverter delays, which is not fine enough in many applications. Secondly, since the desired signal is not necessarily from the last tap, the delay line needs to provide a large phase coverage which inevitably requires more delay cells. Thirdly, basic inverters have poor power supply rejection ratio (PSRR), which leads to more jitter in the output signal. Therefore, digital DLLs employing basic inverters in the VCDL are not suitable for applications with stringent timing requirements.

To overcome the limitations of conventional digital DLLs, several techniques have been proposed in the literature to improve the DLL's performance.

A portable digital DLL was proposed in [22] for clock alignment in the interface cells of a high-speed memory system. Two complementary delay lines are used in the digital DLL to improve the phase resolution by a factor of two. An end-of-cycle (EOC) detector is also employed to facilitate the switching between the operations of the two delay lines. In addition, phase blenders are used to further increase the phase resolution of the DLL.

Another improved digital DLL was proposed in [23]. All the components in this DLL are built from digital parts. The DLL uses a 128 tap delay line. A 128-to-1 selector is employed to select the appropriate output signal. An attractive component of this DLL is a digital lead-lag PD which can achieve zero jitter in the locked state.

2.2.2 Analog DLLs

In the literature, most DLLs used in stringent timing applications are of the analog type. For example, analog DLLs are widely used in high speed random access memory (RAM).

Since analog DLLs' architectures and their components vary significantly from one to the other, the following sections will first discuss the implementations of the three basic building blocks: the PD, the CP, and the VCDL. After the discussion of the design of the individual components, the overall analog DLL architectures will be introduced.

2.2.2.1 Phase detector

The PD is one of the most critical components within a DLL. One of the structural differences between an analog and a digital DLL is that the inputs to the PD are different. Both an analog DLL's PD and a digital DLL's PD take the input reference clock as one input. For the other input to the PD, a digital DLL usually uses the output of a digital tap whose phase is the closest to the input reference signal, and this tap is not necessarily the last tap of the VCDL. While for an analog DLL, since the delay of the entire VCDL must be one clock cycle in the locked state, the other input to the PD is directly taken from the end of the VCDL.

The job of the PD is to detect the phase difference between the output signal from the VCDL and the input reference signal, and to pass the phase error information to the subsequent circuits. The basic implementation of the PD is an XOR gate, which is shown in Figure 2-6.

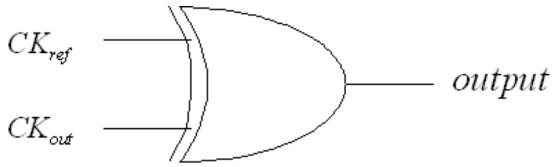


Figure 2-6 A PD based on XOR gate

In an XOR PD, the dc value of the PD output signal is linearly proportional to the phase difference of the two input signals. In addition, the average output signal becomes zero when the two inputs are 90° out of phase. As a result, an XOR PD is often used in quadrature locking where the two input signals to the PD are 90° out of phase in the locked state. The simple XOR PD suffers from several drawbacks. Firstly, there is only one output from the XOR PD, which makes it difficult to interface with the subsequent CP circuit. It is worth mentioning that we have proposed a solution to solve this problem. The circuit we designed is shown in Figure 2-7.

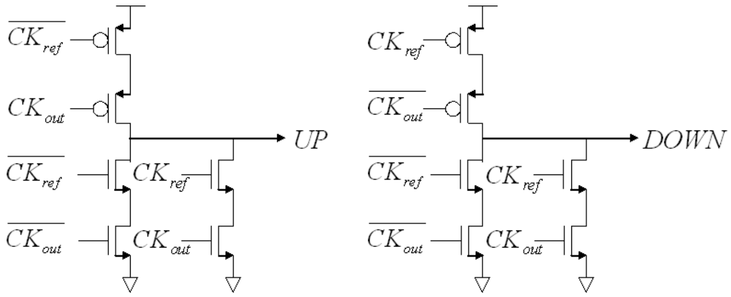


Figure 2-7 A modified PD that provides two outputs

Secondly, an XOR gate is essentially a signal level detector, i.e., the output of an XOR gate is dependent upon the duty cycle of the two input signals. Consequently, XOR

PD may generate incorrect phase error information unless duty cycle correction circuits are used [24].

An improved PD is based on flip-flops. Since flip-flops offer edge detection, the duty cycle dependence problem with the XOR gate PD can be avoided. Figure 2-8 shows a commonly used flip-flop based linear PD, which consists of two flip-flops and one NAND gate [25].

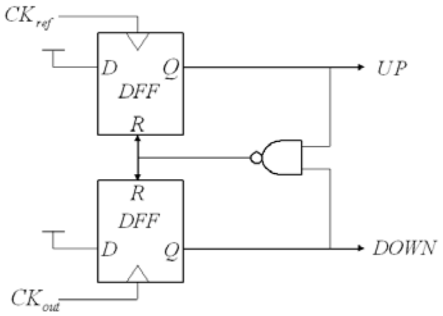


Figure 2-8 A PD based on flip-flops

The flip-flop based PD is capable of detecting both the phase and the frequency difference, which helps to increase the acquisition range and the locking speed. The disadvantage about the flip-flop based PD is that the reset time may limit the speed of the PD, which may further limit the operating frequency and the acquisition speed of the DLL.

A third type of PD, which is widely used recently in high-speed DLL designs, is a dynamic PD. The basic structure of a dynamic PD includes two blocks, which are used to generate the UP signal and the DOWN signal, respectively. The two blocks have exactly the same design, except that the two input signals are switched in position. Each block consists of two cascaded stages with a precharge PMOS in each stage. The precharge activity of the second stage is often controlled by the output of the first stage, as shown in Figure 2-9 [26]. The dynamic PD eliminates flip-flops and has the advantages of simple structure and a fast transition time. However, dynamic PD needs to be carefully designed in order to minimize the dead zone.

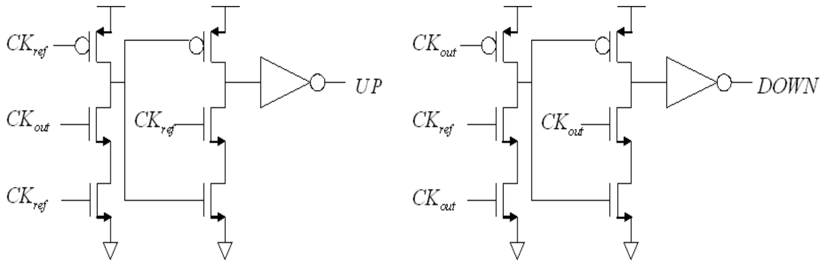


Figure 2-9 A dynamic PD

2.2.2.2 Charge pump

In a DLL, the phase error between the input reference clock and the VCDL output clock is sensed by the PD and transferred to the CP in the form of voltage pulses or current pulses. The CP performs the function of adjusting the voltage of the loop filter and thereby altering the VCDL delay according to the phase error information from the PD. In principle, the CP simply consists of two controlled switches, one current source, and one current sink, as shown in Figure 2-10.

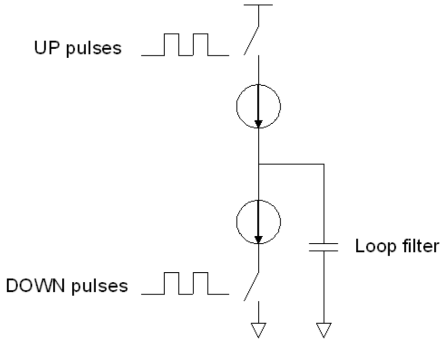


Figure 2-10 A simplified CP diagram

The two switches are controlled by the UP pulses and the DOWN pulses, respectively. Once the switch is closed, the current source or sink will start adding charge onto or removing charge from the loop filter (capacitor). This charging or discharging process will continue until lock is achieved. In the locked state, the voltage (charge) of the loop filter is kept constant. It is possible that equal charging and discharging will still happen in the locked state. In fact, it is desired to have such activities to minimize jitter [16]. However, the charging and discharging currents must be identical as well as very narrow so that the voltage of the loop filter will not be disturbed.

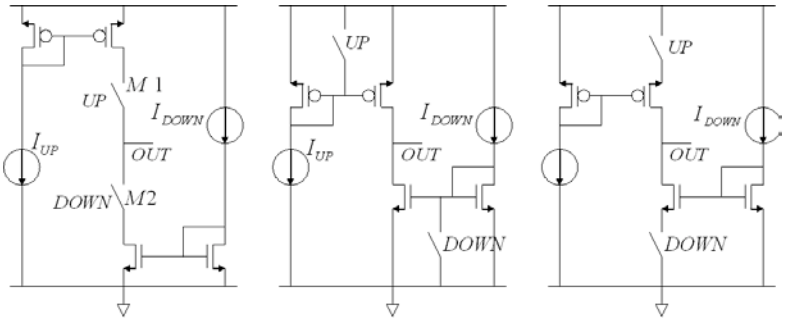
In the actual implementation, the basic structure in Figure 2-10 contains several non-ideal effects that may lead to time jitter [27]. Firstly, the mismatch between the charging and discharging currents I_{up} and I_{down} contributes to time jitter. The mismatch is especially detrimental when both charging and discharging operations happen in the locked state. The amount of the undesired phase shift caused by CP mismatch can be approximated by the following expression [27]

$$\phi_{offset} = 2\pi \frac{\Delta t_{on}}{T_{REF}} \frac{\Delta i}{I_{CP}}, \qquad (2\text{-}15)$$

where Δt_{on}, Δi, and T_{REF} are the turn-on time (the pulse width of the UP and DOWN pulses), the current mismatch, and the period of the input reference clock, respectively. To suppress the effect of current mismatch, the turn-on time Δt_{on} must be minimized. The second non-ideal effect of the CP is the timing mismatch caused by the PMOS and NMOS switches. Due to different characteristics of NMOS and PMOS switches, timing mismatch may occur during charging and discharging operations. Similar to the current mismatch effect, the pulse width Δt_{on} must be reduced to suppress timing mismatch.

Both a single-ended topology and a differential topology CP exist in practical implementations. A single-ended topology has the advantages of smaller area and less power dissipation, but is more vulnerable to supply and substrate noise compared to a differential topology. Nevertheless, a single-ended topology is still popular in CP designs, because a single-ended CP does not need an additional loop filter and its power consumption is lower.

There are three basic configurations for a single-ended CP: switching in the source, switching in the drain, and switching in the gate [27]. These three topologies are illustrated in Figure 2-11 [27].



(a)Switching in drain (b)Switching in gate (c)Switching in source

Figure 2-11 Three single-ended CP configurations

Among the three configurations, switching in source is preferred due to its simpler structure, lower power dissipation and faster switching time [27]. Furthermore, studies show that in CMOS circuits, current switching provides a faster switching speed than voltage switching, if all the other conditions are the same [28]. In addition, several new techniques are proposed to further improve the three basic CP structures. The configuration in Figure 2-11(a) suffers from the charge sharing between the common drain of M1 and M2 and the loop filter when the switch is on (closed). A structure with an active unity-gain amplifier was proposed in [7] to solve this problem. Another enhancement to the basic CP is the adding of two additional current steering switches [29], which greatly improve the switching speed. The third proposed technique is to use only NMOS switches to avoid the mismatch between NMOS and PMOS [27].

A high-performance CP that combines several of the above techniques was proposed in [27]. In this structure, the mismatch problems are avoided by redesigning the NMOS and PMOS switches. Cascode current mirrors are used at the output to increase the output resistance so that the charging or discharging current are not be significantly disturbed.

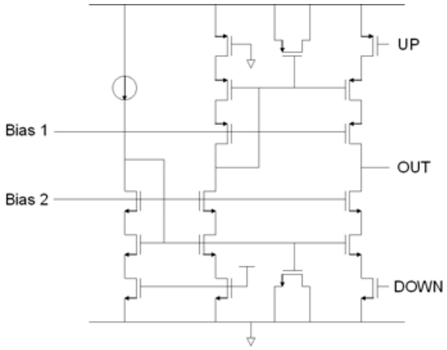


Figure 2-12 A high-performance single-ended CP

A fully differential CP consists of a set of NMOS switches, a set of PMOS switches, two loop filters, and some common-mode feedback circuitry. Although differential CPs are not as widely used as single-ended CPs, they do possess several unique advantages [29]. Firstly, the fully differential structure offers better noise immunity to common-mode noise sources such as supply and substrate noise. Secondly, mismatch between the PMOS and NMOS switches in single-ended CPs does not exist in fully differential CPs. Lastly, the output voltage range can be doubled if the voltages of both loop filters are used. These advantages are achieved at the expense of double chip area and higher power dissipation.

Generally, a single-ended CP can be converted to a fully differential CP by adding duplicate switches and duplicate loop filters. For example, a differential version of Figure 2-12 is shown in Figure 2-13 [27].

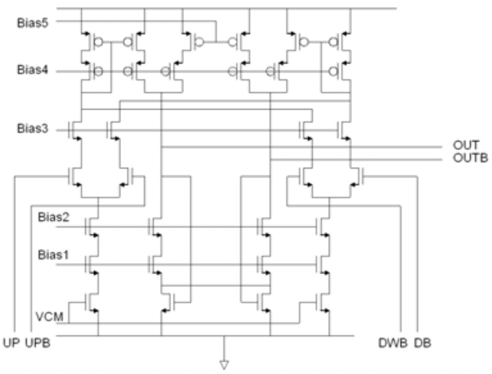


Figure 2-13 A fully differential version of Figure 2-12

2.2.2.3 Voltage controlled delay line

Besides the PD and the CP, the VCDL is also considered to be one of the most critical blocks within a DLL because the output signal of the DLL is directly taken from the VCDL. The VCDL's performance considerably affects the jitter of the output signal and the stability of the DLL.

A VCDL typically consists of a number of delay stages (cells) which are connected in series. A VCDL is an open loop configuration by itself, so it does not oscillate and thereby is different from the voltage-controlled oscillator (VCO) in a PLL.

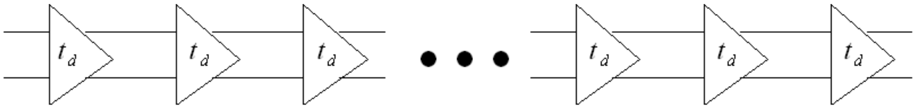


Figure 2-14 A typical VCDL configuration

For analog DLLs, the total delay of the VCDL must be equal to one clock period T_{REF} (or a phase shift of 360°) in the locked state. Theoretically, all the delay stages in the VCDL are identical. Therefore, each delay stage contributes a time delay of T_{REF}/n (or a phase shift of $360^\circ/n$) for an n stage VCDL. Using more stages increases the phase resolution, but also increases the minimum VCDL delay. The design parameters of individual delay cells include output signal swing and delay range.

As usual, the delay cells in analog DLLs can be divided into single-ended types and differential types. Unlike digital DLLs that typically use basic inverters as the delay cells, analog DLLs employ many different configurations for their delay cells. Typical single-ended configurations include a current starved inverter [29] and an RC delay stage [30].

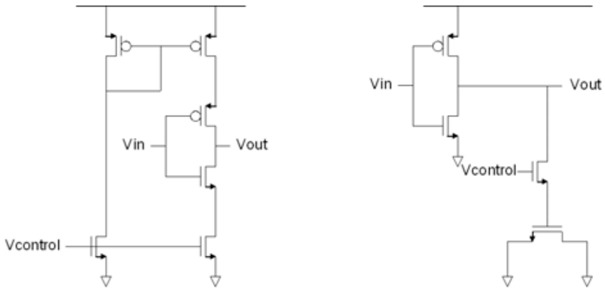
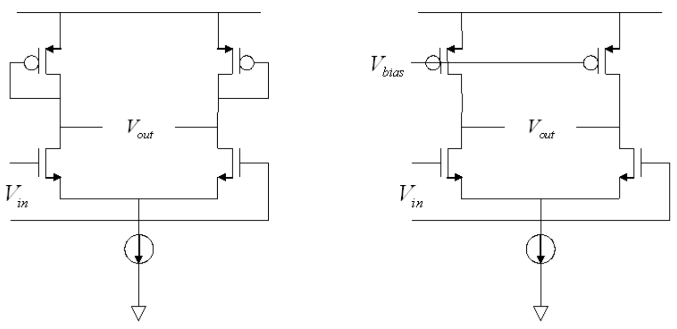


Figure 2-15 Typical VCDL delay stage

Differential delay cells are more widely used due to their inherent advantages such as better immunity to common-mode noise and improved spectral purity. The most popular differential delay stage involves a source-coupled differential pair with two load elements and a biasing tail current source. The load elements are chosen based on design considerations, such as control over the cell delay, the output signal swing, and the dynamic supply noise rejection. Also, an ideal tail current source in the differential delay stage would be highly immune to static supply noise and introduce little voltage headroom.

The MOS load elements in a differential pair can be implemented either in a diode-connected configuration or in a triode configuration, as shown in Figure 2-16 [19].



(a) Diode loads (b) Triode loads

Figure 2-16 Two active loads in VCDL delay cells

For the diode-connected configuration in Figure 2-16(a), the small-signal output resistance r_{out} can be expressed as

$$r_{out} = \frac{1}{g_{mP}} // r_{OP} // r_{ON} \approx \frac{1}{g_{mP}}, \tag{2-16}$$

where g_{mP} is the transconductance of the diode-connected PMOS load, and r_{OP} and r_{ON} are the output resistances for the PMOS transistors and the NMOS transistors, respectively. Since g_{mP} is related to the tail current source I_{SS} by

$$g_{mP} = \sqrt{\mu_p C_{ox} (W/L)_P I_{SS}}, \tag{2-17}$$

the output resistance r_{out} can be written as

$$r_{out} \approx \frac{1}{\sqrt{\mu_p C_{ox} (W/L)_P I_{SS}}}. \tag{2-18}$$

The delay of each delay stage is proportional to the RC constant at the output node, which can be expressed as

$$t_d \approx \sqrt{2} r_{out} C_{out} \approx \sqrt{\frac{2}{\mu_p C_{ox} (W/L)_P I_{SS}}} C_{out}, \tag{2-19}$$

where C_{out} is the effective output capacitance at each output node. As can be seen from (2-18), the delay t_d can be dynamically controlled by adjusting the value of I_{SS}. This structure is attractive due to its simplicity, but several drawbacks prevent it from being attractive. Firstly, since the diode-connected loads consume voltage headroom, the output voltage swing is limited. Secondly, the output DC voltage is not controlled.

The active load elements can also be biased in the triode region by the control voltage $V_{control}$ as in Figure 2-16(b). In this case, the output resistance can be written as

$$r_{out} = R_{on} // r_{ON} \approx R_{on}, \tag{2-20}$$

where R_{on} is the equivalent resistance for the active load in triode region. An approximation for R_{on} is

$$R_{on} = \frac{1}{\mu_p C_{ox} (W/L)_P (V_{DD} - V_{control} - V_{th,P})}, \tag{2-21}$$

where $V_{th,P}$ is the threshold voltage for the PMOS. Based on (2-19), the delay of each stage can be expressed as

$$t_d \approx \frac{\sqrt{2} C_{out}}{\mu_p C_{ox} (W/L)_P (V_{DD} - V_{control} - V_{th,P})}. \tag{2-22}$$

The triode active loads can provide wider output voltage swing than the diode-connected active loads, since in Figure 2-16(a) $V_{out,max} = V_{DD} - V_{th,P}$, whereas in Figure 2-16(b), $V_{out,max} = V_{DD}$. The drawback of this configuration is that it is difficult to maintain the active loads in the triode region.

An improvement to the above two basic delay stages is to use symmetrical loads and a self-biased replica-feedback tail current source bias circuit, as shown in Figure 2-17 and Figure 2-18 [16].

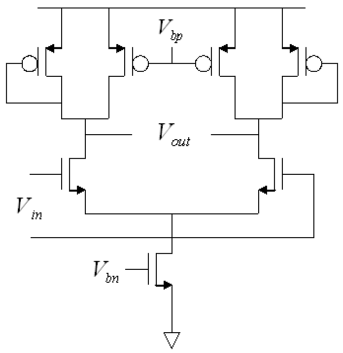


Figure 2-17 Symmetrical loads in VCDL delay cells

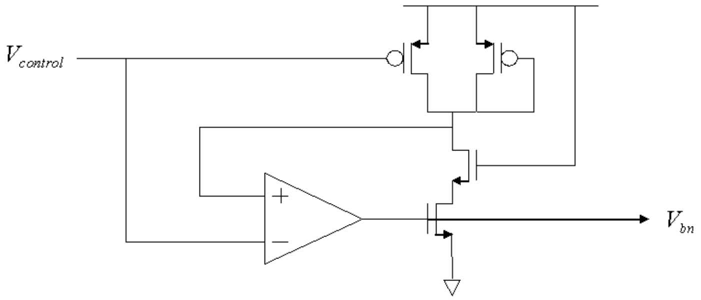


Figure 2-18 Bias circuit for the tail current source

In this structure, each active load consists of a source-coupled PMOS pair. The PMOS pair includes two equal-sized PMOS transistors in shunt, one being diode-connected and the other one being biased in saturation with V_{bp}. The voltage V_{bp} is used to adjust the effective resistance of the symmetrical loads and thereby the buffer delay. It has been proved that the symmetrical loads can provide good control over the cell delay as well as high dynamic supply noise rejection [16]. Furthermore, the bias voltage V_{bn} of the NMOS tail current source is dynamically adjusted so that the tail current is less susceptible to static supply voltage variations [16]

2.2.3 Other DLL structures

Instead of just focusing on redesigning the building blocks of a conventional DLL, many novel DLL architectures have also been proposed in the literature. Many of these DLLs use both analog and digital components to achieve fast speed, low jitter, and wide lock range.

The first DLL design introduced here is a low jitter DLL proposed in [31]. For a conventional DLL, the jitter associated with the input signal can be suppressed by decreasing the loop bandwidth. However, this method can not reduce the jitter originating from the VCDL. To solve this problem, a Butterworth DLL is proposed which can reduce the jitter due to both the input noise and the VCDL noise. The basic idea is to divide the VCDL into several shorter and identical segments so that the VCDL noise does not propagate through the whole VCDL. For example, in a second order Butterworth DLL, the VCDL is divided into two segments, namely, VCDL1 and VCDL2. Theoretical noise analysis shows that the proposed second order and third order DLL can reduce the input noise by a factor of $\sqrt[4]{2}$ and $\sqrt{1.5}$, respectively. The proposed DLL can also reduce VCDL noise by a factor of $\sqrt{2}$ and 2, respectively.

Next, several dual-loop DLL structures will be introduced which have been proven to have a number of advantages over conventional DLLs at the expense of additional power and chip area consumption. The dual-loop DLL in [32] is based on a cascade of a core loop and a peripheral loop. The VCDL in the core loop consists of six delay stages and produces six clocks which are evenly spaced by 30°. The peripheral loop includes a phase selection and inversion circuit, a phase interpolator, a PD, and a digital FSM. The six clocks from the core VCDL are selectively interpolated in the peripheral loop to generate the desired output clock under the control of a FSM. This dual-loop DLL has unlimited phase shift which is largely due to the effective phase capture algorithm implemented by the FSM.

Another dual-loop DLL with multiple VCDLs was proposed to extend the DLL's delay range [33]. The proposed DLL consists of a main loop and a reference loop. The reference loop generates four quadrature clocks, which are delayed by four VCDLs and then multiplexed to generate the desired output clock. This output clock is sent to the main loop for phase comparison and the phase error signal is used to adjust the delays of the four VCDLs. This dual-loop structure allows unlimited delay range.

Another dual-loop structure has been proposed to increase a DLL's operating frequency range in [34]. The operating frequency range of conventional DLLs can be described by the following inequality in terms of the period T_{ref} of the input clock signal [34]

$$\max\left(D_{VCDL}(\min), \frac{2}{3} \times D_{VCDL}(\max) \right) < T_{ref} < \min\left(2 \times D_{VCDL}(\min), D_{VCDL}(\max) \right). \tag{2-23}$$

To extend the operating range, the dual-loop DLL employs an eight-stage core VCDL and a replica delay line. The replica delay line consists of only one delay cell, a current steering PD, and a low pass filter. In the locked state, the delay of the replica delay cell is forced to be one eighth of the reference clock period T_{ref}, and the delay of the core VCDL line is equal to T_{ref}. Thus, if the replica delay cell can be made to have a wide frequency range, the core DLL will have a wide operating frequency range as well. By carefully designing the replica delay cell, the operation range of the core DLL can be extended to

$$Max\left(D_{VCDL}(\min), \frac{1}{7} \times D_{VCDL}(\max) \right) < T_{ref} < D_{VCDL}(\max), \tag{2-24}$$

which is significantly larger than that of conventional DLLs.

Finally, a fast-locking and low-jitter DLL using a digitally controlled half-replica delay line (DHDL) was proposed in [35]. In conventional DLLs, some non-ideal effects such as leakage current may disturb the control voltage $V_{control}$, which further leads to jitter in the VCDL output signal. To solve this problem, the proposed DLL incorporates a DHDL structure to provide a stable bias voltage for the CP so that the control voltage is less affected by these non-ideal effects. As a result, the DLL achieves low jitter operation and at the same time maintains the property of bandwidth tracking.

In summary, there are many techniques and architectures currently available for the design of high-performance DLLs. These methods have been proven to be effective in their respective applications. After comparison of the pros and cons of the available design techniques, a single loop, low-jitter, and wide-lock-range DLL with a newly designed PD, CP, and VCDL is proposed for the BIST application.

CHAPTER 3

OVERALL APPROACH

3.1 DLL for BIST application

As discussed in Chapter 1, the test of digital integrated circuits often involves expensive test equipment in order to obtain the capability of precise timing control. To avoid the cost of using expensive test equipment, we propose to use a DLL-based BIST circuit to generate the necessary delays on chip.

For example, in the testing of setup and hold time, a reference clock and a series of digital test data are supplied to the chip I/O pins concurrently. The reference clock drives the DLL to generate two new clock signals on chip, which are the delayed versions of the reference clock. With the help of the DLL, the delays are set to be aligned with the setup and hold time specifications. Specifically, the setup time test clock is produced by delaying the input clock by the setup time specification. The hold time test clock is generated by delaying the input clock by one clock cycle minus the specified hold time. The newly generated clock signals and the input test data become the inputs to the input registers. The outputs of the input registers are read out and compared with the input test data. If the output data matches the input test data, the register is considered to function correctly and the chip is considered to pass the setup or hold time test. Otherwise, the chip is considered to fail the setup or hold time test.

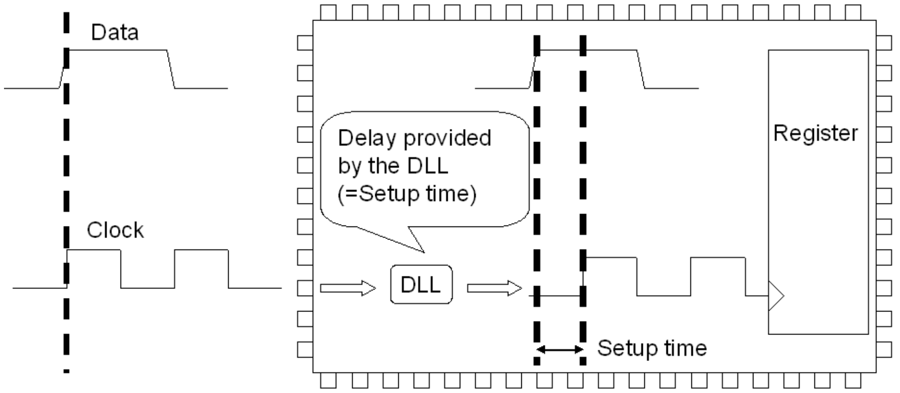


Figure 3-1 Proposed test of setup time

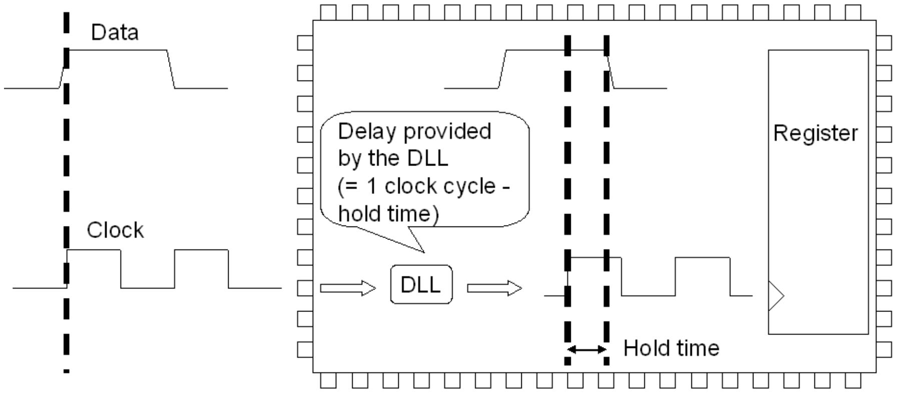


Figure 3-2 Proposed test of hold time

In order to design for realistic specifications, we have selected a test application for the BIST circuit. The DUT is a Fairchild 74ALVC162838 Low Voltage 16-Bit Selectable Register/Buffer [36]. The ALVC162838 contains sixteen non-inverting selectable buffered or registered paths. The device can be configured to operate in a registered mode or a flow-through buffer mode by utilizing the register enable (REGE) and Clock (CLK) signals. The ALVC162838 is suitable for SDRAM DIMM memory modules. A simplified logic diagram of the ALVC162838 is shown in Figure 3-3.

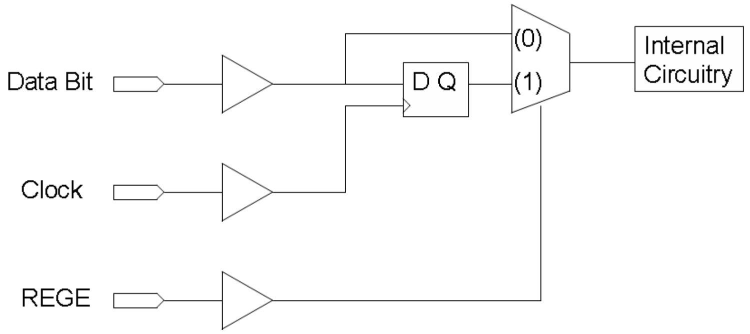


Figure 3-3 A simplified diagram of ALVC162838

At the operating voltage of 1.8V and the operating frequency of 250MHz, the ALVC162838 has a minimum setup time of 2.5ns and a minimum hold time of 1.0ns. The proposed BIST circuit uses an eight-stage VCDL with each delay stage having a delay of 0.5ns (or a phase shift of 45°). Therefore, to test the ALVC162838 setup and hold time specifications, two new test clocks with the delay of 2.5ns and 3ns need to be generated. During the test process, the DLL delay generator will lock to the 250MHz input main clock and produce the desired delay (or the new test clock signals) from the fifth and the sixth stage of the VCDL, respectively.

The test begins with the assertion of the *START* signal, which triggers the DLL locking process. After the *LOCK INDICATOR* signal becomes high (or two times the worst case of the simulated locking time), it is believed that the DLL has achieved lock and the desired delays have been successfully generated. Then the test data is supplied. In order for the BIST circuit to be able to fit a wide range of digital test needs, the DLL must have a wide lock range (or wide operating frequency range). Furthermore, as time jitter significantly affects the quality of the generated time delays, jitter must be minimized. Lastly, as the desired clock signal is taken from some stage of the VCDL, the VCDL delay stages must be matched and the delay of each stage must be equalized.

3.2 DLL for other clock management applications

As was discussed in Chapter 1, DLLs are also widely used in many other timing applications. Different applications often have different focuses in the DLL design. For example, in high speed communication systems or memory applications, jitter is the primary concern. Low power consumption is required for battery-powered mobile devices. Military products need to be able to operate in severe working conditions. The design focus is dependent upon the specific applications.

CHAPTER 4

DESIGN OF THE DLL

Wide lock range, low jitter, and fast locking time are the basic design goals of the DLL. Design trade-offs exist among the three requirements and must be properly handled so that the best overall performance can be achieved.

4.1 DLL architecture design

The first design decision that must be made is the architecture of the DLL. As discussed before, there are two major options: a single-loop architecture and a dual-loop architecture. Although the dual-loop structures provide some advantages over single-loop DLLs, they inevitably introduce more power and area consumption. More importantly, dual-loop DLLs bring in more design complexity. On the other hand, single-loop DLLs can still provide acceptable jitter performance, and by redesigning the individual components, they can also achieve wide lock range and equal cell delays. Based on the above considerations, a single-loop architecture is chosen for the proposed DLL.

4.2 DLL component design

Major redesigns have been done to improve the performance of the DLL. For example, the proposed DLL uses a combined PD and CP circuit for increased speed and reduced jitter. The proposed DLL also employs an eight-stage shift averaging VCDL to improve the matching between delay stages and thus to equalize the delay of each individual stage. Moreover, a start-control circuit is introduced to alleviate the locking failure or false lock to harmonics problems. With the help of the above techniques, the proposed DLL is able to achieve lock as long as the minimum VCDL delay $D_{VCDL}(\min)$ is less than one reference clock cycle T_{REF}. This is the *largest possible* lock range that can be realized by the DLL, since it would be impossible for the DLL to achieve lock if

$D_{VCDL}(\min)$ were larger than T_{REF}. In the next section, a detailed discussion of the design of the BIST circuit will be presented.

4.2.1 Circuit description

After a study of current available DLL circuit structures, a DLL which includes a start-control circuit, a combined PD and CP circuit, a loop filter, and a shift averaging VCDL is proposed. An improved s-domain model which is different from the traditional analysis is also proposed to analyze the DLL's stability. The following sections will discuss the design and the operation of the DLL in detail.

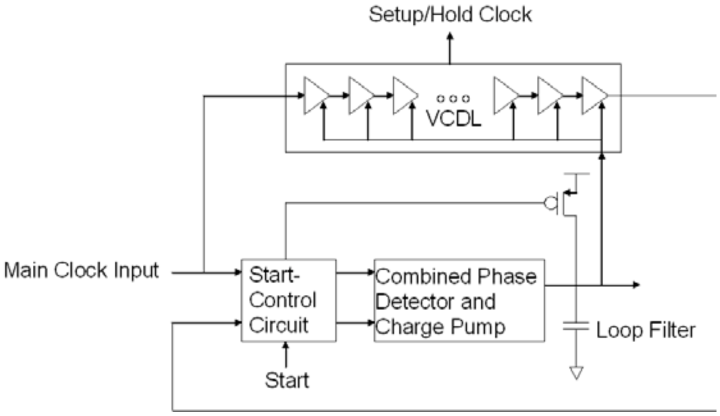


Figure 4-1 The proposed DLL delay generator

4.2.2 DLL lock range

Before the discussion of the details of the DLL delay generator, it is helpful to introduce the concept of lock range again. As discussed before, lock range refers to the minimum and maximum delays of the VCDL which enable a DLL to achieve lock. Since the VCDL can only delay the input reference signal by a limited amount and since it lacks the frequency tuning capability like that of a voltage controlled oscillator (VCO), it is possible for a DLL to run into the problems of failing to lock or locking to harmonic frequencies. Generally, the following two conditions must be satisfied in order to avoid locking failure or false lock to harmonics [21] for a conventional DLL

$$0.5\times T_{REF} < D_{VCDL}(\min) < T_{REF} \tag{4-1}$$

$$T_{REF} < D_{VCDL}(\max) < 1.5\times T_{REF} \tag{4-2}$$

where $D_{VCDL}(\min)$ and $D_{VCDL}(\max)$ are the minimum and the maximum VCDL delays, respectively. The parameter T_{REF} represents the period of the input reference signal. The lock range of conventional DLLs is limited by conditions (4-1) and (4-2), which may be very narrow.

In order to extend the lock range, the proposed DLL adopts several new techniques. The DLL employs a start-control circuit to set the voltage of the loop filter to its maximum value $V_{\max}$ at the beginning of the locking process. As a result, the delay of the VCDL will start from its minimum value $D_{VCDL}(\min)$ and then gradually increase until it reaches one clock period T_{REF}. In addition, the combined PD and CP circuit also helps to further increase the lock range. Owing to the above techniques, the lock range of the proposed DLL is extended to

$$D_{VCDL}(\min) < T_{REF}\,. \tag{4-3}$$

Equation (4-3) is the *largest possible* lock range which can be achieved by the DLL, since it would not be possible for the DLL to achieve lock if the minimum VCDL delay $D_{VCDL}(\min)$ were larger than T_{REF}. Another advantage of setting the initial delay to a minimum is that less jitter will be induced for the VCDL since the delay cell has a faster slew rate when the delay is smaller [28].

4.2.3 Design of the start-control circuit

The operation of the DLL clock generator is started by the assertion of a *START* signal. Before the DLL begins the locking process, the *START* signal is low and the initial voltage of the loop filter is set to a maximum. Consequently, the initial VCDL delay is set to a minimum. After the locking process is started, the delay of the VCDL will gradually increase until lock is reached. As discussed before, setting the VCDL delay to a minimum at the beginning helps to avoid false lock to harmonics and to extend the lock range. The other purpose of the start-control circuit is to preprocess the waveforms of CK_{ref} and CK_{out} for the subsequent phase detection operation. The output signals R_CK_{ref} and R_CK_{out} from the start-control circuit become the inputs to the succeeding PD and CP. The phase difference between CK_{ref} and CK_{out} is the same as the phase difference between R_CK_{ref} and R_CK_{out}.

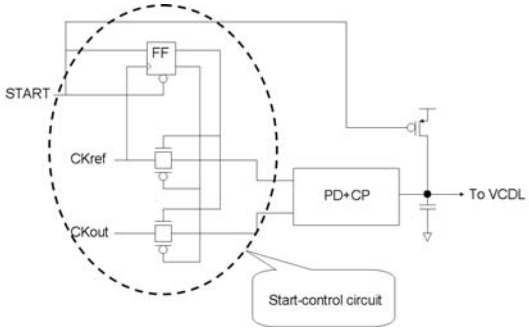


Figure 4-2 The start-control circuit

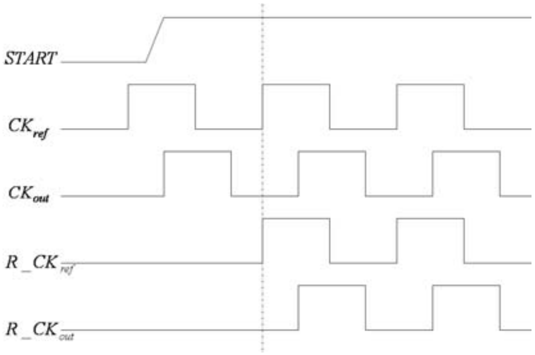


Figure 4-3 The waveforms for the start-control circuit

4.2.4 Design of the combined phase detector and charge pump

As discussed in Section 2.2.2.1, a dynamic PD is often preferred in high speed designs. We have designed a dynamic PD as shown in Figure 4-4.

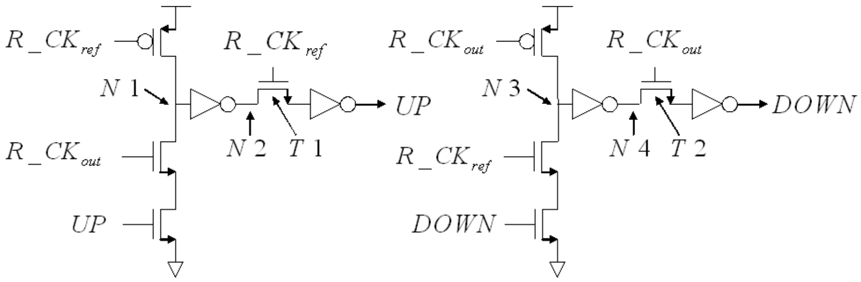


Figure 4-4 The proposed dynamic PD circuit

The behaviour of the proposed PD can be described as follows. Assume that R_CK_{ref} leads R_CK_{out} in phase. In the beginning, node N1 is precharged to high and node N2 is pulled down to low. When the rising edge of R_CK_{ref} arrives, transistor T1 is turned on and an UP pulse is generated accordingly. At a later time, when the rising edge of R_CK_{out} arrives, node N1 is discharged and the UP signal is deasserted after some short delay t_{reset} in the reset path. During this reset delay, the DOWN signal is also asserted. Hence, in every clock cycle the PD asserts both the UP and DOWN signals, and the difference between their pulse widths indicates the phase difference between R_CK_{ref} and R_CK_{out}.

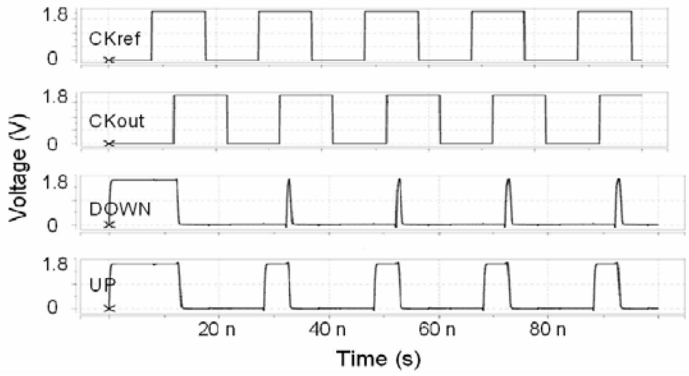


Figure 4-5 The outputs of proposed dynamic PD

After the two inputs R_CK_{ref} and R_CK_{out} become equal in phase, i.e., in the locked state, the PD asserts both the UP and DOWN signals for an equal and short period of time on every cycle, which is illustrated in Figure 4-6. The identical UP and DOWN pulses will trigger equal charging and discharging currents in the CP, which cancel each other and thereby transfer no net charge to the loop filter. The width of such pulses is determined by the delay in the reset path.

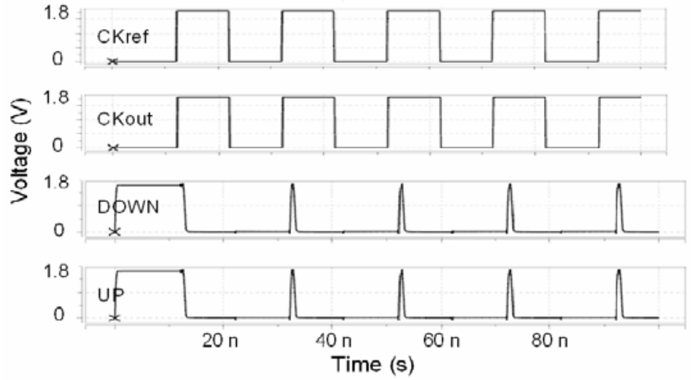


Figure 4-6 The simultaneous pulses in the locked state

The proposed dynamic PD must work together with a separate CP in the DLL circuit. The UP and DOWN outputs of the PD are connected to the charging current switch and the discharging current switch in the CP, respectively. In this manner, the phase difference information detected by the PD is transferred to the CP and is used to control the delay of the VCDL.

In practical designs, it is often desired to integrate the PD and the CP together in order to minimize the die area and the power dissipation. Such an approach was previously taken in [37]. However the design in [37] suffers from two drawbacks. Firstly, the charging or discharging current varies during the charging or discharging process, which may slow down the DLL locking process. Secondly, the charging and discharging currents are not very well matched, which results in higher phase noise and larger clock skew.

Based on the above considerations, a combined PD and CP circuit has been designed which provides a fast switching speed and good matching between charging and discharging currents. The proposed structure also helps to extend the DLL lock range.

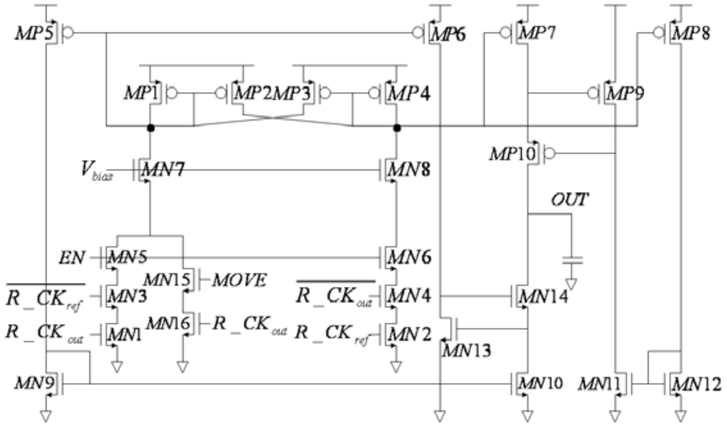


Figure 4-7 The proposed combined PD and CP circuit

The proposed circuit is composed of three major functional blocks: the phase error detection block, which includes transistors MN1 to MN6 and MN15 to MN16; the current switching block, which includes transistors MN7 to MN8 and MP1 to MP4; and the current mirror block, which includes transistors MN9 to MN14 and MP5 to MP10. The phase error detection block (MN1 to MN6 and MN15 to MN16) contains three AND gates, as shown in Figure 4-8.

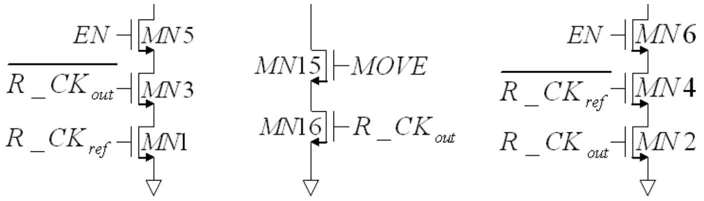


Figure 4-8 Three pseudo AND gates

Each AND gate is composed of two or three series-connected NMOS transistors. During the locking process, one of the AND gates will be turned on corresponding to the phase relationship between R_CK_{ref} and R_CK_{out}. The corresponding current source, MN7 or MN8, will be activated and a fixed current will be produced to charge or discharge the loop capacitor C. As can be seen from Figure 4-8, the three AND gates are controlled by R_CK_{ref}, R_CK_{out}, their complements, and the control signals *MOVE* and *EN*. The signals *MOVE* and *EN* are generated by the following circuits.

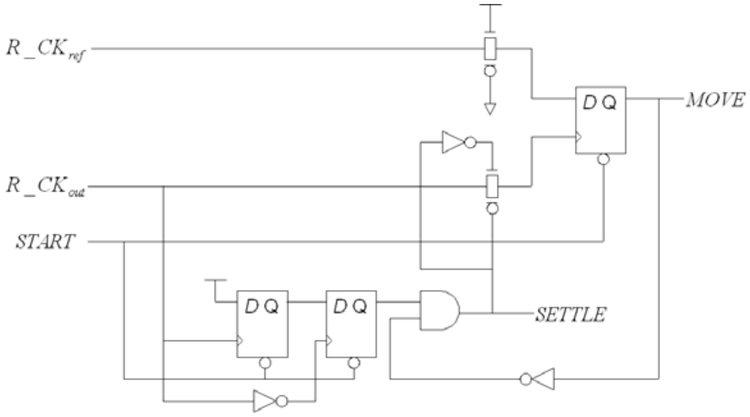


Figure 4-9 The generation of MOVE and SETTLE signals

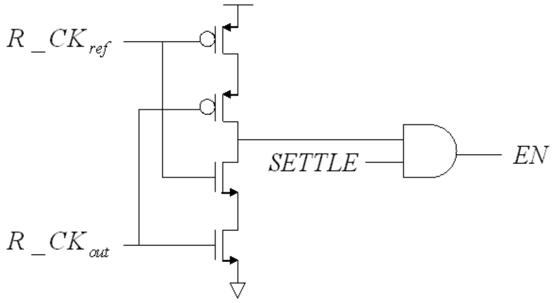


Figure 4-10 The generation of EN signal

If the phase difference between R_CK_{ref} and R_CK_{out} is less than half of the clock period, the values for *MOVE* and *SETTLE* are LOW and HIGH, respectively.

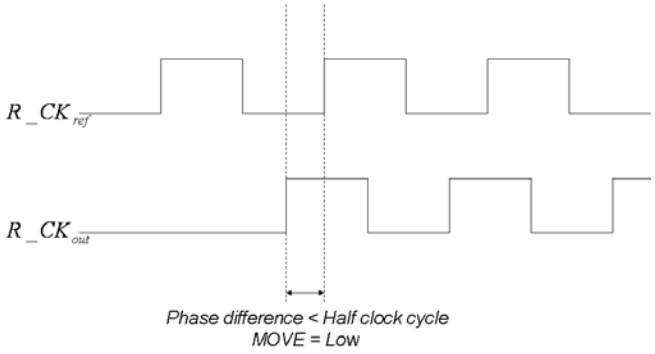


Figure 4-11 The phase difference is less than half the clock period

In this case, the AND gate controlled by *MOVE* and R_CK_{out} will be disabled, and the other two AND gates will work together with the current sources MN7 or MN8. As can be seen in Figure 4-10, the *EN* signal is asserted as soon as both CK_{ref} and CK_{out} become low and will remain asserted until both of them become high. In response to the asserted EN signal, the phase error detection block will detect the phase difference between CK_{ref} and CK_{out}, and turn on the corresponding current source to start the charging or discharging process. Due to the negative feedback mechanism, the phase difference will become smaller and smaller, eventually reaching zero.

On the other hand, if the phase difference between R_CK_{ref} and R_CK_{out} is greater than half the clock period, the *MOVE* signal will be asserted and the SETTLE signal will become LOW.

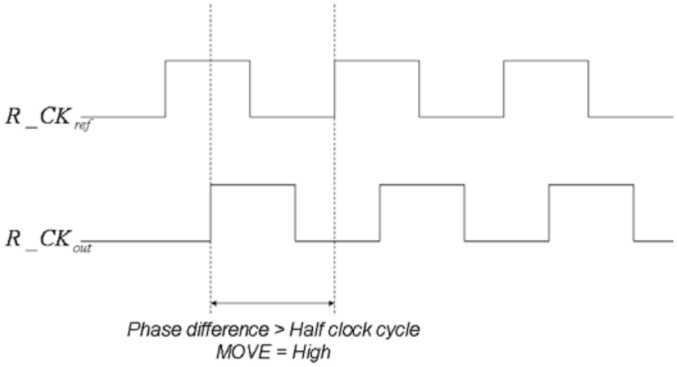


Figure 4-12 The phase difference is larger than half the clock period

Because the *MOVE* signal is HIGH, the AND gate which is controlled by *MOVE* and R_CK_{out} takes over. This AND gate together with the current source MN7 will produce a discharging current to increase the phase delay of R_CK_{out}, i.e., to reduce the phase difference between R_CK_{ref} and R_CK_{out} from greater than half the clock period to less than half the clock period. Once the phase difference between R_CK_{ref} and R_CK_{out}

becomes less than half the clock period, the *MOVE* signal will be deasserted and the other two AND gates will take over, just like the first case.

Owing to the above mechanism, the DLL can achieve lock as long as D_{VCDL}(min) is less than one clock period T_{REF} , which is significantly wider than existing designs.

After the loop is locked, i.e., in the steady state, the CP will produce simultaneous charging and discharging current pulses I_{up} and I_{down} of equal duration for every clock cycle. Such concurrent current pulses are desired to reduce the dead-band region of the PD and thus to avoid additional input tracking jitter [16]. The simultaneous pulses I_{up} and I_{down} must be identical as well as narrow in order not to disturb the voltage of the loop filter in the steady state. According to [37], the current pulses generated by the series transistor structure are significantly narrower than what is possible with a voltage signal. Moreover, the current switching block in the proposed circuit can further narrow the current pulses.

The second functional block is the current switching block. In the proposed circuit, the charging and discharging currents are provided by two identical current sources with the same PMOS loads. Assuming that transistors MN7 and MN8 both work in the saturation region, the charging current and the discharging current can be approximated as (with channel-length modulation ignored)

$$I_{ch\arg ing} \approx \frac{1}{2}\mu_n c_{ox}\left(\frac{W}{L}\right)_8 (V_{bias} - V_{th})^2 \tag{4-4}$$

and

$$I_{disch\arg ing} \approx \frac{1}{2}\mu_n c_{ox}\left(\frac{W}{L}\right)_7 (V_{bias} - V_{th})^2 . \tag{4-5}$$

Since the focus of the BIST application is on accuracy rather than speed, a small charge pump current (about 10μA) is used which leads to a smaller voltage ripple in the loop filter and less jitter in the output signal. Moreover, a larger pump current may result in a large loop gain, which will compromise the stability of the DLL.

Two pull-up transistors, MP2 and MP3, are added to further enhance the switching speed. Current switching happens when the charging activity immediately follows the discharging activity, or vice versa. For example, in the case that the discharging process happens immediately after the charging process, the current source MN7 is turned on to supply the discharging current I_{down}. At the same time, both MP1 and MP2 will be turned on since they have the same source-gate voltage. Transistor MP1 is used to conduct the discharging current I_{down} while MP2 is used to charge the gate of MP4 so that MP4 can be turned off very quickly. In this way, the transition time from the charging process to the discharging process is reduced. A similar scenario happens when the charging process immediately follows the discharging process.

The third functional block is the current mirror block. In the proposed circuit, high-precision current mirrors are used so that the charging current I_{up} and the discharging current I_{down} can be matched very well. Transistors MN13 and MP9 are used to enhance the matching accuracy and to boost the output impedance. Specifically, the introduction of current branch MN13 guarantees that the drain-source voltages of MN9 and MN10 are exactly the same. Similarly, the introduction of current branch MP9 guarantees that the drain-source voltages of MP7 and MP4 are exactly the same. Therefore, MN9 and MN10 have the same gate-source voltage as well as the same drain-source voltage. Consequently, the currents flowing through MN9 and MN10 are exactly equal to each other. Similarly, the currents flowing through MP7 and MP4 are also the same.

High output impedance can reduce the disturbance from the output to the current mirror. Assuming that the voltage gains of MP9 and MN13 are A_{P9} and A_{N13}, respectively, that the drain-source resistances of MP7, MP10, MN14, and MN10 are r_{P7}, r_{P10}, r_{N14}, and r_{N10}, respectively, and that the transconductances of MP10 and MN14 are g_{mP10} and g_{mN14}, respectively, the output resistance can be estimated as

$$R_{out} \approx g_{mN14} r_{N14} r_{N10} A_{N13} // g_{mP10} r_{P10} r_{P7} A_{P9} . \qquad (4\text{-}6)$$

4.2.5 Design of the voltage-controlled delay line

In order to minimize the sensitivity to supply and substrate noise and to achieve a wide tuning range, the delay stage in [16] is used in the proposed DLL. This delay stage is built with a differential topology using symmetrical loads and replica-feedback biasing, which has been discussed in detail in Chapter 2. A direct result of dynamic biasing is that the output signal swing becomes from the control voltage V_{bp} to the top supply V_{DD}. Due to such output swing limitations, the output signal will demonstrate severe distortion when V_{bp} is close to V_{DD}, and therefore will become unusable. Since the voltage V_{bp} is generated from the CP control voltage $V_{control}$, the unusable region of the output signal will in turn pose a limit on the useful range of the control voltage $V_{control}$, which effectively limits the operating frequency range of the VCDL in practice.

As mentioned before, the DLL was designed so that as long as the minimum VCDL delay $D_{VCDL}(\text{min})$ is smaller than T_{REF}, the DLL will be able to achieve lock. Simulation shows that $D_{VCDL}(\text{min})$ is around 180ps. Therefore, the design theory predicts that the DLL will be able to operate as long as the operating frequency is less than 700MHz. However, due to the output signal swing limitation discussed above, the actual operating frequency range of the DLL is from 160MHz to 700MHz.

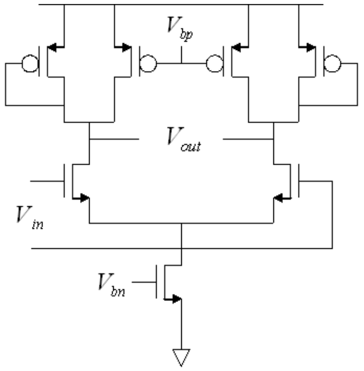


Figure 4-13 Delay stage of VCDL

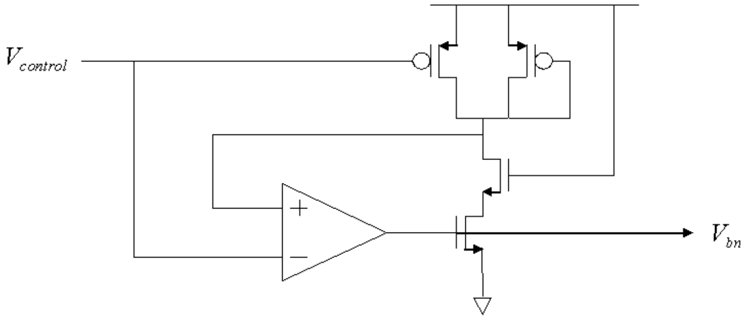


Figure 4-14 Replica-feedback biasing for the VCDL

In the DLL clock generator, the accuracy of the generated setup clock and hold clock signals relies on the matching between the delay stages. In order to improve the matching, a shift-averaging VCDL [38] is used in the design. The shift-averaging technique can equalize the delay of each delay stage as well as improve the duty cycle of the generated clock signals [38]. This technique requires the VCDL to have an even number of delay stages. The shift-averaging VCDL is shown in Figure 4-15.

A design trade-off exists in the number of delay stage in the VCDL. More delay stages can enhance the phase resolution of the VCDL output signals, but at the same time more delay stages will increase the minimum VCDL delay and decrease the high end of the DLL's operating frequency range. On the other hand, fewer delay stages may boost the high end of the DLL's operating frequency range. In the proposed DLL, 8 stages are used.

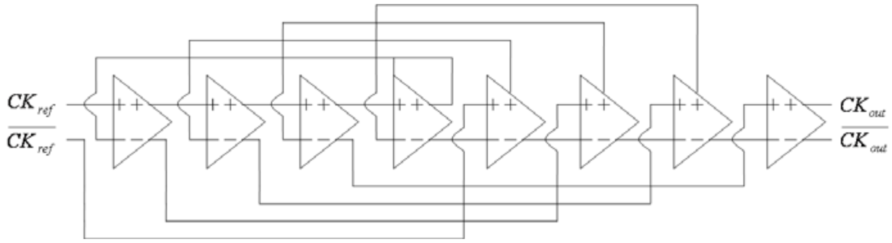


Figure 4-15 Eight-stage shift averaging VCDL

Since the DLL can only generate clocks from a finite number of delay stages, the number of available time delays is limited by the number of delay stages in the VCDL. In other words, the target application influences the choice of the number of stages used in the VCDL. For example, if the operating frequency is 250MHz, the delays that can be generated are 0.5ns, 1.0ns, 1.5ns, 2.0ns, 2.5ns, 3.0ns, and 3.5ns.

4.3 S-domain analysis of the DLL

An improved DLL block diagram is shown in Figure 4-16. The introduction of a delay block in Figure 4-16 reflects the fact that the PD is comparing the current reference clock with the output clock which is derived from the previous reference clock.

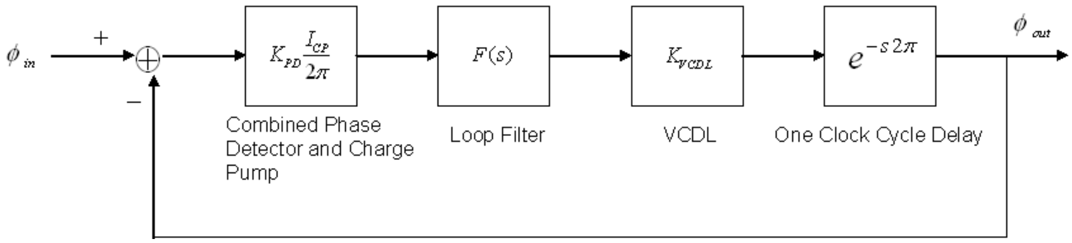


Figure 4-16 An improved DLL block diagram

According to the diagram in Figure 4-16, the DLL transfer function can be expressed as

$$\frac{\phi_{out}}{\phi_{in}} = \frac{e^{-s2\pi} K_{PD} \frac{I_{CP}}{2\pi} F(s) K_{VCDL}}{1 + e^{-s2\pi} K_{PD} \frac{I_{CP}}{2\pi} F(s) K_{VCDL}} . \tag{4-7}$$

Equation (4-7) can be further rewritten as

$$\frac{\phi_{out}}{\phi_{in}} = \frac{G}{G + e^{s2\pi} sC_{loop}} , \tag{4-8}$$

where the parameter G is equal to $K_{PD}K_{VCDL}I_{CP}/2\pi$. The form in Equation (4-8) has a constant in the numerator, and an infinite order polynomial in the denominator. It is clear that the transfer function of the DLL has low pass characteristics. The poles of Equation (4-8) are the roots of the equation

$$e^{s2\pi} s = -\frac{G}{C} . \tag{4-9}$$

The roots of Equation (4-9) or the poles of Equation (4-8) are located in the left half of the complex plane. Therefore, the DLL is stable.

CHAPTER 5

LAYOUT OF THE DLL

The layout of an integrated circuit defines the geometries that appear on the masks used in fabrication [39]. Careful layout helps to achieve the design goals. In this section, we will discuss some of the techniques we used in the layout of the prototype chip.

5.1 Improving matching

Matching is an extremely important issue for analog integrated circuits to achieve the desired performance. Significant mismatch can be very harmful for some sensitive circuit blocks such as differential pairs or current mirrors. In the proposed DLL, the output clocks are directly taken from the delay stages of the VCDL and each delay stage is based on a differential pair. Thus, the matching between the VCDL delay stages and the matching of the transistors within each delay stage are crucial for the proper operation of the DLL. In the layout process, all possible precautions have been taken to minimize mismatch.

Generally, mismatch can be divided into two categories: random mismatch and systematic mismatch. Random mismatch stems from microscopic fluctuations in dimensions, doping levels, oxide thicknesses, and other parameters that influence component values [39]. Random mismatch can not be completely eliminated as it is intrinsic to the fabrication process. However, its impact on circuit performance can be greatly reduced through proper selection of the component values and device dimensions [39]. In contrast, systematic mismatch originates from process biases, contact resistances, nonuniform current flow, diffusion interactions, mechanical stresses, temperature gradients, and a host of other causes [39]. In the layout of analog circuits, maximum design efforts must be made so that critical circuit components are not sensitive to systematic mismatch.

Typically, transistors of different widths and lengths do not match very well. Therefore, a uniform channel length (the minimum channel length) is used for all the transistors in the layout. Moreover, transistors that need be matched in each delay stage are divided into multiple fingers with all fingers being of the same width and length. Identical finger length and width can reduce systematic mismatch of a specific fabrication run. A transistor with multiple fingers also shows reduced junction area and gate resistance [39].

Symmetry is another important factor that affects matching. In the layout of the prototype chip, fully differential signaling is used to reduce the impact of common-mode noise and even-order nonlinearity. The circuit blocks that involve differential signals are placed in a symmetrical position to improve matching.

It has been shown that when all other matching considerations have been addressed, residual random mismatch scales inversely with the square root of device area [39]. Therefore, transistors of relatively large active area are used when possible.

Transistors that must be matched in each delay stage are placed in parallel to one another. In addition, delay stages are also placed in parallel. Transistors that do not lie ın parallel are vulnerable to stress and tilt induced mobility variations, as many steps in lithography and wafer processing behave differently along different axes [39]. The layouts for a single delay stage and for an 8-stage VCDL are shown in Figure 5-1 and 5-2, respectively.

Even in the same die, transistors are vulnerable to gradients in temperature, stress, and oxide thickness. Gradients may lead to significant mismatch. Therefore, matched devices are placed as close as possible [39].

Stress gradients reach the minimum at the center and the maximum in the corners of a die. Therefore, sensitive analog blocks with matched devices are placed in the center of the chip to avoid high stress gradients.

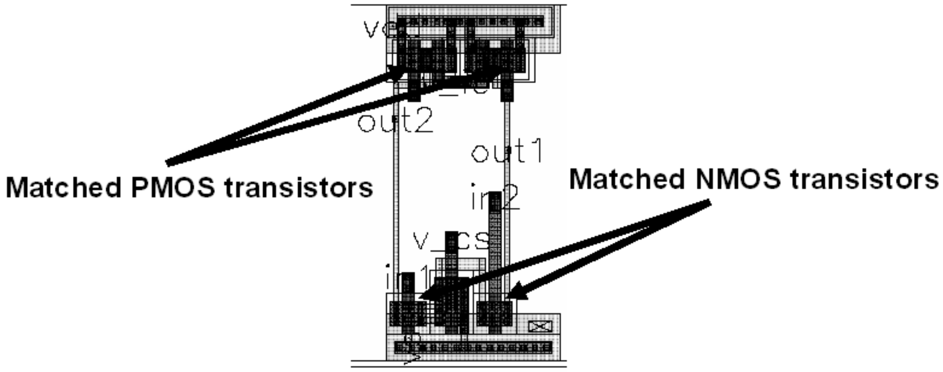


Figure 5-1 Layout of a single delay stage.
Matched transistors are placed in parallel.

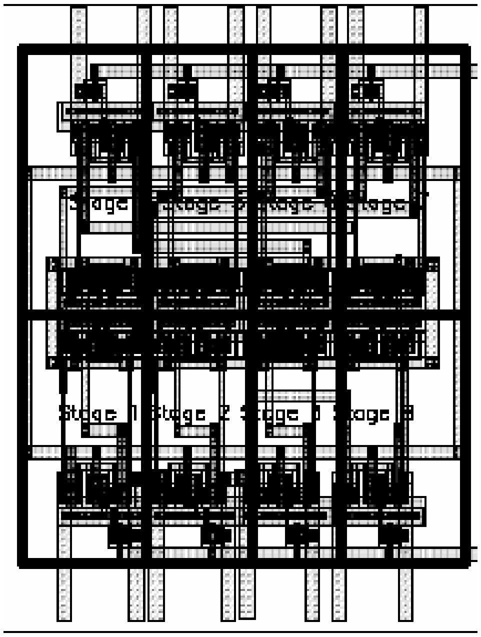


Figure 5-2 Layout of 8-stage VCDL Matched stages are in parallel.

5.2 Reducing substrate and power noise

In the mixed-signal environment, analog circuits are severely affected by the fast switching digital signals which are coupled to analog circuits through low-resistive p+ substrate. Therefore, special methods must be used in the layout to minimize the impact of substrate noise on analog circuits.

The most straightforward way to reduce the effect of the substrate is to separate the analog blocks and the digital blocks as much as possible. In the layout of the prototype chip, the core DLL circuit, which is a sensitive analog block, is placed as far from the digital output drivers, which are significant sources of substrate noise. However, this technique is of limited value in practice as it will inevitably incur long interconnect lines, degrading circuit performance.

Other techniques used in the layout of the prototype chip include using guard rings to isolate the sensitive analog blocks from the substrate noise. In the prototype chip, the sensitive blocks of the DLL are surrounded by guard rings which are continuous substrate ties connected to ground. The substrate ties provide a low impedance path to ground so that the noise will be routed to ground [19]. In addition, n-wells are used outside the substrate ties to further suppress the substrate noise. Figure 5-3 shows a sensitive circuit block, the combined PD and CP, which is surrounded by substrate ties and n-wells.

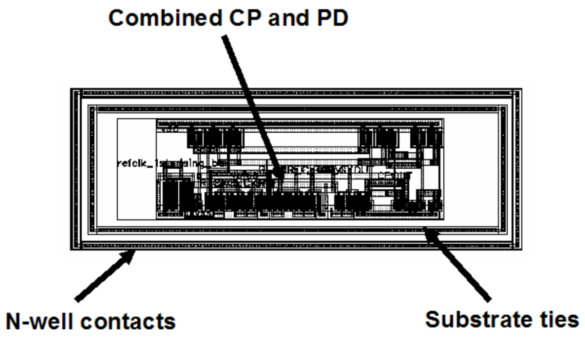


Figure 5-3 PD and CP enclosed by guard rings

Noise on the VDD line (ripples) and GND (GND bounce) can drastically increase clock skew and output jitter [19]. In the layout of the prototype chip, the following methods were used to minimize the effect of power supply noise. Firstly, the VDD line and the GND line are run in parallel to each other, so that opposite voltage glitches cancel each other. Secondly, separate power supplies are used for digital blocks and analog blocks.

5.3 Minimizing crosstalk

Crosstalk refers to the capacitive and inductive coupling between adjacent signal lines [39]. The most effective and most straightforward way to minimize crosstalk is to separate signal lines as much as possible. In particular, critical signal lines, such as clocks, must be isolated from other signal lines as well as power lines [39]. Another method which was used in the layout of the prototype chip is to run differential signal lines in parallel [39]. In addition, careful planning is done in the routing of signal lines. For example, adjacent signal lines are routed on different metal layers. Analog signal lines are kept as short as possible and are kept from crossing any digital tracks. Figure 5-4 shows the start-control circuit and some other circuits as well as the connections between them.

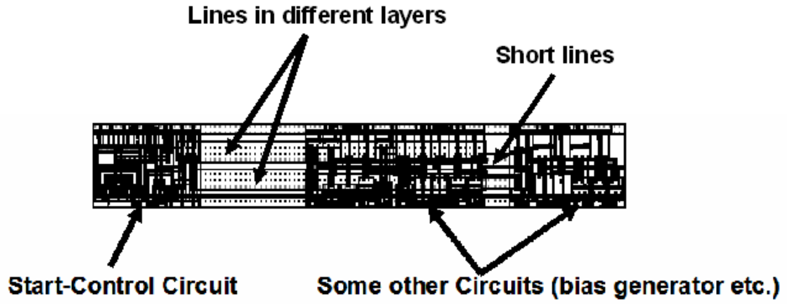

Lines in different layers
Short lines
Start-Control Circuit
Some other Circuits (bias generator etc.)

Figure 5-4 Start-control circuit and some other circuits

Figure 5-5 shows the layout of the core circuit, and Figure 5-6 shows the layout of the entire chip.

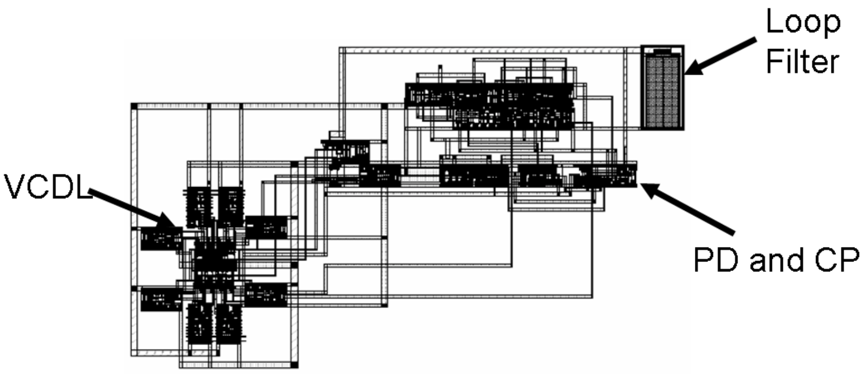


Figure 5-5 Layout of the core circuit

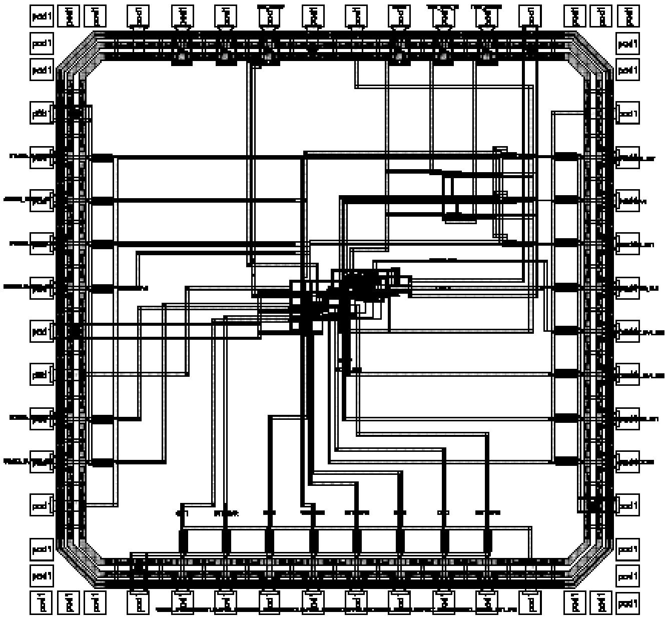


Figure 5-6 Layout of the entire chip

CHAPTER 6

PERFORMANCE EVALUATION

6.1 Performance summary

The proposed BIST circuit is fabricated with the TSMC 0.18-μm one-poly six-metal CMOS process (CL018). Extensive simulations have been done to test the functionality of the prototype chip using device and process parameters provided by TSMC. The following table summarizes the performance of the prototype chip.

Table 6-1 DLL Performance summary

Technology	TSMC 0.18um 1P6M CMOS
Power supply	1.8V
Active die area	0.06 mm^2
Operating frequency	160MHz – 700MHz
Charge pump current	≈ 10uA
Loop bandwidth	≈ 0.07× ω_{REF}
Lock-in Time	11 cycle @ 250MHz

6.2 Process variation

In the semiconductor fabrication process, the characteristics of a MOS transistor can vary substantially from wafer to wafer and even from location to location on the same wafer. This variation is primarily due to the non-uniformity of the processes such as impurity concentration density, oxide thickness, and diffusion depth. Specifically, the following transistor parameters can vary with processes.

(1) The threshold voltage V_{th0}

The threshold voltage of an NMOS transistor is given as

$$V_{th0} = \Phi_{MS} + 2\Phi_F + \frac{Q_{dep}}{C_{ox}}, \tag{6-1}$$

where Φ_{MS} is the difference between the work functions of the polysilicon gate and the silicon substrate, $\Phi_F = (kT/q)\ln(N_{sub}/n_i)$, Q_{dep} is the charge in the depletion region, and C_{ox} is gate oxide capacitance per unit area. As can be seen from equation 6-1, the threshold voltage V_{th0} may vary with the changes in the silicon substrate, polysilicon gate, doping concentration, surface charge, or oxide thickness. Similarly, the threshold voltage of PMOS transistor also varies as a function of the above parameters.

(2) Process transconductance K'

Process transconductance K' can be expressed as

$$K' = \mu C_{ox}, \tag{6-2}$$

where μ is the mobility of the carrier. The main cause for variations in process transconductance is the change in oxide thickness t_{ox}.

(3) Dimensions W and L

Limited resolution of lithographic processes can cause variations in the dimension of transistors (W and L). As a result, the W/L ratio of transistors may be affected.

The circuit is first simulated using the transistor models at corner conditions provided by TSMC, namely, TT (typical NMOS, typical PMOS), FF (fast NMOS, fast PMOS), SS (slow NMOS, slow PMOS), FS (fast NMOS, slow PMOS), and SF (slow NMOS, fast PMOS). These transistor models represent the different characteristics of the transistors at extreme process corners. The following tables list some of the transistor parameters that change at process corners.

Table 6-2 NMOS parameters at different process corners

	V_{th0}	t_{ox}	U0
TT	0.4452004	4.08e-9	0.04590562
FF	0.3452004	3.947e-09	0.04780528
SS	0.5452004	4.213e-09	0.04281001
FS	0.3452004	4.08e-09	0.04633423
SF	0.5452004	4.08e-09	0.04463334

Table 6-3 PMOS parameters at different process corners

	V_{th0}	t_{ox}	U0
TT	-0.4379811	4.08e-9	0.01091231
FF	-0.3709811	3.947e-09	0.01112962
SS	-0.5049811	4.213e-09	0.009606076
FS	-0.5049811	4.08e-09	0.01004072
SF	-0.3709811	4.08e-09	0.01027813

The proposed circuit functions correctly under TT, FF, SS, FS, SF process corners. Figure 6-1 and 6-2 show the transient response of the DLL.

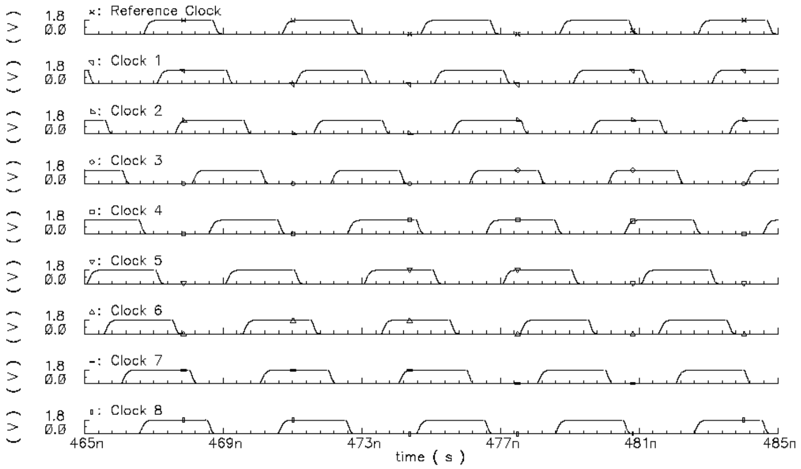


Figure 6-1 Eight DLL output clocks in a typical process @ 250 MHz

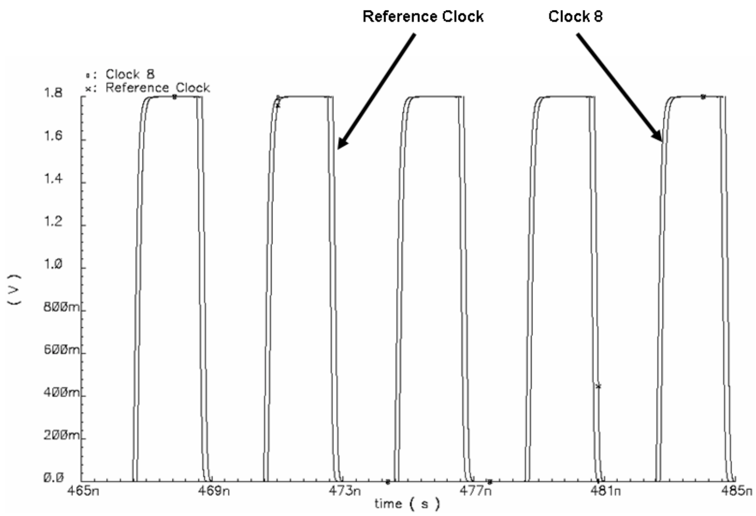


Figure 6-2 DLL phase alignment in a typical process @ 250 MHz

6.3 Temperature variation

Many parameters of MOS transistors vary with temperature. Generally, the following parameters are considered to be temperature-dependent: threshold voltage V_{th0}, built-in potential of S/D junctions, carrier concentration of silicon n_i, and the carrier mobility μ [19]. The DLL was designed to operate in the standard commercial temperature range (0 °C to 70 °C). Simulations have been done at the two temperature extremities to check the functionality of the chip in this temperature range. Figure 6-3 through Figure 6-6 show the output clock waveforms of the DLL at temperature extremities.

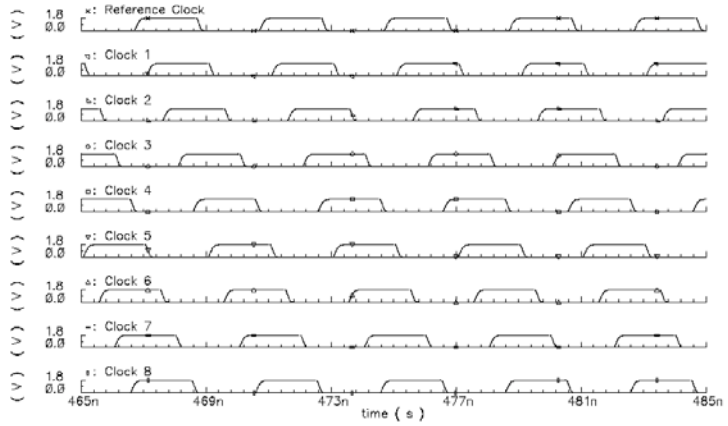


Figure 6-3 Eight DLL output clocks in a typical process and 0 °C @ 250 MHz

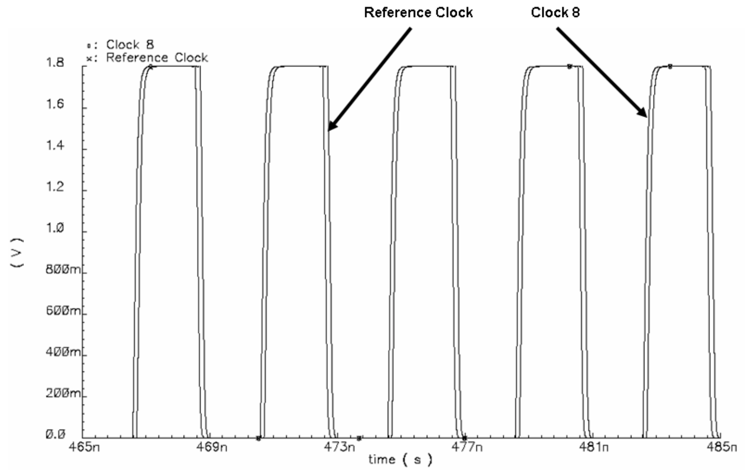


Figure 6-4 DLL phase alignment in a typical process and 0 °C @ 250 MHz

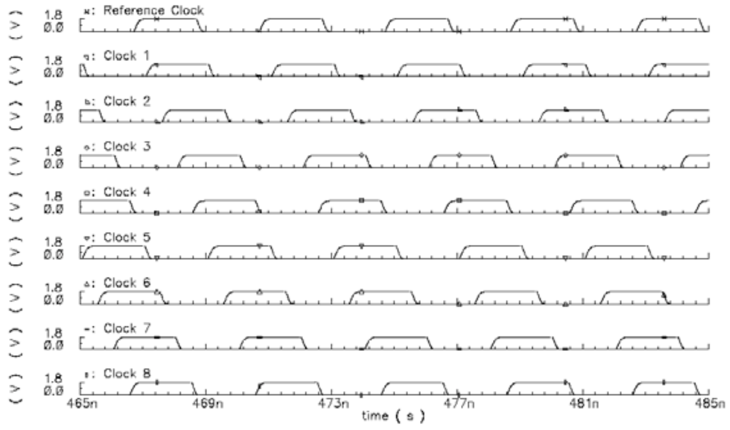


Figure 6-5 Eight DLL output signals in a typical process and 70 °c @ 250 MHz

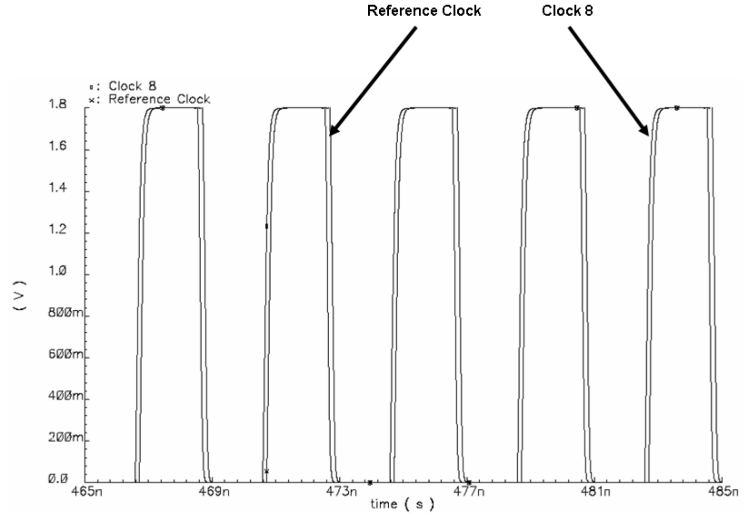


Figure 6-6 DLL Phase alignment in a typical process and 70 °c @ 250 MHz

6.4 Power grid glitch

In reality, power grid is often disturbed by glitches. We simulated the performance of the DLL in case of power supply glitches. In the simulation, a power grid glitch is modeled as a pulse whose magnitude is 15% that of the normal power supply voltage and whose length is one clock cycle. Figure 6-7 and 6-8 demonstrate the output clock signals of the DLL under the influence of such power grid glitch in different process corners and different temperatures. From Figure 6-7, it can be seen that such a power glitch from 470ns to 474ns leads to some distortion of the output signals but they quickly settle down to normal value. Figure 6-8 shows that the phase alignment of the Clock 8 and the input reference clock is also shortly disturbed by the power glitch but rapidly recovers after the glitch.

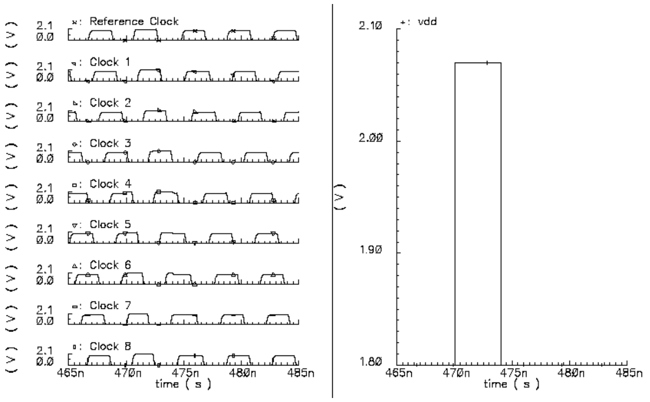


Figure 6-7 Eight DLL output clocks under the influence of power grid glitch @ 250 MHz

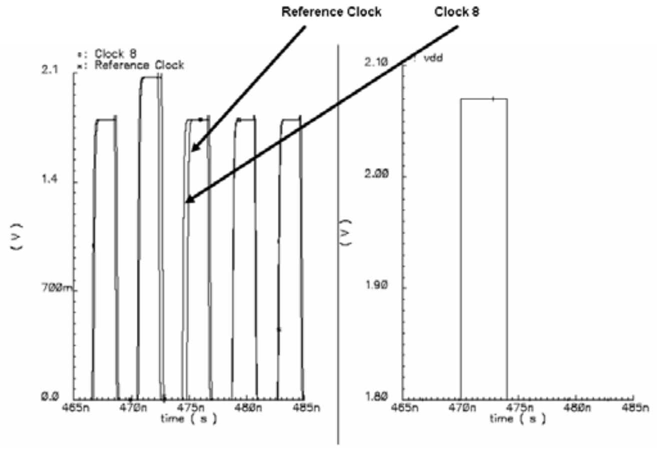


Figure 6-8 DLL Phase alignment under the influence of power grid glitch @ 250 MHz

Process, temperature, and power glitches all affect the performance of the DLL. In the following sections, their influence on jitter, static phase error, lock range, and lock time are discussed in detail.

6.5 Mismatch

Mismatch may affect the performance of the DLL. In the DLL, the transistor pairs that need to be critically matched include the transistor pairs in the current mirror of the combined PD and CP, as well the transistor pairs in each stage of VCDL. The critical transistor pairs are shown in Figure 6-9 and Figure 6-10.

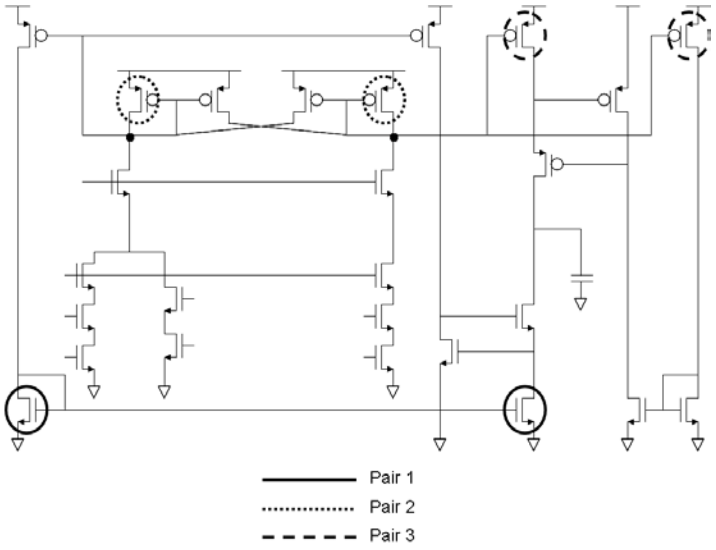


Figure 6-9 Critically matched transistor pairs in the combined PD and CP

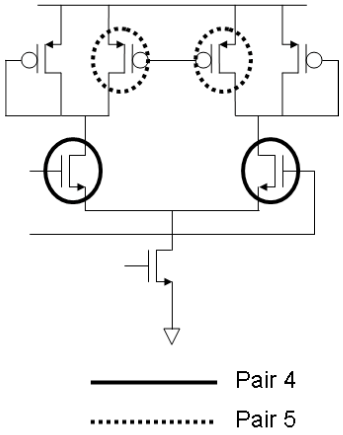


Figure 6-10 Critically matched transistor pairs in the VCDL stage

One common mismatch is the variation of the transistor length. The simulated transistor length mismatch effects are shown in Table 6-4 through Table 6-8.

Table 6-4 Effects of gate length mismatch of transistor pair 1

	Jitter (ps)	Static phase error (ps)	Lock time (clock cycle)	Lock range (MHz)
Matched	25	80	11	160-700
5% Mismatch	25	80	11	160-700
10% Mismatch	25	80	12	160-700
15% Mismatch	25	90	12	160-700
20% Mismatch	25	90	12	160-700

Table 6-5 Effects of gate length mismatch of transistor pair 2

	Jitter (ps)	Static phase error (ps)	Lock time (clock cycle)	Lock range (MHz)
Matched	25	80	11	160-700
5% Mismatch	25	80	11	160-700
10% Mismatch	25	90	11	160-700
15% Mismatch	25	90	11	160-700
20% Mismatch	25	90	11	160-700

Table 6-6 Effects of gate length mismatch of transistor pair 3

	Jitter (ps)	Static phase error (ps)	Lock time (clock cycle)	Lock range (MHz)
Matched	25	80	11	160-700
5% Mismatch	25	80	11	160-700
10% Mismatch	25	80	11	160-700
15% Mismatch	25	80	11	160-700
20% Mismatch	25	80	11	160-700

Table 6-7 Effects of gate length mismatch of transistor pair 4

	Jitter (ps)	Static phase error (ps)	Lock time (clock cycle)	Lock range (MHz)
Matched	25	80	11	160-700
5% Mismatch	30	80	11	160-700
10% Mismatch	40	90	11	160-700
15% Mismatch	40	90	11	160-700
20% Mismatch	50	90	11	160-700

Table 6-8 Effects of gate length mismatch of transistor pair 5

	Jitter (ps)	Static phase error (ps)	Lock time (clock cycle)	Lock range (MHz)
Matched	25	80	11	160-700
5% Mismatch	35	80	11	160-700
10% Mismatch	35	90	12	160-700
15% Mismatch	35	90	12	160-700
20% Mismatch	40	90	12	160-700

6.6 Jitter

Jitter of a DLL is an important performance specification. In a DLL, the output signal/signals are directly taken from certain delay stage of the VCDL. According to [40], the jitter of a DLL's output signal is primarily affected by the jitter of the input reference signal and the jitter contributed by the VCDL. The jitter associated with the input reference signal directly propagates through the VCDL and thereby contributes to the jitter of the output signal. Such jitter can be reduced by adjusting the loop bandwidth of the DLL. As the jitter of the input signal is determined by the previous driving circuit (usually a high Q crystal oscillator), the focus of our design is to reduce the jitter caused by the delay stages of the VCDL [40].

We have investigated the influence of device noise on the jitter performance of the DLL. As discussed in Chapter 5, for active MOSFET transistors, device noise mainly consists of thermal noise and flicker noise. To simulate device noise, we used the method proposed in [41]. Specifically, for each circuit component in the DLL, the effect of the device noise source associated with each transistor is replaced by an equivalent white noise source and an equivalent flicker noise source at the output nodes. For example, for each delay stage of the VCDL, the output referred noise sources are shown in Figure 6-11.

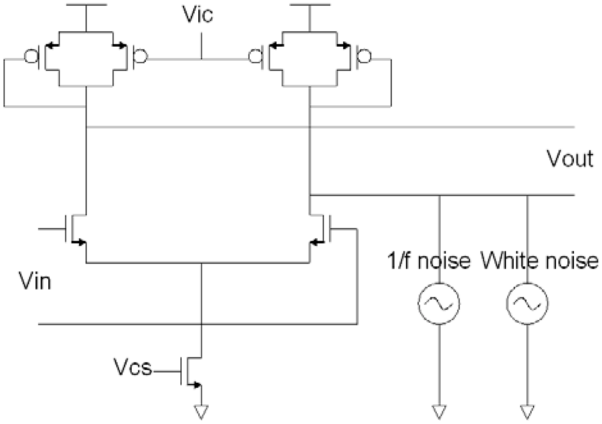


Figure 6-11 Equivalent noise voltages at the output nodes

Next, the equivalent noise voltages are converted to equivalent noise currents. After that, the equivalent noise currents are injected into the signal path and a transient analysis is performed with the equivalent noise currents. However, the simulator we are using (Spectre) can not directly calculate the jitter of the output clock signals. Therefore, we used a method proposed in [41]. Specifically, after running the Spectre transient analysis, we extract the transient analysis data from the Spectre output file and load it into MATLAB. Then, we identify the zero crossing points using data interpolation in MATLAB. After that, the cycle-to-cycle jitter is calculated as

$$\Delta T_{cc} = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(T_{n+1} - T_n)^2} \ , \tag{6-3}$$

where N is the number of clock cycles ($N = 100$ in this case). Table 6-9 shows the calculated cycle-to-cycle jitter of the DLL output signals.

Table 6-9 Cycle-to-cycle jitter of the DLL output signal @ 250 MHz (in ps)

Temperature &	Room	Room	Room	0°C	0°C	0°C	70°C	70°C	70°C
Vsupply	1.8V	2.07V	1.53V	1.8V	2.07V	1.53V	1.8V	2.07V	1.53V
TT	25	27	27	13	14	15	53	53	55
FF	33	33	34	15	16	16	55	56	57
SS	38	38	39	16	17	17	56	58	58
FS	37	38	39	16	16	17	57	57	57
SF	36	38	39	16	16	17	57	57	58

For MOS transistor, thermal noise (white noise) can be modeled as a noise current source connected between the drain and source terminals, with a spectral density of

$$\overline{I_n^2} = 4kT\gamma g_m \, . \tag{6-4}$$

Similarly, flicker noise (1/f noise) can also be modeled as a noise current source connected between the drain and source terminals, with a spectral density of

$$\overline{I_n^2} = \frac{K}{C_{ox}WL}\frac{1}{f}g_m^2 \, . \tag{6-5}$$

The dominant noise source can be determined by comparing (6-4) and (6-5) at the frequencies which interest us. A rough calculation reveals that for transistors from the TSMC 0.18μm process, thermal noise dominates when the operation frequency is higher than about 10MHz. As can be seen from Table 6-4, temperature has a large effect on the simulated jitter performance. The supply voltage does not play a big role due to the dynamic biasing mechanism of the VCDL.

6.7 Static phase error

Static phase error refers to the phase difference between the output signal of the last stage of the VCDL and the input reference signal. In the ideal case, after a DLL is locked, the phases of these two signals should be perfectly matched. However, due to the limited phase resolution of the PD and CP, some static phase error may exist. Table 6-10 shows the averaging static phase errors of the DLL.

Table 6-10 Static phase errors @ 250 MHz (in ps)

Temperature & Vsupply	**Room & 1.8V**	**0°C & 2.07V**	**0°C & 1.53V**	**70°C & 2.07V**	**70°C & 1.53V**
TT	80	70	80	80	80
FF	60	60	60	70	70
SS	110	100	100	110	110
FS	80	70	80	90	90
SF	70	70	70	80	90

6.8 Lock range

Lock range refers to the frequency range in which a DLL is able to achieve lock. Although it is often desired to have lock range as wide as possible, the lock range of a particular DLL is usually designed for a specific application. The lock range of our design is shown in Table 6-11.

Table 6-11 Lock range (in MHz)

Temperature & Vsupply	**Room 1.8V**	**Room 2.07V**	**Room 1.53V**	**0°C 1.8V**	**0°C 2.07V**	**0°C 1.53V**	**70°C 1.8V**	**70°C 2.07V**	**70°C 1.53V**
TT	160-700	170-750	150-640	160-700	170-750	150-640	160-700	170-750	150-630
FF	170-740	180-760	160-650	170-740	180-760	160-650	170-740	180-760	160-650
SS	150-670	160-730	150-620	160-670	170-730	150-620	160-670	170-720	140-620
FS	160-690	170-720	150-640	160-700	170-720	150-640	160-700	170-720	150-640
SF	170-720	170-730	150-640	160-720	180-730	150-640	160-730	170-730	150-630

As can be seen from Table 6-11, the lowest frequency that the DLL can go is 140MHz and the highest frequency that the DLL can reach is 760MHz. Fast NMOS and fast PMOS transistors provide the highest limit. This is because they offer the shortest delay of the VCDL. On the other hand, slow NMOS and slow PMOS set the lowest limit since slow devices cause the longest delay of the VCDL. Another factor that significantly affects the lock range is the power supply. It is clear from Table 6-6 that higher power supply results in higher operation frequency. This is because higher power supply voltage leads to larger charging/discharging currents of the VCDL and thus makes the signal switching faster in the VCDL. It can also be noted that the lower limit of the lock range does not have a strong relationship with the supply voltage and the temperature. This is because that when the operation frequency decreases, the voltage swing of the output signal also decreases. As very small voltage swing significantly affects the validity of the output signal, the major limiting factor of the lower frequency range is the voltage swing.

6.9 Lock time

Lock time refers to the time interval that a DLL takes to achieve lock. Table 6-12 lists the lock times under different temperature, supply and process variations. The longest lock time across all temperatures, supply voltages, and process variations is 13 clock cycles.

Table 6-12 Lock time @ lowest operating frequency (in cycles)

Temperature & Vsupply	Room & 1.8V	0°C & 2.07V	0°C & 1.53V	70°C & 2.07V	70°C & 1.53V
TT	11	12	12	13	12
FF	12	13	13	12	12
SS	12	13	13	13	13
FS	11	12	13	13	12
SF	12	12	13	13	12

CHAPTER 7

DLL BIST

In many applications it is very desirable to be able to test whether a DLL has been functioning correctly. A DLL arrives at a stable or locked state after the phase of the output signal CK_{out} is aligned with the phase of the input signal CK_{ref}. Thus, a fault-free DLL would have a zero phase error between CK_{ref} and CK_{out} in the stable state, and any substantial phase error in the stable state would indicate some faults in the DLL.

To address this need, a BIST circuit is proposed to test the functionality of a DLL. A DLL and the BIST circuit are shown in Figure 7-1.

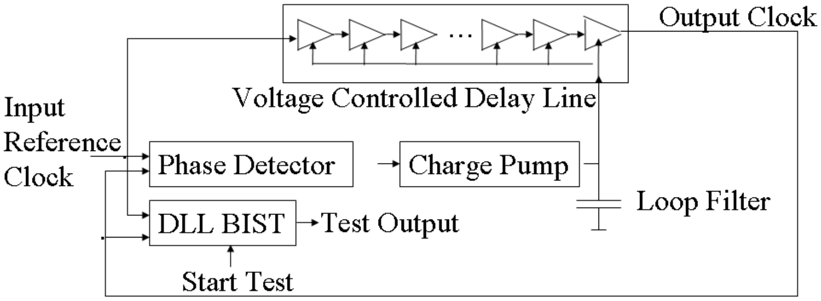
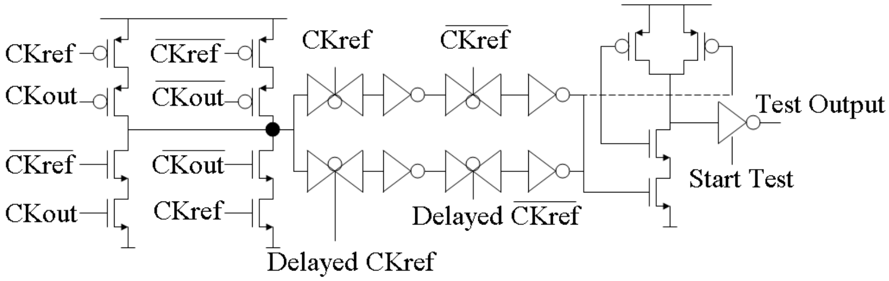


Figure 7-1 Block diagram of a DLL and the BIST for the DLL

The most straightforward and reliable way to check whether two signals are in phase is to use an XNOR gate that generates an output of logic 1 when both inputs are of the same phase, and an output of logic 0 otherwise. Therefore, the BIST circuit we proposed is based on an XNOR gate which accepts both CK_{ref} and CK_{out} as input signals. The output of the XNOR gate is sampled so that the final output is a pure digital value which increases the test reliability. The XNOR gate does introduce a small loading to CK_{ref} and CK_{out}. However, as can be seen from Figure 2, the loading effect to CK_{ref} can be safely ignored since CK_{ref} comes from another on-chip circuit with high driving capability. Also, the loading effect to CK_{out} is not very significant since the input capacitance of the XNOR gate can be made very small. This test method does not require any external analog test signals, and it does not alter any internal circuits of the DLL. The detailed structure of the DLL BIST circuit is shown in Figure 7-2.



CKref
$\overline{\text{CKref}}$
CKout
$\overline{\text{CKout}}$
$\overline{\text{CKref}}$
$\overline{\text{CKout}}$
CKout
CKref
CKref
$\overline{\text{CKref}}$
Test Output
Start Test
Delayed $\overline{\text{CKref}}$
Delayed CKref

Figure 7-2 The proposed DLL BIST circuit

The output of the BIST circuit is enabled after the DLL has achieved lock, which is signified by a *Start Test* signal. The *Start Test* signal is set to be twice the maximum simulated DLL lock time, or even longer if necessary. The test output, which is a stable digital signal, can be scanned out using an IEEE 1149.1 boundary scan path or any other available scan path.

We have tested the proposed DLL BIST circuit for the fault-free case as well as four possible fault cases: CK_{ref} and CK_{out} are in phase; CK_{ref} leads CK_{out} in phase; CK_{ref} lags CK_{out} in phase; CK_{out} is stuck at logic 1; and CK_{out} is stuck at logic 0. The results for all these cases are shown in the following figures.

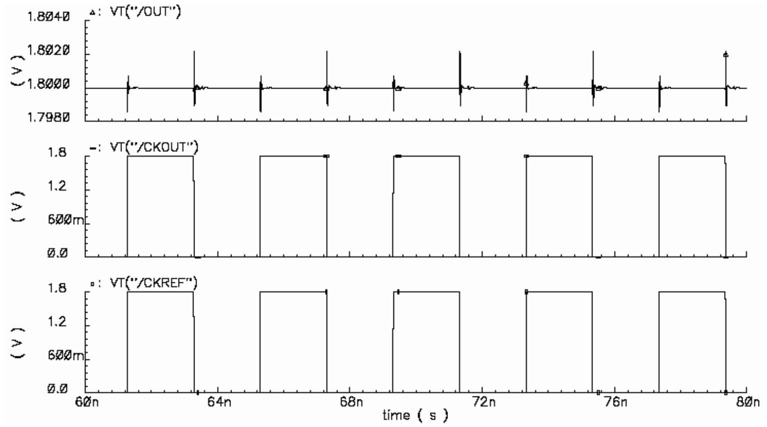


Figure 7-3 Test output of logic 1 when CKref and CKout are in phase

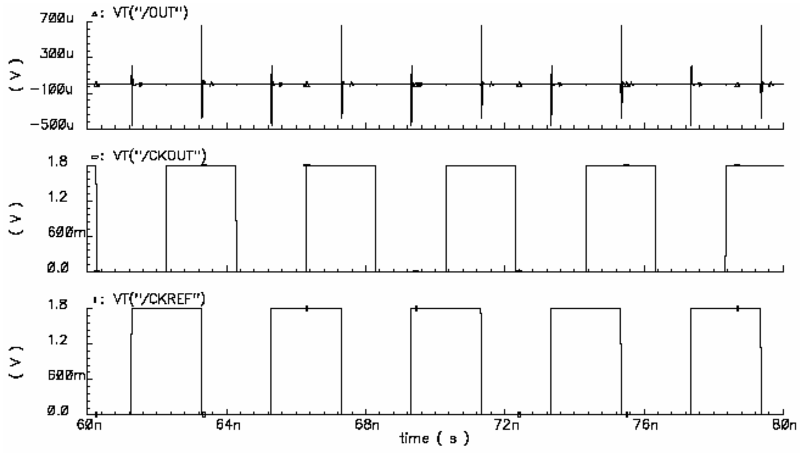


Figure 7-4 Test output of logic 0 when CKout lags CKref in phase

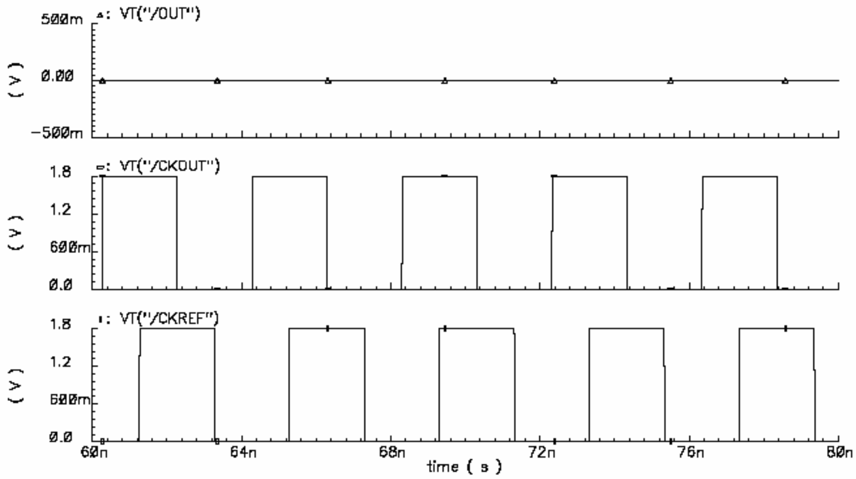


Figure 7-5 Test output of logic 0 when CKout leads CKref in phase

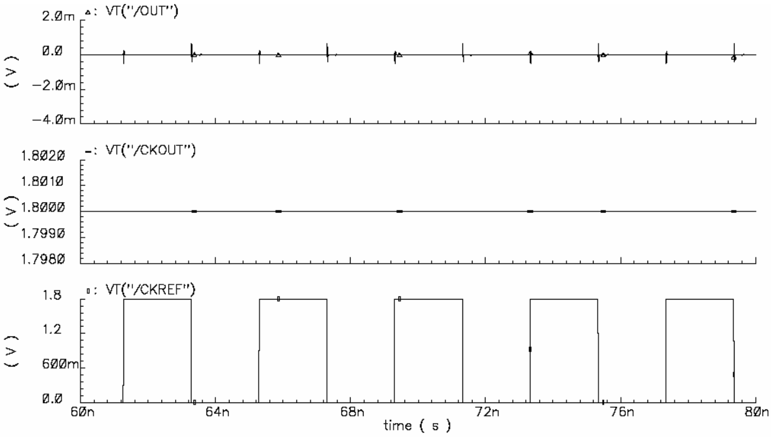


Figure 7-6 Test output of logic 0 when CKout is stuck at logic 1

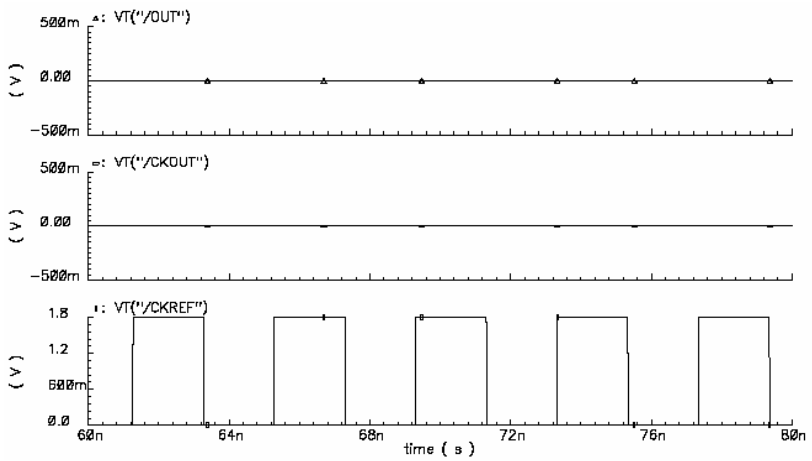


Figure 7-7 Test output of logic 0 when CKout is stuck at logic 0

We have also tested the BIST circuit under process corners and supply voltage or temperature variations. The results are summarized in the following table.

Table 7-1 Outputs of the DLL BIST under supply/temperature variations

Temperature & Vsupply	**0°C & 1.8V**	**0°C & 2.07V**	**0°C & 1.53V**	**70°C & 2.07V**	**70°C & 1.53V**
CKref & CKout in phase	1	1	1	1	1
CKref leads CKout	0	0	0	0	0
CKref lags CKout	0	0	0	0	0
CKout stuck at 0	0	0	0	0	0
CKout stuck at 1	0	0	0	0	0

To evaluate the effectiveness of the proposed BIST circuit, we have built a structural fault model of the above DLL which includes a complete set of catastrophic (hard) faults. The structural fault model we use includes short-fault and open-fault in PD, CP, loop filter, and VCDL. Specifically, the short faults include (1) gate-to-source shorts (GSS), (2) gate-to-drain shorts (GDS), (3) drain-to-source shorts (DSS), and (4) capacitance-shorts (CS). The open faults include (1) gate-opens (GO), (2) drain-opens (DO), and (3) source-opens (SO). For these faults, we used the most common fault models. A short fault is modeled by a 1Ω resistance between the shorted terminals. An open fault is modeled as a 10MΩ resistance between the opened terminals.

The total number of structural faults in the DLL is 205 which include the faults in the PD, CP, loop filter and VCDL (only one stage considered). To demonstrate the effectiveness of the proposed BIST technique, simulations have been conducted to evaluate the fault coverage using all 205 faults. Through simulations, it is confirmed that the output signals due to catastrophic faults will be one of the four possible cases, i.e., CK_{ref} leads CK_{out} in phase; CK_{ref} lags CK_{out} in phase; CK_{out} is stuck at logic 1; and CK_{out} is stuck at logic 0. For the first two cases, the DLL BIST circuit reports a fault as long as the phase difference between CK_{ref} and CK_{out} is beyond the resolution of the BIST circuit. For the last two cases, the DLL BIST would positively report a fault.

The resolution of the DLL BIST circuit (i.e., the minimum phase difference between CK_{ref} and CK_{out} for the DLL BIST to raise a fault signal) is shown in Table 7-2 across all process corners, temperature, and power supply. The fault coverage test results are shown in Table 7-3 and 7-4.

Table 7-2 Resolution of the DLL BIST (in ps)

Temperature & Vsupply	**0°C & 1.8V**	**0°C & 2.07V**	**0°C & 1.53V**	**70°C & 2.07V**	**70°C & 1.53V**
TT	40	45	45	45	45
FF	30	30	30	35	35
SS	60	60	60	60	60
SF	60	60	60	60	60
FS	60	60	60	60	60

Table 7-3 Structural fault coverage results

Fault Type	Fault Coverage (%)
Gate Open	89%
Source Open	100%
Drain Open	100%
Gate-Drain Short	100%
Gate-Source Short	100%
Drain-Source Short	97%
Capacitor Short	100%
Overall	97%

Table 7-4 Structural fault coverage results sorted by circuit blocks

Fault Type	Fault Coverage (%)
PD & CP	97%
Loop Filter	100%
VCDL	97%

The undetected gate open faults are in the PD&CP. They are shown in Figure 7-8. The reason for the undetected gate open faults is that the undetected four transistors are able to provide the necessary currents even if their gate terminals are set open, although at a slower rate. The undetected drain-source short faults are in the VCDL. They are shorts of drain-source nodes in diode connected transistors. Therefore, they do not affect circuit functionality at all.

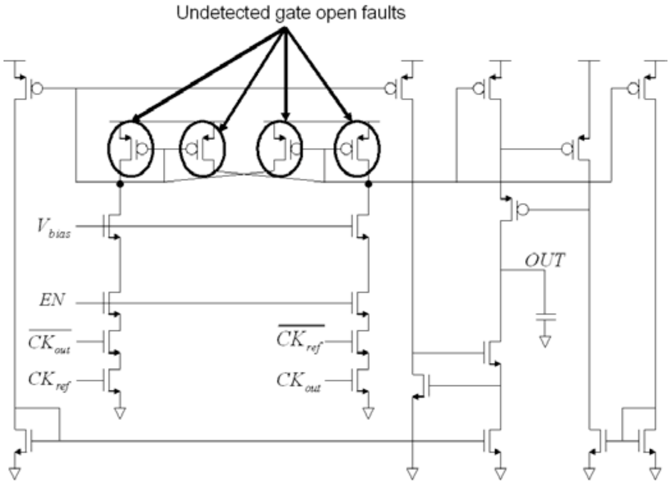


Figure 7-8 Undetected gate open faults

CHAPTER 8

CONCLUSION

This thesis presents our work in the design of a DLL for the generation of multiple clock phases/delays. A novel DLL design is proposed with several new techniques to achieve wide lock range, short locking time, and reduced jitter. The DLL can be used for a variety of applications which require precise time intervals or phase shifts.

The phase detector (PD), charge pump (CP), and voltage-controlled delay line (VCDL) are the three most important blocks in a DLL. The PD serves the purpose of detecting the phase difference between the input reference signal and the output signal from the last stage of the VCDL. The phase difference information from the PD is provided to the CP so that the corresponding charging or discharging current can be activated to control the voltage of the loop filter. The phase resolution of the PD and the switching speed of CP directly affect the performance of the DLL. In our research, we have proposed a novel structure which combines the functionality of both the PD and CP. By using this structure, a fast switching speed and good matching between charging and discharging currents can be achieved. Moreover, the combined PD and CP also lead to reduced chip area and better jitter performance. A novel phase detection algorithm is developed and implemented in the combined PD and CP structure. This algorithm also involves a start-control circuit to avoid locking failure or false lock to harmonics. With the help of this algorithm, the proposed DLL is able to achieve lock as long as the minimum VCDL delay $D_{VCDL}(\min)$ is less than one reference clock cycle T_{REF}, which is the *largest possible* lock range that can be achieved by a DLL. The VCDL, whose delay is controlled by the voltage of the loop filter, uses fully differential signaling to minimize jitter. The delay stage of the VCDL is built with a differential topology using symmetrical loads and replica-feedback biasing, which provides a low sensitivity to supply and substrate noise and a wide tuning range. In addition, a shift-averaging technique is used to improve the matching between delay stages and thus to equalize the delay of each individual stage.

Several techniques are also used in the layout of the prototype chip to improve device matching, minimize power and substrate noise, and reduce crosstalk. Extensive simulations have been conducted to evaluate the performance of the DLL under a variety of process corners, temperatures, and power supply glitches. The DLL shows satisfactory performance under these conditions.

REFERENCES

[1] J. G. Maneatis and M. A. Horowitz, "Precise Delay Generation Using Coupled Oscillators," IEEE J. Solid-State Circuits, vol. 28, pp. 1273-1282, Dec. 1993.

[2] Cypress Semiconductor, CY7C1310V18 Sync SRAM Datasheet, Aug. 2002.

[3] Semiconductor Industry Association, International Technology Roadmap for Semiconductors, 2004.

[4] E. R. Hnatek, *Digital Integrated Circuit Testing from a Quality Perspective*, Van Reinhold, New York, NY, 2000.

[5] G. Chien and P.R. Gray, "A 900-MHz local oscillator using a DLL-based frequency multiplier technique for PCS applications," IEEE J. Solid-State Circuits, vol. 35, pp. 1996-1999, Dec. 2000.

[6] S.-I. Liu, J.-H. Lee, and H.-W. Tsao, "Low-Power Clock-Deskew Buffer for High-Speed Digital Circuits," IEEE J. Solid-State Circuits, vol. 34, pp. 554-558, Apr. 1999.

[7] M. G. Johnson and E.L. Hudson, "A Variable Delay Line PLL for CPU-Coprocessor Synchronization," IEEE J. Solid-State Circuits, vol. 23, pp. 1218-1223, Oct. 1988.

[8] M. Mansuri and C.-K. K. Yang, "Jitter Optimization Based on Phase-Locked Loop Design Parameters," IEEE J. Solid-State Circuits, vol. 37, pp. 1375-1382, Nov. 2002.

[9] B.W. Garlepp *et al.*, "A portable digital DLL for high-speed CMOS interface circuits," IEEE J. Solid-State Circuits, vol. 34, no. 5, pp. 632–644, May 1999.

[10] K. Minami *et al.*, "A1 GHz portable digital delay-locked loop with infinite phase capture ranges," in IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers, Feb. 2000, pp. 350–351.

[11] J.-B. Lee *et al.*, "Digitally-controlled DLL and I/O circuits for 500 Mb/s/pin□16 DDR SDRAM," in IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers, vol. 431, Feb. 2001, pp. 68–69.

[12] T. Matano *et al.*, “A 1-Gb/s/pin 512-Mb DDRII SDRAM using a digital DLL and a slew-rate-controlled output buffer,” IEEE J. Solid-State Circuits, vol. 38, no. 5, pp. 762–768, May 2003.

[13] J.-T. Kwak *et al.*, “Low cost high performance register-controlled digital DLL for 1 Gbps□32 DDR SDRAM,” in Symp. VLSI Circuits Dig. Tech. Papers, 2003, pp. 283–284.

[14] K-H. Cheng, *et al.*, “A 2.2 GHz programmable DLL-based frequency multiplier for SOC applications,” in Proc. Asia-Pacific Conf. on Advanced System İntegrated Circuits, 2004, pp. 72-75.

[15] M.-J. E. Lee, *et al.*, “Jitter Transfer Characteristics of Delay-Locked Loops – Theories and Design Techniques,” IEEE J. Solid-State Circuits, vol. 38, pp. 614-621, Apr. 2003.

[16] J. Maneatis, “Low-Jitter Process-Independent DLL and PLL Based on Self-Biased Techniques,” IEEE J. Solid-State Circuits, vol. 31, pp. 1723-1732, Nov. 1996.

[17] A. Chandrakasan, W. J. Bowhill, and F. Fox, *Design of High-Performance Microprocessor Circuit*, New York: IEEE Press, 2001.

[18] M.-J. E. Lee, W. J. Dally, and P. Chiang, “Low-Power, Area Efficient, High Speed I/O Circuit Techniques,” IEEE J. Solid-State Circuits, vol. 35, pp. 1591-1599, Nov. 2000.

[19] B. Razavi, *Design of Analog CMOS Integrated Circuits*, McGraw-Hill, New York, NY, 2001.

[20] Y. Tsividis, *Operation and Modeling of the MOS Transistor*, Second Ed., McGraw-Hill, Boston, 1999.

[21] Y. Moon, D.-K. Jeong, and G. Ahn, “A 0.6-2.5-GBaud CMOS Tracked Oversampling Transceiver with Dead-Zone Phase Detection for Robust Clock/Data Recovery,” IEEE J. Solid-State Circuits, vol. 36, pp. 1974-1983, Dec. 2001.

[22] B. W. Garlepp et al., “A Portable Digital DLL for High-Speed CMOS Interface Circuits,” IEEE J. Solid-State Circuits, vol. 34, pp. 632-644, May 1999.

[23] A. Efendovich, et al., “Multifrequency Zero-Jitter Delay-Locked Loop,” IEEE J. Solid-State Circuits, vol. 29, pp. 67-70, Jan. 1994.

[24] B. Razavi, *Monolithic Phase-Locked Loops and Clock Recovery Circuits: Theory and Design*, IEEE Press, New York, NY, 1996.

[25] M. Mansuri, D. Liu and C.-K. K. Yang, “Fast Frequency Acquisition Phase-Frequency Detectors for GSamples/s Phase-Locked Loops,” IEEE J. Solid-State Circuits, vol. 37, pp. 1331-1334, Oct. 2002.

[26] H. Notani, H. Kondoh, and Y. Matsuda, “A 622-MHz CMOS Phase-Locked Loop with Precharge-Type Phase Frequency Detector,” VLSI Symp. Dig. Tech. Papers, June 1994, pp. 129-130.

[27] W. Rhee, “Design of High Performance CMOS Charge Pumps in Phase-Locked Loops,” Proc. IEEE Intl. Symp. on Circuits and Systems, vol. 1, 1999, pp. 545-548.

[28] *High Speed Design Techniques*, Analog Devices Technical Reference Books, Analog Devices, Inc., 1996.

[29] I. A. Young, J. K. Greason, and K. L. Wong, “A PLL Clock Generator with 5 to 110MHz of Lock Range for Microprocessors,” IEEE J. Solid-State Circuits, vol. 27, pp. 1599-1607, Nov. 1992.

[30] S. Liu, J. Lee, and H. Tsao, "Low-Power Clock-Deskew Buffer for High-Speed Digital Circuits," IEEE J. Solid-State Circuits, vol.34, pp. 554-558, April 1999.

[31] H.-H. Chang, C.-H. Sun, and S.-I. Liu, “Low Jitter Butterworth Delay-Locked Loops,” Int. Symp. on VLSI Circuits Dig. Tech. Papers, June 2003, pp. 177-180.

[32] S. Sidiropoulos, M. A. Horowitz, “A Semidigital Dual Delay-Locked Loop,” IEEE J. Solid-State Circuits, vol. 32, pp. 1683-1692, Nov. 1997.

[33] Y.-J. Jung, S.-W. Lee, D. Shim, W. Kim, C. Kim, S.-I. Cho, “A Dual-Loop Delay-Locked Loop Using Multiple Voltage-Controlled Delay Lines,” IEEE J. Solid-State Circuits, vol. 36, pp. 784-791, May 2001.

[34] Y. Moon, J. Choi, K. Lee, D.-K. Jeong, M.-K. Kim, “All All-Analog Multiphase Delay-Locked Loop using a Replica Delay Line for Wide-Range Operation and Low-Jitter Performance,” IEEE J. Solid-State Circuits, vol. 35, pp. 377-384, Mar. 2000.

[35] H.-H. Chang, J.-W. Lin, and S.-I. Liu, “A Fast Locking and Low Jitter Delay-Locked Loop Using DHDL,” IEEE J. Solid-State Circuits, vol. 38, pp. 343-346, Feb. 2003.

[36] *PC100 SDRAM Memory Driver Solutions*, Application Note, Fairchild Semiconductor, 2003.

[37] R. Farjad-Rad et al., “A Low-Power Multiplying DLL for Low-Jitter Multigigahertz Clock Generation in Highly Integrated Digital Chips,” IEEE J. Solid-State Circuits, vol. 37, pp. 1804-1812, Dec. 2002.

[38] H.-H. Chang, C.-H. Sun, and S.-I. Liu, “A Low-Jitter and Precise Multiphase Delay-Locked Loop Using Shifted Averaging VCDL,” Dig. Tech. Papers, IEEE ISSCC, Feb. 2003.

[39] A. Hastings, *The Art of Analog Layout*, Princeton Hall, New York, NY, 2000.

[40] G. Chien and P. R. Gray, “A 900-MHz local oscillator using a DLL-based frequency multiplier technique for PCS applications,” IEEE J. Solid-State Circuits, vol. 35, pp. 1996-1999, Dec. 2000.

[41] C.W. Zhang, X.Y. Wang and L. Forbes, “Simulation technique for noise and timing jitter in electronic oscillators,” IEE Proc. Circuits Devices Syst., vol. 151, no. 2, pp. 184-189, April 2004.

VITA

CHENG JIA

Cheng Jia’s main research interests include circuit and system design, signal timing analysis, noise, jitter, and stochastic process. When he is not working on his research, he enjoys basketball, swimming, and travel.