High-Speed Clocking Deskewing Architecture

by

David Li

A thesis
presented to the University of Waterloo
in fulfilment of the
thesis requirement for the degree of
Master of Applied Science
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2007

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

David Li

I understand that my thesis may be made electronically available to the public.

David Li

Abstract

As the CMOS technology continues to scale into the deep sub-micron regime, the demand for higher frequencies and higher levels of integration poses a significant challenge for the clock generation and distribution design of microprocessors. Hence, skew optimization schemes are necessary to limit clock inaccuracies to a small fraction of the clock period. In this thesis, a crude deskew buffer (CDB) is designed to facilitate an adaptive deskewing scheme that reduces the clock skew in an ASIC clock network under manufacturing process, supply voltage, and temperature (PVT) variations. The crude deskew buffer adopts a DLL structure and functions on a 1GHz nominal clock frequency with an operating frequency range of 800MHz to 1.2GHz. An approximate 91.6ps phase resolution is achieved for all simulation conditions including various process corners and temperature variation. When the crude deskew buffer is applied to seven ASIC clock networks with each under various PVT variations, a maximum of 67.1% reduction in absolute maximum clock skew has been achieved. Furthermore, the maximum phase difference between all the clock signals in the seven networks have been reduced from 957.1ps to 311.9ps, a reduction of 67.4%. Overall, the CDB serves two important purposes in the proposed deskewing methodology: reducing the absolute maximum clock skew and synchronizes all the clock signals to a certain limit for the fine deskewing scheme. By generating various clock phases, the CDB can also be potentially useful in high speed debugging and testing where the clock duty cycle can be adjusted accordingly. Various positive and negative duty cycle values can be generated based on the phase resolution and the number of clock phases being "hot swapped". For a 500ps duty cycle, the following values can be achieved for both the positive and negative duty cycle: 224ps, 316ps, 408ps, 592ps, 684ps, and 776ps.

Acknowledgements

First of, I would like to thank Dr.Manoj Sachdev for his great support, guidance, and mentoring as my research supervisor. His advice and support are greatly appreciated. I would also like to thank Professor Nairn and Professor Opal for being my thesis readers. Thank you for your positive and valuable comments and suggestions.

I would also like to specially thank Muhammad Nummer for being a second mentor and good friend in this research project, Phil Regier for solving all the computer problems, Dave Rennie for constantly helping me solving various Cadence issues, and everyone else in the CMOS Design and Reliability Group at the University of Waterloo for their support.

Also I would like to say a special thanks to Chen Hu, Phillip Woo, Jannie Mak, and Shawn Zhang for being my best friends at the University of Waterloo. Without their help and support, this thesis would not have been possible.

Finally, I would like to thank my family for their support and encouragement throughout my academic careers.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

As described by Moore's Law, the downscaling of minimum dimensions enables the integration of an increasing number of transistors on a single chip. In fact, Moore also predicted that the MPU (microprocessor unit) performance doubles 1.5 to 2 years. The high performance of today's microprocessors has required the generation and distribution of very high-quality clock signals. However, increased frequencies and higher levels of integration for microprocessors have posed significant challenges for clock generation and distribution. Large die areas along with aggressive technology scaling have caused the distribution path to be long and widely dispersed across the die. Such distribution produces large latency in the clock path which is greatly impacted by variations in loading on clock lines, temperature shifts, voltage swings, cross-talk, and across die manufacturing process variations [1]. All of these factors have combined to create large clock skew and jitter that effectively shorten the clock cycle. Hence, the ability to maintain stable and reliable clock signals has become a popular research topic for many VLSI researchers.

Many techniques have been used in the clock network design for high performance microprocessors. Clock networks typically include a network that is used to distribute a global reference to various parts of the chip and a final stage that is responsible for local distribution of the clock to various loads. The most common type of clock distribution scheme is the H-tree network where the clock is first routed to a central point on the chip,

and then this reference clock is distributed to the various leaf nodes with matched interconnect and buffers. However, managing a perfectly balanced clock design in a complex microprocessor is difficult to achieve due to many floorplans and the clock loading constraints [2]. In addition, a balanced tree structure does not take into account the effects of all the within-die process variations that affect the clocking elements. Hence, an active deskewing scheme should be adopted in conjunction with a combined balanced clock tree and clock grid.

The aim of this research work is to explore the circuit techniques and architectures that are able to minimize the clock skew at various parts of the Application Specific Integrated Circuits (ASICs) caused by a variety of factors. In this thesis, we present the design and application of a digitally-controlled crude deskew buffer based on delay-locked loop (DLL) architecture in achieving the active deskewing scheme for certain real application clock networks. In addition, we will also be illustrating the potential application of the deskew buffer in the area of high-speed testing and debugging. These circuits have been designed and simulated using 0.13μm CMOS technology.

1.2 Thesis Organization

This thesis is organized in the following manner. Chapter 2 reviews some of the more widely used clock generation and distribution network schemes in current high speed ASICs/VLSI systems. Chapter 3 provides the background information on clock skew, the effects of different within-die variations that contribute to it, the need for adaptive deskewing schemes, and the proposed deskewing methodology. Chapter 4 offers detailed analysis on the design of the crude deskew buffer. Chapter 5 introduces the concept and the simulation results of phase swapping between clock phases using the crude deskew buffer to manipulate the clock duty cycles. Chapter 6 presents the simulation results on the application of the crude deskew buffer in reducing the clock skew in a sample ASIC clock network under the effects of process, temperature, and supply voltage variations. Finally, concluding remarks will be given in Chapter 7.

Chapter 2

Clock Generation and Distribution in High-Speed ASICs

The demand for higher frequencies and higher levels of integration poses a significant challenge for the clock generation and distribution design in high performance microprocessors. Phase-Locked Loop (PLL) and Delay-Locked Loop (DLL) are common clock generation architectures that achieve the desired clock frequencies as well as eliminating the timing variation between the internal clock and the reference clock signals. As for clock distribution, the H-tree network, clock grid, and clock spine are some of the popular approaches in high performance microprocessor designs.

2.1 Clock Generation

2.1.1 Phase Locked Loop

In today's VLSI systems, high performance digital circuits require clock frequencies in the gigahertz range. A crystal oscillator is an electronic circuit that uses the resonance of a vibrating crystal of piezoelectric material to create an electrical signal with a very precise frequency [3]. However, crystals only generates periodic clock signal in the megahertz range. In order to achieve the frequencies required in high performance digital circuits, a phase-locked loop takes an external crystal frequency and multiplies by a factor of N.

A PLL is a complex and non-linear feedback circuit, as illustrated in **Figure 2-1** [1]. It consists of the following components: phase detector (PD), charge pump, loop filter, voltage-controlled oscillator (VCO), and a divider. The reference clock signal (*REF_CLK*) shown in the figure is usually generated from an off-chip crystal oscillator while the local clock signal (*LCLK*) comes from the divided version of the system clock. The phase detector then compares these two clock signals to produce an *Up* and *Down* signal depending on their arrival time. For example, the *Up* signal is generated when *LCLK* lags the *REF_CLK* signal, and vice versa for the *Down* signal. The outputs of the phase detector are fed into a charge pump circuit that translates the digital encoded control information into an analog voltage [1]. The analog control voltage will increase to speed up the *LCLK* signal if *Up* is generated. On the other hand, the *Down* signal slows down the VCO and eliminates the leading phase of the *LCLK* signal. The main purpose of the loop filter shown in **Figure 2-1** is to remove the high-frequency components from the VCO control voltage and smooths out its response, which reduces the amount of jitter present in the PLL system [1]. When the PLL is under the locking condition, the system clock signal (*SYS_CLK*) is N time the reference clock frequency.

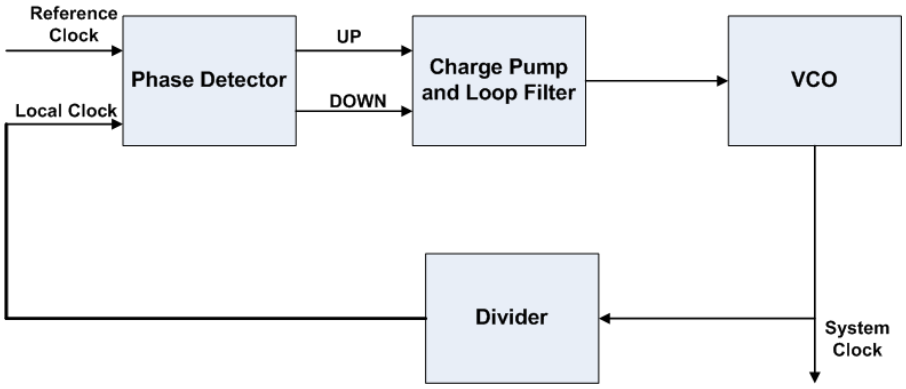

Figure 2.1: Block Diagram of PLL

An example of clock generation using PLL architecture is demonstrated in the design of a multi-gigahertz clocking scheme for the Pentium-4 microprocessor [4]. In this particular design, two separate PLLs are used to generate the core and I/O clocks, as shown in **Figure 2-2**. The two PLLs provide a total of six different frequencies distributed to different logic

blocks in the die. Based on a 100MHz system clock and a divide ratio of 20, the core PLL synthesizes a 2GHz clock and injects it into the core clock network. Local clock drivers (LCD) then generate 1, 2, and 4GHz core clocks as well as 100 and 200MHz I/O clocks. At the same time, the I/O PLL generates and synthesizes a 400MHz clock from a 100MHz system clock.

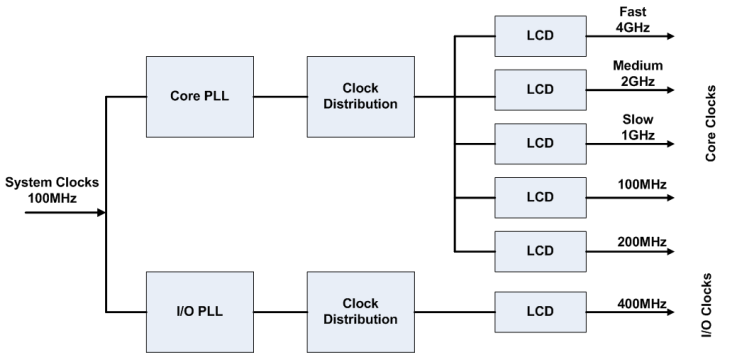

Figure 2.2: Clock Generation in Pentium-4 Microprocessors

A PLL is an analog circuit that is potentially very sensitive to noise and interference mainly due to the loop filter and the VCO [1]. A major source of interference is the noise coupling through the supply rails and the substrate. In general, PLL system is a complex and sensitive component that requires a lot of expertise and experience in order to achieve optimal performance. As such, a variation of the PLL structure, Delay-Locked Loop (DLL), is sometimes used for clock generation in recent high performance clocking microprocessors.

2.1.2 Delayed Locked Loop

The block diagram of a DLL is shown in **Figure 2-3**. It is clear that the architecture of a DLL is similar to that of a PLL except the VCO is replaced with a voltage-controlled delay line (VCDL). In most DLL designs, the VCDL consists of a number of adjustable delay elements such that the output clock signal is delayed by exactly one clock period with respect to the reference clock. The locking process of a DLL is almost identical to the PLL in which a phase detector along with a charge pump adjusts the analog control voltage that

is being fed into the VCDL to manipulate the corresponding delay. After many cycles, the phase error between the two clock signals is corrected such that the delayed clock signal ($DCLK$) is exactly one period behind the reference clock signal (REF_CLK).

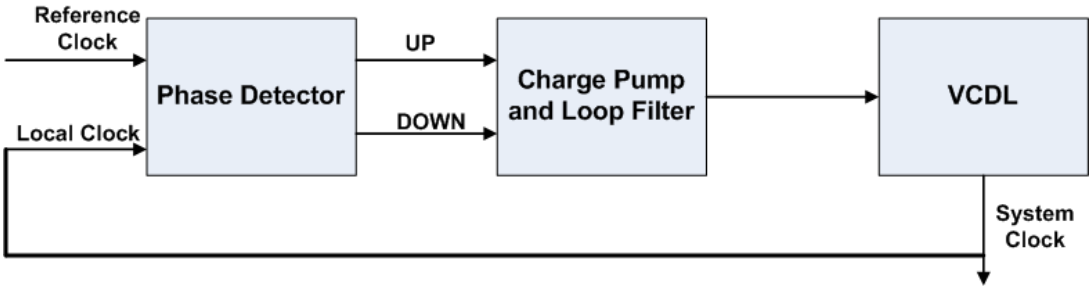

Figure 2.3: Block Diagram of DLL

Research done in [5] has illustrated a clock generation circuit based on a DLL array system. This is shown in **Figure 2-4**. The DLL system takes *CLK_Out1* as the system clock and generates 10 different phases of clock signals called *clk_in* such that each phase is separated by the same margin of Δ. The Segmented Delay Line (SDL) shown in the figure has similar variable delay attributes as the VCDL. Each SDL takes the common external clock and generates the output clock signal labeled *clk_out*. The phase detector detects the phase difference between each pair of *clk_in* and *clk_out* signal. The corresponding charge pump and the low pass filter (LPF) adjust the analog control voltage accordingly to correct the phase error. When each pair of *clk_in* and *clk_out* signal is under the locking condition, each *clk_out* signal will vary with an equal step size of Δ.

Overall the feedback loop of the DLL system compensates for both static and dynamic variations such that the phase error between clock signals can be significantly reduced. This attractive feature along with low jitter and more stability have made DLL a favorable configuration for clock generation in high performance microprocessors.

2.2 Clock Distribution

Due to increasing in chip size, complex functionalities, and higher clock frequencies, interconnects are having a significant impact on microprocessor performance. In the case

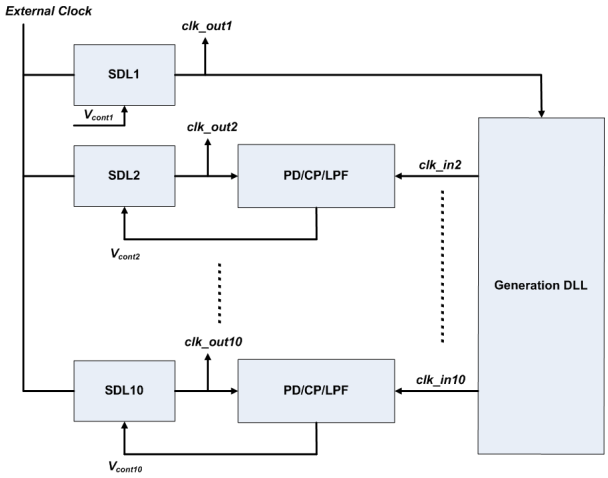

Figure 2.4: Clock Generation using DLL

of on-chip clock distribution where clock signals must be distributed simultaneously to all regions of the chip with accurately known delays, unexpected variations in clock skew will reduce the effective cycle time and can potentially cause functional errors. Hence, accurate modeling and delay minimization through design optimization and technology improvements are required. This section examines some of the common approaches used in clock distribution with the goal of minimizing clock skew.

2.2.1 H-Tree Network

The main idea behind distributing the clock signal to various parts of the chip is to use balanced paths such that the relative phase between two clocking points is minimized, and the H-tree network is the common approach. In the sample 4x4 H-tree illustrated in **Figure 2-5**, the clock is first routed to a central point on the chip through either a PLL or DLL structure. This reference clock signal will then propagate to various leaf nodes through equivalent balanced paths such that both interconnect and the number of buffers used are matched. Under the ideal conditions, the clock skew throughout the entire chip should be zero since each path is perfectly balanced. However, process and environmental variations as well as modeling errors can still cause significant skew to occur even between nearby clocking elements. Furthermore, wiring and tuning tree topologies to drive highly

non-uniform loads with low skew can be difficult [6].

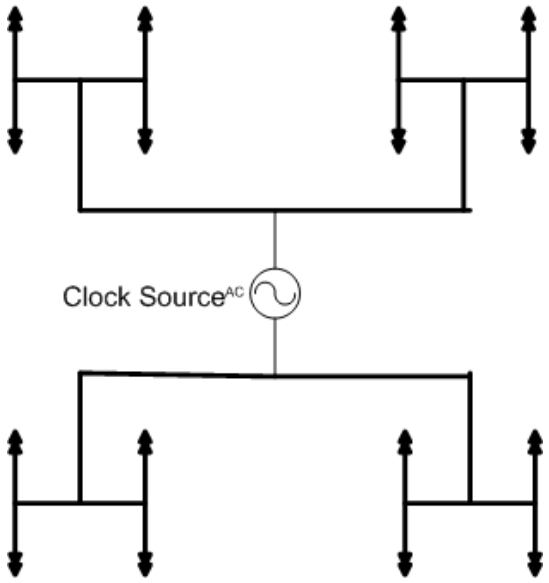

Figure 2.5: 4x4 H-Tree Clock Network

The H-tree configuration is particularly useful for regular array networks in which all elements are identical and the clock can be distributed as a binary tree [1]. A more general approach is the matched RC trees setting where the interconnections carrying the clock signals to the leaf modes are equal in length. A typical example is demonstrated in the design of the IBM S/390 microprocessor [7], as shown in **Figure 2-6**. In this design, a single clock is globally distributed in two levels of balanced H-trees from a centrally located on-chip PLL through a central chip buffer to the lower level local regions [7]. The first-level tree routes the global clock from the central clock buffer to the nine sector buffers where the interconnections are of equal length. Similarly, the second-level RC-matched tree is used to drive 580 clock pins on functional blocks. Overall, with optimized on-chip wiring, the clock skew is reduced to 30ps for a 400MHz CMOS microprocessor.

2.2.2 Clock Grid Network

Another clock distribution approach is the grid structure shown in **Figure 2-7** [1]. A clock grid is a mesh of horizontal and vertical wires driven from the middle and delivers the clock

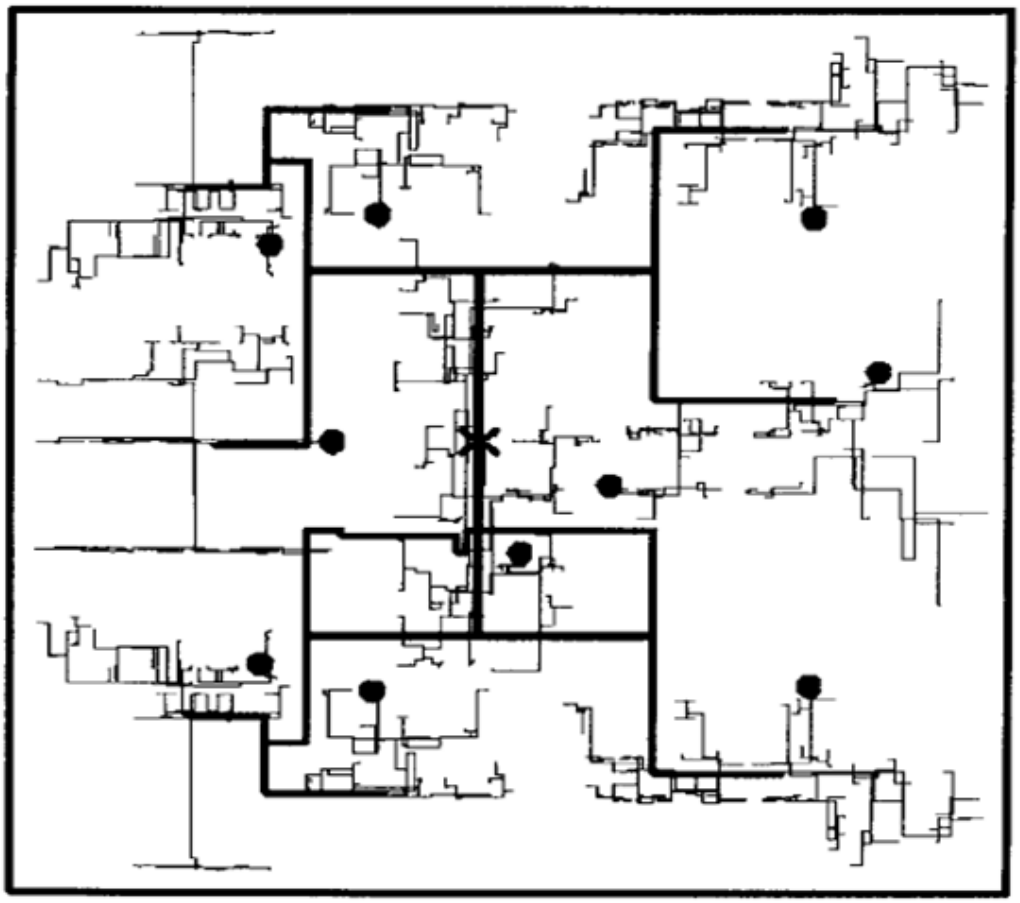

Figure 2.6: RC-Matched H-Tree in an IBM Microprocessor [7]

signal to the leaf nodes nearby the clocking elements. Since the resistance is low between any two neighboring leaf nodes in the mesh system, the skew will also be small between nearby clocking elements. The grid structure also compensates for much of the random skew because shorting the clock together makes variations in delays irrelevant. Hence, the absolute delay from the final driver to each clocking element is not matched but rather minimized. This approach is fundamentally different from the balanced H-tree approach [1]. It allows for late design changes since the clock is accessible at variation points on the chip. However, the grid system does have significant systematic skew between the points

closest to the drivers and the points furthest away. Furthermore, its power consumption is extremely high due to a large amount of metal resources and hence high switching capacitances.

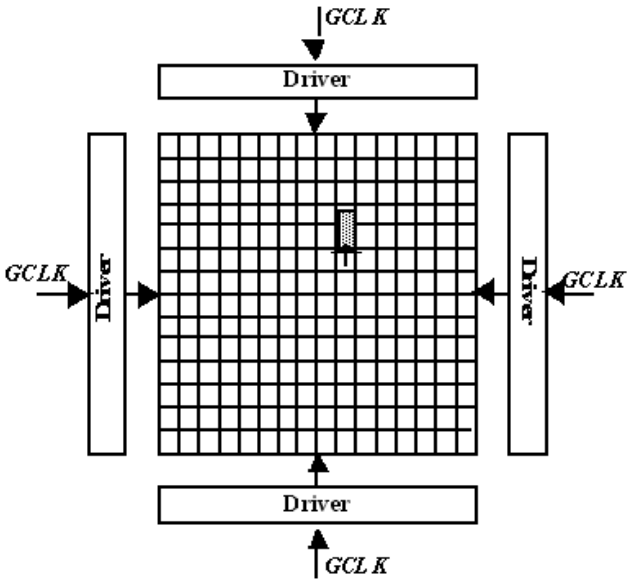

Figure 2.7: Clock Grid Structure [1]

The DEC Alpha series of microprocessor uses a grid-based clock distribution driven by one or more lines of buffers. As shown in **Figure 2-8**, this design uses a single-node, gridded, and two phase global clock named GCLK that covers the entire die [8]. In addition, this implementation uses a hierarchy of clocks such that local clocks and local conditional clocks are driven several stages past GCLK. **Figure 2-8** also illustrates the locations of the clock drivers along the global distribution network. A clock signal generated by PLL is routed to the center of the die and distributed by RC matched H-trees to 16 distributed GCLK drivers. It is from these drivers that the clocking elements in the grid receive the clock signals. Within the GCLK grid, all the clock interconnects are laterally shielded with either VDD or VSS interconnects. The GCLK grid was simulated under the worst case conditional loading, and 72ps of total skew was observed for a 600MHz microprocessor.

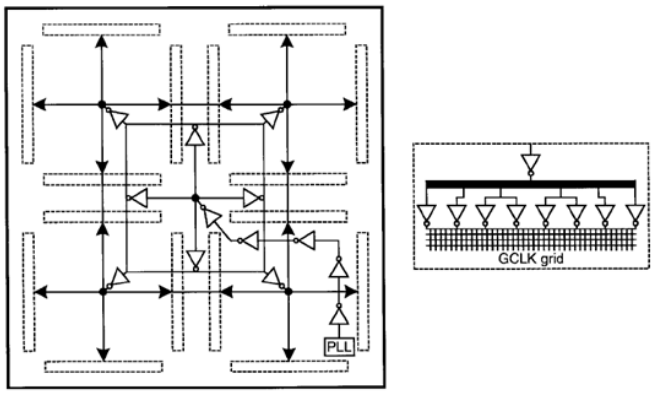

Figure 2.8: Clock Grid Structure for DEC Alpha Series Microprocessors [8]

2.2.3 Clock Spines

Figure 2-9 shows a clock distribution scheme using a pair of spines. In this design, the spines are made of clock buffers located in a few rows across the chip that drive length-matched serpentine wires to the individual group of clocking elements. If the loads are uniform, the spine avoids the systematic skew that is present in the grid structure by matching the length of the clock wires. The serpentine is easy to design and each load can be tuned individually for optimized performance. However, a complex microprocessor design with many clocking elements may require a large number of serpentine routes, and thus leading to high area and undesirable capacitive and inductive effects for the clock network.

The Pentium microprocessors designed by Intel adopt the clock spine distribution scheme. For example, Pentium II and III use a pair of clock spines while Pentium IV adds a third clock spine to reduce the length of the final clock wires [4] [9]. In the Pentium IV design shown in **Figure 2-10**, the global clock buffers distributing the clock to the three spines driving 47 independent clock domains result in zero systematic skew.

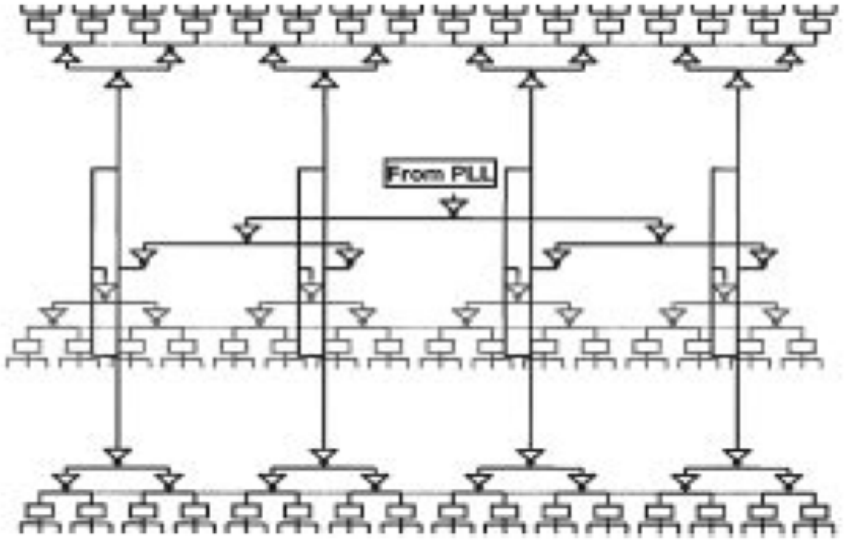

Figure 2.9: Clock Spine Structure [4]

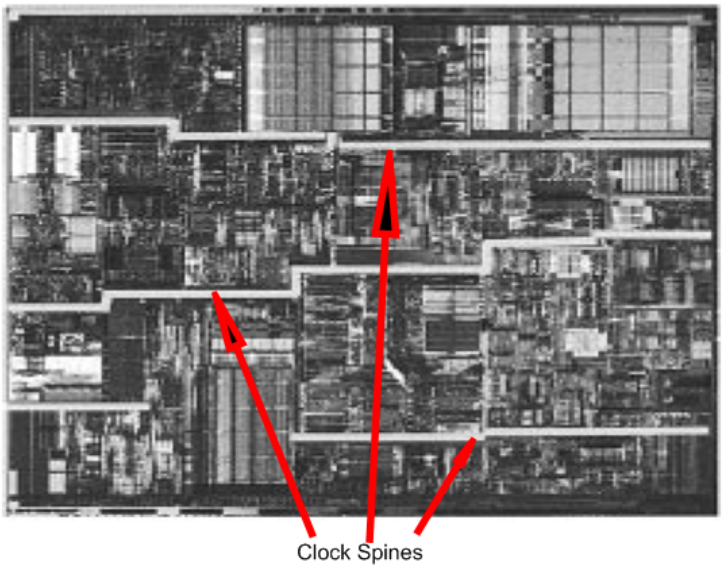

Figure 2.10: Clock Spine Structure in Pentium-4 Microprocessor [4]

Chapter 3

Background Information on Clock Skew

The previous chapter examines some of the common approaches of clock generation and distribution in high performance microprocessor designs. However, within-die variations due to technology scaling have made it virtually impossible to distribute a low-skew clock signal without some kind of skew-reduction circuitry. This chapter provides some important background information on clock skew. It will describe the impact of clock skew on digital synchronous systems, the factors that cause clock skew to exist, the need for an adaptive deskewing scheme, and a proposed deskewing methodology.

3.1 Introduction to Synchronous Systems

In digital logic design, the flow of data in synchronous systems is synchronized with the clock signal such that the data can be sampled directly without any uncertainty. The concept of a positive edge-triggered synchronous system is shown in **Figure 3-1**.

For the system shown in the figure, all the data is sampled at the rising edge of the clock signal for the register. Here, the data signal D_1 is sampled by register R_1 to yield the output signal Out_1. In turn, Out_1 passes through the combinational logic block and produces D_2 after a certain propagation delay. Finally in synchronization with the clock, Out_2 becomes valid after D_2 is sampled by register R_2. The worst propagation delay in the

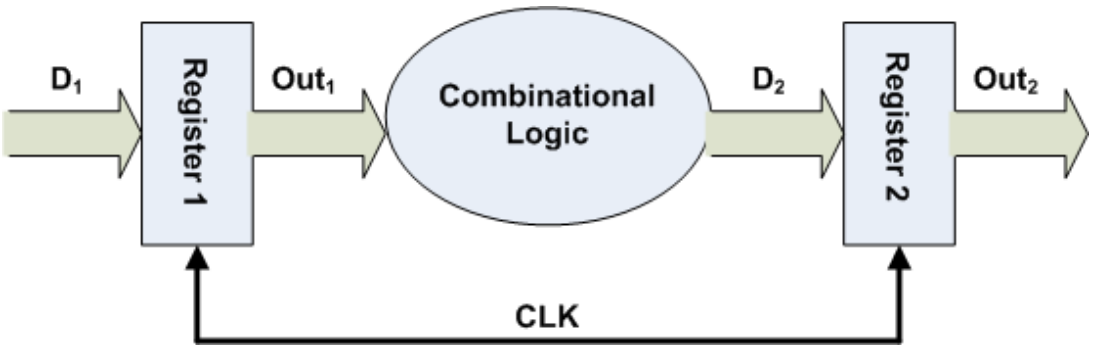

Figure 3.1: Block Diagram of a Synchronous System

combinational logic block, or the longest time it would take for D_2 to become valid, places an upper bound on the performance of the synchronous system. The requirement for the minimum clock period is discussed in more detail in the following paragraph.

There are two important timing parameters in any synchronous system: (i) setup and (ii) hold time associated with a register [1]. The setup time (t_{su}) is the time that the data inputs (D) must be valid before the clock transition. The hold time (t_{hold}) is the time the data input must remain valid after the clock edge. The other timing parameters that must be considered in a synchronous system include the maximum propagation delay of the register (t_{reg}) and the maximum delay of the combinational logic (t_{logic}). Under the ideal conditions, the phase of the clock signal at various locations of the system should be exactly identical where the clocks at registers 1 and 2 shown in **Figure 3.1** should have the same period and transition at the exact same time. Under such ideal assumption, the minimum clock period must be long enough for the data to propagate through the registers and logic and be set up for the destination register before the next rising edge of the clock [1]. This requirement is shown in Equation [3.1].

$$T > t_{reg} + t_{logic} + t_{su} \tag{3.1}$$

Similarly, Equation [3.2] shows that the hold time of the destination register must be shorter than the minimum propagation delay through the logic network.

$$t_{hold} < t_{reg} + t_{logic} \tag{3.2}$$

The timing analysis provided in this section is based under the absence of clock skew and jitter. In reality, process and environmental variations cause the clock signal to have both spatial and temporal variations, which lead to system performance degradation and possible circuit malfunction [1].

3.2 Impact of Clock Skew on Synchronous Systems

Clock skew is often defined as the spatial variation in the arrival time of a clock transition on an integrated circuit [1]. It can be positive or negative depending on the routing direction and location of the clock source. The temporal variation of the clock signal is often referred to as clock jitter where the clock period can randomly reduce or expand on a cycle-to-cycle basis. In this thesis, clock jitter will be ignored since the main focus of the research work is on reducing the clock skew. With respect to **Figure 3-1**, both *CLK1* and *CLK2* should arrive to the respective register at the exact same moment under ideal conditions. Due to static mismatches in the clock paths and differences in the clock load, however, *CLK1* may arrive earlier than *CLK2* by δ, which denotes the skew between the two clock signals. Clock skew δ is constant from cycle to cycle, which means if in one cycle *CLK2* lags *CLK1* by δ then it will lag by the same amount in the next cycle. The idea of clock skew is illustrated in **Figure 3-2** [1]. Clock skew can have significant impact on the performance of the synchronous system. In this section, we will examine the effect of both positive and negative skew on the clock period. For simplicity and clarity, we will assume a positive skew ($\delta > 0$) if *CLK2* lags *CLK1* by δ, and vice versa for negative skew ($\delta < 0$).

Figure 3-2 illustrates the concept of positive skew. The time available for a signal to propagate from R_1 to R_2 is increased by δ if we assume data input D_1 is sampled by R_1 at edge #1 and propagates through the combinational logic and be sampled by R_2 on edge #4. Hence, Equation [3.3] is a modification to Equation [3.1] that takes into account of the clock skew δ.

$$T > t_{reg} + t_{logic} + t_{su} - \delta \tag{3.3}$$

It actually suggests the minimum clock period required to ensure correct functionality is reduced with increasing clock skew. While this observation is indeed true, a high skew value will cause race conditions to occur and possible malfunction on the circuit. The race

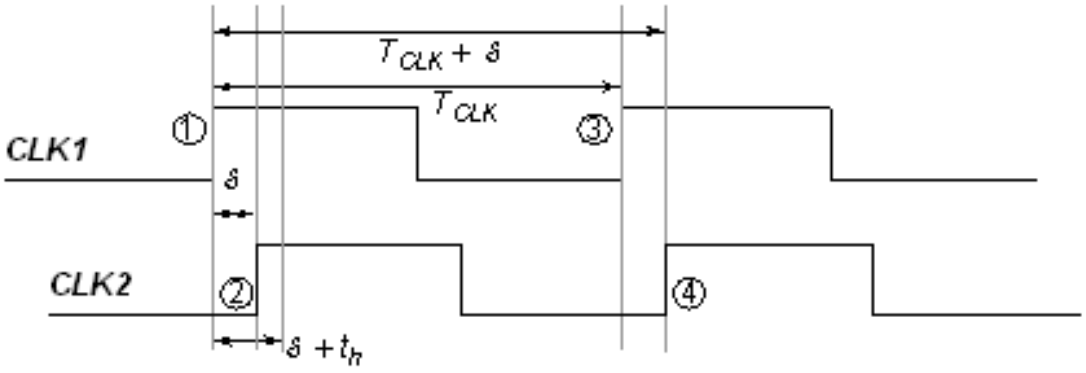

Figure 3.2: Example of Positive Skew [1]

condition occurs when the minimum delay through the combinational block is small such that the inputs to R_2 may change before the current input data is sampled correctly by the register. Hence, the constraint shown in Equation [3.4] must be satisfied such that the minimum propagation delay is long enough so that the input to R_2 is valid for an appropriate hold time.

$$\delta < t_{reg} + t_{logic} - t_{hold} \tag{3.4}$$

Figure 3-3 [1] shows an example of a negative skew where *CLK1* lags *CLK2* by δ. In this case, it is clear that the skew has a negative effect on the performance of the synchronous system as it increases the effective clock period by δ. On the other hand, the race condition will never occur in the synchronous system for a negative clock skew.

In general, positive skew occurs when a clock is routed in the same direction as the flow of the data in a synchronous pipelined datapath system, and vice versa for negative skew [1]. Since the flow of data can occur in either direction, the designer must carefully design the system such that the worst case skew conditions are taken into account.

3.3 Sources of Clock Skew

Minimizing clock skew is one of the key performance requirements in today's microprocessor design. In order to reduce clock delay and skew, the clock tree network is composed of large buffers driving interconnect and loads over the entire chip. Ideally, the clock generation

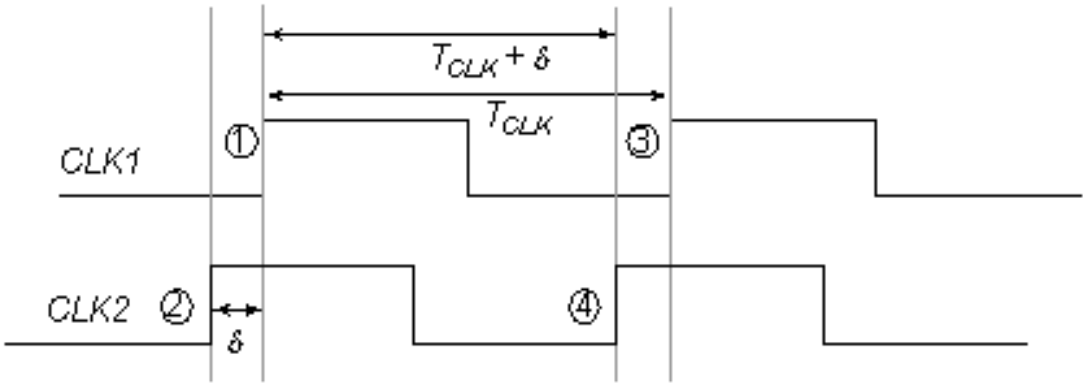

Figure 3.3: Example of Negative Skew [1]

circuit is expected to drive thousands of registers such that all the clock signals should reach the register inputs at the exact same time. In reality, however, various factors have caused the clock to be non-ideal where all the clock signals arrive to the registers at different times, and thus resulting in clock skew. This is shown in **Figure 3-4**. The uncertainty in the clock signal can be divided into two main categories: systematic and random [1].

3.3.1 Systematic Errors in Clock Skew

The systematic components of the clock skew may include variation in the total load capacitance in each clock path as well as layout parameters such as interconnect variation. This concept is illustrated in **Figure** 3-5 [10]. The global clock is distributed along two wires through two buffers to the respective register as two separate signals *CLK1* and *CLK2*. Although the two buffers are identical, the distance in which the clock signal has to travel to each register is different. In addition, one buffer drives a lumped load of 10pF while the other drives a load of 15pF. All of these factors have contributed to the systematic component of the clock skew where *CLK1* and *CLK2* do not arrive at their respective registers at the exact same time.

Systematic errors are mostly predictable and normally identical from chip to chip, and as such they can be modeled and corrected at design time [1]. Using deterministic variation models in circuit simulation can improve circuit performance in at least two ways. If the variation is greater than desired, post-design correction can be performed on the layout

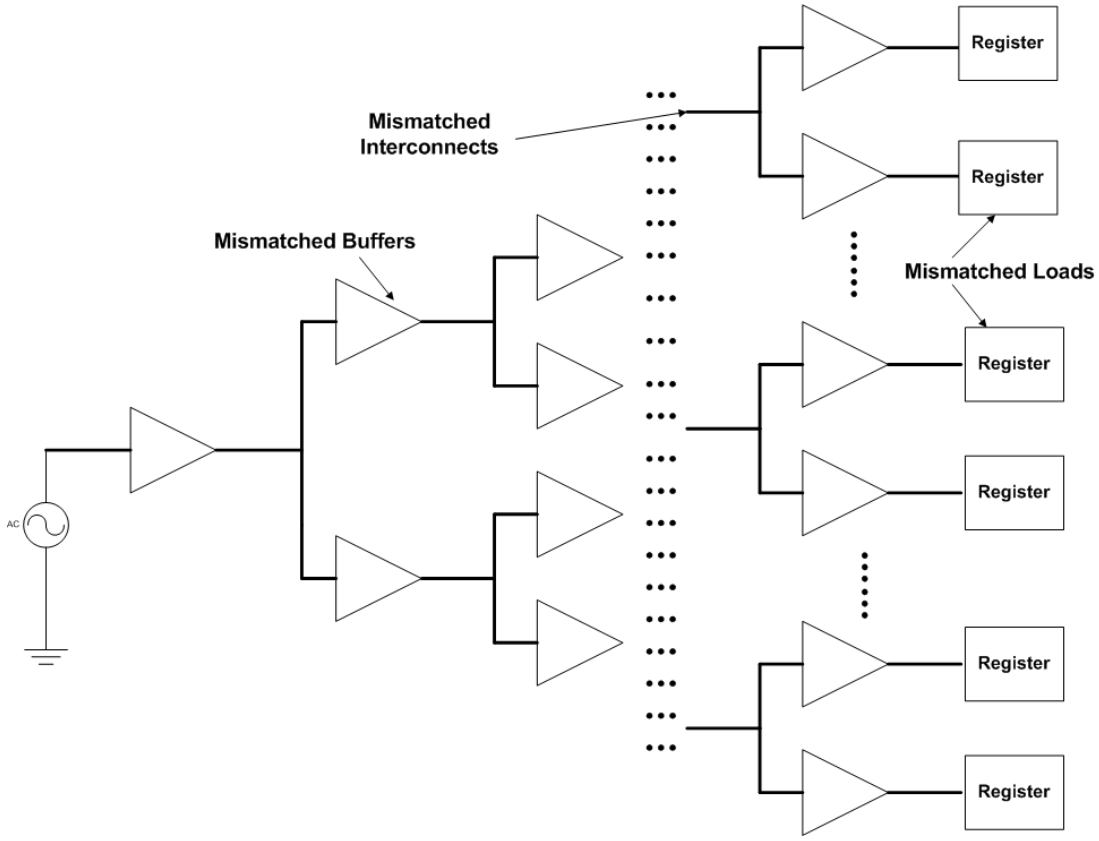

Figure 3.4: Mismatches in the Clock Distribution Network

where possible. Furthermore, if the actual variation is less than the worst case tolerance, design uncertainty can be reduced through better modeling.

3.3.2 Random Errors in Clock Skew

Unlike systematic errors, the random components of the clock skew are due to manufacturing process variations and circuit parameter tolerances that are difficult to model and eliminate. These variations can be divided into three categories: device parameter variations, interconnect parameter variations, and system parameter variations [11].

During the IC fabrication process, all the device parameters will deviate from their nominal values by a small margin. These parameters may include threshold voltage (V_{th}),

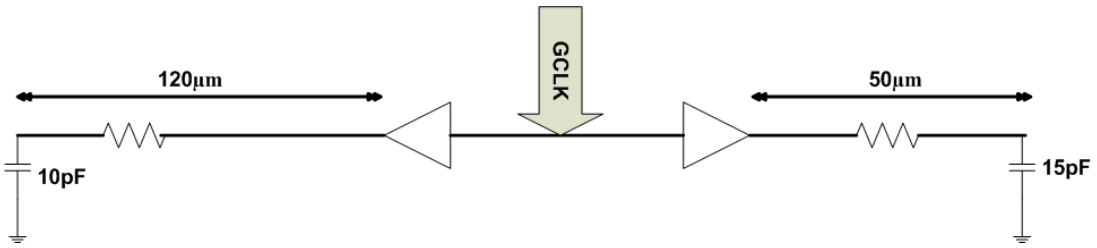

Figure 3.5: Mismatch in Wire Length and Load

gate oxide thickness (t_{ox}), and effective channel length (L_{eff}) [11]. For instance, dopant variation can affect the junction depth and dopant profiles which in turns cause electrical parameters such as V_{th} and parasitic capacitances to vary. Furthermore, the effective channel length of the transistor, L_{eff}, can be affected by the orientation of the polysilicon during fabrication. According to Equation [3.5] and Equation [3.6], all of these variations will in turn affect the amount of current flowing in the transistors and consequently the propagation delay.

$$I_{avg} = \mu C_{ox} \frac{W}{L}(V_{GS} - V_t)V_{DS} - \frac{V_{DS}^2}{2} \tag{3.5}$$

$$t_p = \frac{C_L V_{swing}}{I_{avg}} \tag{3.6}$$

Since the clock distribution network consists of numerous buffers and repeaters to drive both the register loads as well as various interconnects, matching of devices in the buffers along the multiple clock paths is critical in minimizing clock skew; any variations in the device parameters will result in some mismatch in the arrival time of the clock signal at the respective registers.

As the die size is increasing rapidly in today's microprocessor designs, long metal wires are necessary to distribute the clock signal across the entire chip. As such, random interconnect variations can have considerable impact on the clock skew. This may include variation in interconnect width (W), thickness (H), as well as inter-level dielectric (ILD). As copper is becoming the dominant choice for interconnect material, the IC fabrication process has adopted chemical mechanical polishing (CMP) as the main process to achieve planarization throughout the wafer. Although using CMP greatly reduces the metal and ILD non-uniformity in multi-layer structures, issues such as metal dishing and oxide erosion

will still result in ILD and interconnect thickness variations even after CMP [11]. Equation [3.7][3.8]and [3.9] show a rough estimation of the interconnect delay (t_p) using Elmore delay model as well as the calculation for interconnect resistance (R) and capacitance (C).

$$t_p = \frac{RC}{2} \tag{3.7}$$

$$C = \frac{\epsilon}{t} WL \tag{3.8}$$

$$R = \frac{\rho L}{HW} \tag{3.9}$$

From the equations above, it is apparent that any random deviation in the interconnect geometries from the nominal values will result in variations in the resistance and capacitance values, and consequently affects the delays of the distributed clock signals.

Other than process and interconnect parameter variations, system level fluctuations such as power supply voltage and temperature variations can also result in random clock skew. Variations in power dissipation across the chip have resulted in temperature gradients at various parts of the die. Since device parameters such as threshold voltage and electron mobility are a strong function of temperature, temperature variations can cause the buffer delay for a clock distribution network to vary drastically from path to path [1]. In addition to temperature fluctuation, IR drop and supply voltage variation across the chip also adds complexity to the clock network design in minimizing the skew. In modern VLSI systems, the power and ground distributions are done through a multi-layer mesh of metal wires [12]. Hence, the power and supply voltages seen by the transistors will deviate from the ideal supply voltage due to voltage drop along the wires caused by the resistance of wires to the current flow. This is illustrated in **Figure 3-6**. Some other factors that result in supply voltage variation include parasitic inductance of package pin, bond wires, and on-chip wires [13]. The fluctuation in the supply voltage causes variation in the delay, skew, and slew rates of the clock signals.

3.3.3 Impact of Technology Scaling on Clock Skew

As described by Moores Law, the downscaling of minimum dimensions of CMOS technologies enables the integration of an increasing number of transistors on a single chip. The

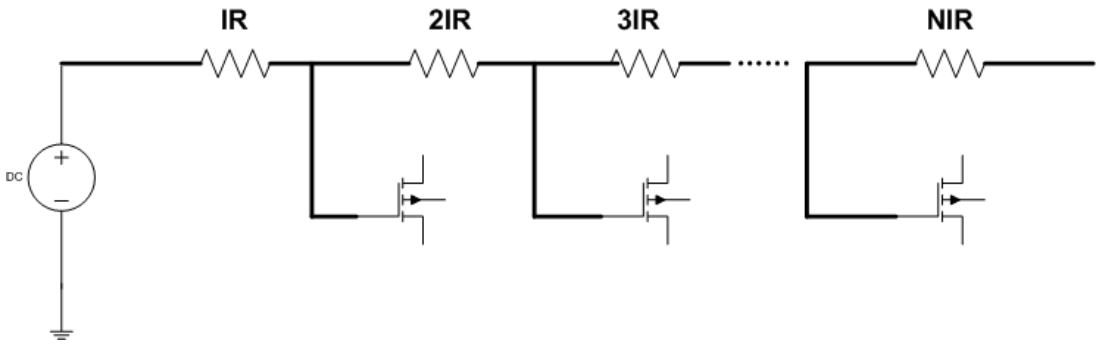

Figure 3.6: Effect of Wire Resistance on Power Distribution

performance of the microprocessor is also rapidly increasing with the scaling of transistor dimensions. **Figure 3-7** shows the projected clock frequency for microprocessors from 2005 to 2020 [14]. From the figure, it is clear that the projected clock period in the year 2020 will be approximately 13.67ps. Hence, the need to reduce and correct clock skew is even greater since a few Pico-seconds can be a large fraction of the clock cycle and can result in significant impact on the performance of the microprocessors. Furthermore, technology scaling has made within-die variations significant factors in high performance microprocessor designs.

In [15], the impact of technology scaling on delay variation is studied extensively. **Table 3-1** [15] illustrates the relative impact of device and interconnect variations on delay as a function of technology scaling. Based on the data presented, it is clear that as technology continues to scale into the deep sub-micron (DSM) regime, interconnect continues to play an important role in the overall circuit performance. Although the intrinsic delay in the devices is decreasing, the delay variations due to fluctuation in device parameters are slightly increasing. However, interconnect delay actually increases and is becoming a dominant factor due to smaller metal dimensions and metal pitch. Furthermore, resistivity is becoming one of the dominant factors in causing delay variability. As a result, technology scaling has caused both IR drop and interconnect delay variation to become significant factors in affecting clock skew in high performance microprocessor designs.

Table 3-2 [16] shows the simulated clock skew data in a sample H-tree for 180nm and 50nm technologies. It is clear that when technology scales from 180nm to 50nm, clock skew

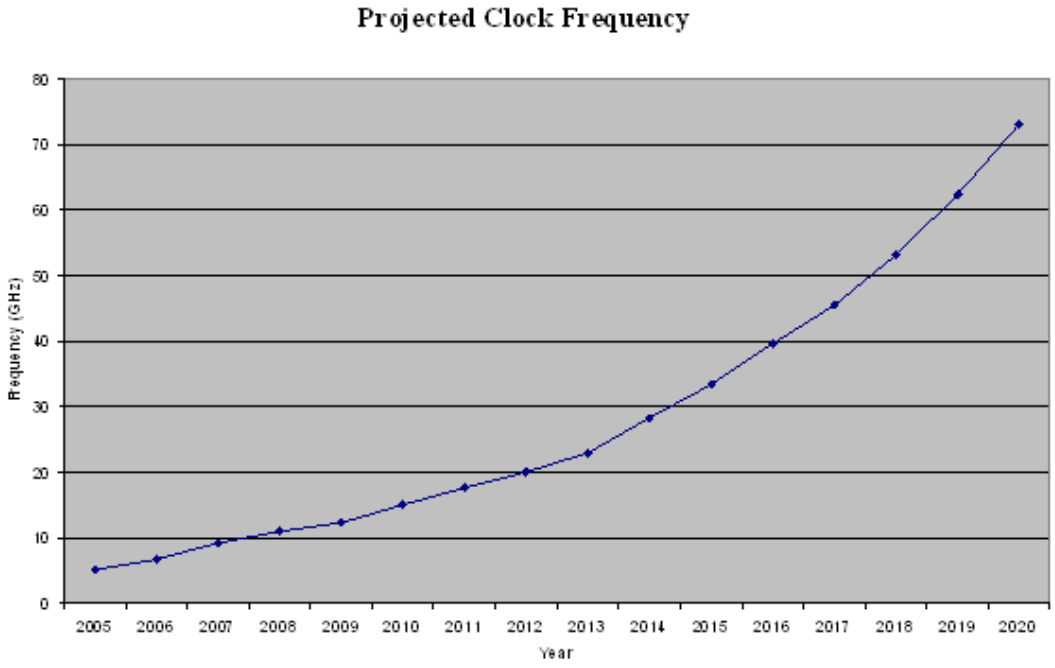

Figure 3.7: Projected Microprocessor Clock Frequency

can increase approximately from about 15% to 30% of the clock cycle. With increasing clock frequency and the dominating effects of device and interconnect variations due to technology scaling, the need to minimize clock skew is becoming a critical factor in any high performance microprocessor design.

3.4 Adaptive Deskewing

3.4.1 Need for Adaptive Deskewing

In section 2.2, some of the common approaches in distributing the clock signal across the chip in an effort to minimize the variations in clock skew between clocking elements are examined. However, none of these architectures can simultaneously eliminate both the systematic and the random component of the skew. With rapid technology scaling,

Table 3.1: Impact of Technology Scaling on Delay Variation [15]

	1997 (L_{eff}= 250nm)	1999 (L_{eff}= 180nm)	2002 (L_{eff}= 130nm)	2005 (L_{eff}= 100nm)	2006 (L_{eff}= 700nm)
Supply Voltage					
V_{dd}	9.5%	10.8%	10.0%	9.5%	8.9%
Device					
T_{ox}	1.3%	2.5%	3.2%	3.9%	4.9%
V_t	3.8%	5.3%	5.5%	6.5%	7.2%
L_{eff}	32.4%	28.3%	25.5%	24.6%	23.8%
Wire					
W	13.3%	12.0%	11.7%	11.4%	10.5%
S	9.3%	9.4%	9.9%	9.5%	9.4%
T	6.8%	7.0%	8.0%	8.2%	8.2%
H	7.8%	8.0%	8.1%	8.3%	7.1%
ρ	16.0%	16.6%	17.9%	18.4%	20.1%

Table 3.2: Impact of Technology Scaling on Clock Skew [16]

Variation Source	**Clock Skew in 180nm** (% of Clock Period)	**Clock Skew in 50nm** (% of Clock Period)
None	4.55%	6.62%
Devices and Supply Voltage	9.7%	15.59%
Interconnect	3.69%	4.44%
Total Skew	16.21%	27.96%

process and environmental variations have caused random skew a significant factor in the total clock skew despite the presence of the aforementioned clock distribution schemes.

In the research work published in [17], the effects of parameter variation and interconnect coupling on the delay of a clock signal propagating in a two-level H-tree clock distribution network are studied in a great detail. First of all, power supply variations will affect the current drive of the clock buffers within a given clock distribution network. A ±5% VDD variation from the nominal value of 1.8V results in 1% and 4% delay variation in the first and second level of the H-trees respectively. Secondly, with increasing die area, circuit density, and on-chip power dissipation, the variation of temperature across the die becomes significant. As temperature is increasing from the center of the network to the leaf nodes, the variations in the clock delay also increase from approximately 7% to 10%. Thirdly, rapidly scaling on the feature transistor size has caused manufacturing variations such as gate oxide thickness (t_{ox}) to become more prominent in affecting clock delay. Simulation results have shown that a ±5% variation in t_{ox} will affect the delay variation by approximately 1%. Finally, technology scaling has also reduced the distance between adjacent interconnect lines, and hence the capacitive coupling effects have increased significantly. It has been illustrated that coupling closer to the leaf nodes of the H-tree network has a greater effect on the delay variation of the clock signal.

Another study published in [18] has illustrated the effects of process and environmental variations in clock delays on a 1GHz Itanium-2 microprocessor using an H-tree network for clock distribution. The distribution network uses four levels of buffering elements between the PLL and the clocking elements. The detail impact of the skew sources on clock variations is listed in **Table 3-3**. In addition to the variations listed below, jitter comes from the power supply noise on the PLL and clock buffers is also a major problem for the H-tree network.

As CMOS technology continues to scale into the DSM regime, the effects of the process and environmental variations described above will have a more significant impact on the clock skew. Hence, designing a clock distribution network based on equalizing the delay time between the clock generator and the clock receivers is simply not enough to correct and minimize the skew present in a multi-gigahertz microprocessor. An adaptive deskewing scheme must be adopted in order to compensate for mismatches in clock distribution along

Table 3.3: Effects of Process and Environmental Variations on Itanium Microprocessors

Parameter	**Description**	**Delay Variation**
Supply Voltage (VDD)	100mV variation on a 1.2V supply	13%
Temperature	Variation of 20°C across the core	1.5%
Threshold Voltage	Non-constant variation for different transistors	2%
Channel Length	±12.5nm variation from a nominal length of 180nm	10%

various clock paths.

3.4.2 Examples of Deskew Buffer Designs

In recent microprocessor designs, adjustable delay buffers are inserted at various locations in the clock tree to compensate for mismatches in clock distribution along various paths.

In the design of Intel IA-64 64-bit microprocessor [2], variable delay buffers (DSKs) are strategically placed in the clock regions to minimize the skew. **Figure 3-8** shows the schematic circuit for the DSK. It is essentially a digitally-controlled analog delay line consisting of a 20-bit delay control register, a two-stage variable delay circuit, and a push-pull style output buffer. The 20-bit delay control was selected based on the delay step size resolution and the total buffer range. The measured delay range of the DSK is 170ps with a step size of 8.5ps. Finally, a deskew buffer controller compares the reference and the feedback clock signals and sends the appropriate information to the DSK for deskewing purposes.

A similar adaptive deskewing scheme is adopted in the Pentium IV processor [4] where a phase comparator checks the arrival times of the physical clocks and adjusts the digitally-controlled delay lines to make all clocks arrive simultaneously. The main deskewing circuit composed of 47 adjustable delay domain buffers and a phase detector network of 46 phase detectors. The schematic of the digitally-controlled delay domain buffer is shown in **Figure 3-9**. The delay lines can be adjusted to reduce systematic and random skew to approximately ±8ps as compared to 64ps before adjustment.

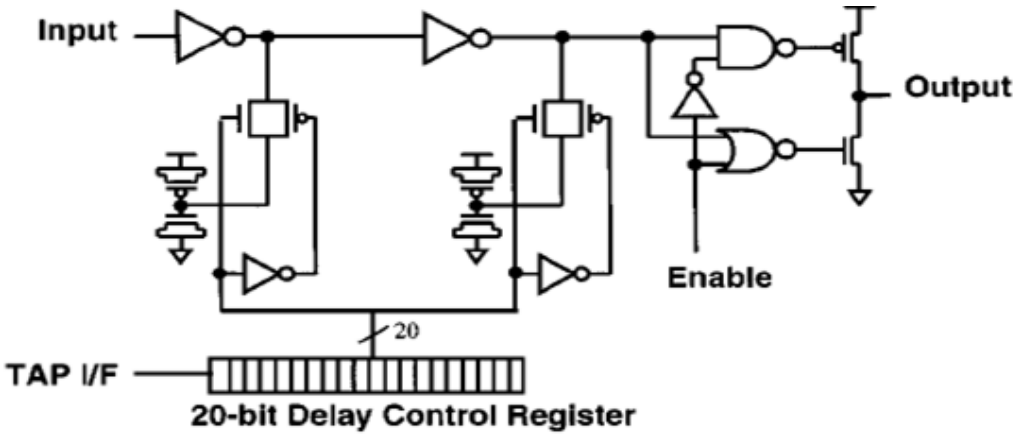

Figure 3.8: Digital Deskew Buffer in IA-64 Microprocessor [2]

3.5 Proposed Deskewing Methodology

Skew in a microprocessor can be divided into two main categories: intra-region skew and inner-region skew. Intra-region skew refers to the variation in the arrival time of the clock signals to the input node of each clock domain distributed by the clock generation circuit such as the PLL. Inner-region skew, on the other hand, is present between different clocking elements at the leaf nodes within a given clock domain. The deskew range of the deskew buffer designs presented in the previous section are good to correct the inner-region skews. For intra-region skews, however, the deskew range may not be adequate to synchronize all intra-region clock signals to a certain limit.

The complete proposed deskewing methodology, shown in **Figure 3-10**, involves the design and application of both the crude (red) and fine deskew buffer (blue). The crude deskew buffers are aimed at correcting the intra-region skews at lower strategic levels while the fine deskew buffers are designed to minimize the inner-region skews at the higher tree branches. The rest of this thesis will be dedicated towards the design and application of the crude deskew buffer with the goal of reducing the intra-region clock skew to a minimum. In addition, a sample clock tree in a real ASIC application will be analyzed in order to achieve the optimal performance of the deskew buffers.

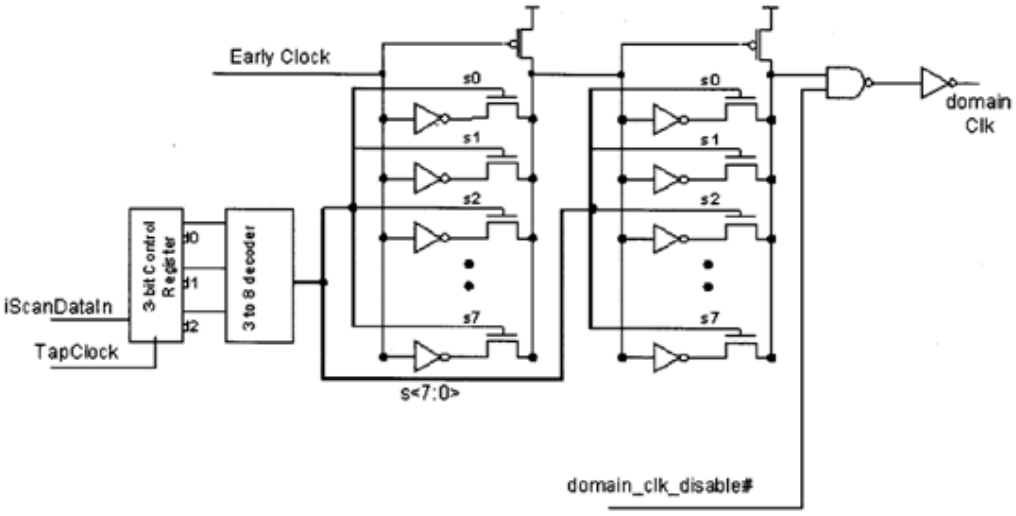

Figure 3.9: Deskew Buffer in Pentium-4 Microprocessor [4]

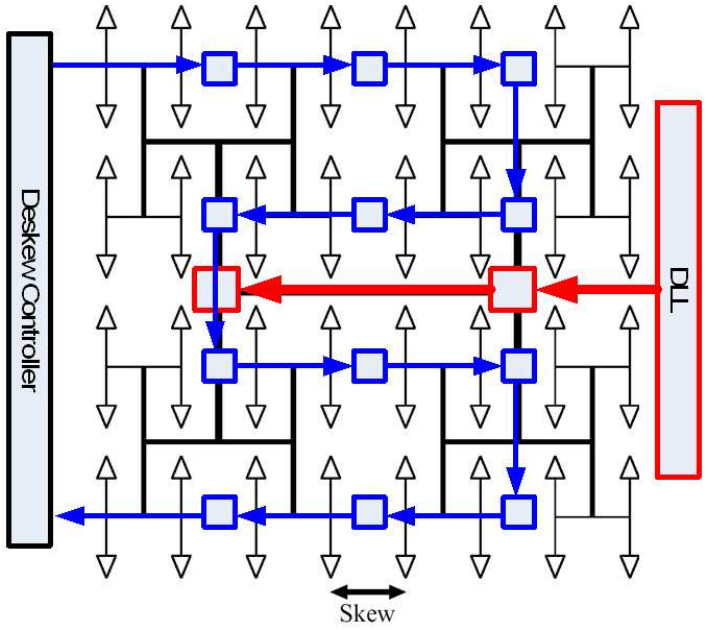

Figure 3.10: Proposed Deskewing Methodology

Chapter 4

Design of the Crude Deskew Buffer

The design of the crude deskew buffer (CDB) is based on the DLL architecture. It is composed of the following components: a phase detector, a charge pump, a voltage controlled delay line (VCDL) made up of current-starved delay elements, and a multiplexer. Once the DLL is under the locking condition, each delay element in the VCDL will produce a clock phase that is equal in delay step size. A digitally-controlled multiplexer will then select the appropriate clock phase(s) to be distributed to the rest of the clock network. Since the purpose of the CDB is to minimize the clock skew caused by process, voltage, and temperature (PVT) variations, the design is simulated and must function properly under the conditions listed in **Table 4-1**. Condition 1 is called the best condition because the transistor performance is at its optimum due to fast process corners and low temperatures. Condition 3, on the other hand, is at the opposite end of the spectrum because a high temperature and slow process corners will yield poor transistor performance. Finally, Condition 2 is the typical scenario with room temperature and normal process corners. In this thesis, both the best and worst simulation conditions will be referred to as the extreme conditions. This section will examine the criteria and techniques involved in the design of the crude deskew buffer.

Table 4.1: Simulation Conditions for the Crude Deskew Buffer Design

	Process Corner	**Temperature**	**Supply Voltage**
Condition 1: Best	Fast-Fast (FF)	0°C	1.2V
Condition 2: Typical	Typical-Typical (TT)	27°C	1.2V
Condition 3: Worst	Slow-Slow (SS)	110°C	1.2V

4.1 Delay Element

A delay element is a circuit that produces an output waveform that is similar but delayed version of its input waveform. It is an important part of the CDB design because it dictates the delay step size between each clock phase coming out of the deskewer, and thus determining the deskew range for the clock network. There are several architectures of delay element proposed in previous research work [19]. In this thesis, the transmission gate, cascaded inverters, and voltage-controlled delay elements will be briefly discussed and analyzed.

4.1.1 Comparison of Various Delay Elements

A transmission gate is a bi-directional switch consisting of a parallel connection of an NMOS and a PMOS transistor that are controlled by complimentary signals, as illustrated in **Figure 4-1**. The delay of a transmission gate is determined by the effective load capacitance C_L at the output as well as the equivalent resistance R_{eq} of the two transistors connected in parallel. By using the Elmore delay model, the delay of a network of n transmission gates is approximated by Equation [4.1].

$$t_p = 0.69 R_{eq} C_L \frac{n(n+1)}{2} \tag{4.1}$$

Due to the linear relationship between transistor length L and R_{eq}, delay is increased by increasing L of the transistors from the minimum size. Similarly, delay can be reduced by increasing the width W of the transistors; however, this approach is limited because the diffusion capacitances of the transistors will also increase, and thus contributing to the overall load capacitance C_L. Since a single transmission gate delay element only requires two

transistors, both the power consumption and area cost are kept to a minimum. However, signal integrity deteriorates quadratically with number of transmission gates in a chain [19].

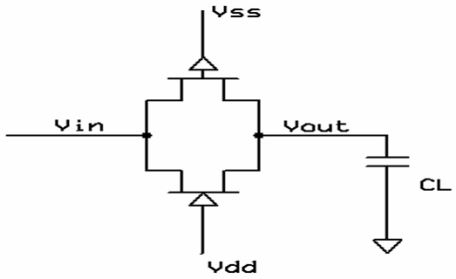

Figure 4.1: Transmission Gate-Based Delay Element

Another type of delay element, as shown in **Figure 4-2**, is the cascaded inverters where the delay is equal to the combined propagation delays of the two inverters. The approximate delay of an inverter is given by Equation [4.2].

$$t_p = \frac{C_L}{2V_{DD}}(\frac{1}{k_p} + \frac{1}{k_n}) \tag{4.2}$$

Similar to the transmission gate delay element, the propagation delay of an inverter is proportional to the effective resistance R_{eq} and load capacitance C_L. Equation [4.3][4.4][4.5][4.6] shows the power consumption of the inverter-based delay element consists of three components: static power, dynamic power, and short-circuit power.

$$P_{total} = P_{static} + P_{dynamic} + P_{short-circuit} \tag{4.3}$$

$$P_{static} = I_{leakage}V_{DD} \tag{4.4}$$

$$P_{dynamic} = C_L V_{DD}^2 f \tag{4.5}$$

$$P_{short-circuit} = \frac{t_r + t_f}{2} I_{short-circuit} V_{DD} f \tag{4.6}$$

As shown in the equation, the power consumption is a strong function of the supply voltage and the load capacitance. Furthermore, the static power will become an even more prominent factor in the overall power consumption due to the increasing effect of

leakage power from technology scaling. The area for the inverter-based delay element requires two times that of the transmission gate based delay element for the same number of stages. Finally, the signal integrity depends on the same factors that are affecting the propagation delay. Hence, careful transistor sizing is required in order to achieve the optimal performance for the inverter-based delay element.

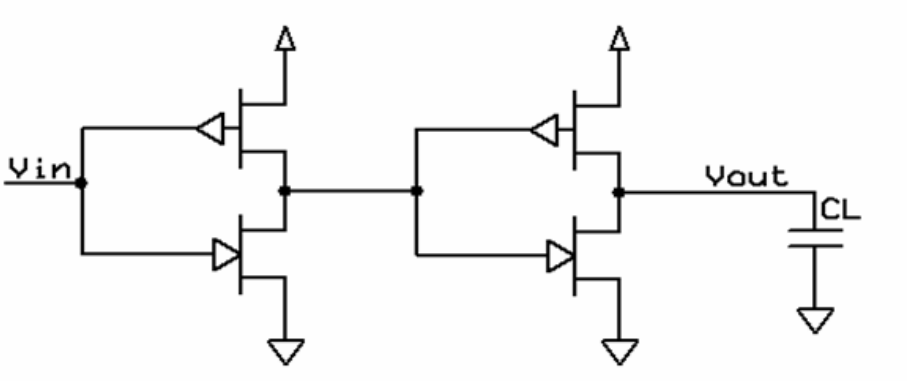

Figure 4.2: Inverter-Based Delay Element

Figure 4-3 illustrates a voltage-controlled delay element. It consists of a cascaded inverter pair with an additional series-connected NMOS transistor in the pull down path controlled by a control voltage called V_c [20]. There are several ways to change the delay of this delay element. Methods such as varying the transistor size and altering the effective resistance and capacitance are similar to those described for the transmission gate and the cascaded inverter delay element. The best and most accurate approach, however, is through the manipulation of the control voltage V_c. During the transition of the input signal V_{in}, one of the two inverters in the delay element will be discharging through a voltage-controlled transistor while the other charging through a PMOS transistor in the inverter. Therefore, the overall delay of the delay element is the sum of the normal inverter delay and a controlled inverter delay, as indicated by Equation [4.7].

$$t_p = C_L(\frac{1}{k_p V_{DD}} + \frac{V_{DD}}{k_n + V_c^2}) \tag{4.7}$$

Both the signal integrity and power consumption of the voltage-controlled delay element are similar to those for the cascaded inverter delay element. However, there are minor

differences. For example, the fall time of the output signal will be affected due to the dependency of the controlled voltage transistor in the pull-down path. Also, there will be a slight increase in the power consumption because of the additional diffusion capacitances of the controlled transistors that contribute to the total load capacitance C_L. In terms of area cost, the voltage-controlled delay element requires two extra transistors when compared to the cascaded inverter delay element.

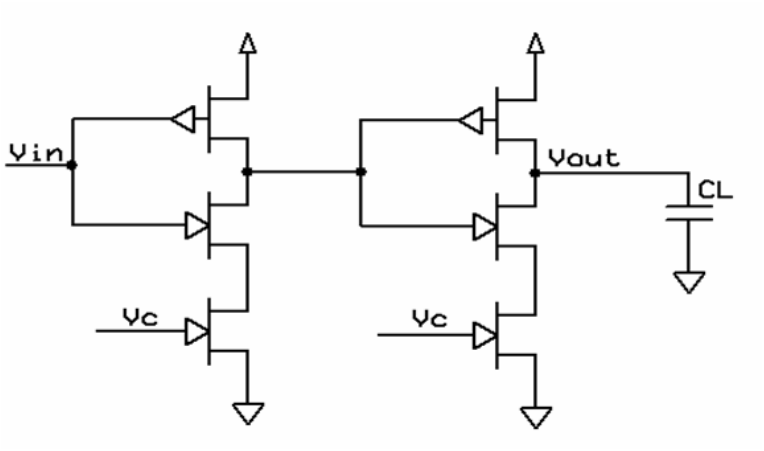

Figure 4.3: Voltage-Controlled Delay Element

4.1.2 Current Starved Delay Element

Since the crude deskew buffer uses a DLL structure that adopts a feedback loop, the voltage controlled delay element is the most logical choice for this design. With the feedback loop, the control voltage V_n will continuously fine tune the delay of each delay element until the DLL reaches the locked condition. A current-starved voltage controlled delay element is illustrated in **Figure 4-4**. As seen from the figure, the delay element consists of two cascaded stages to ensure both the rising and the falling edge of the signal is delayed by an equal amount such that the output is symmetric. Each stage includes a static inverter with current I_1 and an additional parallel pull-down path composed of voltage-controlled transistors with current I_2. By controlling the bias voltage V_n, the rise and fall times of the delay element can be changed to achieve the desired delay. The range of the delay element

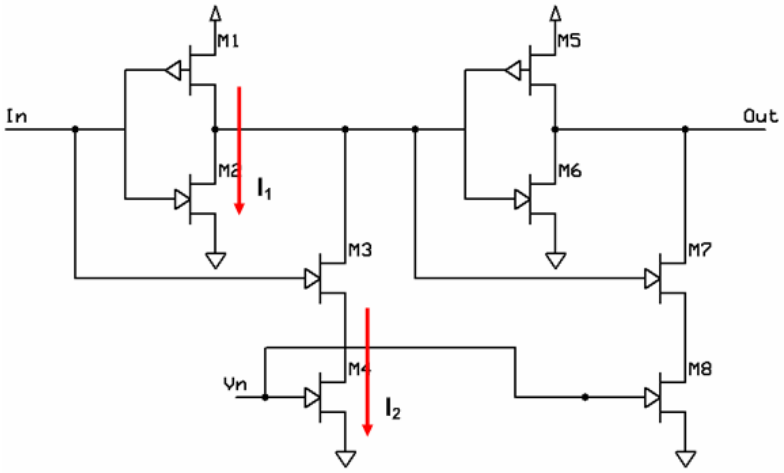

Figure 4.4: Current-Starved Delay Element

and its delay precision are determined by the relative size of I_2 to that of I_1. For example, a large I_2 allows the delay to be adjusted over a wide range of values but makes precision coarser and more dependent on V_n. Conversely, a smaller I_2 sacrifices some delay range for finer precision and less susceptibility to fluctuations in the bias voltage.

For the purpose of the CDB design, the control V_n will be constant once the DLL is in the locking condition. Hence, transistor M_3 and M_4 in the delay element are bigger than M_1 and M_2 such that I_2 is made bigger than I_1 in order to achieve a wide locking range. The simulation results of the delay element characteristic are shown in **Figure 4-5**.

The delay characteristic of the current-starved delay element corresponds to the inverse quadratic relationship between the delay and the controlled voltage given by Equation [4.7]. The three curves shown in the figure represent the delay characteristics under the three simulation conditions mentioned in the beginning of this chapter. Since the main goal of the CDB is to reduce the effects of PVT and interconnect variations, the DLL should be able to lock under both the typical and the extreme conditions described in **Table 4-1**. As such, the transistors in the delay element are carefully sized so that a common delay value is achieved for all three simulation conditions. The minimum and maximum delay of the delay element under each simulation condition is listed below in **Table 4-2**. As described in the next section, these values will have a significant impact on the locking range in the

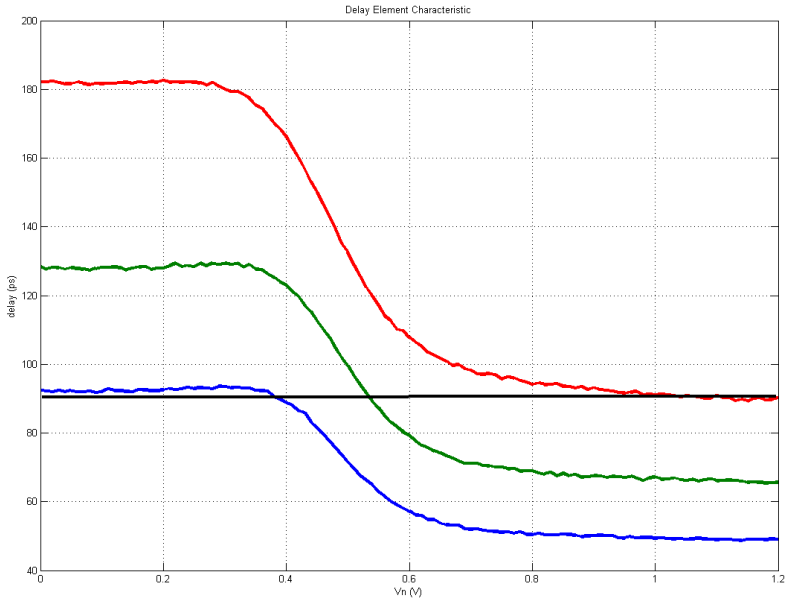

Figure 4.5: Delay Element Characteristics

design of the voltage controlled delay line. Furthermore, the horizontal line shown in the figure indicates the theoretical control voltage value for each simulation condition when the DLL is under the locking condition based on an 11-delay element voltage controlled delay line (VCDL) for a 1GHz clock signal; these values are also listed in **Table 4-2**.

Table 4.2: Delay Element Characteristics

	Maximum Delay (ps)	Minimum Delay (ps)	Theoretical Locking Voltage
Best	93.49	49.09	380mV
Typical	128.47	65.62	530mV
Worst	182.14	91.54	980mV

4.2 Voltage Controlled Delay Line

The voltage controlled delay line (VCDL) is considered to be one of the most important blocks within the DLL. Its performance directly affects the jitter of the output signal and the stability of the DLL. The VCDL in the design of the CDB is an open-loop system consisting of a number of delay elements connected in series. The inputs to the VCDL are the reference clock signal *REF_CLK* and a control voltage V_n. V_n will continuously adjust the delay of the *REF_CLK* signal until the DLL reaches the locked state such that its output, delayed clock signal *DCLK*, is exactly one full clock period T_{ref} (i.e. a phase shift of 360°) lagging the *REF_CLK* signal. The basic block diagram of the VCDL is illustrated in **Figure 4-6**.

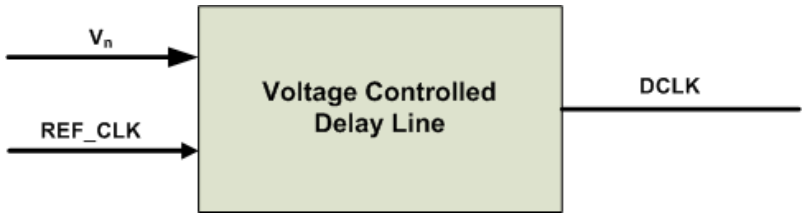

Figure 4.6: Block Diagram of VCDL

Typically, the DLL is designed for a specified clock period T_{ref} with two design parameters: the number of delay elements (n) and the delay each delay element (t_p) contributes. Theoretically, all the delay elements in the VCDL are identical, and hence each contributes a delay of $\frac{T_{ref}}{n}$ for an n-stage VCDL, as indicated in **Figure 4-7**. Usually, parameter n is chosen because it will dictate the number of clock phases coming out of the CDB to deskew the clock trees. Once both T_{ref} and n are determined, the delay element will be sized accordingly such that the desired propagation delay value $\frac{T_{ref}}{n}$ can be achieved for all three simulation conditions.

Another important aspect that must be determined during the design of the DLL is the locking range. The locking range refers to the minimum and maximum delays of the VCDL which enable a DLL to establish the locking condition. Since the VCDL can only delay the input reference clock signal by a maximum of one clock period and it lacks the frequency tuning capability a voltage-controlled oscillator possesses, it is possible for a

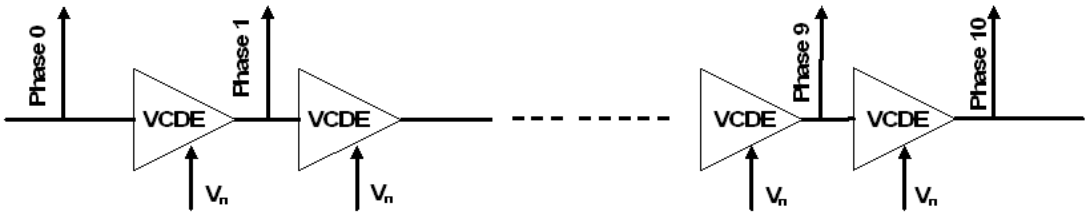

Figure 4.7: Delay Elements in VCDL

DLL to run into problems such as failing to lock or false locking. **Figure 4-8** and **Figure 4-9** illustrate an example of correct and false locking respectively.

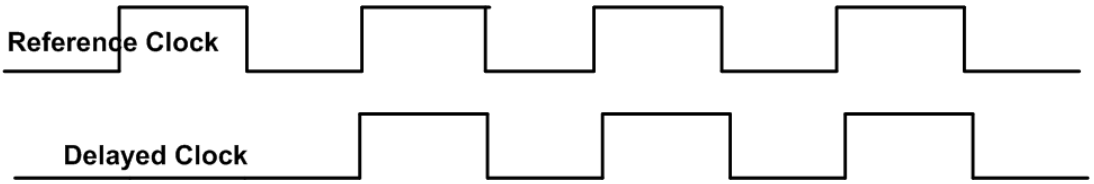

Figure 4.8: Example of Correct DLL Locking

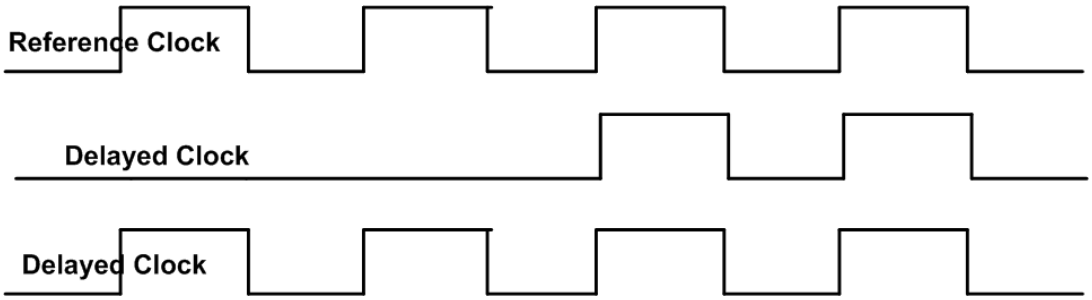

Figure 4.9: Examples of False DLL Locking

Under the correct locking condition, the first rising edge of the delayed clock signal *DCLK* should align with the second edge of the reference clock signal *REF_CLK*. As a result, the phase resolution in the DLL outputs is equal to the delay step size determined by each delay element. A possible false locking scenario occurs when the first edge of *DCLK* aligns with the third edge of *REF_CLK*. This occurs when the total maximum propagation

delay in the VCDL is greater than $1.5T_{ref}$. Although the two clock signals are aligned, the phase resolution in this case is twice as much as the desirable and original delay step size of the delay element. A similar false locking scenario occurs when the minimum VCDL delay is less than $0.5T_{ref}$. In this case, the first edge of the *DCLK* signal will try to align with the first edge of the *REF_CLK* signal, and the phase resolution will be zero. Overall, the frequency range of the input signal in which the DLL operate properly can be derived from the following criterion shown in Equation [4.8][4.9][4.10] [21] [22].

$$T_{VCDL,min} < T_{ref} < T_{VCDL,max} \tag{4.8}$$

$$T_{VCDL,min} > 0.5T_{ref} \tag{4.9}$$

$$T_{VCDL,max} < 1.5T_{ref} \tag{4.10}$$

Furthermore, equations shown above can be combined into an inequality relationship shown in Equation [4.11].

$$Max(T_{VCDL,min}, \frac{2}{3}T_{VCDL,max}) < T_{ref} < Min(T_{VCDL,max}, 2T_{VCDL,min}) \tag{4.11}$$

If a DLL can satisfy the above equations, it will operate correctly under the locking condition. Otherwise, it will fail to lock or falsely lock to two or more periods of delay [23].

As mentioned previously, the VCDL is designed for one particular clock frequency only. However, the clock input of the DLL often comes from the output of another block such as the PLL where the frequency of the signal can vary from the nominal value. For instance, a PLL may generate a nominal clock frequency of 1GHz (f_{ref}), but due to process variations and stability factors, the actual frequency may vary anywhere from 800MHz (f_{min}) to 1.2GHz (f_{max}). The delay element characteristic shown in **Figure 4-5** suggests that even though the DLL can lock correctly under the typical simulation condition for the given frequency range, but the phase resolution will vary for different clock frequencies. Furthermore, depending on the degree of variation from the nominal value, the DLL may not be able to lock properly when simulated under the extreme conditions due to the violation of the criteria stated in Equation [4.11]. A numerical example demonstrating the relationship between clock frequency, the number of stages in the delay line, the corresponding phase resolution, and the locking decisions is presented for all three simulation conditions

in **Table 4-3**. The decision for proper locking is based on the data shown in **Table 4-2**. For example, proper locking can be achieved for a given simulation condition if the phase resolution value can be found along the corresponding delay element characteristic curve.

Table 4.3: Locking Decision for Various Clock Frequencies in a given VCDL

	Frequency	Number of Delay Elements	Phase Resolution	Locking Decision
Best	800MHz	11	113.6ps	No
	1GHz	11	90.9ps	Yes
	1.2GHz	11	75.7ps	Yes
Typical	800MHz	11	113.6ps	Yes
	1GHz	11	90.9ps	Yes
	1.2GHz	11	75.7ps	Yes
Worst	800MHz	11	113.6ps	Yes
	1GHz	11	90.9ps	Yes
	1.2GHz	11	75.7ps	No

In this research work, a slight modification is done to the DLL where some bypass lines are inserted to the VCDL, and a multiplexer is used to select the appropriate delay line depending on the incoming clock frequency. This is illustrated in **Figure 4-10**. In the

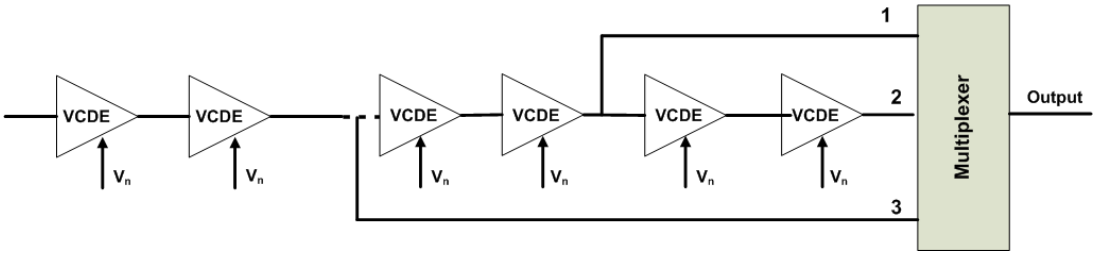

Figure 4.10: VCDL with Multiplexing Feature

figure above, three delay lines are shown in the VCDL design. The characteristic of each delay line is listed in **Table 4-4**. By referring to the minimum and maximum delay values shown in **Table 4-2**, it is clear that the DLL will lock properly under all the simulation

conditions for any frequency within the given range. Furthermore, the variation in the phase resolution due to changing in the clock frequency is also reduced significantly.

Table 4.4: Theoretical Phase Resolution of VCDL with Multiplexing Feature

	Frequency	Number of Delay Elements	Phase Resolution
Delay Line #1	800MHz	14	89.3ps
Delay Line #2	1GHz	11	91ps
Delay Line #3	1.2GHz	9	92.6ps

Another important aspect of designing the VCDL is to keep identical loads for each delay element in order to achieve equal delay steps and similar rise/fall time. As a result, dummy inverter loads are inserted at various locations both inside and outside the VCDL to equalize the loading effort of each delay element.

4.3 Phase Detector

The phase detector (PD) is another critical component in the design of the DLL because it dictates the accuracy of the locked clock signals. It detects the phase difference between the output signal from the VCDL, *DCLK*, and the input reference signal, *REF_CLK*, and produces output phase error information to the subsequent circuits. There are several common phase detector architectures used in the design of DLL.

The simplest phase detector is an XOR gate, as shown in **Figure 4-11**. The XOR PD detects the relative phase difference between the two clock signals and the output signal is linearly proportional to the phase difference of the two input signals, as evident in **Figure 4-11**. For this phase detector, however, a deviation in a positive or negative direction from the aligned position produces the same change in duty factor, and the average output signal becomes zero when the two input signals are 90° out of phase. The linear phase range for an XOR PD is only 180°. Thus, it is often used in quadrature locking where the two input signals are 90° out of phase in the locked state. The simple XOR PD suffers from several drawbacks. First of, only one output signal is generated by the XOR PD, and this makes it difficult to interface with the subsequent circuits. Secondly, the output of an XOR gate

is dependent upon the duty cycle of the two input signals, and hence incorrect phase error information maybe generated unless duty cycle correction circuits are used [24].

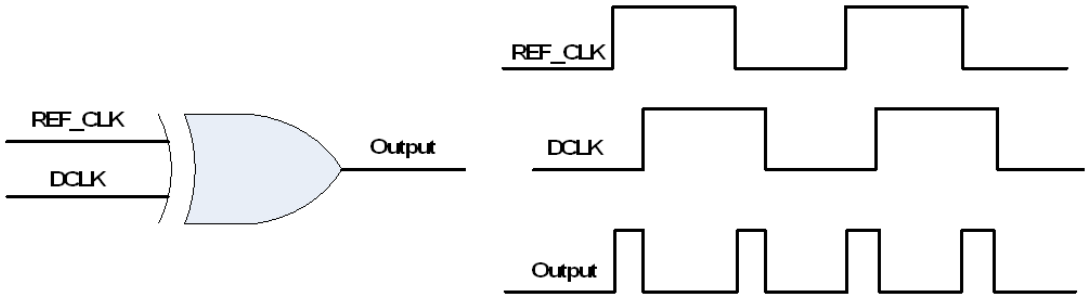

Figure 4.11: The XOR Phase Detector and the Corresponding Waveform

Another type of phase detector is the bang-bang phase detector, which can be implemented by various circuit techniques. The most common technique, however, is to use back-to-back D flip-flops (DFF) to sample the incoming clock signals. The detailed circuit implementation is shown in **Figure 4-12** where two identical C^2MOS DFFs are used. With this approach, the detector phase range is extended over the entire clock period of 360°. As shown in **Figure 4-13**, if *DCLK* is behind *REF_CLK*, an *UP* pulse is generated until the next rising edge of the *DCLK* signal. Similarly, a corresponding *DOWN* pulse is produced when *DCLK* is ahead of the *REF_CLK* signal until the next rising edge of the latter signal. When the two signals are perfectly aligned, no pulses are generated for either the *UP* or *DOWN* signal. The main disadvantage of this design is that the setup time of the flip flops will cause a certain dead zone where the PD cannot accurately and correctly detect the phase difference between the incoming signals. Such dead zone will cause some limitation on the locking of the DLL. For example, the minimum phase difference between the *DCLK* and the *REF_CLK* signals will be approximately 50ps if the phase detector is composed of DFFs that have a setup time of 50ps. Hence, optimal transistor sizing is required to minimize the setup time of the flip-flop, and consequently reducing the dead zone of the phase detector.

The phase frequency detector (PFD) is the most common used form of phase detector in high-speed DLL designs. As shown in **Figure 4-14**, it is composed of two flip-flops, an AND gate, and an inverter. When *REF_CLK* lags *DCLK*, an *UP* pulse is generated on

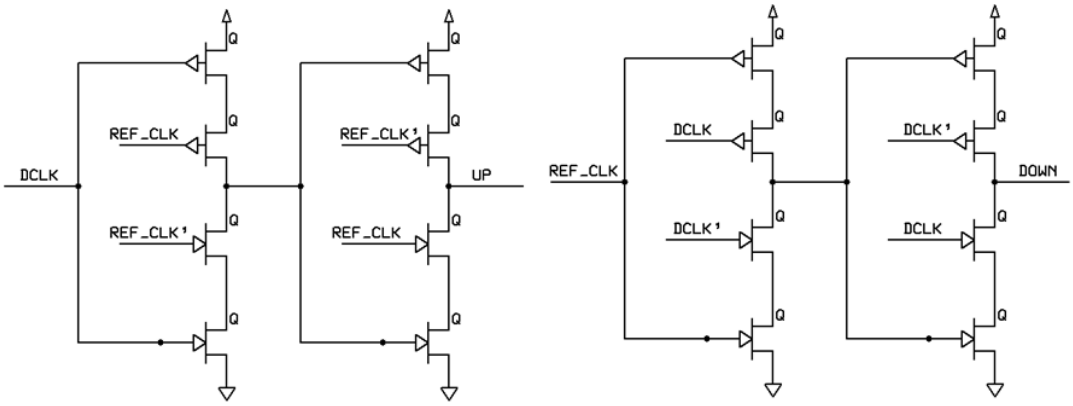

Figure 4.12: Bang-Bang Phase Detector

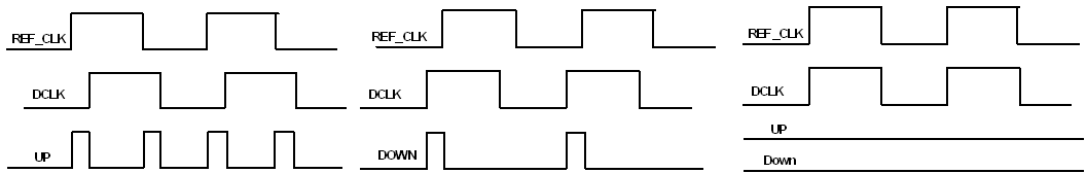

Figure 4.13: Waveforms for Bang-Bang Phase Detector

the rising edge of *REF_CLK*. The *UP* pulse remains until a low-to-high transition occurs on *DCLK*. When such transition happens, the *DOWN* signal is asserted and causes both flip flops to reset asynchronously. A very small pulse exists for the *DOWN* signal that is equivalent to the delay through the AND gate and the reset circuit. The pulse width of the *UP* signal is equal to the phase difference between the two clock signals. The same analysis can be done if the *REF_CLK* signal leads *DCLK*. When the two signals are perfectly aligned, short pulses will be generated for both the *UP* and *DOWN* outputs. The output waveforms of the PFD are illustrated in **Figure 4-15**.

Under the locking condition, both the location and pulse width of the *UP* and *DOWN* signal should be exactly identical. From **Figure 4-14**, the *UP* signal is generated with an extra inverter delay and additional load capacitance than the *DOWN* signal. Hence, additional inverters are added in the path of the *UP* and *DOWN* signal such that both the delay and pulse width are balanced and equalized, as evident in **Figure 4-16**. Since the

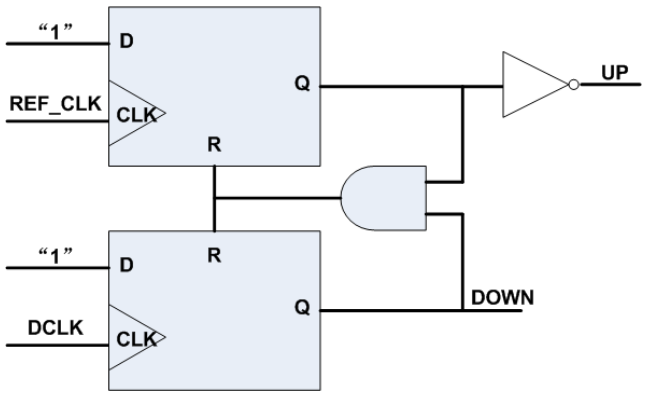

Figure 4.14: Phase Frequency Detector

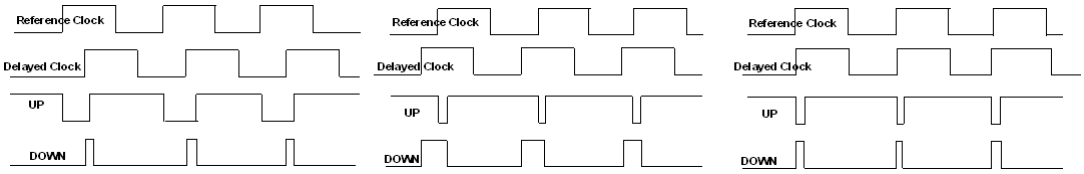

Figure 4.15: Waveforms for the Phase Frequency Detector

data input of both flip-flops is always a logic 1, the setup time of the flip flop is essentially zero. Hence, the phase locking limitation caused by the dead zone does not exist in this design.

If designed properly, the skew offset between the reference and the delay clock signal, *REF_CLK* and *DCLK*, can be minimized. As illustrated in **Figure 4-17**, the skew offset is referring to the phase difference between *REF_CLK* and *DCLK*. Ideally, this skew offset value should be zero indicating the *REF_CLK* and *DCLK* signals are exactly locked. However, due to design restrictions, an absolute zero skew offset value is difficult to achieve for all simulation conditions. Hence, the aim in this research work is to reduce the skew offset to a similar minimum value for all three simulation conditions. The skew offset will have a direct impact on the delay step size, or the phase resolution of the DLL. For example, an offset value of 50ps on a 1GHz clock frequency means the total delay in the VCDL is either 950ps or 1050ps. For an 11 delay element VCDL, this means the phase resolution will either be 86.4ps or 95.5ps, and thus deviates from the nominal value of

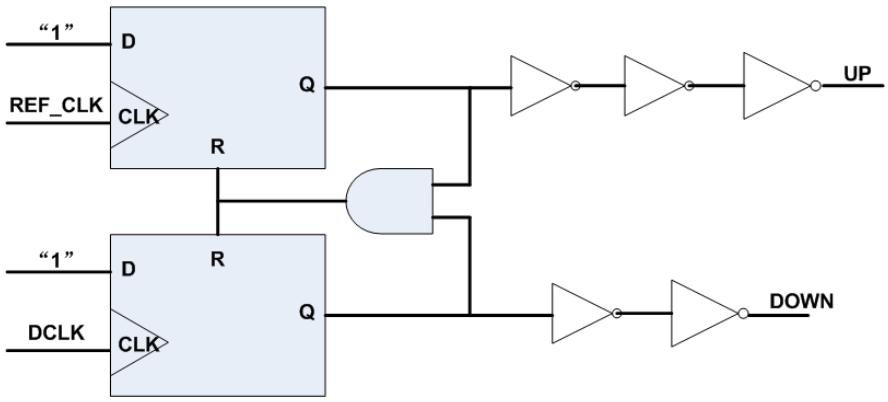

Figure 4.16: Modified Phase Frequency Detector

91.6ps. Hence, it is extremely important to minimize the skew offset in the CDB design in order to achieve more accurate phase resolution values.

In [25], the skew offset $\Delta\phi$ is modeled by Equation [4.12] where the pulse width of the *UP* and *DOWN* signal, the reference clock period, and the current mismatch of the charge pump are denoted by Δt_{on}, T_{ref}, and Δ_i.

$$|\Delta\phi| = 2\pi \frac{\Delta t_{on}}{T_{ref}} \frac{\Delta_i}{I} \tag{4.12}$$

4.4 Charge Pump

The phase error between the reference clock and the delayed clock generated by the PD is sent to the charge pump (CP) in the form of voltage or current pulses. The CP converts the pulses to an analog voltage that adjusts the VCDL delay based on the phase error information from the PD. In general, the charge pump architecture consists of two controlled switches, a current source, a current sink, and a loop filter consists of a capacitor, as shown in **Figure 4-18**.

Depending on the phase error information from the PD, the charge pump will either be charging or discharging the capacitor. For instance, *UP* pulse is generated if the *REF_CLK* signal lags behind the *DCLK* signal; this means that the charge pump must add charge to the loop filter in order to increase the analog control voltage V_n and decrease the delay

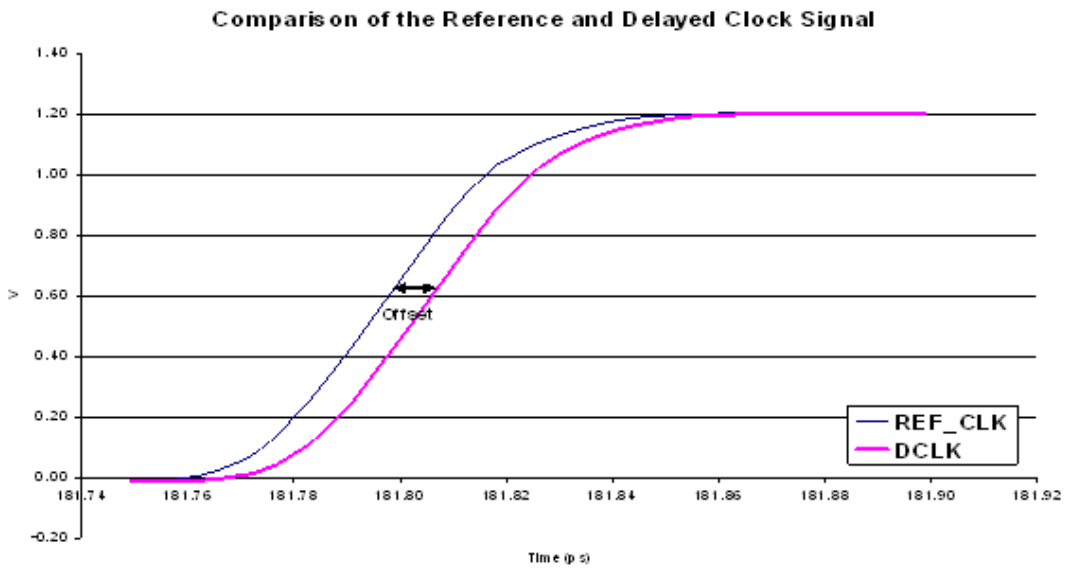

Figure 4.17: Illustration of Skew Offset

in the VCDL. The exact opposite scenario occurs when the *REF_CLK* signal leads the *DCLK* signal. As mentioned previously, both the *UP* and *DOWN* pulses are generated during the locked state, and thus simultaneously charging and discharging the capacitor. Hence, a key issue in properly designing the charge pump circuit is to equalize the charging and discharging currents such that the voltage of the loop filter is kept constant and thus ensuring the delay through the VCDL does not vary under the locked condition. From Equation [4.12], it is clear that the skew offset can also be minimized by reducing the current mismatch in the charge pump Δ_i. This can be achieved with careful sizing of transistor M2 and M6 shown in **Figure 4-18**.

4.5 Multiplexer

The previous sections have illustrated the detailed design of the individual components in a DLL. Once the DLL is under the locked state, the delay step between each clock phase will be fixed to a certain value. The final component in the CDB, the multiplexer, will

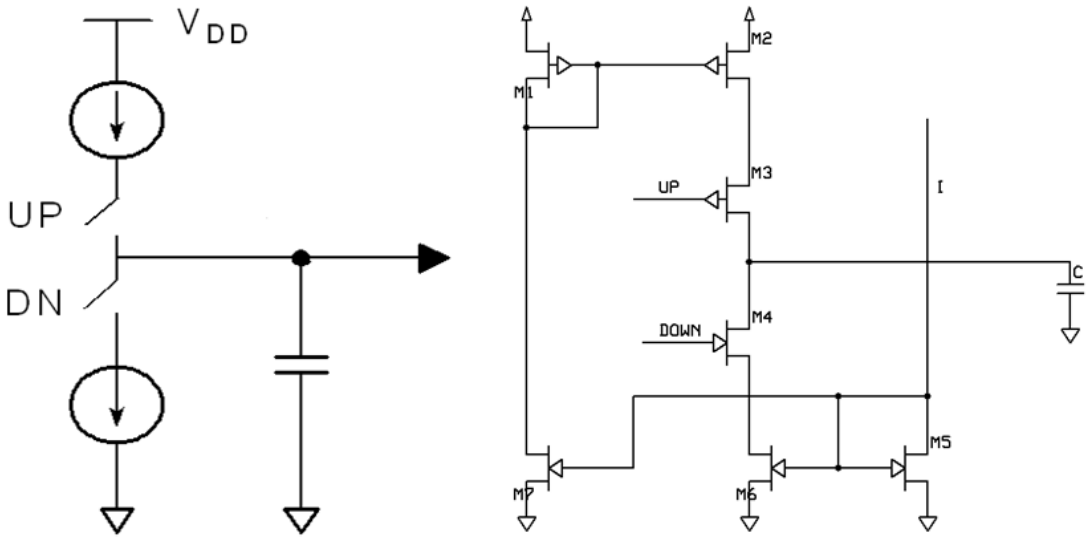

Figure 4.18: Schematic of Charge Pump

then be used to select the appropriate output clock phases to deskew the clock trees. The block diagram of the multiplexer is shown in **Figure 4-19**.

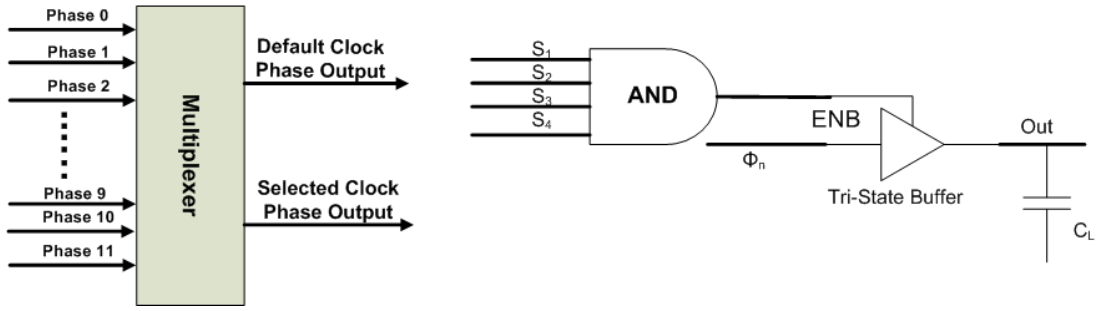

Figure 4.19: Multiplexer Design

The inputs to the multiplexer are essentially the different clock phases from the DLL. There are two outputs to the multiplexer: a default clock phase and a digitally selected clock phase. The design of the multiplexer consists of logic gates and tri-state buffers. The former is used to decode the incoming selection signals and produce an “Enable” signal that is sent to the tri-state buffer; the tri-state buffer takes both the ”Enable” signal and a clock phase from the DLL as its inputs. If the ”Enable” signal to a particular tri-state buffer is

true, then the corresponding clock phase will be selected as the output of the multiplexer. As shown in **Figure 4-19**, the selection of each clock phase is essentially constructed with identical logic decoder and a tri-state buffer. The output of all the tri-state buffers is connected to a common node, which is the final output of the entire multiplexer.

The process described above involves the selection of the clock phase based on the incoming selection signals. As mentioned before, the other output of the multiplexer is the default clock phase. This clock phase is always present regardless of the incoming selection signals to the multiplexer. Hence, the multiplexer is designed such that the default clock phase has the same propagation delay as the selected clock phase in order to have a synchronous phase difference with the selected clock phase. This is achieved by balancing the capacitive load at the output node of the default clock phase, as illustrated in **Figure 4-19**.

In addition to selecting the appropriate clock phase for deskewing the clock trees, the multiplexer component of the CDB can facilitate additional features such as "hot swapping" clock phases. Hence, a modified version of the multiplexer design will be presented in Chapter 5.

4.6 Overall CDB Performance

4.6.1 CDB Performance without Multiplexing VCDL

Figure 4-20 shows the block diagram for the entire CDB design. In this thesis, the CDB is designed for a nominal reference clock frequency of 1GHz. The delay characteristic of the delay element is shown in **Figure 4-5**. From the figure, the delay step size under all simulation conditions is approximate 91.6ps. Consequently, the VCDL consists of 11 delay elements, and a total of 12 clock output phases will be generated. These 12 clock phases are the inputs to the multiplexer where the appropriate phases will be selected to deskew the clock trees.

Upon power-up in any circuit, the initial conditions of all signals should be zero. Based on the delay characteristic shown in **Figure 4-5**, false locking will occur under the worst case simulation condition if the initial control voltage V_n is zero. Hence, V_n is initialized

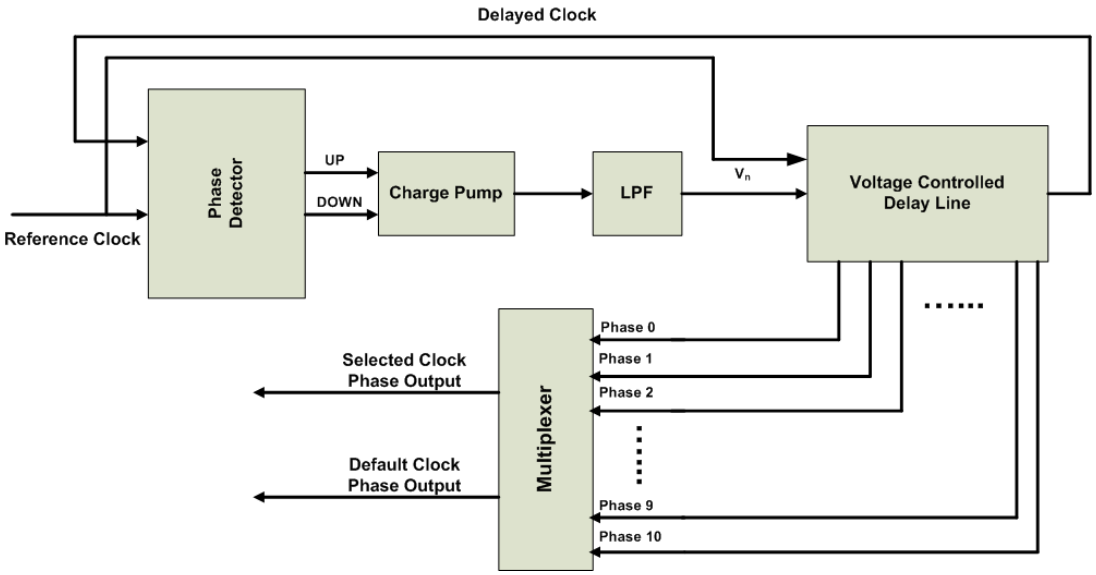

Figure 4.20: Block Diagram of the Crude Deskew Buffer

to 420mV so the conditions described in Equation [4.11] are satisfied for all simulation conditions. Once the initial condition is established, the phase detector and charge pump are carefully designed to minimize the skew offset between the reference and the delay clock signal, *REF_CLK* and *DCLK* for all simulation conditions. **Table 4-5** shows the skew offset data for 1000 rising edges of the *REF_CLK* and *DCLK* signal for the three simulation conditions.

Table 4.5: Skew Offset Values for Different Simulation Conditions

	Minimum Skew Offset (ps)	Maximum Skew Offset (ps)	Mean Skew Offset(ps)
Best Condition	6.49	7.42	6.95
Typical Condition	6.87	7	6.95
Worst Condition	6.292	6.294	6.293

For all simulation conditions, the skew offset is kept under 10ps. With an 11-delay element VCDL, a skew offset of 10ps is relatively tolerable and insignificant since, on average, the phase resolution is affected by less than 1ps. Hence, the locking condition

of the DLL is reached when the skew offset is under 10ps. **Figure 4-21** illustrates the histogram for the skew offset values for 1000 rising edges of the clock signal simulated under each condition. Both from the figure and the data shown in the table, it is clear that the skew offset value has been reduced to a minimum for all the simulation conditions. Once the DLL is under the locked condition, **Table 4-6** summarizes the overall performance of the DLL design based on a 1GHz reference clock frequency.

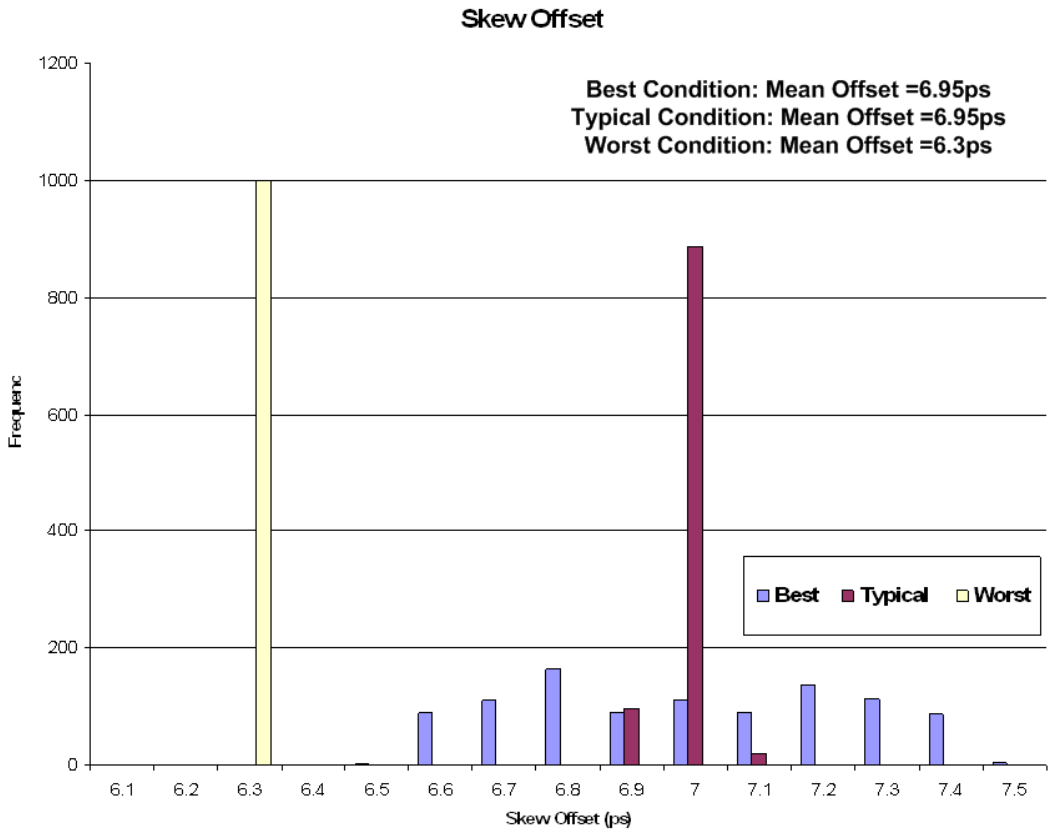

Figure 4.21: Skew Offset Histograms

From the data shown in **Table 4-6**, a phase resolution of approximately 91.6ps is achieved for the typical condition. The slight deviation in the phase resolution for the other two conditions is due to the different skew offset values achieved during the DLL

Table 4.6: CDB Performance Summary without Multiplexing VCDL

	Best Condition	Typical Condition	Worst Condition
Mean Skew Offset	6.95ps	6.95ps	6.3ps
Locking Voltage	378mV	539mV	983mV
Phase Resolution	90.3ps	91.6ps	91.5ps
Power Consumption	4.82mW	4.9mW	5.2mW
Lock Time	48 cycles	16 cycles	28 cycles

locking process. For example, the exact delay achieved in the best condition is 993ps, which is equivalent to a phase resolution of 90.3ps instead of the 91.6ps achieved in the typical condition where the clock signal is delayed by 1008ps. The measured locking voltage for each simulation condition is very similar to the theoretical values shown in **Table 4-2**. The locking time in each scenario will vary significantly since the locking voltage for each condition is different. For the purpose of the CDB design, the locking time of the DLL is relatively insignificant. **Figure 4-22** illustrates the locking voltage waveform for each of the simulation condition. **Figure 4-23** shows the delay step achieved from each delay element for each simulation condition. Finally, **Figure 4-24** shows the corresponding simulated waveforms of 11 sample clock phases with a phase resolution of approximately 91.6ps under the typical simulation condition.

4.6.2 CDB Performance with Multiplexing VCDL

As described earlier, the VCDL has a multiplexing feature that allows various reference clock frequencies to be locked by the DLL. **Table 4-7** summarizes the data for the performance of the DLL design with the multiplexing feature.

From the data shown, it is clear that the skew offset is kept under 10ps in almost all cases, and thus indicating the DLL is under the locked state. The only exception is the worst simulation condition for the 800MHz input clock signal scenario where the average skew offset value is approximately 14ps. In this particular case, the theoretical locking voltage is approximately 1.2V, as indicated by the delay characteristic shown in **Figure 4-5** and the theoretical values calculated in **Table 4-4**. In order for the locking voltage to

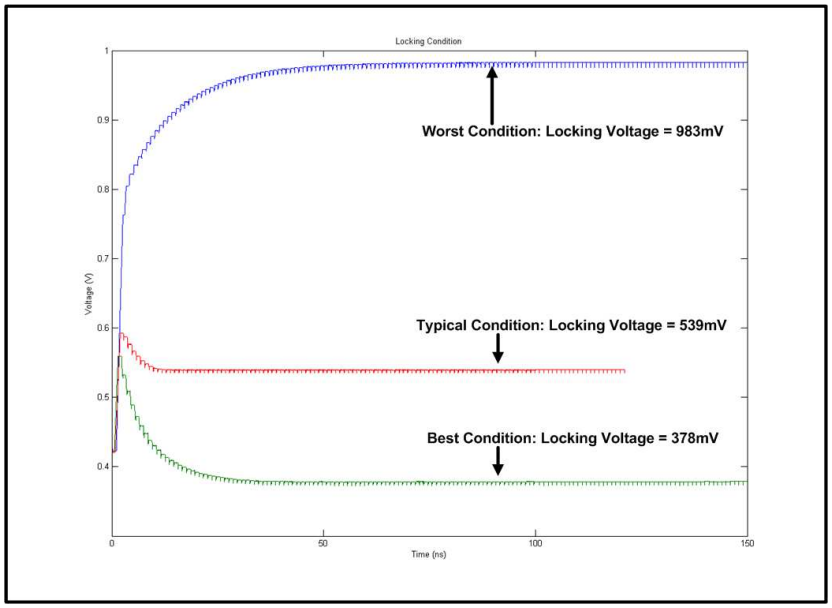

Figure 4.22: Locking Voltage Waveforms

Table 4.7: CDB Performance Summary with Multiplexing VCDL

		Best Condition	Typical Condition	Worst Condition
800MHz	Mean Skew Offset	8.51ps	5.79ps	13.94ps
	Locking Voltage	395mV	534mV	1.04V
	Phase Resolution	90ps	89.04ps	90.3ps
	Power Consumption	4.22mW	4.36mW	4.66mW
	Lock Time	22 cycles	11 cycles	80 cycles
1.2GHz	Mean Skew Offset	9.86ps	7.7ps	8.88ps
	Locking Voltage	374mV	520mV	975mV
	Phase Resolution	91.4ps	93.3ps	91.6ps
	Power Consumption	5.43mW	5.69mW	6.14mW
	Lock Time	30 cycles	8 cycles	65 cycles

reach 1.2V, the charging current in the charge pump must dominate over the discharging current. In doing so, however, the locking voltage in the best and typical condition will be

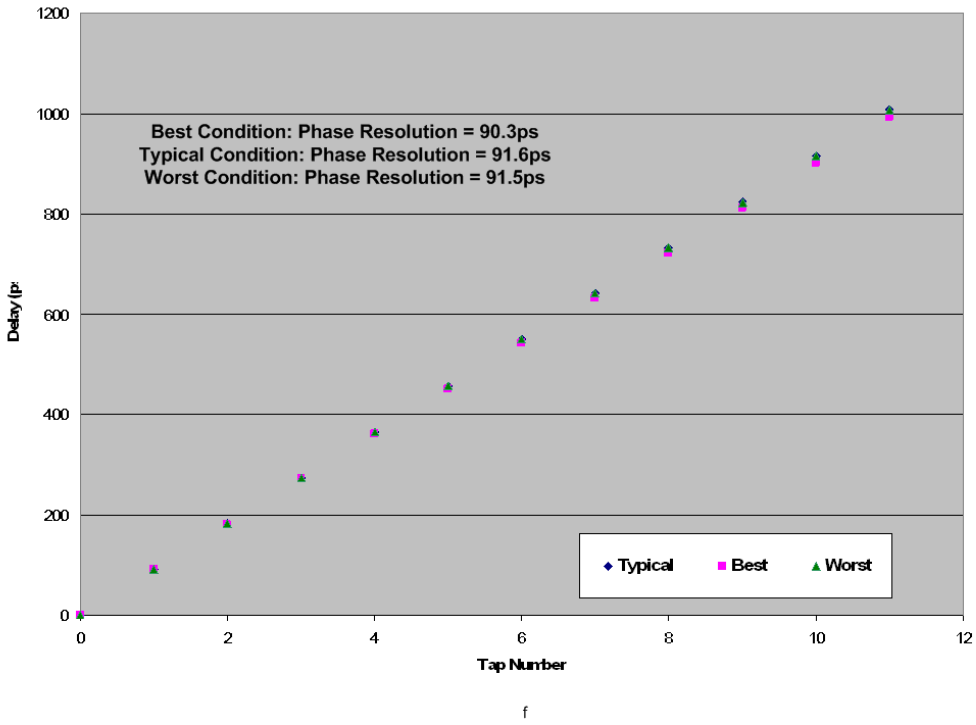

Figure 4.23: Phase Resolution of the CDB

affected because the discharging current plays a more significant role in these two cases. For the 800MHz clock frequency, a skew offset of 14ps is tolerable because the VCDL consists of 14 delay elements instead of 11 in the nominal 1GHz clock frequency. Hence, on average, the phase resolution will not be affected by more than 1ps. The locking voltage in each scenario deviates slightly from the nominal clock frequency due to the variation in the phase resolution. Due to the dependency of the switching power on the clock frequency, the power consumption is the highest with a fast clock frequency and lowest when the clock frequency is the slowest. With different skew offset values, the phase resolution in each scenario deviates slightly from the theoretical values shown in **Table 4-4**. For the three different input clock frequencies in all simulation conditions, the maximum phase resolution deviation is only 4.26ps. For the purpose of crude deskewing the clock networks, this deviation is not significant and should not affect the performance of the CDB in

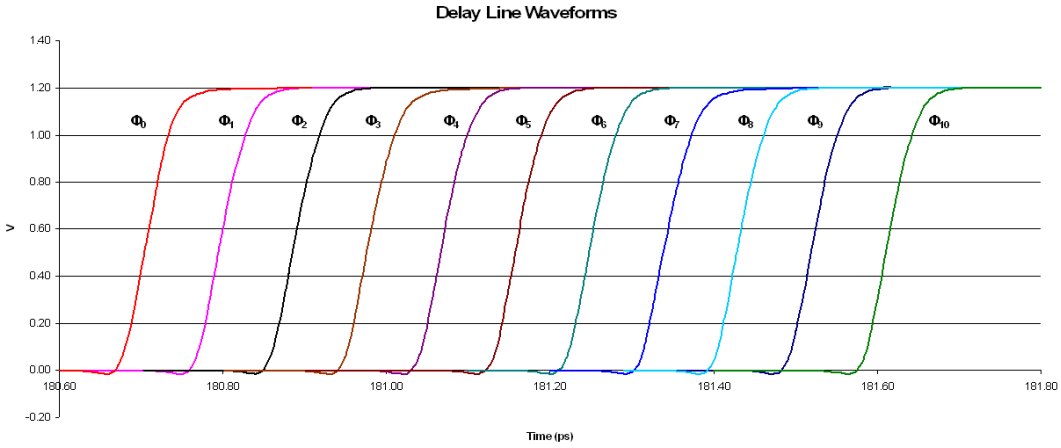

Figure 4.24: Waveform of 11 Clock Phases

deskewing the clock networks.

In conclusion, the design of the crude deskew buffer (CDB) based on a DLL structure requires careful analysis in choosing the appropriate architecture for each critical component such as phase detector, charge pump, voltage controlled delay line, and multiplexer in order to achieve the desired performance. In this thesis, a CDB is successfully designed and simulated with a frequency operation ranging from 800MHz to 1.2GHz at a nominal 1GHz clock frequency. Including the original clock phase, a total of 12 clock phases are generated with a phase resolution ranging from 89ps to 92ps depending on the input clock frequency. Each of the clock phases generated by the CDB will be used accordingly to deskew the clock network at the input source node. The addition of the multiplexing VCDL feature adds more robustness to the design of the CDB by allowing a range of clock frequency to be locked by the DLL instead of just one single frequency. Yet, the CDB performance is not affected with the addition of the VCDL multiplexing feature, as indicated by the data shown in **Table 4-7**. Furthermore, the CDB has very similar performance for each of the simulation conditions described in **Table 4-1**.

Chapter 5

"Hot Swapping" Clock Phases

In hardware systems, the term "hot swapping" is usually referring to the ability to remove and replace components of a machine while it is operating [26]. In this thesis, the term "hot swapping" refers to the ability of changing the duty cycles of the clock signals during the operation of the crude deskew buffer (CDB). This is accomplished by manipulating either the rising or falling edges of the output clock phases generated from the DLL. The variation in the duty cycle can be potentially useful in high speed debugging and delay testing.

5.1 Background Information and Motivation

As modern ICs are growing more complex in terms of logic gates and operating frequency, the deep-submicron (DSM) effects are becoming more prominent with shrinking technology, thereby increasing the probability of time-related defects [27][28]. In general, these defects can be divided into three main types: static, transition, and path delay faults. Static faults, or often referred to as stuck at faults, are defects that cause the circuit to malfunction at any speed. The transition and path delay faults, on the other hand, provide a relatively good coverage for delay-induced defects [28]. Path delay model targets the cumulative delay through the entire list of gates in a pre-defined path while the transition fault model is aimed at each gate output in the design for a slow-to-rise and slow-to-fall delay fault [29][30].

In the stuck-at model, a faulty gate input is modeled as a stuck at zero (SA0) or stuck at one (SA1). These faults most frequently occur due to gate oxide or metal-to-metal short circuits [6]. As technology continues to scale into the DSM regime, however, stuck-at fault tests alone cannot ensure high quality level of chips. Hence, both transition and path delay fault testing become increasingly important as more robust at speed techniques are necessary to reduce the number of timing-related defects [31][32].

Any test for path delay faults checks if the propagation time of signal transition through combinational paths exceeds the clock period. These tests, however, are generally complicated to design properly. Most chips have a large number of clock domains of different sizes. The commonly used industry methods test either the whole chip at the slowest speed or separately create tests for each domain, which requires specific knowledge of the design and running multiple test generation processes [33]. These practices require the manual entry of the time at which each clock event must occur in certain pattern and verification by a separate static timing analysis process [34]. To simplify the design of test generation for delay fault testing, a procedure known as the variable-clock method is often adopted. In this method, both fast and slow combinational or sequential logics can be tested using different clock frequencies. For instance, the clock can be slowed down to make the faulty circuit behave as a fault-free circuit until some desired state is reached. By increasing the clock frequency, stress testing is often performed under certain logics to examine their functional behaviour under harsh conditions. In addition, shorter clock periods can be applied on faster logics to determine the absolute minimum clock cycle requirement; this feature would help the designers to optimize chip performance by allocating appropriate skew budget into various parts of the chip. All in all, the potential of applying the variable-clock method for debugging and testing purposes is enormous. The most popular approach in generating a variable clock period signal is through the manipulation of the appropriate clock edges.

One method of changing the effective duty cycle of the clock is proposed in [2] where manipulation of the clock edges is done using the On-Die Clock Shrink (ODCS) feature implemented within the clock generation unit. In this work, the ODCS block consists of the ODCS controller, an ODCS delay buffer, and the ODCS bus block generation circuitry. The ODCS controller contains three registers that will specify the rise and the fall delays

of the ODCS buffer at any three consecutive cycles. Clock edge manipulation is done by the ODCS controller that will determine the specific cycle transition with appropriate rise and fall times. The hardware overhead associated with this design is quite large due to the additional circuitry required to implement the ODCS feature. In addition, the amount in which the clock edges can be manipulated is limited by the rise/fall time parameters stored in the registers.

The design proposed in this thesis requires no additional circuitry other than the design of CDB discussed in Chapter 4; only a slight modification is made to the multiplexer design in order to facilitate the "hot swapping" feature. The manipulation of clock edges is achieved through swapping between the initial and the final clock phases, and it is controlled by the selection signals from the multiplexer. Instead of relying on rise/fall time values stored in the registers, the clock duty cycle is changed according to the phase resolution from the CDB design.

5.2 The Operation of "Hot Swapping"

As mentioned previously, "hot swapping" is the altering of the clock duty cycle such that the edges of the clock signals are manipulated. **Figure 5-1** shows the waveform of ten sample clock phases generated from the DLL with each phase separated by ϕ. If N represents the numerical difference between the final and initial clock phase, and $\theta_{initial}$ is the normal duty cycle of the clock signal, the new duty cycle of the clock signal θ_{final} after the "hot swapping" process can be calculated by Equation [5.1].

$$\theta_{final} = \theta_{initial} \pm N\phi \tag{5.1}$$

Figure 5-1 also illustrates a numerical example of the phase swapping concept such that the delay step size ϕ is 90ps while the positive and negative duty cycle is 540ps and 460ps respectively. With respect to changing the positive duty cycle, the rising edge of the final output clock signal *Out* is initially the rising edge of ϕ_0. For the falling edge, however, the *Out* signal is following that of ϕ_2 instead of ϕ_0. Hence, the *Out* signal is a result of swapping two clock phases such that the effective positive duty cycle has been increased by 180ps. A similar analysis can be done on the changing of the negative cycle. In the

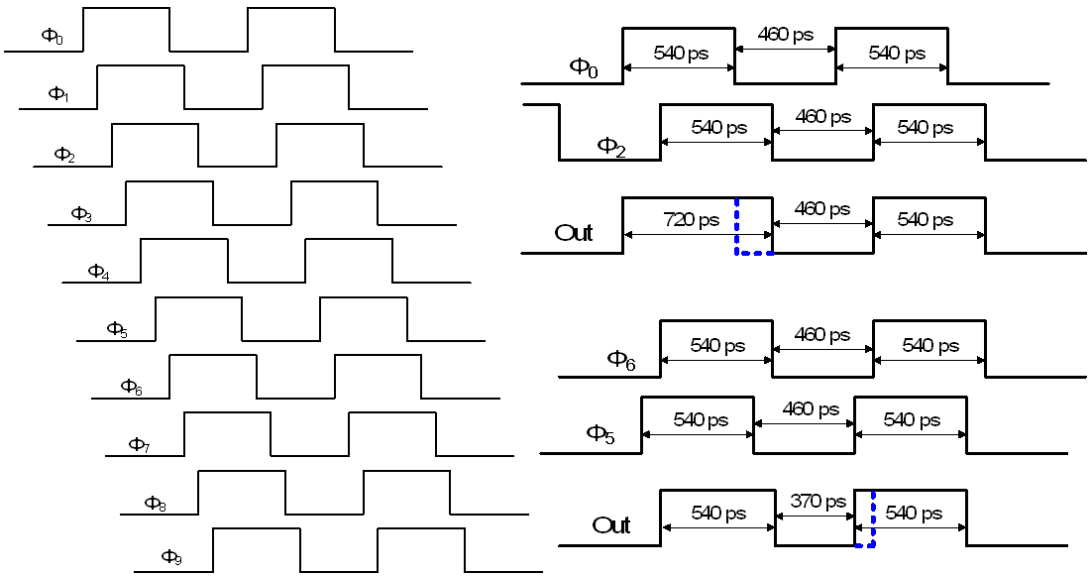

Figure 5.1: Illustration of "Hot Swapping"

example shown above, the clock phase has been swapped from ϕ_6 to ϕ_5, and the effective duty cycle has been reduced from 460ps to 370ps.

5.3 Implementation of the "Hot Swapping" Feature

The implementation of the "hot swapping feature" in the crude deskew buffer (CDB) is facilitated by some modification to the design of the multiplexer. **Figure 5-2** illustrates the original 12-to-1 multiplexer designed described in Chapter 4. With 12 input clock phases, the capacitance load associated with 12 tri-state buffers at the output node of the multiplexer is large. This large load capacitance can result in unacceptable rise/fall time at the output node, which might also lead to inaccurate phase swapping in terms of reducing or increasing the effective duty cycle. Hence the one-stage 12-to-1 multiplexer is implemented in two stages: three 4-to-1 multiplexers and one 3-to-1 multiplexer, as shown in **Figure 5-3**. With this approach, the effective charging and discharging capacitance has been reduced significantly. **Figure 5-4** illustrates an example of accurate phase swapping with acceptable rise/fall time as well as the opposite scenario with unacceptable rise/fall

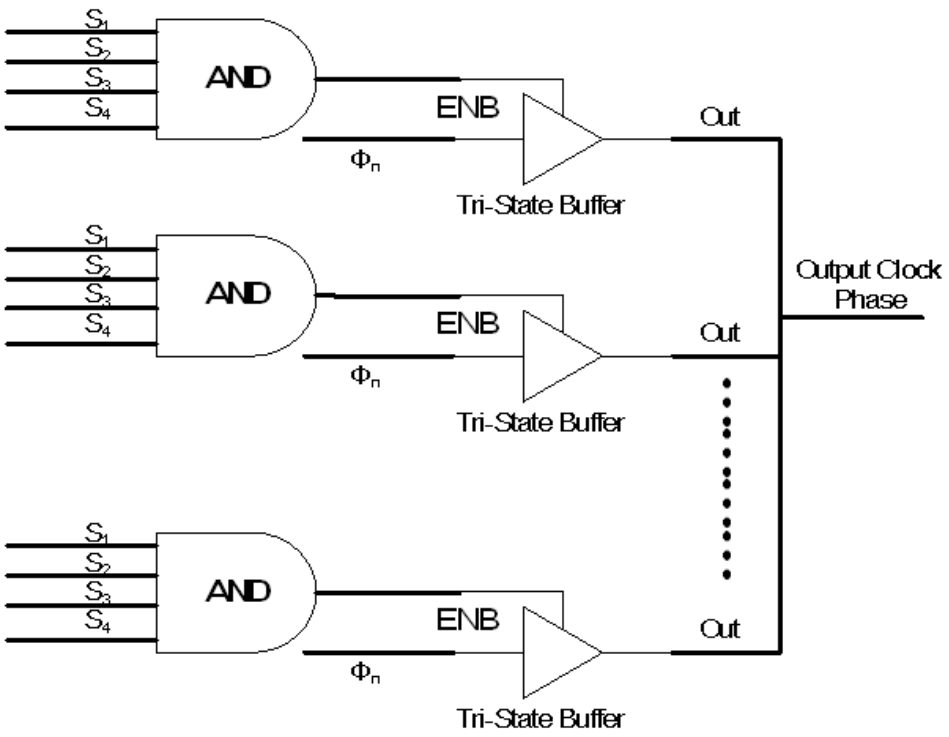

Figure 5.2: One Stage 12-to-1 Multiplexer Design

time. In the left diagram, the output node of the multiplexer has been fully discharged, and hence resulted in the correct duty cycle value after buffering. In the right diagram, the output node of the multiplexer has only been charged up to 1.08V instead of the full 1.2V. Hence, the desired duty cycle value is not reached even though buffering makes the signal appeared to have an altered duty cycle.

Figure 5-5 illustrates the actual "hot swapping" mechanism between different clock phases. Before the activation of the "Enable" signal, the final output clock phase (ϕ_{out}) resembles the waveform of the initial selected phase ($\phi_{initial}$). When the "Enable" signal becomes true, the final output clock phase will now follow the waveform of the newly selected clock phase (ϕ_{final}). Since the rising and falling edge of ϕ_{out} comes from $\phi_{initial}$ and ϕ_{final} respectively, the effective positive duty cycle of ϕ_{out} has been changed. For the changing in the negative duty cycle of ϕ_{out}, the falling and rising edge of ϕ_{out} comes from

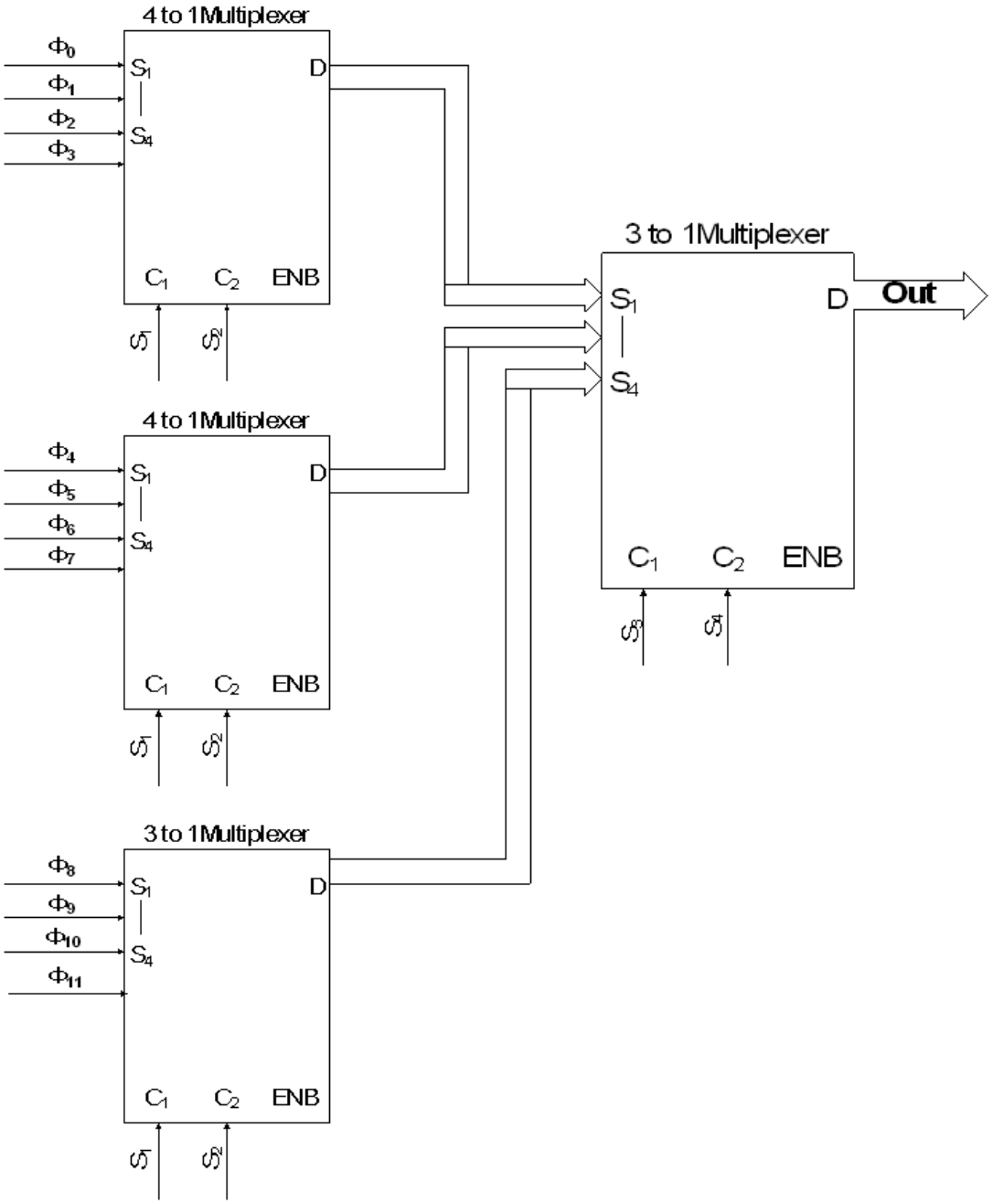

Figure 5.3: Two Stage 12-to-1 Multiplexer Design

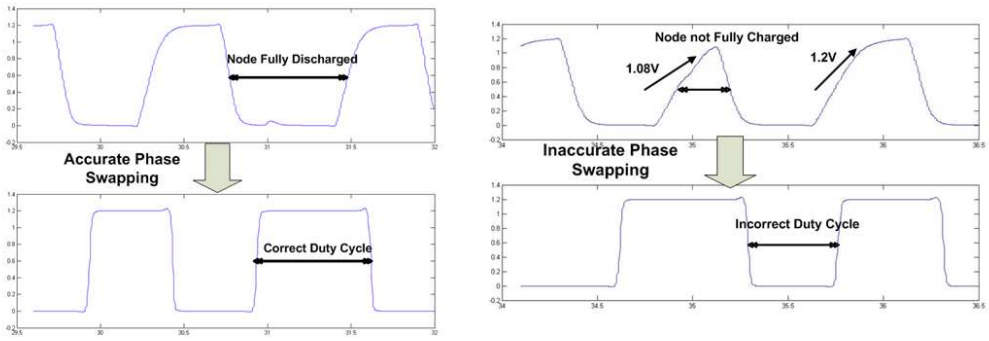

Figure 5.4: Effects of Rise/Fall Time on Correct and Accurate Phase Swapping

$\phi_{initial}$ and ϕ_{final} respectively. For Equation [5.1] to be valid and therefore establishing an accurate and predictable phase swapping, both $\phi_{initial}$ and ϕ_{final} must be at the same duty cycle when the "Enable" signal is activated. **Figure 5-6** illustrates an example when the "Enable" signal becomes true as $\phi_{initial}$ is at the negative duty cycle and ϕ_{final} is in the positive duty cycle phase. In this particular example, the falling edge of ϕ_{out} still follows that of $\phi_{initial}$. The rising edge, however, comes from the charging of the capacitance load at the output node of the multiplexer. Because the rising edge does not come directly from ϕ_{final}, the duty cycle of ϕ_{out} becomes inaccurate and unpredictable. In this thesis, the phase swapping illustrated in **Figure 5-5** is referred as a valid swap while the swap shown in **Figure 5-6** is invalid.

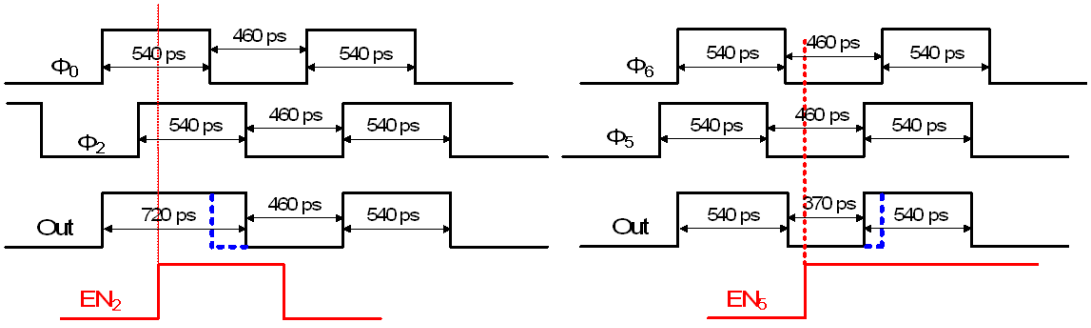

Figure 5.5: Examples of Valid Phase Swapping

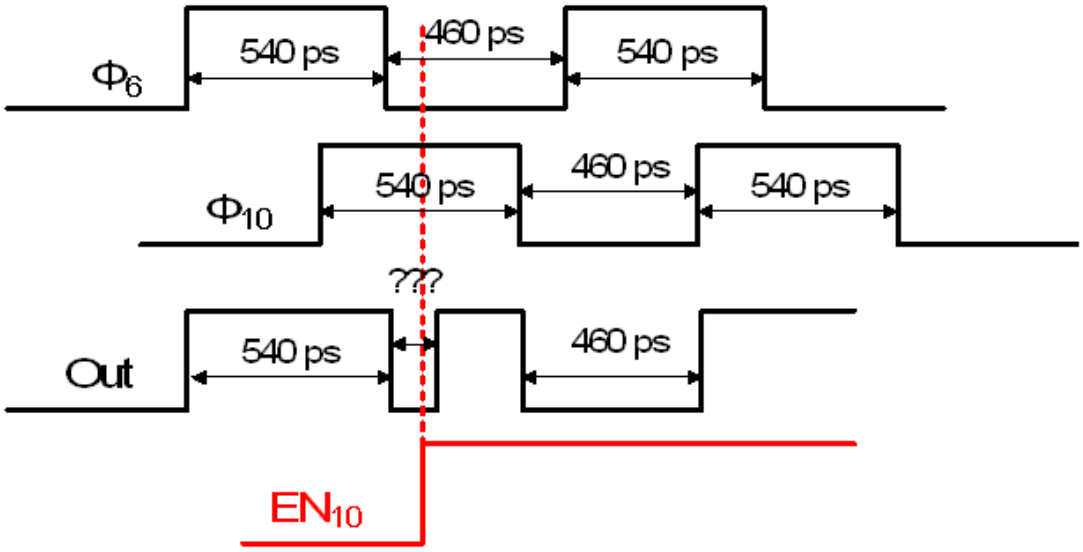

Figure 5.6: Example of Invalid Phase Swapping

In order to establish predictable swapping patterns, the "Enable" signal should have a fixed timing relationship with all the clock phases. Hence, a negative-edge triggered flip-flop is inserted between the AND gate and the tri-state buffer for each clock phase. The corresponding schematic diagram and waveform are illustrated in **Figure 5-7**. The clock signal used for the flip-flop is one of the clock phases coming out of the CDB. Thus, the timing relationship between each clock phase and the "Enable" signal is simply the sum of the propagation delay in the flip-flop and the phase difference between the flip-flop clock and each clock phase.

After the timing relationship is established, all cases of valid and invalid swapping can be determined based on the waveforms of each clock phase and the "Enable" signal. As mentioned previously, a valid swap only occurs when the "Enable" signal is activated such that both $\phi_{initial}$ and ϕ_{final} are on the same duty cycle. Furthermore, a valid swap can be divided into four different cases listed below in **Table 5-1**.

For all the cases listed above, the theoretical maximum number of phases (N) that can be swapped between two clock phases is given by Equation [5.2] where $\theta_{initial}$, $t_{rise/fall,1}$, and $t_{rise/fall,2}$ denote the initial duty cycle, the rise/fall time at the first multiplexer output, and the rise/fall time at the second multiplexer output respectively. Since the "hot swapping"

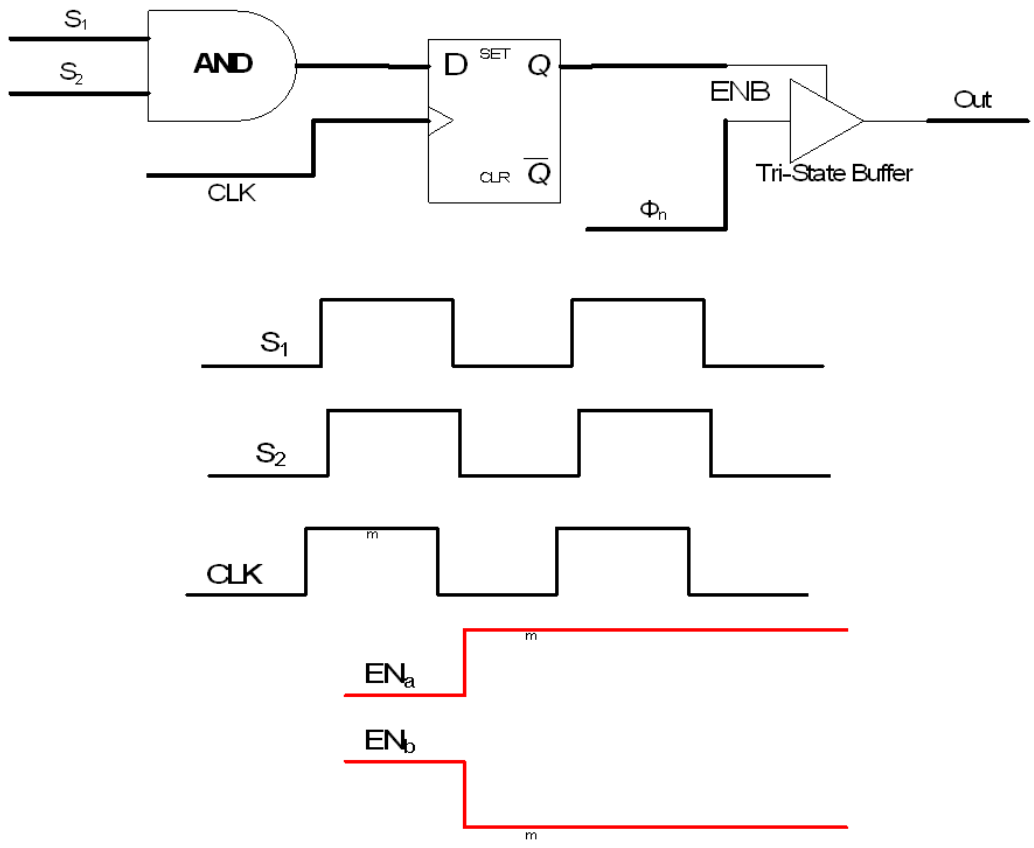

Figure 5.7: Timing Relationship of the "Enable" Signal

Table 5.1: Different Scenarios for Valid Phase Swapping

Case 1	Increasing the Positive Duty Cycle
Case 2	Decreasing the Positive Duty Cycle
Case 3	Increasing the Negative Duty Cycle
Case 4	Decreasing the Negative Duty Cycle

concept deals with precise timing mechanisms, the rise/fall time in Equation [5.2] will be referred to as 0%-100% instead of the normal 10%-90% criteria.

$$N = \frac{\theta_{initial} - t_{rise,1} - t_{fall,1} - t_{rise,2} - t_{fall,2}}{\phi} \tag{5.2}$$

Equation [5.2] illustrates an important criterion for achieving predictable and accurate phase swapping: *the initial and final clock edge must be given enough time to be fully charged and/or discharged at the output multiplexer node.* Even a slight deviation from this condition will result in a phase swap where the resultant duty cycle value is inaccurate. Equation [5.2] is only valid when the two clock phases to be swapped are coming from different multiplexers in the first stage (i.e. ϕ_0 to ϕ_5). Otherwise, it can be simplified into the form shown in Equation [5.3] where the rise/fall time of the second stage multiplexer is eliminated (i.e. ϕ_0 to ϕ_2).

$$N = \frac{\theta_{initial} - t_{rise,1} - t_{fall,1}}{\phi} \tag{5.3}$$

If both the initial and final clock phase are the inputs to the same multiplexer in the first stage, then the second stage multiplexer does not require any time to switch between clock phases because "hot swapping" is essentially performed in the first stage multiplexer. Consequently the corresponding load capacitance charging/discharging at the output node is eliminated.

From the equations given above, it is clear that N can be increased by increasing $\theta_{initial}$ or decreasing ϕ, $t_{rise/fall,1}$, or $t_{rise/fall,2}$. However, $\theta_{initial}$ is usually 50% of the clock period and does not change very often. Meanwhile, $t_{rise/fall,1}$ and $t_{rise/fall,2}$ are limited by the given technology and can only be optimized to a certain limit. Hence, the only realistic parameter that can be changed to vary N is the phase resolution ϕ, which can be accomplished by re-designing the delay element and the VCDL. It is important to realize that Equation [5.2]and Equation [5.3] only provide the theoretical maximum number of phase swapping; the actual swapping mechanism depends on the timing relationship between the "Enable" signal and each of the clock phases. Finally, by combining Equation [5.2] or Equation [5.3] with Equation [5.1], the effective duty cycle of the clock output after "hot swapping" is determined by three parameters: $\theta_{initial}$, ϕ, and N.

5.4 Results of "Hot Swapping"

Based on the design of the CDB given in Chapter 4, the "hot swapping" feature is implemented based on the parameters listed in **Table 5-2**. As stated previously, any clock phase with rising or falling edge coincides with the activation of the "Enable" signal should not be considered for the "hot swapping" process. Based on the waveforms from the CDB design, **Table 5-2** also summarizes the usage of each clock phase in the "hot swapping" mechanism.

Table 5.2: Parameters and Clock Phases in "Hot Swapping" Implementation

Initial Positive Duty Cycle	499ps
Initial Negative Duty Cycle	501ps
Phase Resolution	92ps
Rise Time at the Multiplexer Output	108ps
Fall Time at the Multiplexer Output	95ps
Clock Phases used to Manipulate Positive Duty Cycle	ϕ_2, ϕ_3, ϕ_4, ϕ_5
Clock Phases used to Manipulate Negative Duty Cycle	ϕ_7, ϕ_8, ϕ_9, ϕ_{10}
Clock Phases not Used for Hot Swapping	ϕ_0, ϕ_1, ϕ_6, ϕ_{11}

From Equation [5.2], it can be calculated that the maximum number of clock phases that can be correctly swapped if the input clock phases are coming from different first-stage multiplexer is *one*. By using Equation [5.3], however, a maximum of *three* phases can be correctly swapped for any of the cases listed in **Table 5-1**. Due to this reason, the inputs to the first stage multiplexer is arranged somewhat differently than the one shown previously in **Figure 5-3**. All the clock phases that are responsible for positive duty cycle manipulation are the inputs into the first multiplexer while those responsible for negative duty cycle manipulation are grouped into the second multiplexer, and the unused clock phases for hot swapping are inputted into the final multiplexer. With this approach, accurate duty cycles can be achieved with more confidence from phase swapping since more timing slacks are now provided.

Table 5-3 lists all possible valid cases of phase swapping and the resultant duty cycle.

For a 500ps nominal duty cycle, it is clear that both the positive and negative duty cycle can be increased up to approximately 786ps and reduced to a minimum of 216ps respectively. Theoretically, the duty cycle values should be identical for swapping the same number of phases. With cycle-to-cycle jitters, however, it is impossible to achieve the exact same duty cycle value calculated from Equation [5.1].

Table 5.3: Simulation Results of "Hot Swapping"

	# of Phase Swap	**V**alid Swapping Cases			**R**esultant Duty **Cycle**		
Case 1	1	$\phi_2 \rightarrow \phi_3$	$\phi_3 \rightarrow \phi_4$	$\phi_4 \rightarrow \phi_5$	591ps	592ps	592ps
	2	$\phi_2 \rightarrow \phi_4$	$\phi_3 \rightarrow \phi_5$		685ps	685ps	
	3	$\phi_2 \rightarrow \phi_5$			781ps		
Case 2	1	$\phi_5 \rightarrow \phi_4$	$\phi_4 \rightarrow \phi_3$	$\phi_3 \rightarrow \phi_2$	403ps	404ps	402ps
	2	$\phi_5 \rightarrow \phi_3$	$\phi_4 \rightarrow \phi_2$		309ps	310ps	
	3	$\phi_5 \rightarrow \phi_2$			216ps		
Case 3	1	$\phi_7 \rightarrow \phi_8$	$\phi_8 \rightarrow \phi_9$	$\phi_9 \rightarrow \phi_{10}$	591ps	591ps	594ps
	2	$\phi_7 \rightarrow \phi_9$	$\phi_8 \rightarrow \phi_{10}$		693ps	691ps	
	3	$\phi_7 \rightarrow \phi_{10}$			786ps		
Case 4	1	$\phi_{10} \rightarrow \phi_9$	$\phi_9 \rightarrow \phi_8$	$\phi_8 \rightarrow \phi_7$	405ps	406ps	405ps
	2	$\phi_{10} \rightarrow \phi_8$	$\phi_9 \rightarrow \phi_7$		314ps	314ps	
	3	$\phi_{10} \rightarrow \phi_7$			218ps		

Table 5-4 compares the theoretical and the average simulated duty cycle values for each swapping scenario based on a 92ps phase resolution, 499ps positive duty cycle, and 501ps negative duty cycle. For the most part, the simulated values match the theoretical values very well with a maximum percentage difference of only 3.14%. However, this small deviation still needs to be considered carefully during real testing since a precise duty cycle value is often required. **Figure 5-8** and **Figure 5-9** demonstrate the simulated waveforms for "hot swapping" on positive and negative clock edges respectively. In addition, the instances where "hot swapping" occurs are also indicated in the figures when the select signals are changed. **Figure 5-10** and **Figure 5-11** illustrate manipulation of the positive and negative duty cycles when compared to the original duty cycle value.

Table 5.4: Comparison between Theoretical and Simulated Phase Swapping Values

	# of Phase Swap	Theoretical Value	Average Simulated Value	% Difference
Case 1	1	591ps	591.67ps	0.11%
	2	683ps	685ps	0.29%
	3	775ps	781ps	0.77%
Case 2	1	407ps	403ps	0.98%
	2	315ps	309.5ps	1.75%
	3	223ps	216ps	3.14%
Case 3	1	593ps	592ps	0.17%
	2	685ps	692ps	1.02%
	3	777ps	786ps	1.16%
Case 4	1	409ps	405.3ps	0.90%
	2	317ps	314ps	0.95%
	3	225ps	218ps	3.11%

With the ability to change the duty cycle value for up to three phases based on the phase resolution of the CDB, any defects along a given digital path can be easily detected and analyzed. By shrinking the duty cycle values, for example, stress testing can be performed on certain logic blocks to determine the conditions in which failures will occur. Also, duty cycle can be expanded to detect the exact location of the logic failure along a path. Overall, the "hot swapping" feature of the CDB is very attractive in high-speed testing and debugging.

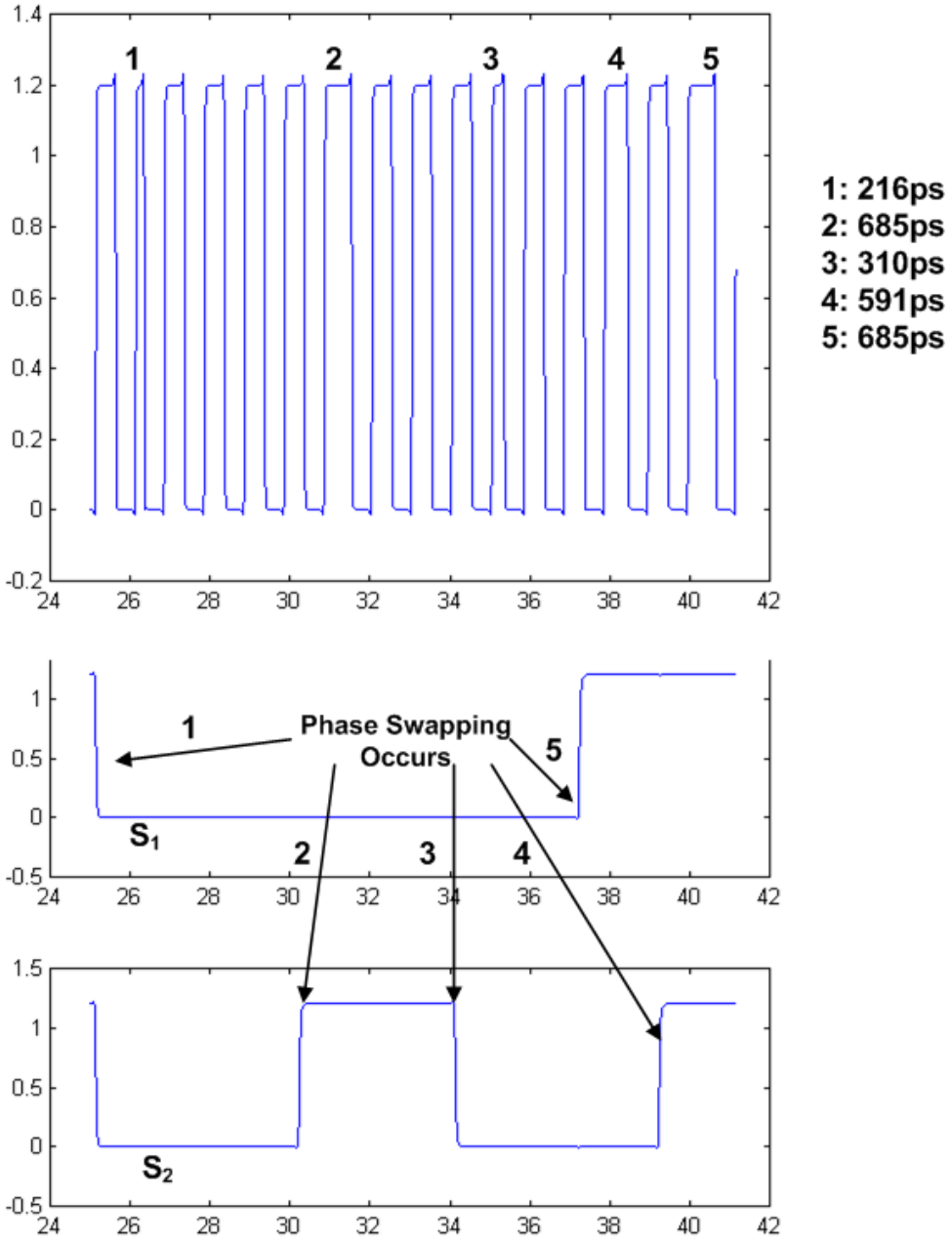

Figure 5.8: Simulated Waveforms for Positive Duty Cycle Phase Swapping

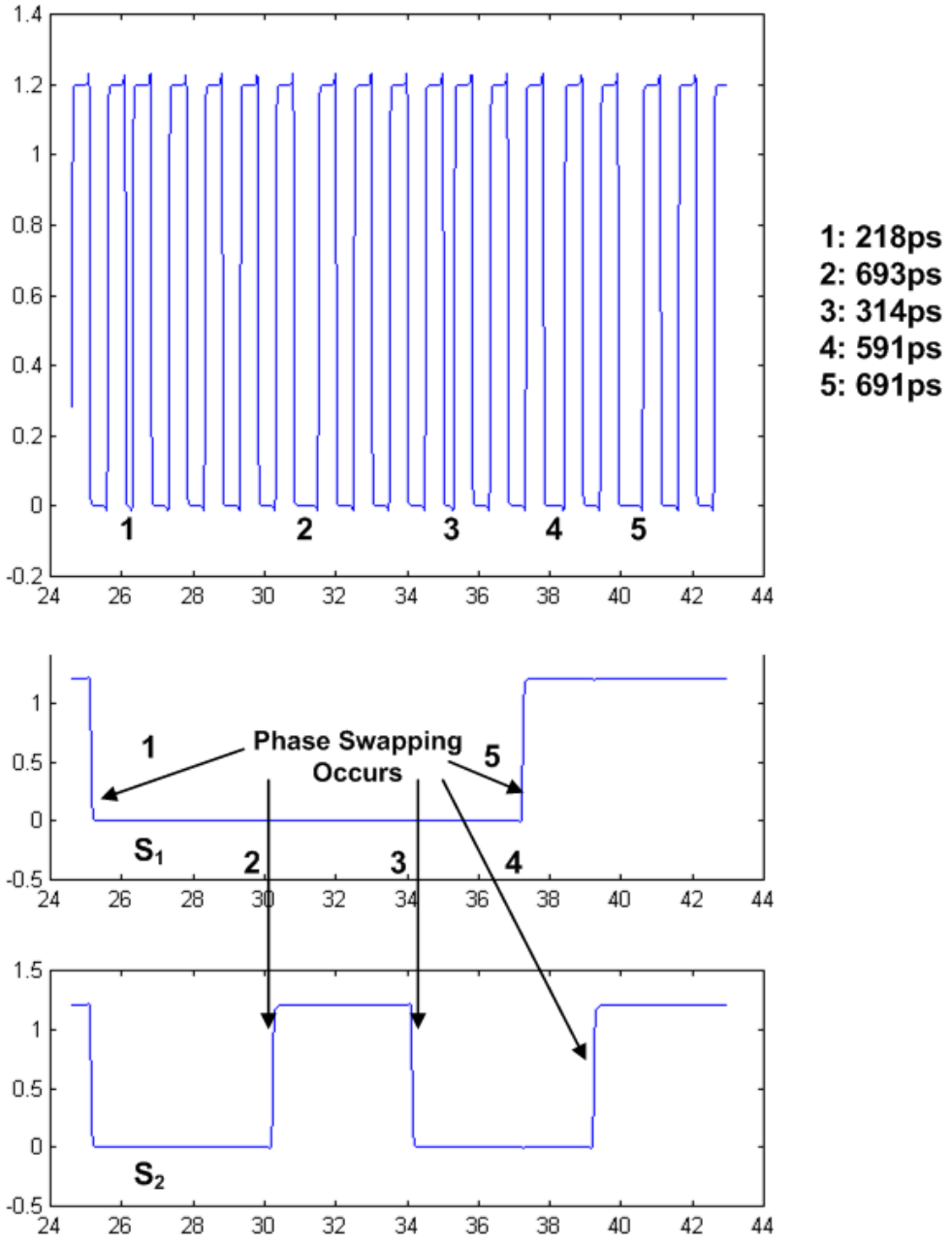

Figure 5.9: Simulated Waveforms for Negative Duty Cycle Phase Swapping

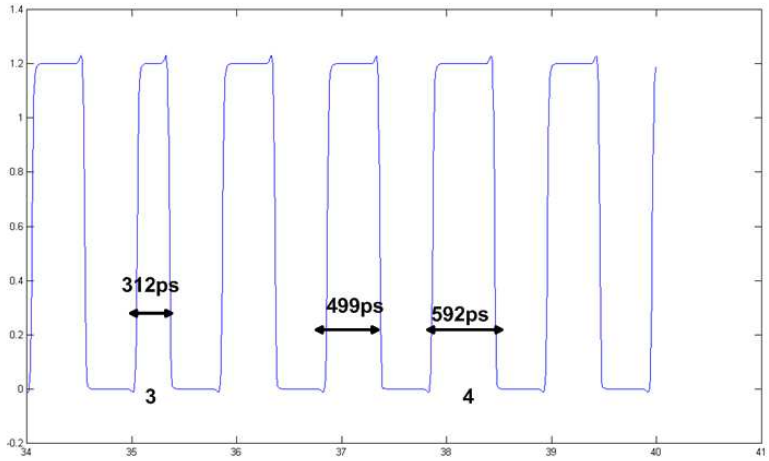

Figure 5.10: Resultant Changes in Positive Duty Cycle

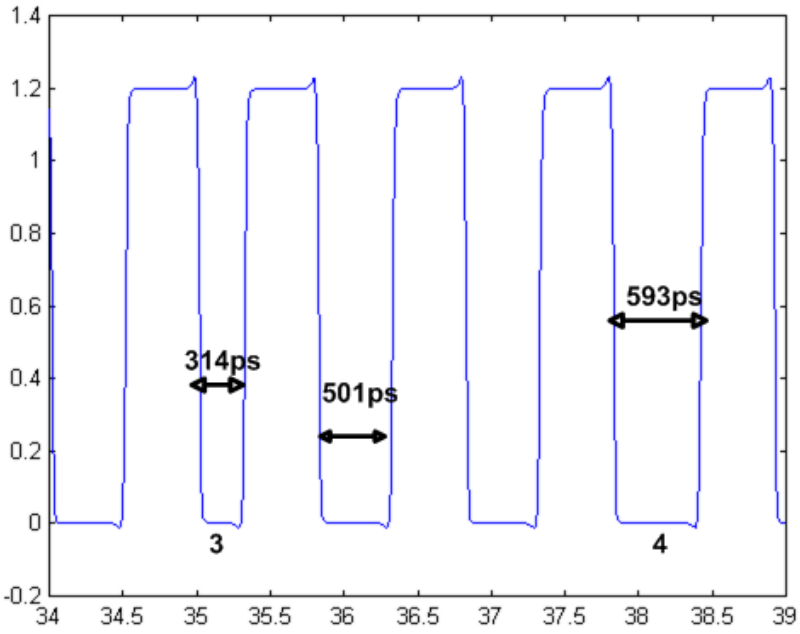

Figure 5.11: Resultant Changes in Negative Duty Cycle

Chapter 6

Application of the Crude Deskew Buffer

The main purpose of the CDB is to deskew the clock tree at the input source node to minimize the clock skew caused by process, voltage, and temperature (PVT) variations. In this section, the impact of PVT variations on clock skew is illustrated through a high-level analysis on a high-speed ASIC clock network. After that, detailed procedures of deskewing methodology and the corresponding simulation results will also be discussed. The entire deskewing methodology is simulated under the worst condition, as given in **Table 4-1**, using the crude deskew buffer designed in Chapter 4.

6.1 Clock Tree Investigation

For research purposes, a detailed high-level study on a realistic clock tree used in high-speed ASICs is performed. The clock tree is constructed using an H-tree type structure and consists of 18 levels of buffering; it has a similar structure as the clock tree diagram shown in **Figure 3-4**. After the last level of buffering, there are a total of 1881 clock signals driving appropriate loads such as registers. The input source clock is 1GHz with 50% duty cycle.

An HSpice simulation is performed on the clock tree netlist shown above, and the propagation delay from the input source node to each clock signal at the last level of

buffering is measured. The clock signal with the minimum delay will be used as the reference value, and the difference between this reference delay value and the propagation delay of each remaining clock signal will be referred to as the clock skew. The skew value of all the clock signals is plotted in **Figure 6-1**.

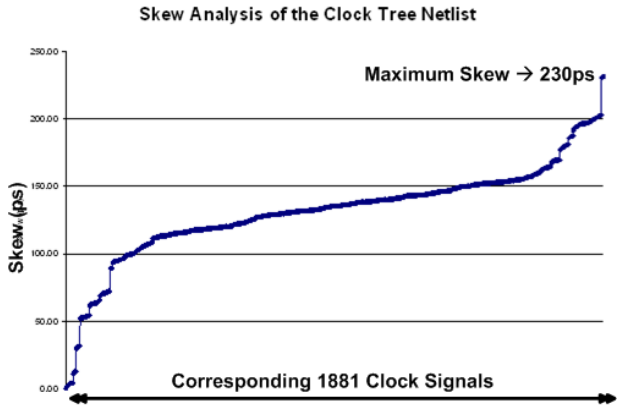

Figure 6.1: Skew Analysis for an ASIC Clock Network

The simulation performed in HSpice was done under the worst condition (Slow-Slow Corner, 110°C) with constant supply voltage and temperature as well as the absence of process variations. Yet, a maximum skew of 230ps still exists in the clock tree, as illustrated in the figure. Hence, interconnect variations after extraction and possible unbalanced loads of the clock signals are some of the factors that have resulted in the clock skew.

Under realistic ASIC environments, PVT variation can cause even identical H-trees to have different clock skew behaviour. Such behaviour is summarized in Equation [6.1] where δ, ΔP, ΔV, and ΔT represents the clock skew, process, voltage, and temperature variation respectively.

$$\delta = f(\Delta P, \Delta V, \Delta T) \tag{6.1}$$

To simply the simulation process, the PVT variation has been lumped into just one variation parameter: the supply voltage, as shown in Equation [6.2]. The variation of the supply voltage will have similar effect on the clock skew simulation as in the case stated in Equation [6.1] because propagation delay of the clock signals can be changed accordingly

depending on the amount of variation.

$$\delta = f(\Delta V) \tag{6.2}$$

Furthermore, two clock trees have been simulated to compare the skew behaviour as a result of supply voltage variation. The netlist of these two clock trees are exactly identical in terms input clock source and the number of buffering levels in an H-tree like structure, but one is simulated under nominal supply voltage (Vdd) while the other one is simulated with Vdd-Δ. This concept is shown in **Figure 6-2**.

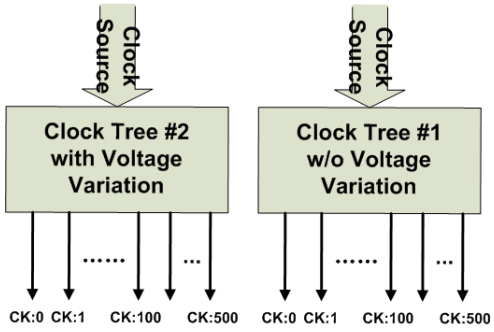

Figure 6.2: Initial Clock Network Simulation without Deskewing

By varying the supply voltage in the clock tree netlist such that Δ is 0.1V and -0.1V , the new and the original clock skew behaviour is plotted in **Figure 6-3**. It is evident that a variation in supply voltage will cause the skew curve to shift from the original position. A similar simulation is repeated for the same clock tree netlist where a different supply voltage variation is injected in each case. Essentially, the set of simulations performed is equivalent to the diagram shown in **Figure 6-4** where the same input clock source is driving seven different H-tree based clock networks. In terms of architecture, each clock network is identical to the netlist shown in **Figure 3-4**. To illustrate the effect of supply voltage variation on clock skew, however, each clock network is simulated under a different supply voltage. By injecting supply voltage variation into each clock network, it is equivalent of having each network experiencing different PVT variations during realistic ASIC environment. The corresponding simulation results of the clock skew behaviour for each clock network is illustrated in **Figure 6-5**.

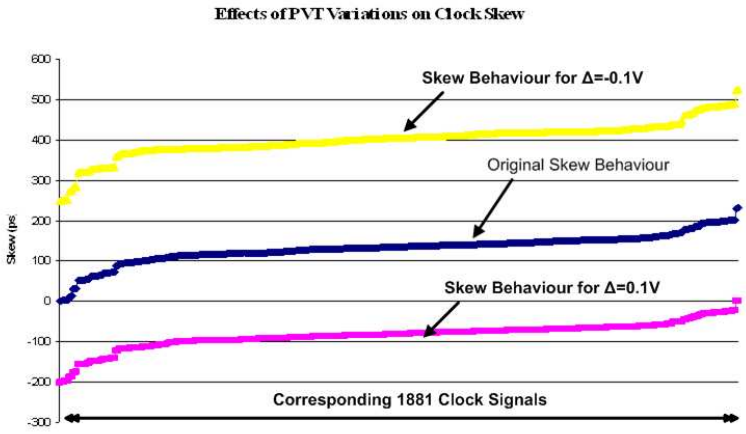

Figure 6.3: Clock Skew Behaviour under Supply Voltage Variation

Each line in **Figure 6-5** represents the skew behaviour for each of the seven clock networks simulated under supply voltage variation. All the skew values are calculated with respect to the reference value, which is derived from the clock signal with the absolute minimum propagation delay among all the clock networks. The line in the middle represents the skew for the clock network simulated with no supply voltage variation, and such clock network will be referred to as the reference network. As the supply voltage is varied for each clock network, the skew behaviour is shifted accordingly. The maximum positive and negative clock skew value from the reference value is 755ps and -202.1ps respectively. The absolute phase difference between these two clock signals is 957.1ps, which is almost a full clock period for a 1GHz input clock. Although these values are retained based on extreme supply voltage variation conditions, it does illustrate the potential harmful impact of PVT variation on clock skew.

6.2 Deskewing Methodology and Results

Ideally, the skew behaviour for each clock network displayed in **Figure 6-5** should all be aligned vertically as closely as possible despite variations in supply voltage. This can be accomplished by inserting the crude deskew buffer (CDB) to select the appropriate clock

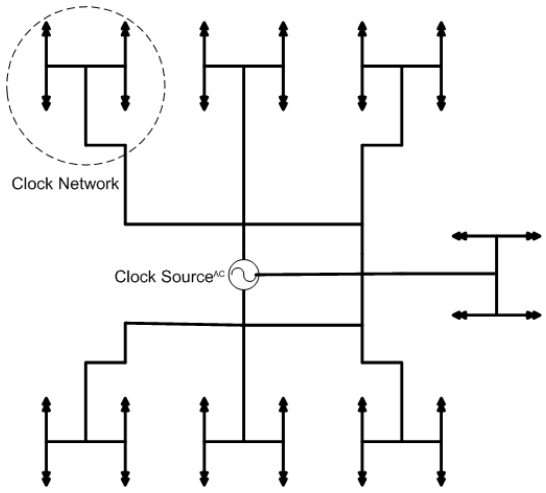

Figure 6.4: Complete Clock Network Simulation without Deskewing

phase to deskew at the input nodes of each clock network, as shown in **Figure 6-6**. Since the CDB produces different clock phases with a resolution of ϕ, one can examine the skew behaviour between each clock network and select the appropriate clock phases accordingly as the input clock in order to minimize the variation in skew behaviour.

With the usage of the crude deskew buffer, the same set of simulation performed in **Figure 6-4** is repeated for each clock network. However, instead of having a common input source clock driving each clock network, the crude deskew buffer selects the appropriate clock phases to drive different clock networks based on the skew behaviour shown in **Figure 6-5**. For instance, ϕ_5 is the input clock source to the reference clock network where the supply voltage is nominal. With its skew behaviour deviates the most from the reference network, ϕ_{10}, which is delayed by 460ps from ϕ_5, is used to drive the clock network with the greatest supply voltage variation.

The corresponding simulation results are shown in **Figure 6-7**. Despite different supply voltage variation in each clock network, the variation in skew behaviour has been reduced significantly, as evident by a much better vertical alignment of all the lines shown in the figure. **Table 6-1** shows the absolute maximum skew value of each clock network for both the case of using and not using the CDB. As mentioned previously, the maximum skew value in the reference clock network without supply voltage variation is 230ps. By injecting

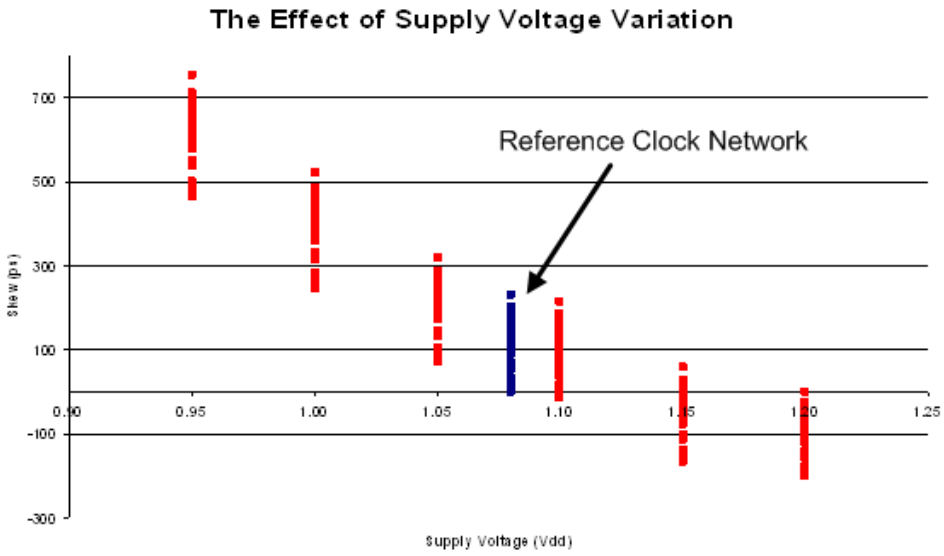

Figure 6.5: Simulation Results for Clock Skew without Deskewing

supply voltage variation into each clock network, however, the respective maximum clock skew has been increased significantly for most clock networks. With the usage of CDB, however, the maximum clock skew has been reduced drastically despite the presence of supply voltage variation. Furthermore, the maximum skew in each clock network has been brought much closer to the value in the reference clock network. **Figure 6-8** below shows the percentage reduction in absolute maximum clock skew for each clock network as well as the similarities achieved between each reduced maximum clock skew and the maximum clock skew in the reference clock network. All the values shown have been normalized with respect to the reference maximum clock skew value. Another way to interpret the results shown in **Figure 6-8** is that the usage of the CDB can also synchronize clock signals between different clock networks. Without CDB, the maximum phase difference between clock signals in all clock networks is 957.1ps. This difference is reduced to 311.9ps after the CDB is inserted to deskew the clock networks due to a maximum positive skew and negative skew of 248.7ps and -63.2ps respectively. By synchronizing all the clock signals in different clock networks such that they are all within a reasonable amount of phase

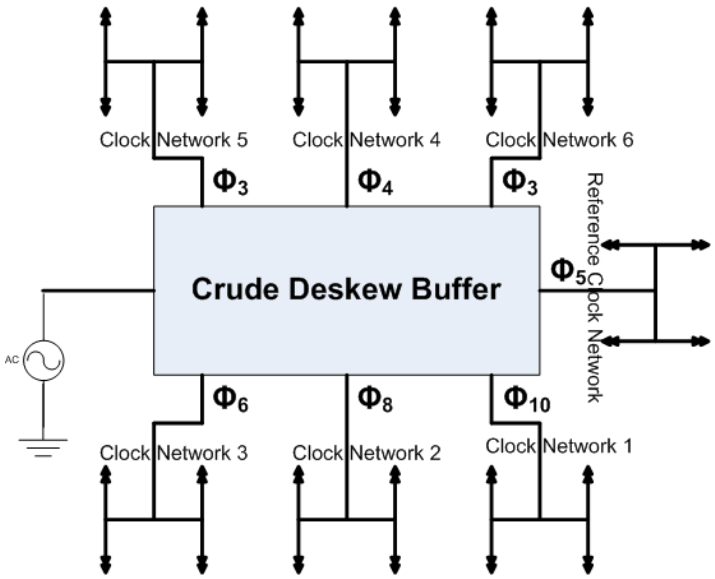

Figure 6.6: Complete Clock Network Simulation using CDB

difference of each other, the fine deskewing methodology can be implemented with more regular and predictable pattern.

Table 6.1: Reduction of Clock Skew using CDB

	Absolute Maximum Skew w/o CDB (ps)	Absolute Maximum Skew with CDB (ps)
Clock Network # 1	755	248.7
Clock Network # 2	524.4	238.3
Clock Network # 3	320	234.1
Clock Network # 4	215.1	187.4
Clock Network # 5	-167.2	151.6
Clock Network # 6	-202.1	184.4
Clock Network # 7 (Reference)	231	231

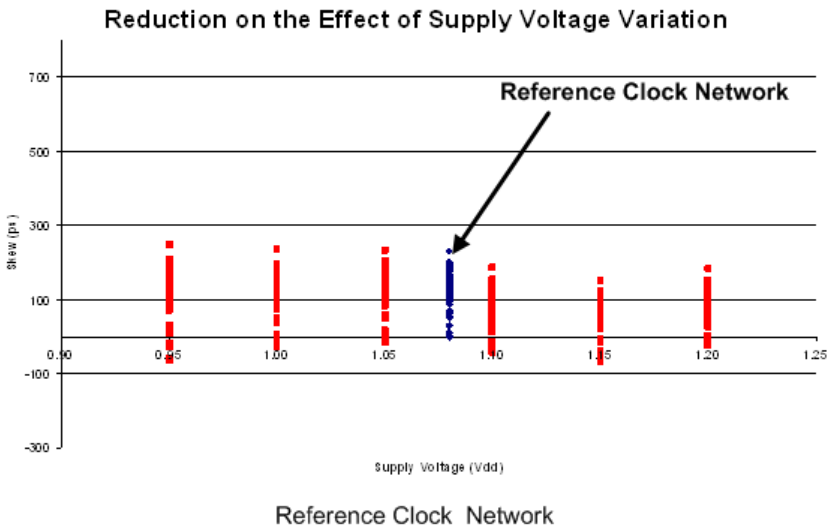

Figure 6.7: Simulation Results for Clock Skew using CDB

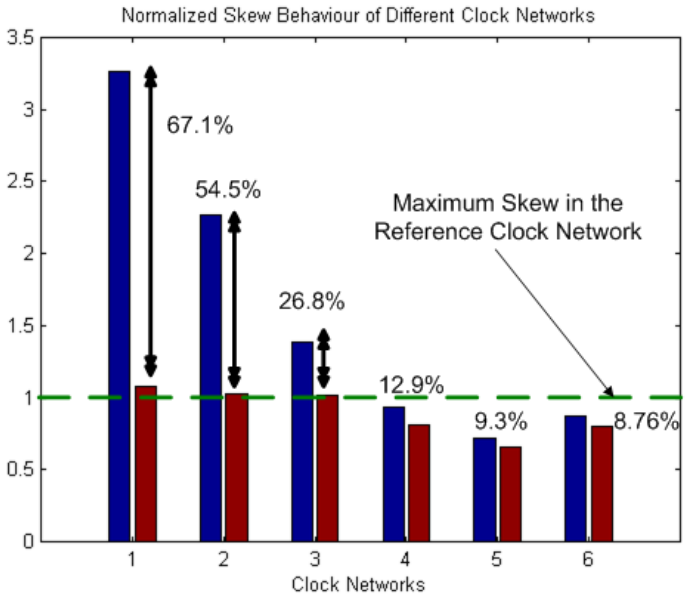

Figure 6.8: Synchronization of Clock Signals and Reduction in Clock Skew using CDB

Chapter 7

Concluding Remarks

7.1 Conclusions

As the CMOS technology continues to scale rapidly, clock skew is quickly becoming a prominent factor in the design of high performance microprocessor. While traditional methods such as adopting an H-tree structure are able to reduce the systematic component of clock skew, process and environmental variations have caused random skew a significant factor in the total clock skew. In this thesis, an active deskewing methodology is proposed where a crude deskew buffer is successfully designed and simulated in 0.13μm CMOS technology. The simulation conditions include all process corners with various temperature conditions. The following highlights some of the key points mentioned in each chapter.

- A crude deskew buffer (CDB) is designed based on the delay-locked loop (DLL) architecture. The frequency operation of the CDB is from 800MHz to 1.2GHz with a nominal clock frequency at 1GHz. Including the original clock phase, a total of 12 clock phases are generated with a phase resolution ranging from 89ps to 92ps depending on the input clock frequency.

- In high speed ASIC testing, the variable clock method is often adopted where constant changing in clock duty cycle is required. A "hot swapping" feature is implemented in the CDB where the rising and falling edges of the clock signal are manipulated such that the effective clock duty cycle is changed based on the phase resolution

value. For a 500ps duty cycle, both the positive and negative duty cycle values can be increased or decreased up to a maximum of three phase swaps. The theoretical corresponding duty cycle values are 224ps, 316ps, 408ps, 592ps, 684ps, and 776ps. The simulated values are very similar to the theoretical values with a maximum percentage difference of 3.14%.

- The CDB is applied to seven high-speed ASIC clock networks each with different supply voltage variation. A maximum of 67.1% reduction in absolute maximum skew value can be achieved by using the appropriate clock phase from the CDB to deskew the clock network. Furthermore, the maximum phase difference between all the clock signals in all seven clock networks has been reduced from 957.1ps to 311.9ps, a reduction of 67.4%. Overall, the CDB serves two important purposes in the proposed deskewing methodology: reducing the absolute maximum clock skew and synchronizes all the clock signals to a certain limit for the fine deskewing scheme.

7.2 Future Work

Although the usage of CDB reduces the effects of PVT variations, a clock skew of approximately 230ps still exists within the clock network due to factors such as unbalanced loads and interconnect variations. Hence, more adaptive deskewing is required to further minimize the clock skew. For example, a low-level analysis on the clock network is required in order to determine the appropriate locations to insert the fine deskew buffers. With careful placements, the clock skew can be reduced to an absolute minimum. Finally, a stable start-up circuit needs to be designed in order to provide an accurate initial control voltage for the DLL to lock correctly.

Bibliography

[1] Jan M. Rabaey, A. Chandrakasan, and B. Nikolic *"Digital Integrated Circuits: A Design Perspective"*. Prentice Hall, 2nd Edition, 2003.

[2] S. Tam, S. Rusu, U. Desai, R. Kim, J. Zhang and I. Young "Clock Generation and Distribution for the First IA-64 Microprocessor". *IEEE Journal of Solid-State Circuits*, Vol. 35(11): pp.1545-1552, November 2000.

[3] Wikipedia "Crystal Oscillator". *http://en.wikipedia.org/wiki/Crystal_oscillator*, 2007.

[4] N. Kurd, J. Barkatullah, R. Dizon, T. Fletcher, and P. Madland "A Multigigahertz Clocking Scheme for the Pentium© 4 Microprocessor". *IEEE Journal of Solid-State Circuits*, Vol. 36(11): pp.1647-1653, November 2001.

[5] S. Kio, K. Chong, and C. Sechen "A Low Power Delayed-Clocks Generation and Distribution System". *Proceedings of the 2003 International Symposium on Circuits and Systems*, Vol. 5: pp. 445-448, May 2003.

[6] P. J Restle and A. Deutsch "Designing the Best Clock Distribution Network". *1998 Symposium on VLSI Circuits*, Digest of Technical Papers, pp. 2-5, June 1998.

[7] P. J Restle, A. Jenkins, A. Deutsch and P.W. Cook "Measurement and Modeling of On-Chip Transmission Line Effects in a 400MHz Microprocessor". *IEEE Journal of Solid-State Circuits*, Vol. 33(4): pp.662-665, April 1998.

[8] D. W. Bailey and B. J. Benschneider "Clocking Design and Analysis for a 600-MHz Alpha Microprocessor". *IEEE Journal of Solid-State Circuits*, Vol. 33(11): pp.1627-1633, November 1998.

[9] G. Geannopoulos and X. Dai "An Adaptive Digital Deskewing Circuit of Clock Distribution Networks". *45th ISSCC IEEE International Solid-States Circuit Conference*, Digest of Technical Papers, pp. 400-401, February 1998.

[10] N. Weste and D. Harris *"CMOS VLSI Design: A Circuit and Systems Perspective".* Addison Wesley, 3rd Edition, 2005.

[11] P. Zarkesh-Ha and J. D. Meindl "Optimum Chip Clock Distribution Networks". *45th IEEE International Conference Interconnect Technology*, pp. 18-20, May 1999.

[12] A. Chandrakasan, W. Bowhill, F. Fox *"Design of High-Performance Microprocessor Circuits".* IEEE Press, 2000.

[13] M. Zhao, K. Gala, V. Zolotov, Y. Fu, R. Panda, R. Ramkumar and B. Agrawal "Worst Case Clock Skew Under Power Supply Variation". *Motorola Inc, Austin TX, USA, Motorola India Design Center, Noida, India, White Paper*, 2002.

[14] International Technology Roadmap for Semiconductors "International Technology Roadmap for Semiconductors 2005 Edition: Executive Summary". *http://www.itrs.net/Links/2005ITRS/ExecSum2005.pdf*, 2005.

[15] S. R. Nassif "Design for Variablity in DSM Technologies". *Proceedings of IEEE 2000 First International Symposium on Quality Electronic Design*, pp. 451-454, March 2000.

[16] V. Mehrotra and D. Boning "Technology Scaling Impact of Variation on Clock Skew and Interconnect Delay". *Proceedings of IEEE 2001 International Interconnect Technology Conference*, pp. 122-124, June 2001.

[17] I. Chanodia and D. Velenis "Effects of Parameter Variations and Crosstalk on H-Tree Clock Distribution Networks". *48th Midwest Symposium on Circuits and Systems* , Vol. 1: pp. 547-550, August 2005.

[18] D. Harris and S. Naffziger "Statistical Clock Skew Modeling with Data Delay Variations". *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol. 9(6): pp.888-898, December, 2001.

[19] N. R. Mahapatra, S. V. Garimella and A. Tareen "An Empirical and Analytical Comparison of Delay Elements and a New Delay Element Design". *Proceedings of IEEE 2000 Computer Society Workshop on VLSI*, Vol. 1: pp. 81-86, April 2000.

[20] Y. W. Pang, W. Y. Sit, C. S. Choy, C. F. Chan and W. K. Cham "An Asynchronous Cell Library for Self-Timed System Designs". *IEICE Transactions on Information and Systems*, Vol E80-D: pp. 296-305, March 1997.

[21] Y. Moon, J. Choi, K. Lee, D. K. Jeong and M. K. Kim "An All-Analog Multiphase Delay-Locked Loop Using a Replica Delay Line for Wide-Range Operation and Low-Jitter Performance". *IEEE Journal of Solid-State Circuits*, Vol. 35(3): pp.377-384, March 2000.

[22] H. H. Chang, J. W. Lin, C. Y. Yang and S. I. Liu "A Wide-Range Delay-Locked Loop with a Fixed Latency of One Clock Cycle". *IEEE Journal of Solid-State Circuits*, Vol. 37(8): pp.1021-1027, August 2002.

[23] D.J. Foley and M. P. Flynn "CMOS DLL-Based 2-V 3.2-ps Jitter 1-GHz Clock Synthesizer and Temperature-Compensated Tunable Oscillator". *IEEE Journal of Solid-State Circuits*, Vol. 36(3): pp.417-423, March 2001.

[24] J. Cheng *A Delay-Locked Loop for Multiple Clock Phases/Delays Generation*. PhD Thesis, Georgia Institute of Technology, December, 2005.

[25] W. Rhee "Design of High-Performance CMOS Charge Pumps in Phase-Locked Loops". *Proceedings of the 1999 IEEE International Symposium on Circuits and Systems*, Vol. 2: pp. 545-548, June 1999.

[26] Wikipedia "Hot Swapping". *http://en.wikipedia.org/wiki/Hot_swapping*, 2007.

[27] S. Natarajan, M.A. Breuer and S. K. Gupta "Process Variations and Their Impact on Circuit Operation". *Proceedings of the 1998 IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems*, pp. 73-81, 1998.

[28] R. Wilson "Delay-Fault Testing Mandatory, Author Claims". *EE Design*, December 2002.

[29] K. Cheng "Transition Fault Testing for Sequential Circuits". *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 12(12): pp. 1971-1983, December 1993.

[30] T. M. Mak, A. Krstic, K. Cheng and L. Wang "New Challenges in Delay Testing of Nanometer, Multigigahertz Designs". *IEEE Design and Test of Computers*, pp. 241-248, May-June 2004.

[31] G. Aldrich and B. Cory "Improving Test Quality and Reducing Escapes". *Proceedings of Fabless Forum, Fabless Semiconductor Association*, pp. 34-35, 2003.

[32] X. Lin, R. Press, J. Rajski, P. Reuter, T. Rinderknecht, B. Swanson and N. Tamarapalli "High-Frequency, At-Speed Scan Testing". *IEEE Design and Test of Computers*, pp. 17-25, September-October 2003.

[33] P. Gillis, K. McCauley, F. Woytowich and A. Ferko "Low Overhead Delay Testing of ASICs". *Proceedings of 2004 IEEE International Test Conference*, pp. 534-542, 2004.

[34] J. Saxena, K. Butler, et. al. "Scan Based Transition Fault Testing - Implementation and Low Cost Chanllenges". *Proceedings of 2002 IEEE International Test Conference*, pp. 1120-1129, 2002.