A Successive Approximation Register based Digital Delay Locked Loop for Clock and Data Recovery Circuits

by

Priyanka Mittal

A thesis submitted in partial fulfillment for the
degree of Master of Technology

under supervision of
Dr. Pydi Ganga Mamba
Department of Electronics and Communication Engineering
Indraprastha Institute of Information Technology, Delhi

July 2018

A Successive Approximation Register based Digital Delay Locked Loop for Clock and Data Recovery Circuits

by

Priyanka Mittal

A thesis submitted in partial fulfillment for the
degree of Master of Technology

to

Indraprastha Institute of Information Technology, Delhi

July 2018

Certificate

This is to certify that the thesis titled "A Successive Approximation Register based Digital Delay Locked Loop for Clock and Data Recovery Circuits" being submitted by Priyanka Mittal (Roll No.- MT16103) to the Indraprastha Institute of Information Technology Delhi, for the award of the Master of Technology, is an original work carried out by her under my supervision. In my opinion, thesis has reached the standards fulfilling the requirements of the regulations relating to the degree. The results contained in the thesis have not been submitted in part or full to any other university or institute for the award of any degree/diploma.

Date: _____

Dr. Pydi Ganga Mamba
Associate Professor
Department of Electronics and Communication Engineering
Indraprastha Institute of Information Technology, Delhi
New Delhi 110020

Mr. Bhavin Odedara
Sr. Technologist
ASIC DEV Engineering
SanDisk, a Western Digital Brand
Bengaluru 560103

Abstract

The delay locked loop (DLL) is widely used in the electronics industry for implementing clock and data recovery circuits (CDR) in high-speed IOs. DLL contains first-order closed-loop architecture, aligns the output clock to the reference clock, and reduces skew between two clocks across variations in process, voltage and temperature (PVT) by the help of the delay line. This circuit is always stable as it is a single pole system.

DLLs can be broadly classified into two categories: analog DLL and digital DLL. The analog DLL has an analog controlling input to control the delay offered by the delay lines to reduce the skew between the input and the output clock of the DLL system. Digital DLLs on the other hand, have quantized steps for delay change in the delay line. This delay line is controlled by a digital code obtained from the controller. The Digital DLLs can easily adopt to technology changes as they do not have strict voltage headroom requirements like analog DLLs. Also, mostly standard cells are used to design Digital DLL which makes them portable. These characteristics make Digital DLLs an attractive choice for implementing clock and data recovery circuits for very advanced technologies.

Lock time is one of the important parameter while designing DLL. It is decided by the type of controlling mechanism used in implementing DLL. In order to reduce the lock time, in this thesis work, the controller is implemented by Successive Approximation Register which reduces the lock time for proposed DLL by using binary search algorithm.

To track the PVT variations, the SAR used has been modified in such a manner that it uses binary search algorithm for locking the DLL and then turns into a counter, to track the PVT variations after locking. This DLL is designed in TSMC16nm FinFET technology. This technology has its own limitations with respect to the delay offered by the inverters and the amount of current an inverter can support. In order to mitigate the current consumption and hence power consumption, a timing controller has been proposed which helps the delay line achieving a resolution as small as 5ps for a frequency of 400 MHz whereas the state-of-art study shows that for lower frequencies, a delay resolution of at least 10ps has been reported. This circuit has a power consumption of only 240μW across corners whereas the state-of-the work has reported a power consumption in order of mW. The phase accuracy of the designed Digital DLL is 99.5% across corners in locked condition.

Acknowledgements

I would like to thank my supervisors Dr. Pydi Ganga Mamba and Mr. Bhavin Odedara for their guidance and corrections made in the scope of this work. Without their valuable suggestions and assistance, this thesis would not have been possible.

This work is dedicated to my parents Mr. Sunil Mittal and Mrs. Anju Mittal for their teachings and values passed on to me. I thank their understanding even in those moments in which my immaturity prevailed. To my fiance Tejas Tilak and brother Deepanshu Mittal for all their support and friendship throughout these years.

I would like to thank all the team members of the ASIC Analog team at SanDisk for their help on obtaining this works final results. Special thanks to Srikanth Bojja and Akshaya Bhardwaj for their enthusiastic discussions. Performing this work among these outstanding professionals contributed a lot to my evolution as a professional and a human being.

I would also like to thank my friends Vaibhav Agarwal and Tejaswini K Rao for their suggestions and support.

Contents

List of Figures

List of Tables

Dedicated to my beloved parents...

Chapter 1

Introduction

Nowadays, the transportation of real-time audios and videos over the internet has led to an increasing demand for high-speed serial data communication. To support a high bandwidth, the transmission medium has now converted from a simple copper wire medium to optical fiber medium such as Synchronous Optical Network (SONET)/Gigabit Ethernet network and chip-to-chip interfaces such as PCI-Express (PCIe), Serial ATA (SATA) etc. This has motivated the demand for low-cost, low-power and high-speed serializer and de-serializer (SerDes) integrated receiver circuits.

In high-speed serial data transmission systems, the data is often transmitted without any additional timing reference information. However, the receiver should process the data synchronously. Therefore, it is a critical task for receivers to extract the timing information from its incoming serial data stream such that it can sample the data on the serial lines synchronously. This is known as clock recovery. By modifying the transmitted data, the clock information recovery from data stream can be expedited. In order to recover the sampling clock, the receiver uses a reference frequency signal. For generating the recovered clock, the receiver needs to align the phases of the of the sampling clock with the transitions (either positive edges or negative edges) of the incoming serial data stream. The input serial data stream is then sampled with respect to the recovered clock and a bit stream is generated. This is known as data recovery. Together, this is known as clock and data recovery (CDR)[6].

Figure 1.1 shows an example of an optical communication system. In an optical communication system, the parallel incoming electrical data is converted to serial stream of bits by the help of the serializer. This data then converted into optical data by the help of laser diode or other medium, which is then transmitted through an optical fiber channel and received by the optical receiver. A transconductance amplifier at the receiver side converts this optical stream of bits to electrical stream of bits. This information is then

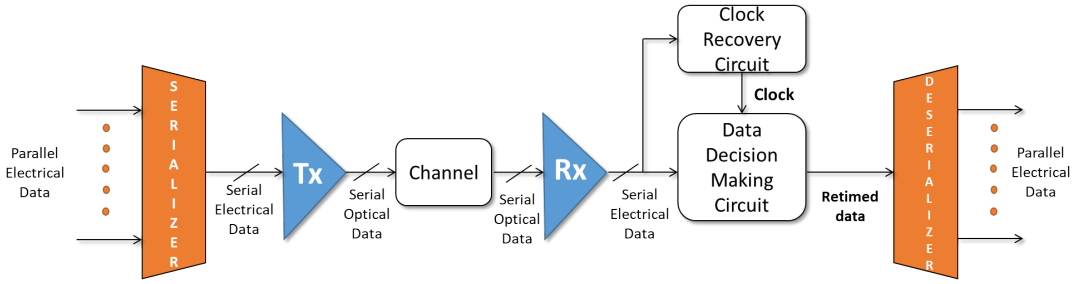


FIGURE 1.1: An optical communication system having clock and data recovery circuit

used by the clock extraction circuit to recover the sampling clock. The data decision making circuit uses this sampling clock and produces retimed data which is sent to de-serializer and converted to parallel bits. Thus, clock and data recovery circuit as shown in fig 1.1 is an important part of high-speed serial data communication system.

To perform this process various methods like phase-locked loop(PLL), delay locked loop (DLL) and oversampling of the data stream etc can be used. Clock data recovery is one of the key design components of the high-speed integrated circuits [7]. In the recent works, the CDR circuits are implemented mostly by PLLs, the analog DLLs or a combination of PLL and analog DLL. In this work, a CDR based on digital DLL has been proposed in TSMC 16nm FinFET technology. For a receiver circuit, a circuit having very low power consumption is desirable. The state-of-the art study shows that digital DLLs implemented in [3], [2], [8], [9] etc. have a power consumption in order of mW, whereas, the proposed DLL in this thesis work has a power consumption in order of hundreds of μW. The recovered clock should have a low jitter possible, as any jitter value present in extracted clock will be passed to retimed data which is obtained by sampling the bit stream at the positive or the negative edge of sampling clock. Therefore, a DLL should have small jitter value. The jitter value obtained in this thesis work is comparable to the jitter values obtained from the state-of-the-work.

1.1 Motivation

The continuous scaling of the Complimentary MOSFET (CMOS) process technologies has aided to the ability to design highly integrated receiver circuits with low cost and low power consumption. In order to have a high-level of integration, the jitter obtained by the power supply variation or noise in power supply should reduce. This makes PLLs a little less attractive here to be used as clock and data recovery circuits because PLLs have an inherent disadvantage of jitter accumulation due to the voltage controlled oscillator (VCO), which keeps on accumulating the jitter in the signal, till it reaches to a sufficient value, where it could be detect by the phase detectors easily. Here, DLLs have

an advantage since they have voltage controlled delay line (VCDL) which does not have any problem of jitter accumulation. DLLs are inherently stable (sine they are first order systems). So, the advantages of DLLs like stability and no jitter accumulation make DLLs an attractive choice for clock and data recovery circuits as compared to PLLs.

When we talk about low-power designs, the voltage supply (V_{DD}) reduces. This means available voltage headroom also decreases, making it difficult for analog and mixed-mode circuits to operate correctly in all possible situations as the voltage headroom requirement is different for different processes. However, this could be an advantage for the digital circuits as they don't have to take care of sufficient voltage headroom across devices. If there is sufficient noise margin available, the digital devices are good to go. Also, with the scaling of CMOS technologies, portable and flexible module and macro cells have become more attractive. The digital circuits are more portable and flexible as compared to analog circuits. These above mentioned points favor the development of digital DLL as compared to analog DLL for lower technology nodes for clock and data recovery circuits.

1.2 Objectives

The objective of this thesis is to develop a digital delay locked loop for clock and data recovery applications. The proposed DLL is to be designed in TSMC 16nm FinFET technology. This DLL should have a lock time less than ($< 1us$), the output clock duty cycle should be 50% across corners, a fine delay resolution value ($< 10ps$) and a low rms jitter value ($< 10ps$). The proposed circuit should work on 0.8V supporting all process, voltage and temperature variations.

1.3 Structure of the Work

This thesis is divided in 5 chapters to reflect the necessary information that lead up to the complete design of a delay locked loop system.

- **Chapter 2: Overview Of DLL**
  This chapter gives the description about the functioning of generic DLL, types of DLL, different configurations to realize the DLL and a short comparison between analog and digital DLL. This chapters explains how digital DLLs are a better choice than analog DLLs

- **Chapter 3: System Design**

  This chapter discusses about the proposed delay locked loop design, detailed analysis of each block that has been used and challenges faced while implementing them. This section also describes how the proposed timing controller helps in achieving a small delay resolution value for a frequency input of 400 MHz where in the state-of-the work, for such a low frequency, a delay resolution of atleast 10ps has reported.

- **Chapter 4: Results**

  This chapter discusses about the simulated findings of the proposed DLL. The values for important parameters like lock time, delay resolution, phase noise etc. are discussed.

- **Conclusions**

  This chapter concludes all the findings and contributions of this work and discusses about possible future work.

Chapter 2

Overview Of DLL

Phase locked Loops (PLLs) and Delay Locked Loops (DLLs) have been widely used in on-chip clock timing functions as clock generators and clock de-skew buffers, in DDR-SDRAM interfaces, high-speed I/O interfaces, clock and data recovery and application-specific integrated circuits. Both PLLs and DLLs are nonlinear negative feedback systems, i.e. compare the output phase with the input phase and have similar circuits except the Voltage Controlled Oscillators (VCOs) used by PLLs are replaced by voltage Controlled Delay Lines (VCDLs) in DLLs. PLLs are attractive to be used as clock multipliers as it can support a programmable multiplication rate and hence provide easy means of frequency multiplication. But under identical noise and circuit conditions, PLLs have more phase accumulation and hence show a high value of jitter as compared to a DLL. In a PLL, a step variation in voltage controlled oscillator is integrated over many cycles, till the loop filter can detect it. This results in a phase error larger than the original phase error (order of three to four times), whereas in a DLL similar step variation in delay line creates a constant offset. A detailed analysis on comparison of jitter behavior of PLL and DLL can be found in these references. Besides no jitter accumulation, a DLL is mostly a single-pole system (always stable), does not rely on a high loop bandwidth to correct for jitter and exhibits negligible jitter peaking. Hence, A DLL based CDR topology is preferable for a high speed synchronous links than a PLL based CDR topology because of the stability and no jitter accumulation property.

2.1 DLL Operation Principle

DLLs are negative feedback systems which align the output (feedback) clock to the reference clock. Figure 2.1 shows a generic DLL showing its basic operation principle. The input reference clock is passed through a delay line controlled by some mechanism

(either analog control voltage or digital controlling bits). The delayed feedback clock is compared with the reference clock by the help of Phase Detector (PD), which compares the phases of two signals and generates the UP/DOWN signal based on whether the reference clock is leading or lagging the feedback clock respectively. These UP/DOWN signals are used by the controlling mechanism to either increase or decrease the delay to align the phases.

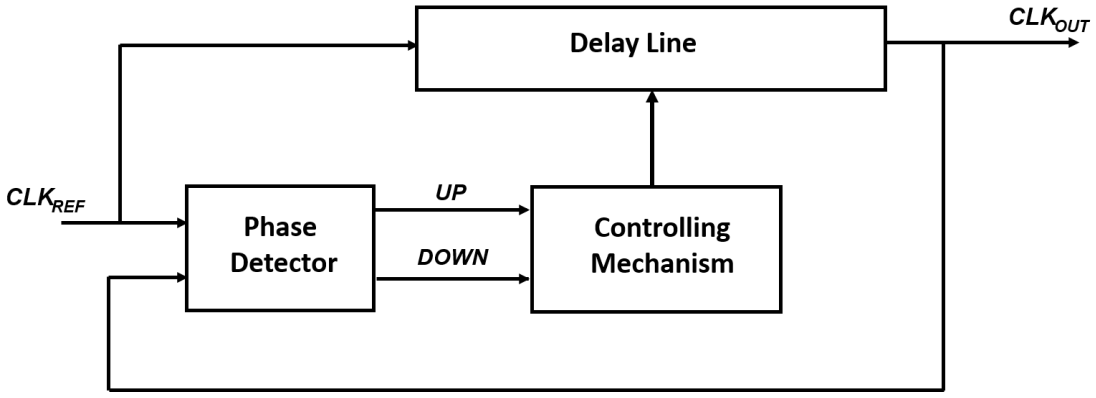


FIGURE 2.1: Block diagram of Delay Locked Loop

When DLL is used as de-skew buffers, the skew is removed by providing extra delay to the reference clock. There are two options for delay addition by DLL, either the delay added could be just the phase difference between the two clock signals or the delay added could be a summation of a multiple of clock signal and the phase errors between the two clock signals.

2.2 Closed Loop Dynamics of DLL

For a generic DLL, the control model looks like:

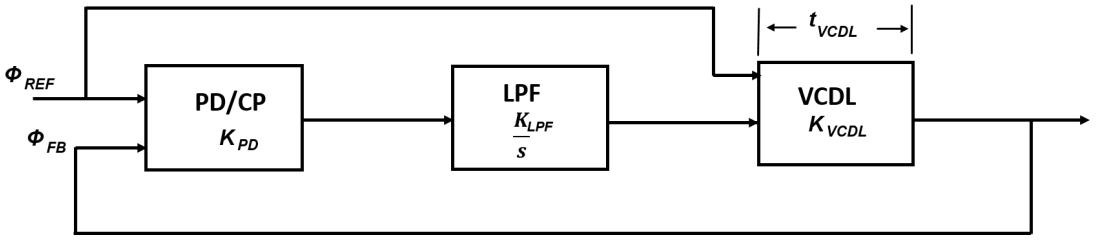


FIGURE 2.2: Generic Control Model of DLL

The open loop transfer function can be defined as:

$$\frac{\phi_{REF}}{\phi_{FB}}, G(s) = \frac{K_{PD}.K_{LPF}.K_{VCDL}}{s} \tag{2.1}$$

Here, there is no feedback element used for simplicity and hence, ϕ_{FB} is same as ϕ_{OUT} (not shown in th image above but it will be present at the output node). The closed loop transfer function can be defined as:

$$\frac{\phi_{REF}}{\phi_{FB}}, H(s) = \frac{K_{PD}.K_{LPF}.K_{VCDL}}{s + K_{PD}.K_{LPF}.K_{VCDL}} \tag{2.2}$$

It is evident from the eq (2.1) and (2.2) that the DLL is essentially a single pole system. So, it always stable. The main sources for jitter in delay locked loop are noise sensitivity of the delay line, noise sensitivity of the clock buffers (if used), device noises (usually negligible).

A delay line is usually made of delay elements in series which are providing a constant (or fixed) amount of delay to the input provided. However, by the help of a controlling mechanism, the delay provided by these elements can be changed. For example, the delay across a cascade of two inverters can be changed by varying the power supply, load capacitance, input slew etc. DLLs can be roughly classified into two categories, namely analog DLL and digital DLL, based on how the delay line is being controlled by the controlling mechanism. An analog DLL has a continuously variable delay line which is controlled by an analog controlling input. An example of such a delay line is Voltage Controlled Delay Line (VCDLs). Whereas, in digital DLL, the delay line is made up of digital devices which have pre-defined quantized delay steps. This delay line is controlled by a digital word which directly maps the delay to a certain value. Both analog and digital DLL have a common element called phase detector (PD) which is used to find the phase error, as digital block.

2.3 Analog DLL

The main components of the analog DLL are Phase Detectors (PD), Charge Pump (CP), Loop Filter (LPF) and Voltage Controlled Delay Line (VCDL) as shown in the below block diagram.

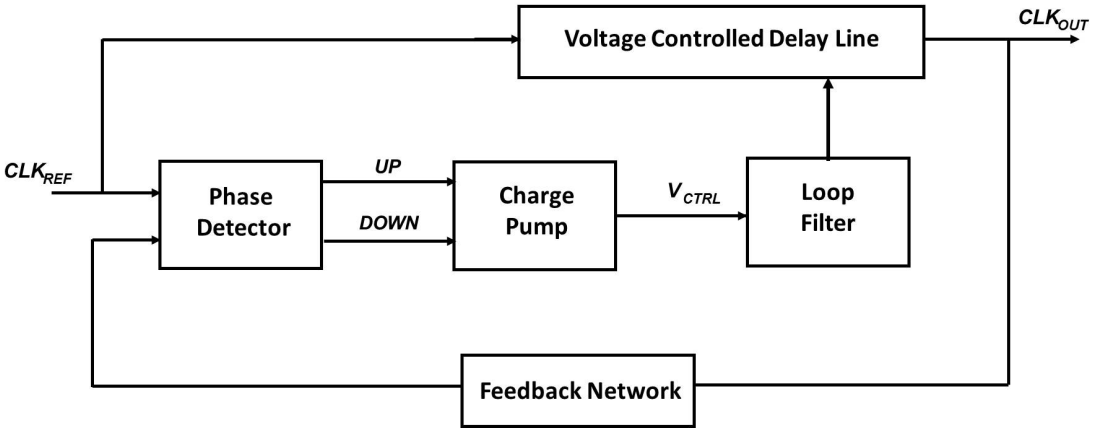


FIGURE 2.3: Block diagram of Analog DLL

2.3.1 Phase Detector

A phase detector is a circuit whose average output ($\overline{V_{out}}$) is directly proportional to the phase error, $\Delta\phi$, between the two inputs applied to the phase detector. An XOR gate is a familiar example of the phase detector, as shown below in the fig. 2.5a.

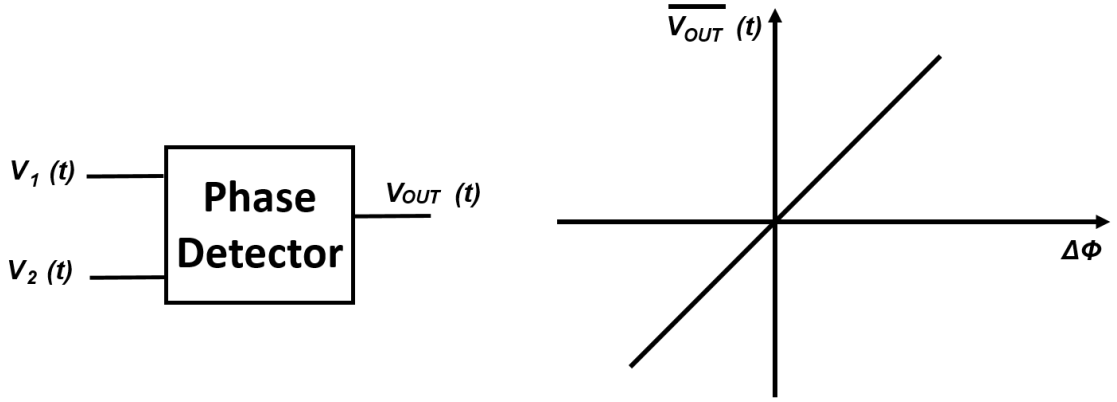
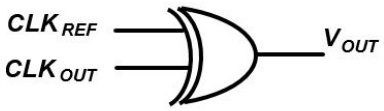
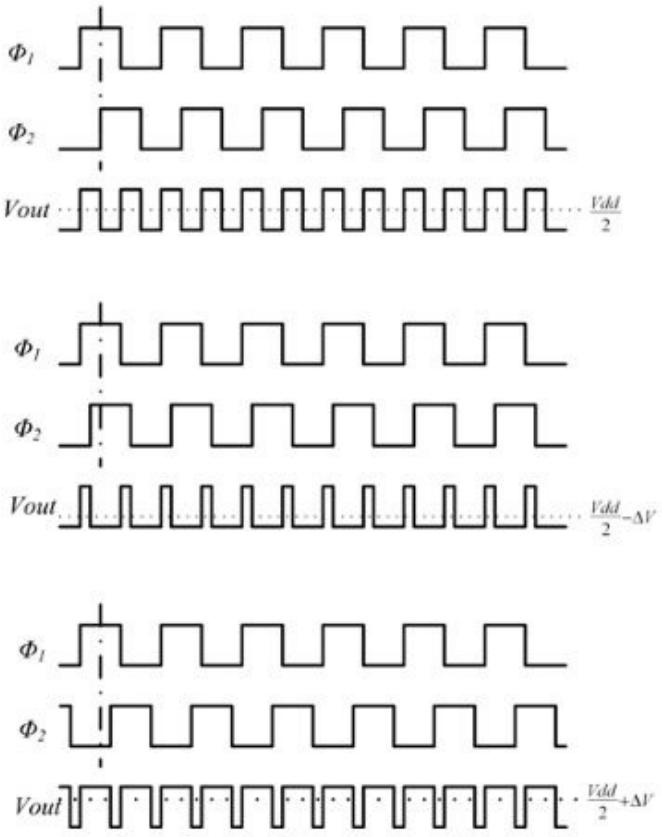


FIGURE 2.4: Definition of phase detector

The XOR circuit produces a DC level proportional to the phase error between its two inputs. As the phase difference between two inputs varies, the width of the output pulse also varies. Some of the phase detectors respond only to either positive or negative transitions, but the XOR based PD responds to both rising and falling edges. When the inputs applied to XOR based PD are 90°out of phase, the average dc value of the output signal produced by XOR gate is Vdd/2 and the phase error defined by $|\phi_1 - \phi_2|$ -90°turns out to be 0. This is why, this type of PD is normally used for detecting quadrature locking i.e. when the input and output signals are 90°out of phase in locked condition. There is only output signal available after comparison which makes this circuit less attractive for to be used as it becomes difficult to interface it with the charge



(A) XOR gate



(B) Output waveforms for different input combinations [4]

FIGURE 2.5: XOR based Phase Detector

pump. Another problem with XOR based PD is that the output is dependent on the duty cycle of the signals as the XOR gate can only detect levels. So to make sure that XOR based PD is correctly, we need to provide an extra duty cycle correction circuit.

Another type of phase detector available is flip-flop based phase detector which detects either the rising or falling edge of the transition of the input signal. In such cases, we often do not need a duty cycle correction circuit. The simplest among these type of PDs are a D-flip flop based PD, which has feedback (or output) clock as Data input, reference clock as the Clock input and produces the difference between two signals as Q output of the flip flop. If the Q is high, this means that the reference clock is leading the feedback clock and the system should decrease the delay in order to achieve a lock, Similarly, if Q is low, the system should increase the delay to match the phases of the input signals. However, this kind of PD has its own disadvantages. The major shortcoming is based on setup and hold time requirement of flip flops. If the basic timing requirement for

a flip flop is not met, it would lead to irregular errors in the output as the decision is entirely based on the setup and hold requirements.

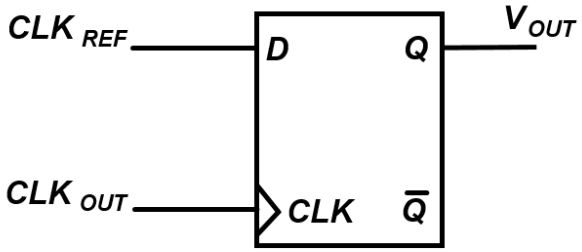


FIGURE 2.6: DFF based PD

Figure 2.7 shows a Phase Frequency Detector (PFD) having two D-flip flops and a reset circuit (NAND gate). This circuit is capable if detecting both frequency locks and phase locks which makes the locking speed faster and increases the acquisition range for our circuit. First, the frequency detector will be active and try to bring the output frequency in acquisition range of input frequency (which is typically in order of loop filter's cut-off frequency). After this, the phase detector will generate signals proportional to the phase difference in order to compensate the phase error and bring the circuit in lock condition.

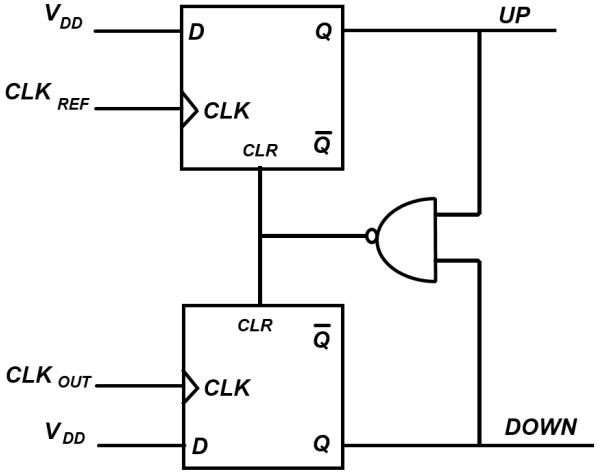


FIGURE 2.7: Phase Frequency Detector (PFD)

To overcome the shortcomings offered by flip flops based phase detectors, dynamic PDs are used. They eliminate the need of flip flops and have simple architecture and fast transition capability which make them suitable for high-speed DLL designs. Figure shows such kind of PD. There are two stages with a pre-charge in each stage. The pre-charge of second stage is controlled by the output of first stage. Extra attention is required to remove the dead zones and align the phases with precision.

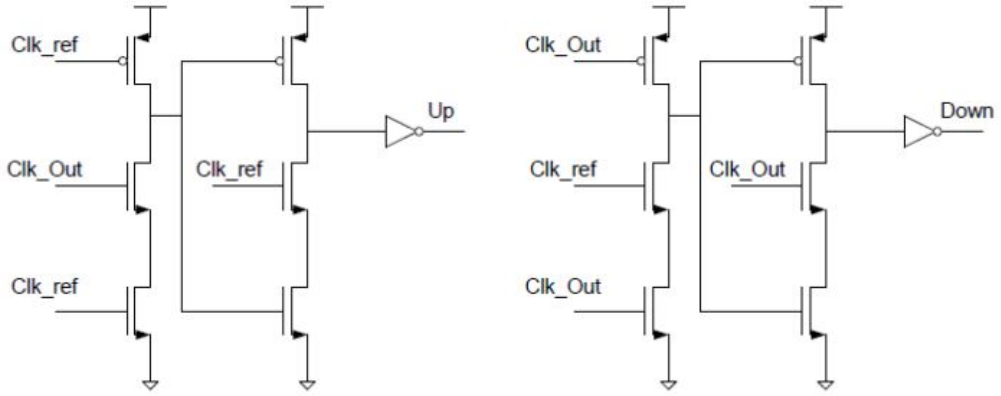


FIGURE 2.8: Dynamic PD [1]

2.3.2 Charge Pump

In a DLL, a charge pump is made up of two controlled switched current sources, one is working as current source and another is working as current sink, responsible for pumping current into or out of the loop filter based on the logical inputs received from the PDs. The figure below shows an example of CP driven by a PFD and driving a simple capacitor based loop filter. The CP receives UP and DOWN signals from PFD, which controls the the two switches of CP respectively. If the switches are closed, the current source or sink will add or remove the charge from the capacitor, therefore adjusting the voltage of the capacitor. This voltage is used as analog controlling signal for controlling the delay in the Voltage Controlled Delay Line (VCDL). Thus the phase error $\Delta\phi$ is mapped to the delay across delay line. This process of charging and discharging the capacitor continues till the DLL comes into locked condition. After it is locked, a constant voltage will appear across capacitor controlling the VCDL at a particular delay value needed to keep the circuit in lock condition.

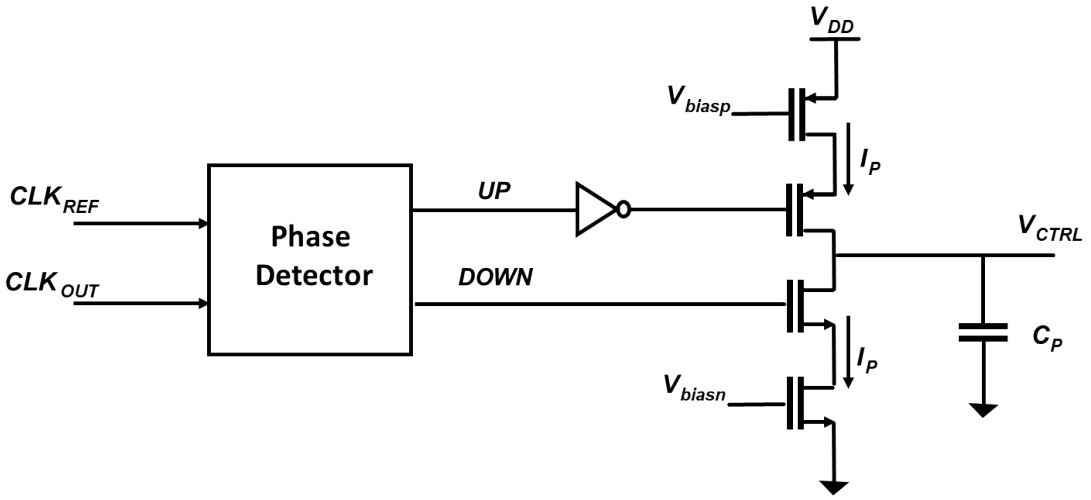


FIGURE 2.9: PFD, Charge Pump and Loop Filter Combination

The transfer function of figure can be easily obtained as

$$\frac{V_{out}}{\Delta\phi}(s) = \frac{I_p}{2\pi C_p}.\frac{1}{s} \tag{2.3}$$

It may be noticed that the charging and discharging process across capacitor continues to happen even if the DLL is in locked condition. This kind of action is desirable in order to minimize the jitter, however, an extra care is needed to make sure that these charging or discharging currents have a small value and does not affect the constant control voltage (V_{ctrl} across the capacitor and disturbs the locking of the DLL.

2.3.3 Loop Filter

It is necessary that the undesirable high-frequency components of the output available at the output node of the charge pump should be suppressed and only the stable dc value is passed to the delay line. For this purpose, a loop filter is interposed between the PFD and CP combination and the VCDL. We generally use a low pass filter, to pass lower frequencies and suppress high frequencies. The design of DLL should be done in such a manner that two most important properties of DLL i.e. jitter and locking time should be as small as possible.

For a generic analog DLL, the control model looks like:

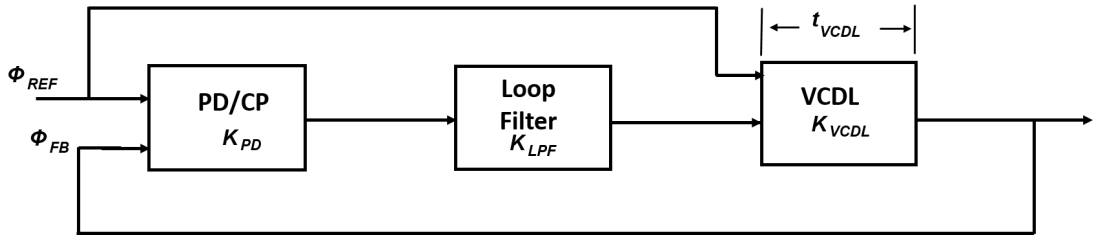


FIGURE 2.10: Generic Control Model of DLL

Here, we have ignored the feedback loop delay for simplicity purpose. The relationship between ϕ_{ref} and ϕ_{out} can be deduced from the following equation:

$$\phi_{out} = 2\pi f.(\phi_{ref} - \phi_{out}).K_{PD}.K_{LPF}.K_{VCDL} \tag{2.4}$$

So, the transfer function of analog DLL can be written as:

$$\frac{\phi_{out}}{\phi_{ref}} = \frac{1}{1 + \frac{1}{2\pi f.K_{PD}.K_{LPF}.K_{VCDL}}} \tag{2.5}$$

Here, we can define the bandwidth of DLL (ω_{DLL}) as:

$$\omega_{DLL} = 2\pi f.K_{PD}.K_{LPF}.K_{VCDL} \tag{2.6}$$

The frequency of loop filter (ω_{loop}) is defined as:

$$\omega_{loop} = \frac{2\pi}{\Delta t_{VCDL}} \tag{2.7}$$

For a proper operation, it is desirable that the minimum delay and maximum delay obtained from the delay should be more than half of the clock period and less than the 1.5 times of clock period respectively, i.e.

$$\frac{T}{2} \leq \Delta t_{VCDL} \leq \frac{3T}{2} \tag{2.8}$$

So, max loop frequency can be:

$$\omega_{loop} = \frac{2\pi.2}{3T} \tag{2.9}$$

For a proper operation of DLL, it is desirable to keep the bandwidth of DLL one decade lower than the loop bandwidth. So, we can define a relation between the bandwidth of DLL and its loop filter as:

$$\omega_{DLL} = \frac{\omega_{loop}}{10} \tag{2.10}$$

This design rule is very important to keep in mind while designing an analog DLL as the two important parameters jitter and locking time depends on the relation defined by eq. 3.8. The jitter is directly proportional to the ratio of bandwidth of DLL and its loop filter i.e. $\frac{\omega_{DLL}}{\omega_{loop}}$ and the locking time is inversely proportional to bandwidth of DLL i.e. ω_{DLL}. Hence, designing a loop filter correctly is of utmost importance. In most of the designs, a loop filter or a low pass filter is simply a capacitor. The voltage across capacitor (V_{ctrl}), controls the delay line and helps set the delay across the DLL. The operation of the delay line is explained next.

2.3.4 Voltage Controlled Delay Line

Voltage Controlled Delay Line is the heart of any DLL. This is the part responsible for providing delay to the input reference signal and produce an output clock that is deskewed and in phase with the input signal. The two main inputs of VCDL are (1) Reference Clock and (2) Control voltage (V_{ctrl}). VCDL comprises of delay elements cascaded in series providing a delay (or phase shift) to the input signal, keeping the frequency intact. Designing of VCDL in correct manner is of utmost importance as it directly provides the output signal of the DLL. The performance of VCDL considerably affects the jitter performance of the DLL.

As shown in the figure below, VCDL typically consists of delay cells connected in series, controlled by an analog controlling voltage. Since it is an open loop configuration, so unlike Voltage Controlled Oscillator (VCO) (used in PLLs), this configuration does not oscillate and hence less susceptible to noise (and hence advantageous over PLLs). Also, in VCDLs the change in control voltage is immediately mapped to change in delay value and hence the transfer function is simply equal to the gain of the VCDL.

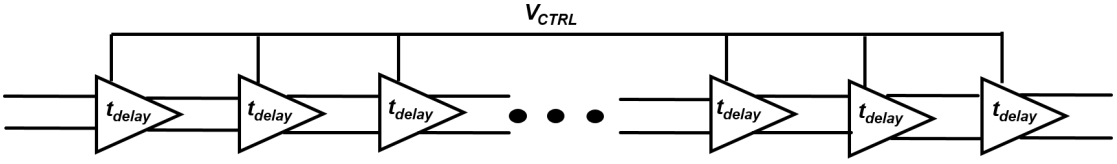


FIGURE 2.11: A simple representation of Differential ended Voltage Controlled Delay Line

Theoretically, the maximum delay offered by the VCDLs should be equal to one clock period (say T_{ref}) i.e. the input signal should undergo a maximum phase shift of 360°. Also, all the delay elements should be identical in nature and hence the delay offered by each stage (t_{delay}) should be (T_{ref}/N) where N is the number of delay elements present in the delay line. The phase resolution for such a delay line is defined by 360°/N, therefore,

by increasing N i.e. by the increasing number of stages, the phase resolution can be improved, but this also increases the delay offered by the delay line. Hence, the phase resolution is limited by the maximum delay that a delay line can offer to the input signal, which in turns depends on the frequency of the input signal. So, the design parameter for a delay line is the delay range.

In analog DLLs, a delay line can be single-ended type or differential-ended type. There are various possible implementations of a delay cell most common being shunt capacitor delay stage, variable resistor based delay stage and current starved inverter (CSI) based delay stage. Figure 2.12 shows a shunt capacitor delay stage. A N-MOSFET M_4, with its drain and source terminal shorted, behaves as a MOS capacitor. The gate voltage (V_{ctrl}) of transistor M_3 controls the charging and discharging current of the capacitor through the inverter (formed by transistors M_1 and M_2). Interested readers can refer to [10] for further details.

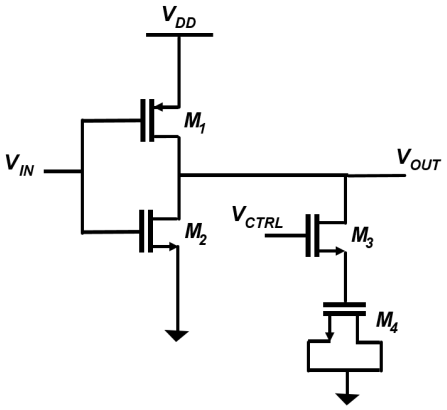


FIGURE 2.12: Shunt Capacitor Inverter

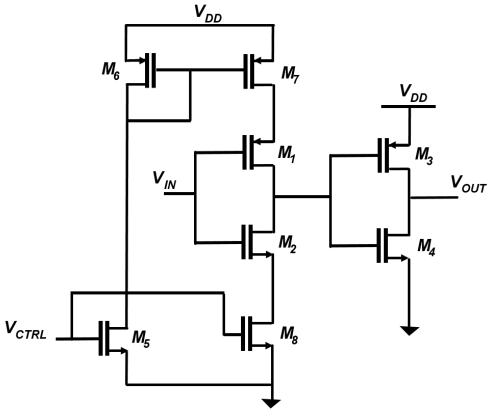
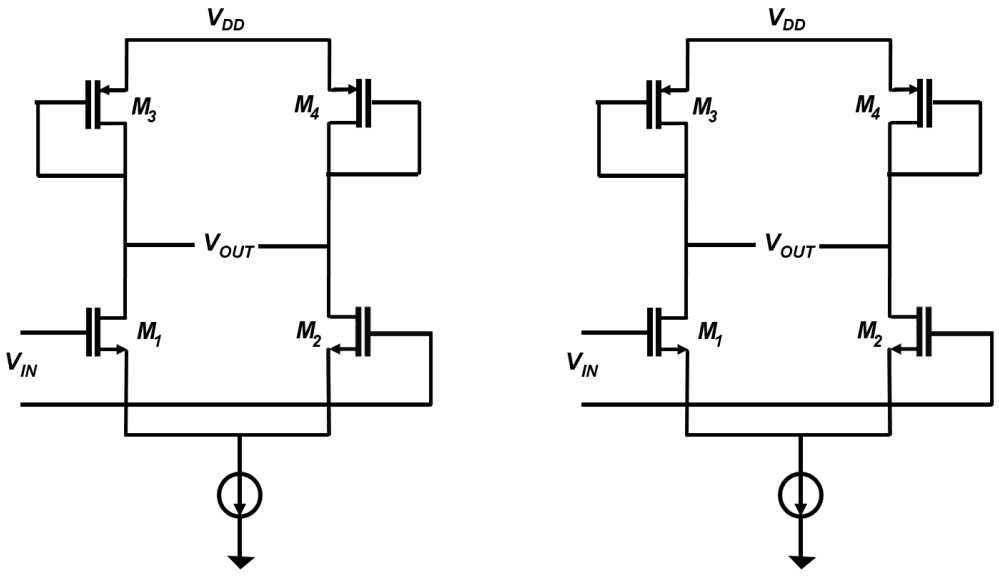


FIGURE 2.13: Current Starved Inverter

Figure 2.13 shows a current starved inverter. In this configuration the charging and discharging current through the output capacitance of first stage is controlled by the PMOS (M_7) and NMOS (M_8) in series with the inverter (formed by the transistor pair M_1 and M_2). A transistor in saturation region, behaves as a current source, controlled by its gate voltage. Here, a current mirror is used for generating the gate voltage for transistors M_7 and M_8. In this manner, we can control the delay through both the rising and falling edges of the output signal. However, if only one edge needs to be controlled, so only one transistor is enough to source or sink the current. For example, in order to control only rising edge, M_7 transistor will be used. We can see that there is second stage available at the output of first stage. This stage is basically added to control the rise and fall times of the output. Many a times, multiple such stages are needed to be cascaded in order to improve the rise and fall time of the output.

It can be seen that in both the cases mentioned above, there is an analog controlling voltage which is controlling the charging and discharging currents through the output node and hence affecting the delay of the circuit.

Single ended type delay stages have high susceptibility to noise and poor power supply rejection ratio. Hence, differential ended type delay stages are more preferable as they have inherent advantage of common-mode rejection ratio and better spectral density. Figure shows one example of such a differential ended delay stage. The source coupled differential pair having load elements and tail current source. This type of circuit can have either diode-connected loads or transistors in linear region (behaving as resistance) [11]. These loads are selected based on desired delay value, output voltage swing, power supply rejection value etc. The ideal tail current source provides a constant current to the circuit and provides stability against supply noise.



(A) Diode Connected Loads (B) Loads in Triode Region

FIGURE 2.14: Differential Delay Elements

For diode connected loads, r_{out} can be written as:

$$r_{out} = r_{onN} // r_{onP} // \frac{1}{g_m} \tag{2.11}$$

We know that for a differential pair, the relation between the trans-conductance g_m and the tail current I_{SS} is given by:

$$g_m = \sqrt{\mu C_{OX}(W/L) I_{SS}} \tag{2.12}$$

The delay of such a cell is given by:

$$\Delta\tau = r_{out}.C_{out} \tag{2.13}$$

where $\Delta\tau$ is the delay offered by each stage and C_{out} is the effective output capacitance offered to the signal.

By putting the value of eq 2.10 in 2.9 and value of 2.9 into 2.11, we can the delay value as:

$$\Delta\tau = \sqrt{\frac{1}{\mu C_{OX}(W/L)I_{SS}}}.C_{out} \tag{2.14}$$

Hence, it can be seen that value of delay ($\Delta\tau$) depends upon the tail current I_{SS}. So, this current if controlled, can alter the delay value offered by the delay cell. To control this current value, a biased transistor will be used which has gate voltage as V_{ctrl}. On the similar lines, a relation between the delay value and the control voltage V_{ctrl} can be drawn for figure (b) as well. Here, V_{ctrl} is used as bias voltages for the triode loads. The relation in such a case turns out to be:

$$\Delta\tau = \frac{C_{out}}{\sqrt{\mu C_{OX}(W/L)(V_{DD} - V_{ctrl} - Vth)}} \tag{2.15}$$

Figure shows another type of delay element used commonly in delay lines. This circuit is an improved version of the delay stages mentioned above, as it has symmetrical loads and self-biased tail current source.

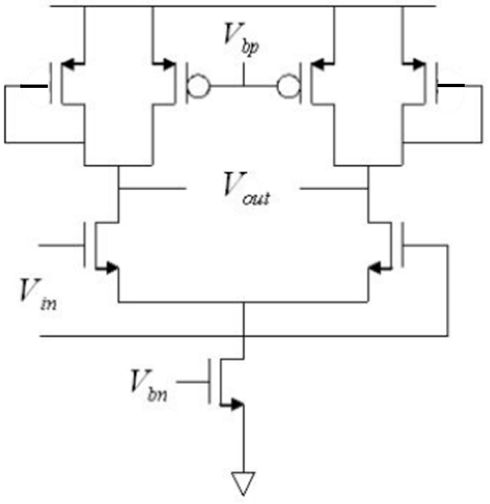


FIGURE 2.15: Delay cell having symmetrical loads [1]

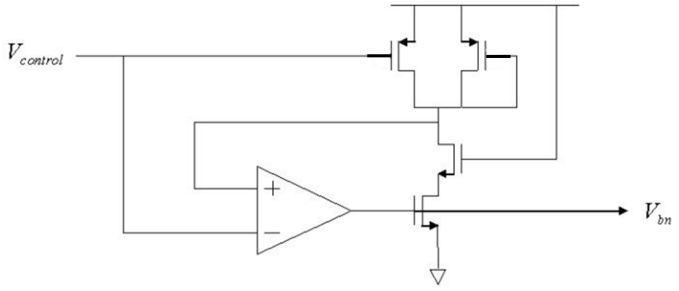


FIGURE 2.16: Self-bias circuit for the tail current source [1]

Thus, an analog DLL is designed. These DLLs has advantage of low jitter value, as an analog controlling signal is used to continuously vary the delay. However, this kind of DLL has some disadvantages which will be discussed in next sections.

2.4 Digital DLL

The main components of the digital DLL (DDLL) are the Phase Detectors (PD), Controlling Mechanism and Digitally Controlled Delay Line (DCDL) as shown in the below block diagram. The digital DLLs are typically made up of simple digital elements, which helps us to design portable delay locked loop which can be easily migrated to other technologies. This is the major advantage of the digital DLLs which make them attractive to use as compared to analog DLLs. Another advantage includes the stability of zero-order transfer function. However, this is at an extra cost of more layout area, moderate phase resolution and degraded jitter performance.

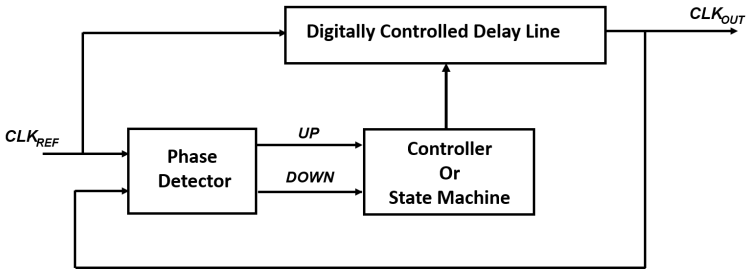


FIGURE 2.17: Digital Delay Locked Loop

2.4.1 Phase Detectors

As described in section 2.3.1, Phase detectors are used to measure the phase error between its two input signals and produce a corresponding output voltage which is directly proportional to the phase error $\Delta\phi$. In digital DLLs also, phase detectors can

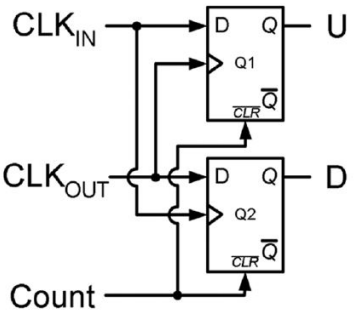


FIGURE 2.18: Phase detector based on flip-flops [2]

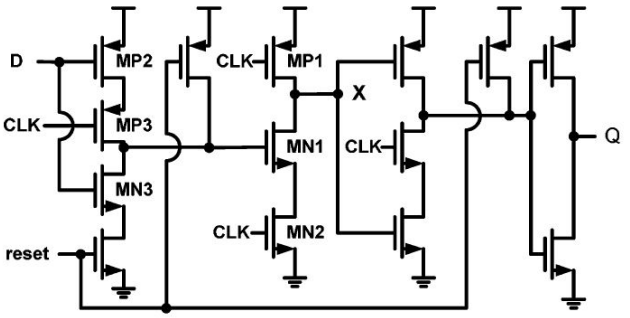


FIGURE 2.19: TSPC DFF [3]

be realized by using using flip flops. Figure 3.4 shows a possible implementation using flip-flops to detect the phases of the input signals. The dynamic phase detector shown in figure is also commonly used for digital DLLs. Apart from this, for detecting phase difference between high frequency inputs, the true-single phase clock type (TSPC) type DFF is used. The advantage of such a flip-flop is that the dithering phenomenon, commonly observed in other type of phase detectors, can be suppressed which helps in jitter reduction [3]. Figure 2.19 shows a possible implementation of TSPC DFF having an extra reset circuit.The choice of using phase detectors basically depends on the type of signal and the frequency of the signals used.

2.4.2 Control Mechanism

The characterization parameter for a digital DLL is the controller of digital DLL, that controls the delay line. Based on how the controller is implemented, the digital DLLs are roughly divided into four broad categories:

- **Register-Controlled Digital DLL (RCDLL)**: In RCDLL, the control is comprised of shift registers, which decides the number of delay elements to be present between the input and the output clock. The register value is set by the help of the output of the phase detector, producing digital bits as output which controls the delay line. The main disadvantage with this kind of digital DLL is that when the control bits increases, then the time to lock the DLL also increases exponentially [12],[13]. This is because the number of delay elements will increase in order to be controlled by the more number of bits. This increases the delay and hence the locking time of the DLL. Figures 2.20, 2.21 shows an example of register controlled DLL.

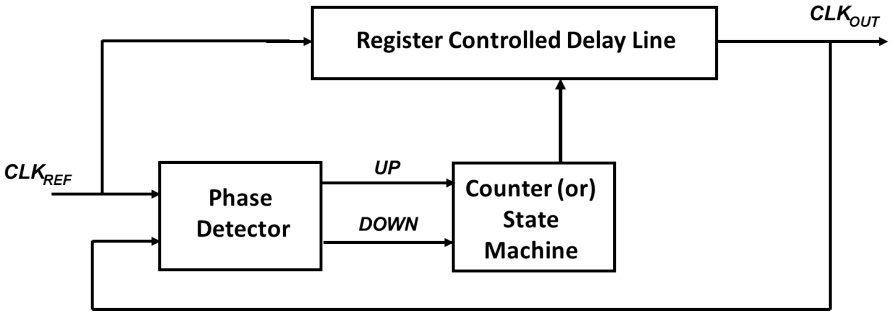


FIGURE 2.20: Register Controlled Digital DLL

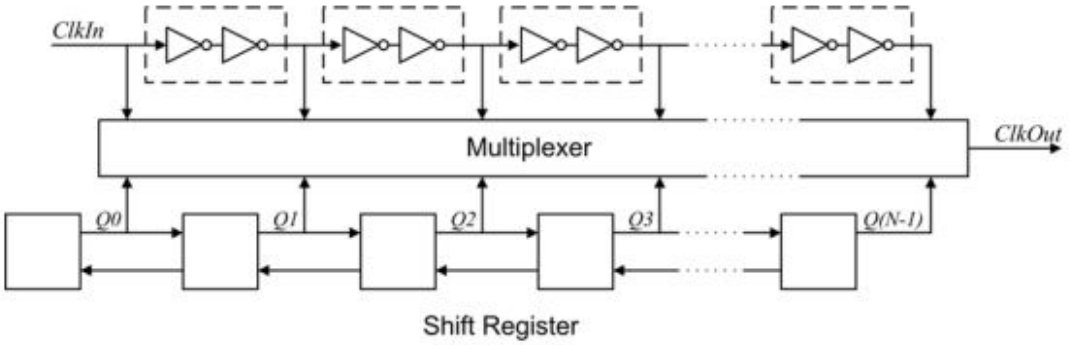


FIGURE 2.21: RCDL block diagram [4]

- **Counter-Controlled Digital DLL (CCDLL)**: This is a commonly used architecture for digital DLL [14], [15]. There are two possible configurations for this type of DLL. As shown in figure 2.22 [5], the control is just a up/down counter or a bidirectional shift register, which is triggered on the UP/DOWN signal provided by the phase detector. The up/down counter decides whether to increase the delay of the delay line or to decrease it to conquer a lock between the inputs of the phase detector.

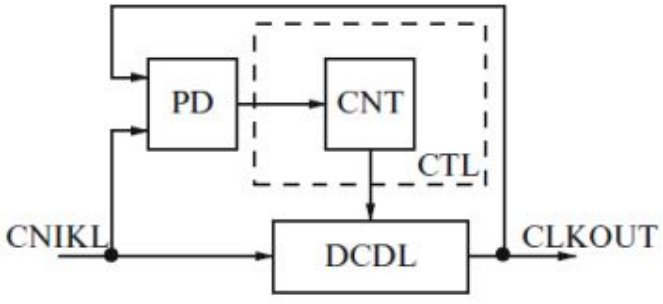


FIGURE 2.22: Simple counter based DLL [5]

Next, a complex control algorithm can be used for counter-controlled DLL if we put a FSM between the outputs of the phase detectors and the input of the up/down

counter. So, now the counter is controlled by the output of phase detector as well as the output of the FSM which depends upon the internal state of the system. This configuration helps in achieving lock faster than the configuration shown in figure 2.22, however, it strongly impacts the DLL performance i.e. the stability, dynamic range and sensitivity etc [5].

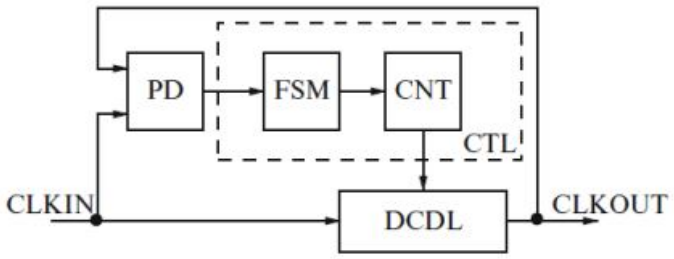


FIGURE 2.23: Counter Controlled DLL having FSM and Counter as control logic [5]

Although, the hardware required to implement the counter controlled DLL is less than the hardware requirement of the register controlled DLL, the CCDLL has same disadvantages as RCDLL. When the control bits, controlling the delay line increases, the number of delay elements and the locking time of the system increases exponentially [12], [13]. Also, in a counter-controlled DLL, a digital code may keep toggling between two values, increasing the output jitter due to dithering phenomenon [13].

- **Successive Approximation Register based Digital DLL (SARDLL)**: In the above mentioned DLLs, the locking time required will increase exponentially as the number of bits increases. A binary search algorithm can be use a remedy to this. The most common circuit level of implementation of binary search algorithm can be realized by Successive Approximation Register method. This structure will be used in this thesis and will be explained in the detail in the next sections.

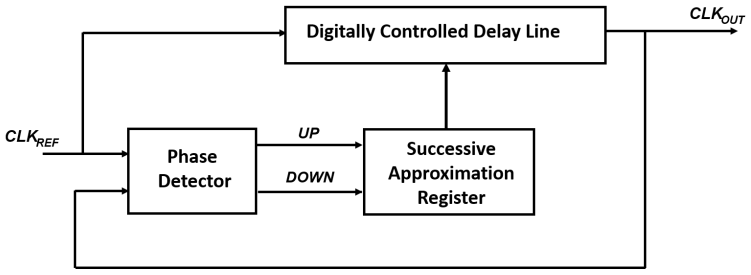


FIGURE 2.24: SAR Digital DLL

- **Time-to-Digital Converter based Digital DLL (TDCDLL)**: Last category is Time-to Digital Converter based DLL. It uses Gray code counter running at a

very high pace, whose values gets updated at a triggering action [16–20]. To obtain a sub-nanosecond resolution with high time interval digitization needs the system to run at very high frequencies. In spite of having simple architecture, the large chip area, wide frequencies, and large power consumption makes this architecture less attractive to chosen.

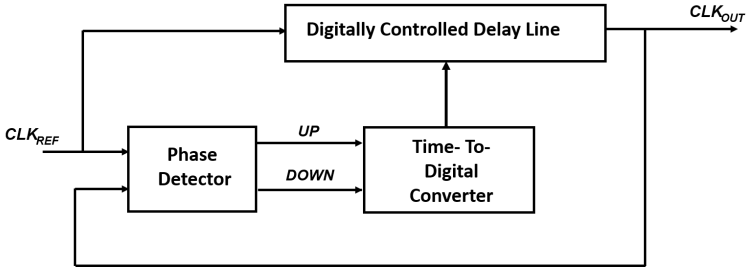


FIGURE 2.25: TDC based Digital DLL

2.4.3 Digital Controlled Delay Line

In digital controlled delay line, the digital control value is mapped to a particular delay value. A sample transfer function is shown in figure 2.26. Here, the x-axis shows the digital value which is is needed to set the desired delay and the y-axis shows the desired delay value at which delay should set. This transfer can be mathematically expressed as:

$$\Delta\tau = D_{min} + (N-1) \times d_r \tag{2.16}$$

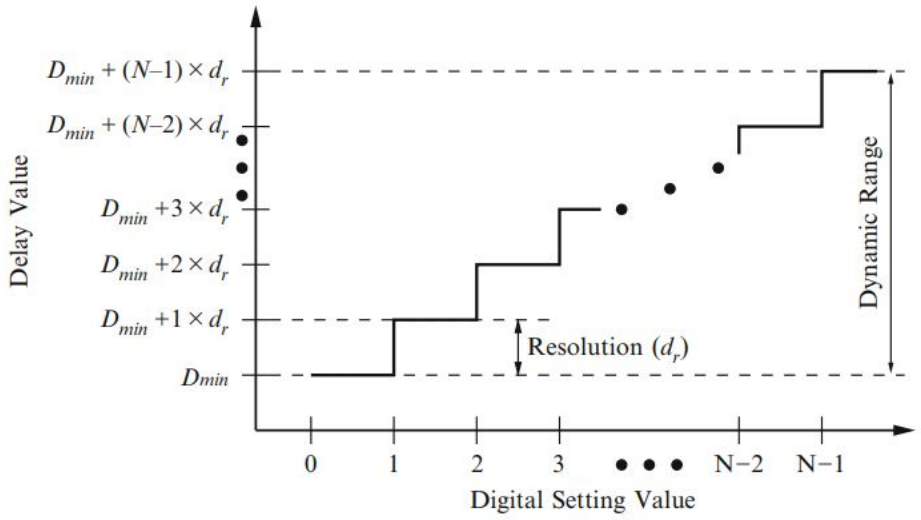


FIGURE 2.26: Digital Controlled Delay Line Transfer Function [5]

It is evident from eq 1.14, that the delay line is characterized by three parameters:

- **Minimum Delay (D_{min})**: It is defined as the minimum delay offered by the delay line. With a delay line controlled by N programmable bits, this minimum delay could be at lowest setting value (0) or highest setting value (N-1) depending on whether the delay line has monotonically increasing relation or monotonically decreasing relation with respect to the digital setting value. In the figure 2.26, the delay value defined at the minimum setting value i.e at 0 is the minimum delay ($Dmin$).

- **Resolution (d_2)**: This is defined as incremental delay per setting value i.e. the difference between two consecutive delay values at two consecutive setting values. In order to achieve a good phase resolution, this parameter should be as least as possible. It affects the DLL's accuracy.

- **Maximum Delay (D_{max})**: With a delay controlled by N programmable bits, the delay corresponding to the $(N-1)^{th}$ setting value or 0^{th} setting value, based on whether the delay line has monotonically increasing relation or monotonically decreasing relation with respect to the digital setting value, is defined as the maximum delay.

The dynamic range for a DLL is defined as the difference between maximum and minimum delays i.e $D_{max} - D_{min}$. This parameter defines the capability of the DLL to track the PVT variations and extension of bits needed for realizing high-frequencies. In digital DLLs, the delay line is divided into two parts: (1) Coarse delay line (also referred as Gate delays), and (2) Fine delay line (also referred as sub-gate delays). The various architecture possible for both the delay lines are described in next subsections.

2.4.3.1 Coarse Delay Line

They are formed by using standard CMOS logic gates. A cascaded chain of standard CMOS logic gates having intermediate outputs tapped which forms input of the multiplexer defines a coarse delay line. A simple structure includes a chain of simple inverters cascaded to form a delay line and outputs being tapped and selected by the multiplexer. The delay cells used in coarse delay line can have a moderate minimum delay value (D_{min}) but the delay resolution value (d_r) should be coarse enough, such that the maximum delay (D_{max}) should be able to match the input frequency or reference clock signal. A very straight forward implementation is shown in figure 2.27. Here, an inverter pair forms a delay element and the outputs are tapped and selected by multiplexer. This delay line has a resolution as $2T_d$ where T_d is average delay of each CMOS inverter. Thus, the minimum delay value is $2T_d + \log N \times \mathrm{T}_d$.

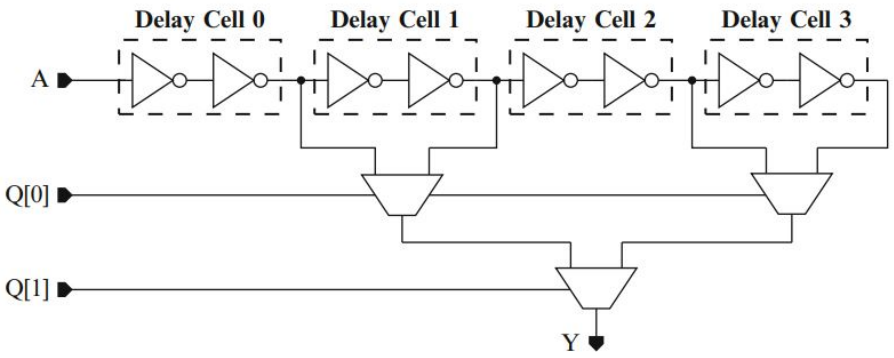


FIGURE 2.27: Inverter based Coarse DCDL [5]

The same implementation can be realized for a differential ended delay line as well as shown in figure !2.28. Here, two inverters, forming alatch is kept at output node of each delay cell. This is done in order to remove the skew value (if any) present the output of the differential ended delay element.

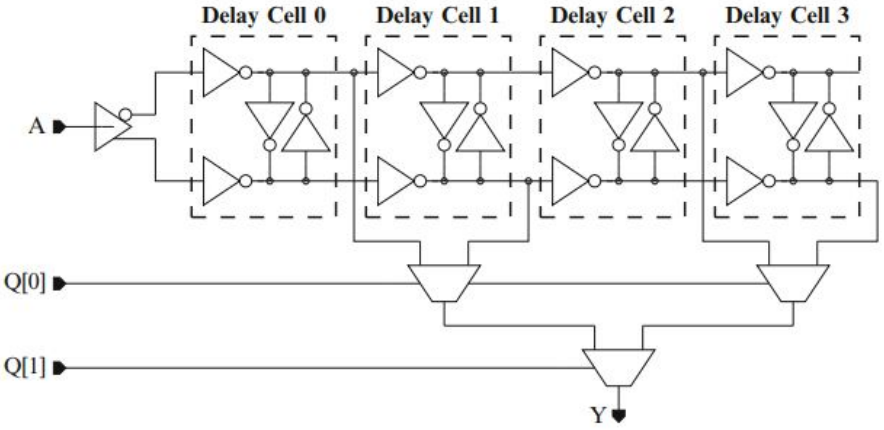


FIGURE 2.28: Inverter based Differential ended Coarse DCDL [5]

Figure 2.29 shows a very popular coarse delay line [21, 22]. This is NAND based register controlled delay line. Here, there are four delay stages between the input A and output Y. The bidirectional register is controlled by one-hot encoded bits (Q[3:0]). The resolution value and the minimum delay for such a circuit is $2T_d$ where T_d is the average delay value offered by each NAND gate. However, this design presents a substantial amount of load to the input clock, which makes the need of clock buffering necessary. By putting buffers at the load of the input clock signal reduces the load capacitance offered to the input signal but increases the minimum delay offered by the circuit slightly by a value of $2T_d + T_{buf}$, where T_{buf} stands for the delay offered by the buffers.

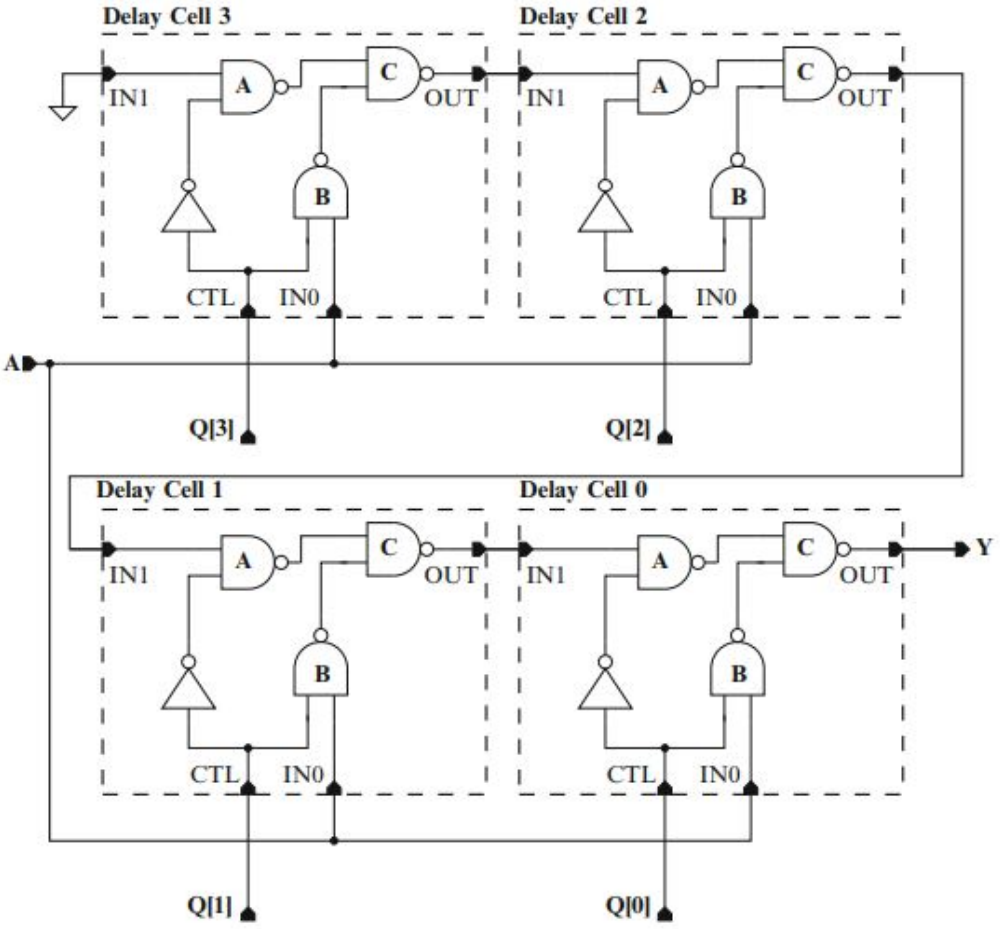


FIGURE 2.29: NAND based Coarse DCDL [5]

To avoid the delays by buffers, a work-around solution is shown in figure 2.30. The architecture shown in 2.29 has been redesigned and now, the delay elements are connected in a telescopic fashion. The controlling signals Q[3:0] can be either one-hot encoded or thermometerically encoded. The minimum delay value (D_{min}) can be easily realize as $2T_d$. When say the controlling values are Q[3:0] = 0001 then only cell A and cell B will be active. The input signal is routed through NAND cell A and NAND cell B and a delayed signal (shifted by two NAND gate delays) is available at output of NAND cell B. Next when controlling signal value is Q[3:0] = 0010, then the delay values of two more NAND cells i.e. NAND cell C and NAND cell D are added to the previous delay value. This way, every time with each bit shift in controlling value, two NAND gate delays will be added to the input clock signal. Hence, both the minimum value and the delay resolution i.e. D_{min} and d_r is $2T_d$.

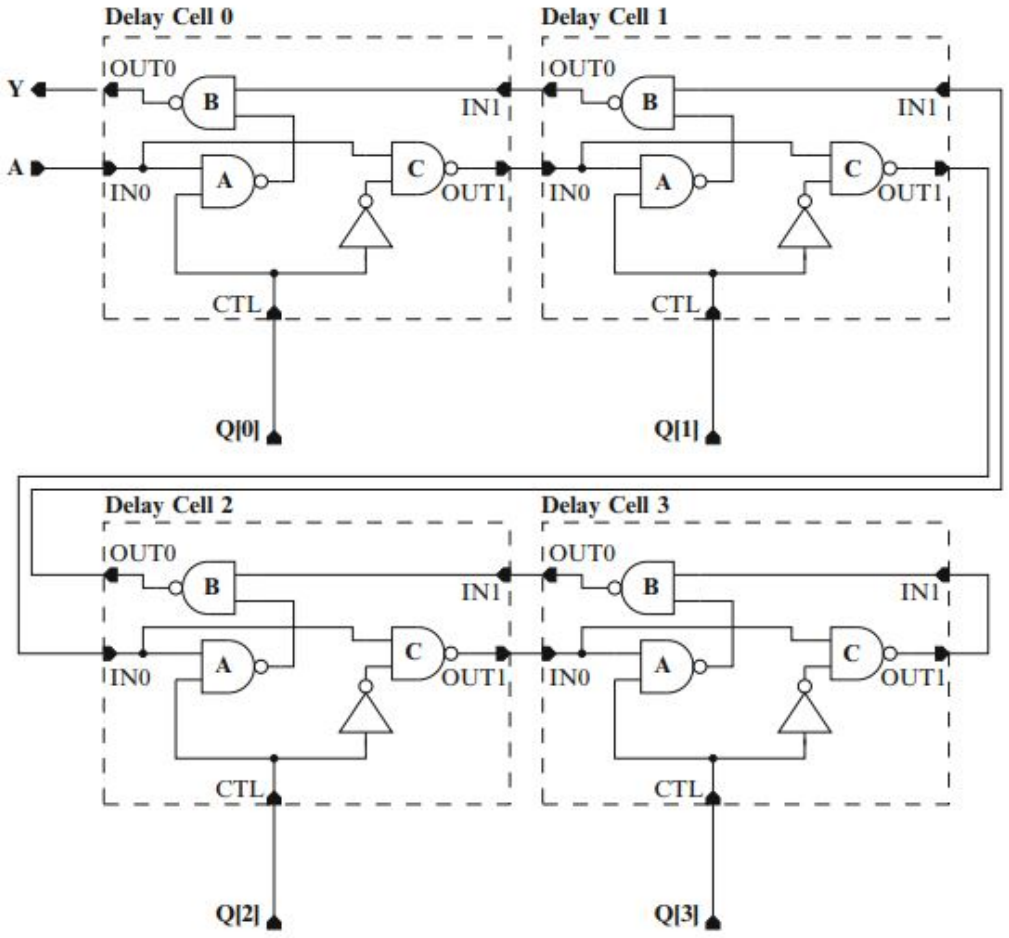


FIGURE 2.30: NAND based Coarse DCDL connected in telescopic fashion [5]

In this thesis, we are using the coarse DCDL implemented by connecting NAND gates in telescopic fashion as it does not require extra buffers to be put at the input stage and also NAND gates offer a coarse enough value for the minimum delay and the delay resolution.

2.4.3.2 Fine Delay Line

This delay line is responsible for defining the phase lock accuracy of the delay locked loop. Usually, a subgate delay line is implemented by variable RC circuits. Figure 2.31a and 2.31b shows two such architecture. The delay in such cases is usually equal to RC and a variable resistance or a variable capacitance value can alter these delay values. Figure 2.31a shows an example of variable resistance and figure 2.31b shows an example of variable capacitance. In both the cases, thermometer encoded bits are used.

Another type of variable RC delay element includes current starved inverter shown in figure 2.13. However, the disadvantages offered by such a circuit is already discussed in

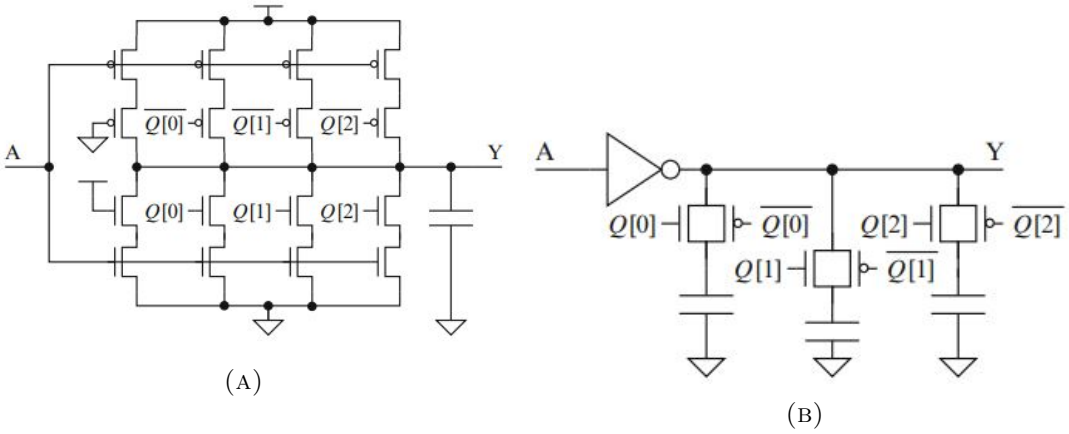


FIGURE 2.31: RC based delay elements

section 2.3.4. In this thesis, we have used a slightly modified version of current starved inverter based scheme. The details of this delay line will be discussed in next section and can also be found in [23]. Interested readers can find a detailed description about delay lines in [5].

Thus, a digital delay locked in loop is designed. The next section describes the basic advantage and source of motivation for using digital delay locked loop in spite of analog delay loop as the topic of thesis work.

2.5 Comparison between Analog DLL and Digital DLL

In the above sections, we have seen a detailed explanation of analog and digital DLL designs, various possible architectures, different type of phase detectors, delay line etc. We have also seen how the basic architecture of analog DLL and digital DLL is different. In this section, we will compare both the DLLs for all the aspects.

- Digital DLL are more adaptable to power supply changes as compared to analog DLL. The reason being the voltage headroom requirement becomes more strict in the analog DLLs when supply voltage decreases. While a digital DLL can work at a very low value of supply voltage as well as long as the supply voltage is sufficient to overcome the noise margins of the devices. Hence, working at a very lower voltage sounds bit tough for analog DLLs.

- Analog DLL involves a complex circuitry which makes it difficult to adapt different technologies as it requires process-specific implementation. If we try to scale down the analog DLLs, it directly impacts the effective output resistance offered by the

circuit and hence affects the delay and intrinsic gain of the circuit. So, reusing the same design over various technologies seems to be difficult. This decreases the capability of the device to be portable. Digital DLLs are more portable compared to analog DLLs.

- Another main advantage is inline with the power consumption. As the supply voltage is decreases. the switching power or the dynamic power decreases. For a digital circuit, power calculation is done as:

$$P = 0.5 \times \alpha \times C \times V^2 \tag{2.17}$$

Hence, power consumption is directly proportional to square of V (voltage supply), whereas in case of analog circuits, power supply consumption is related to V as:

$$P = I \times V \tag{2.18}$$

Hence, it has a linear relation with supply voltage. Since digital DLL can support a lower value of supply voltage, the power consumption also decreases as compared to the analog DLLs.

- Analog DLLs generally provide a better jitter performance as compared to digital DLLs as the accuracy is controlled by a analog signal. But this comes at a price of large chip area, complex circuits and less stable design.

- Digital DLLs are zero-pole systems and hence inherently always stable. It eliminates the need of loop capacitor filter and charge pump, and hence a pole can be avoided, which make a negative feedback system which is always stable.

Although the Digital DLL has more jitter than Analog DLL, its greater simplicity, portability, lower power consumption, its ability to operate at lower voltages, smaller area, more noise immunity, short lock time and synthesizable circuit makes it very attractive for clock alignment applications. [24–26].

Chapter 3

Design of SAR based all digital DLL

This chapter covers the block diagram of the designed delay locked loop along with the detailed description of each module used in the designing. This DLL is designed to have an accuracy of 0.5% across corners, low jitter value, low phase noise value and reduced lock time. This DLL supports a wide range operating frequency range (400 MHz- 1.3 GHz).

3.1 System Architecture

The all digital delay locked loop in this work is designed on TSMC 16nm CMOS technology. The specifications of the design has been shown in the table 3.1.

Table 3.1: Electrical Specification

Technology	TSMC 16nm
Supply Voltage	0.8 V
Reference Clock Frequency	400 MHz - 1GHz
Reference Clock rise/fall time	100 ps
Reference Clock Duty Cycle	50 %
Lock Time	$<$ 1us
Residual Phase Error	$<$ 5 %
R.M.S Jitter	$<$ 10 ps

The figure 3.1 shows the system architecture of the proposed all digital delay locked loop. The SAR based DLL has an advantage of reducing the number of lock cycles as

compared to the conventional counter-based DLL. In this design, the implementation of delay line i.e the coarse blocks and fine blocks are done in such a manner such that amount of current required to control the delay provided by the fine delay element can be reduced and hence the power consumption is reduced. The proposed design has a delay resolution of 5ps, accuracy of ±0.5% and able to produce an output having 50% duty cycle across process, voltage and temperature variations.

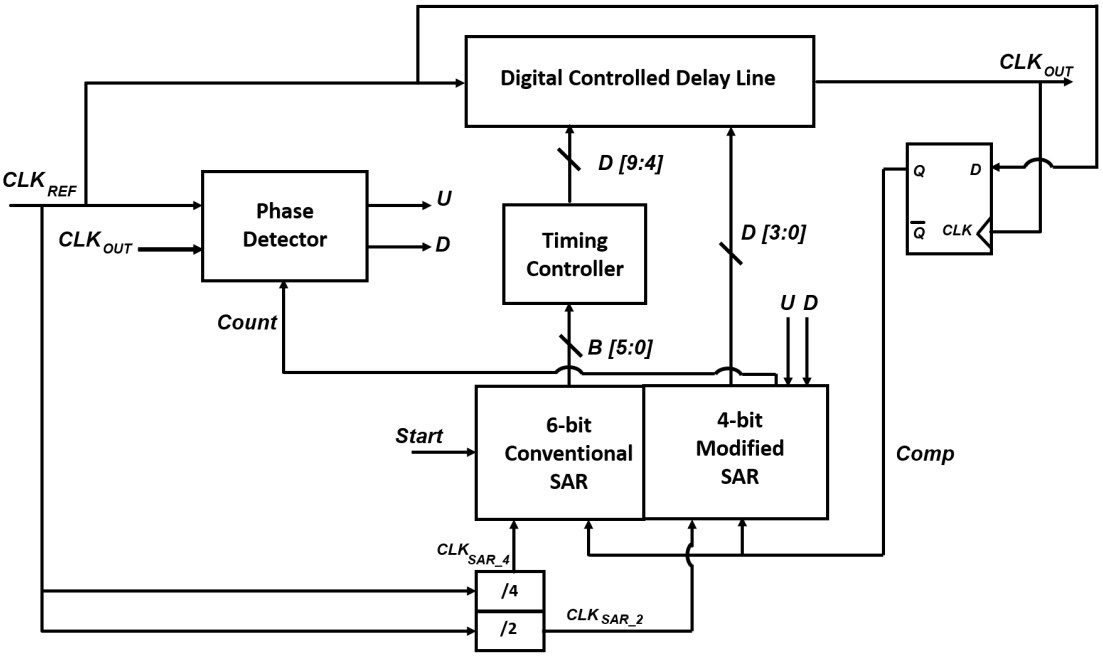


FIGURE 3.1: System Architecture of proposed DLL

3.2 Design Flow of the DLL controller

The flow chart 3.2 explains the complete design flow of the delay lock loop implemented in this work. Initially all blocks are in reset state. All output bits of SAR are initially set to 0, providing the maximum delay to the reference clock input. When the start signal is asserted, the conventional SAR block becomes active. Upon detecting the positive edge of the clock input, the Most Significant Bit (MSB) of SAR i.e. D[9] is set to high. This information is passed to the coarse delay line, which changes the amount of delay offered to the reference clock signal. The output clock from the delay line is compared with the input reference clock through a simple DFF and a COMP signal is generated. This signal decides whether the bit set by the SAR should be held or reset to zero. This way the conventional SAR blocks continues to determine the bits D[9:4]. After deciding the last bit, the SAR asserts a Stop signal, indicating about the completion of its operation. At this point, the output clock is expected to be coarsely locked with respect to reference input clock signal. The timing controller adds an additional

increment of 1 to the bits obtained in order to reduce the delay value by ine unit of its minimum delay offered. This stop signal of conventional SAR acts a start signal for the modified SAR. Modified SAR comprises of a binary search SAR logic and a counter logic. The detailed description about modified SAR will given in subsequent sections. This SAR first works in binary mode and determines the remaining four bits used by fine control delay line which brings the coarsely locked signal in phase alignment with respect to reference signal. The fine delay element is responsible for phase accuracy. The modified SAR also receives the COMP signal and make a decision for the bits. When the stop signal of MSAR is asserted high, the system is expected to be in a locked state. The modified SAR generates a *Count* signal as output after one clock cycle delay with respect to *Stop* signal. This acts as an enable for phase detector. Upon receiving a high Count signal, the phase detector starts operating and samples the reference and output clock signal, compares the signals and generate up and down signals. The value of Up and Down signals is used by the MSAR oeprating in counter mode after the stop signal goes high. If Up is high and Down is low, the counter is incremented and the delay between the two signals is reduced. If the Down signal is high and Up signal is low, the counter value is decremented and the delay is increased. Thus, it works as two way counter. When both the Up and Down signals are either one or zero, the phase detector enters into a hold region and the counter stops counting. At this point, the output clock is expected to be in phase locked with the reference clock cycle and after a small period of time, this output clock can be tapped out as output clock of delay locked loop.

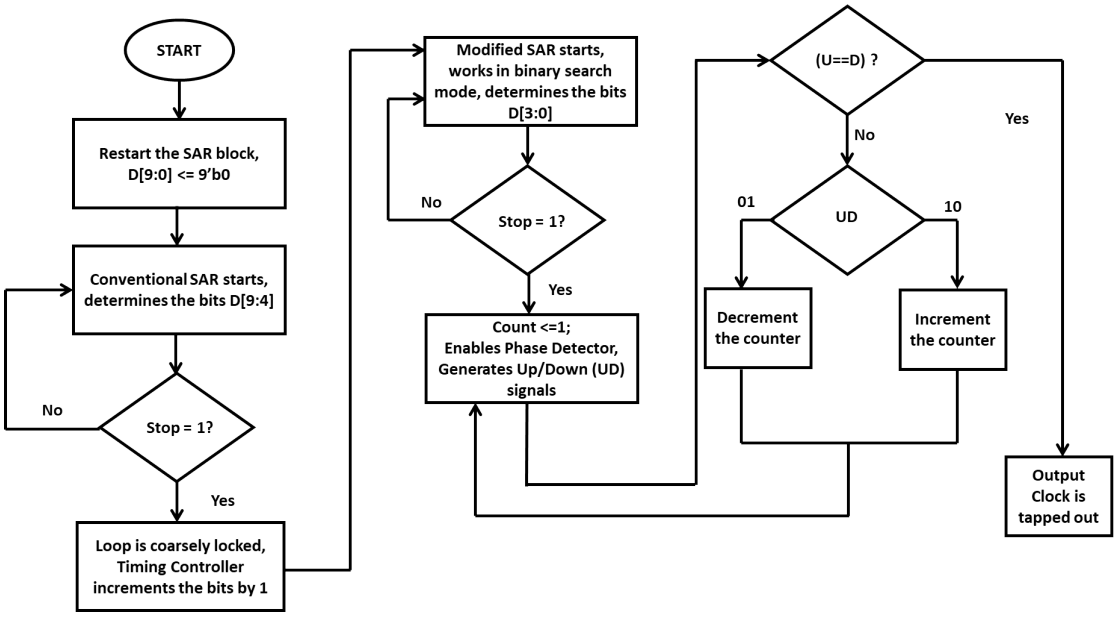


FIGURE 3.2: Design Flow of DLL controller

The figure 3.3 explains the timing diagram of the proposed DLL. The operation works in two modes: binary search mode and sequential search mode. In binary search mode,

conventional SAR will find the code in 6 clock cycles of CLK_{SAR_4}. This CLK_{SAR_4} i generated with the help of divide-by-4 counter as shown in the block diagram. After, conventional SAR completes its operation, the output clock is coarsely locked with the input clock, the stop signal generated by the conventional SAR becomes the start signal for the modified SAR and the MSAR starts its operation to determine the remaining bits. It takes next four clcok cycles of CLK_{SAR_2} to generate the output code value. CLK_{SAR_2} is generated by the help of divide-by-2 frequency divider. When MSAR finishes its operation, the stop signal of MSAR goes high, indicating the end of the binary search mode. In the next clock cycle it enables a high *Count* signal which indicates the start of the sequential search mode. MSAR is now in counter mode and based on the inputs from phase detector (U/D) it starts counting and try to reduce the phase error between the clock signals.

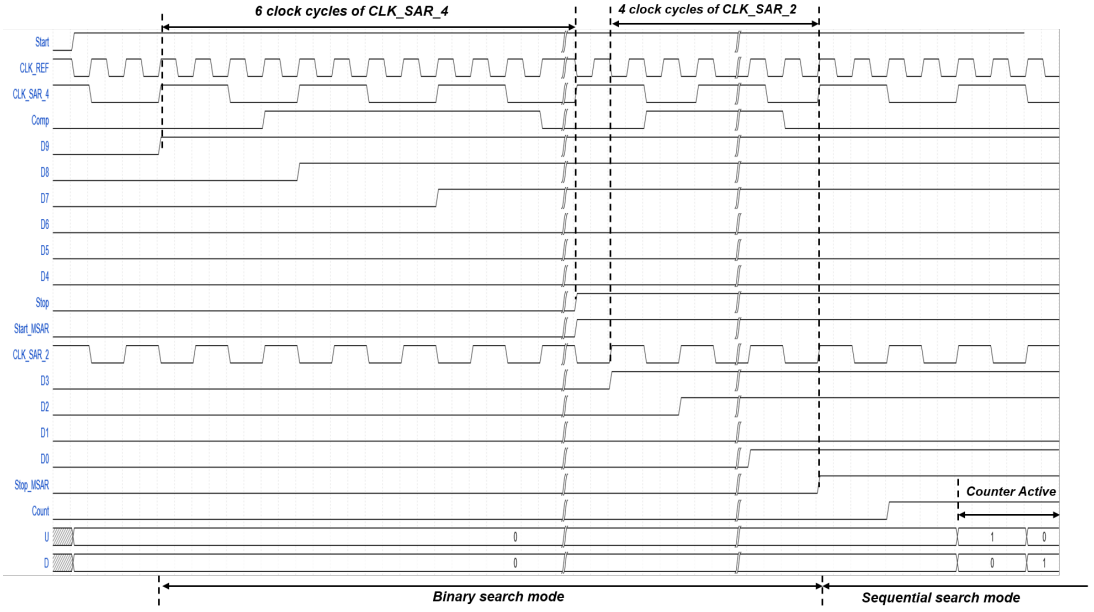


FIGURE 3.3: Timing Diagram of operation of proposed digital DLL

This is how the SAR based all digital delay locked loop works. The following sections will describe about each of the blocks used in this design in depth.

3.3 Phase Detector

The need of the phase detectors, different state-of-the-art techniques used and the various design issues generally encountered to implement phase detectors are discussed in chapter 2. A phase detector can be seen as a black box having two inputs i.e. the reference clock (CLK_{REF}) and the output clock (CLK_{OUT}), generating one or two output signals (namely, Up and Down signals) depending upon what is inside that block box.

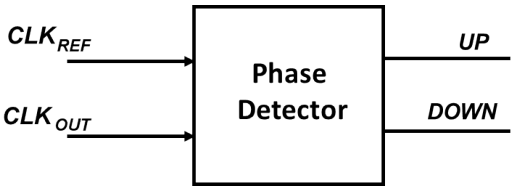


FIGURE 3.4: Phase Detector as black box

A conventional digital phase detector can be a simple DFF having the reference clock as the input signal (D input) and the output clock as the input to clock pin. The phase difference ($\Delta\phi$) between the input and the output clocks is captured at the Q pin of the DFF. An example of such a phase detector is shown in figure 2.6. However, this kind of phase detector has a major drawback. A conventional DLL adjusts by the help of the Up and Down signals produced by the phase detector, to either increase or decrease the delay with respect to the reference signal in order to track the PVT variations and have a closed loop characteristics. The delay cells controlled by digital bits have a finite resolution, i.e. the difference between the delay values of two digital code is a finite number. Due to this, the delay of the DCDL may not be exactly equal to the clock period. This holds true true even for locked condition. In locked condition as well, there can be a small phase error existing between the two signals. Due to this, the phase detector will always generate an error signal and the counter, upon seeing this error value keeps on increment or decrement the count value. Hence, the digital code keeps on changing back and forth. This behaviour is shown in figure 3.5.

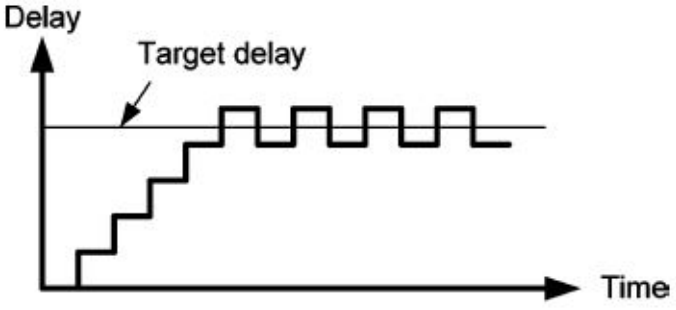


FIGURE 3.5: Code keep on changing back and forth due to finite resolution in a conventional counter-controlled DLL [3]

The horizontal delay line is representing the target delay for the circuit which is equal to one clock period. Since, a finite phase error exists, the counter keeps on changing the digital code value between two values as shown by the pulse around the target delay line. This introduces jitter in the system. In worst case, this jitter value can be equal to twice the minimum resolution of the DCDL. In this case, we have not considered the effect of input clock jitter. When we consider the input jitter in a normal distribution, (say

less than the minimum delay resolution of DCDL), then the target delay is not a single value but becomes a range in which the DLL should lock. However, chances are there that now the digital code, unlike previous case where it was toggling between two values, start changing between two or more values. This is known as dithering phenomenon and shown in figure 3.6. This is responsible for a jitter in the output clock which is equal to the jitter in input clock plus twice the minimum delay resolution. In cases, where input jitter is more than $2T_D$, the code can change between three or more values resulting more jitter at the output node. Thus, a simple DFF as a phase detector is not a good idea, as it cant help in suppressing the dither phenomenon.

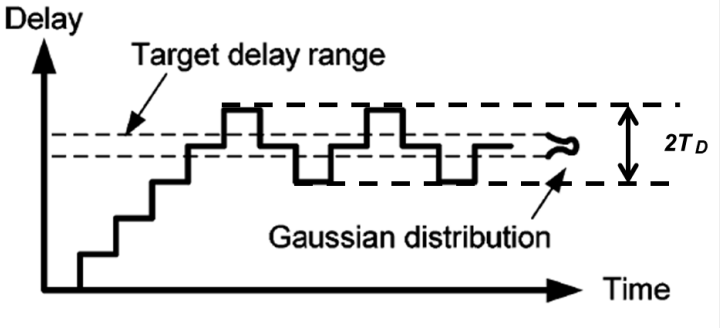


FIGURE 3.6: Output clock jitter and Locking process when input clock has a jitter [3]

To suppress the jitter caused by this dither phenomenon, true single phase clock (TSPC) type DFF are used in phase detectors. TSPC type flip-flops have various advantages like no clock inversion, low clock load capacitance, high speed and reduced clock skew problems. Two such flip-flop are arranged in such a manner that the input of first flip-flop acts as clock for the second flip-flop. Hence, this way we are able to sample the difference between the two signals with respect to each other. This sampling is modelled to Up and Down signals respectively. Figure 3.8 shows the phase detector used in this thesis work. The CLK_{REF} is the reference clock for our system and CLK_{OUT} is the output of the DCDL. The signal *Count* acts a reset to the flip-flops. When *Count* is low, the signals U and D are reset to zero. The phase detector is in disable state. When the *Count* signal is asserted high, the phase detector starts operating. The DFFs sample the CLK_{REF} and CLK_{OUT} and phase error between them is sent to U and D signals.

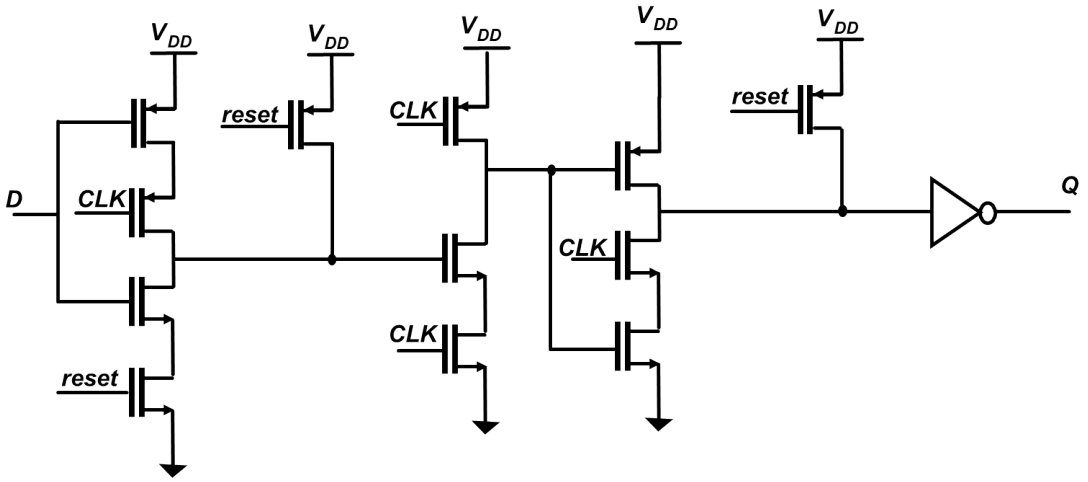


FIGURE 3.7: True Single Phase Clock type DFF with reset

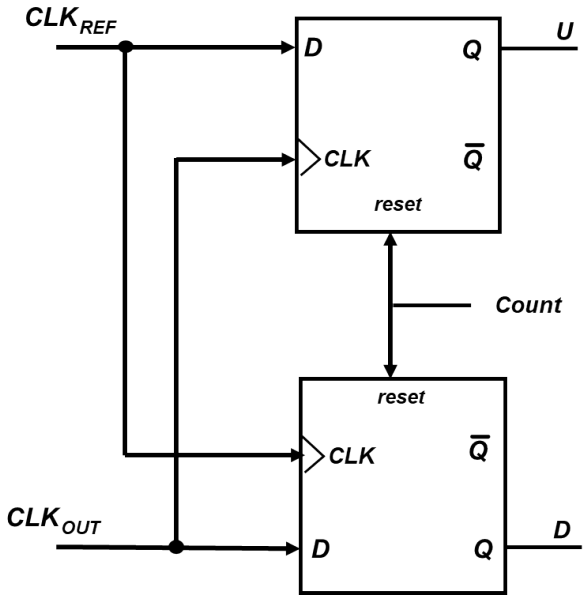


FIGURE 3.8: Phase Detector

When the positive edge of CLK_{REF} comes before the positive edge of CLK_{OUT}, the U signal becomes high. When the positive edge of CLK_{OUT} comes before the positive edge of CLK_{REF}, the D signal becomes high. These signals are input to the modified SAR which upon receiving these inputs acts as a binary counter. When these signals are different, the counter counts up or down. When these signals are same, the counter hold its value and the system comes in hold state. Based on the values of U and D signals and state of counter, we can define a truth table as:

TABLE 3.2: Truth Table

U	**D**	**State**
0	0	Hold
0	1	Down
1	0	Up
1	1	Hold

The hold region depends on the offset of the DFF. Ideally, the transition point for the outputs of DFFs i.e. U and D signals should occur at $\Delta t = 0$, i.e. any change in input signals should be mapped to output signal immediately, however, in practical implementation this does not hold true. This transition depends on how fast the intermediate nodes are changing their values. The driving capabilities of the transistor should be strong enough to affect the gate voltage of the next stage. This causes a timing offset during the transition of the intermediate signal. To avoid this offset, one possible solution could be appropriate aspect ratio of the transistors. But across various process and temperature variations, the transistors will show different driving capabilities and hence affect the transfer characteristics. So, the information from phase detector will be used in such a manner that the counter counts up or down only when up and down signals are different, otherwise it holds the state and does not change the digital code value.

It was stated earlier that the TSPC type DFF is used to suppress the dither phenomenon. Refer to figure 3.9 for the explaination. The two gray afreas represent the hold region for TSPC type flip flop and hence, when due to jitter, the input or output clock enters into that region, the counter is hold mode and suspends its operation. Thus, as shown in the figure, the counter changes its values only outside the gray region and hence chnages its value only between two codes, thus reducing the jitter and hence the dither phenomenon.

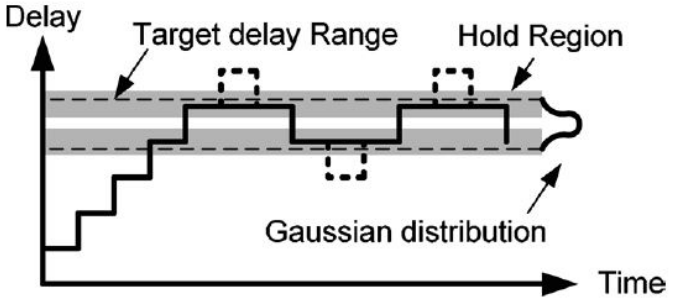


FIGURE 3.9: Dither Suppression phenomenon in TSPC type DFF [3]

Figure 3.10 shows the input and output waveforms of the phase detector. When *Count* is asserted high, phase detector starts sampling the input and output clocks and produce U and D signals. When the negative edge of *Count* is encountered, the phase detector stops working and the values are reset to zero.

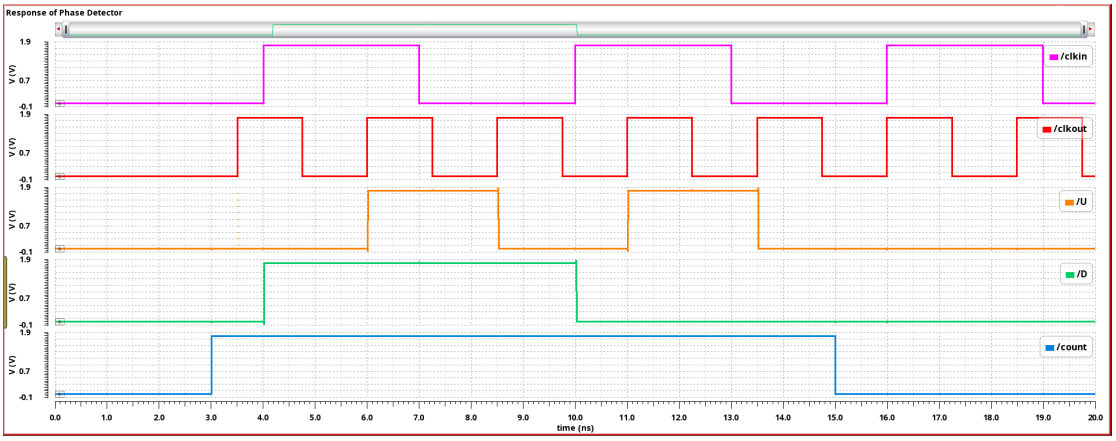


FIGURE 3.10: Input and Output waveforms of the phase detector

3.4 Controller

As mentioned above, the controller is the heart of a digital DLL. It controls the delay line and hence the delay offered by sending the control bits information to the DCDL. In this thesis work, controller comprises of three blocks:

- Conventional Successive Approximation Register
- Modified Successive Approximation Register
- Timing Controller

Each of the above mentioned block are discussed in depth in the below sections.

3.4.1 Conventional Successive Approximation Register

A successive approximation register is presented in [27]. It codes all the possible conversion output for a particular input value, is non-redundant, optimized and simple. The SAR based on the output of a comparison logic, determines the value of each bit of the digital output in a sequential manner. The architecture reported in [28] uses different registers for memory operation and shift operation. While in this thesis work, a non-redundant SAR is used which uses the same register for both the memory and the shift operation and limits the maximum number of flip-flops used.

After receiving a *Start* signal, the SAR sets MSB high and all other bits as zero. So for our 6-bit SAR, B[9:4] = 6'b1000000. This information is passed to the coarse delay line and the delay line sets the delay values as half of the whole delay line. This delayed line is compared with the input clock and based on the output of DFF, the *Comp* signal is set to 1 or 0. If the *Comp* signal is equal to 1, the MSB will retain its value. If the *Comp* signal is equal to 0, the MSB will be reset to 0. So by now, we will be having value of our MSB bit. In the next cycle, SAR operation repeats its operation with a known MSB bit and guessed next to MSB bit. This process repeats until all the bits are known by the help of *Comp* signal. So, the operation of successive approximation register can treated as a sequential finite state machine having state diagram as shown in fig. 3.11.

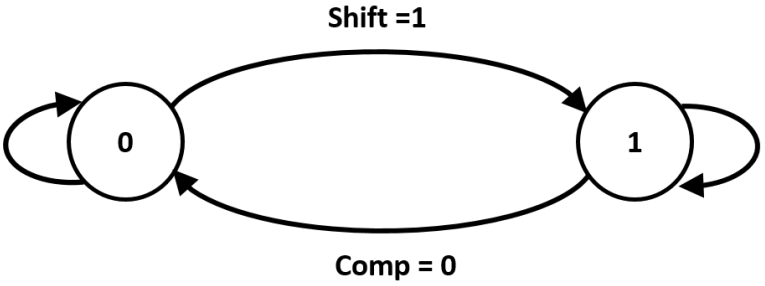


FIGURE 3.11: State Diagram of conventional SAR

The basic structure of the 6-bit SAR is a multi input shift register as shown in figure 3.12, where each multi input shift register is realized by a combination of multiplexer, decoder and a positive-edge triggered DFF. Each state has the possibility of choosing the data inputs among three possible options:

- memorization - the output of the flip flop itself.
- shift right - the output of the $(k+1)^th$ flip-flop
- data load- the output of the comparator.

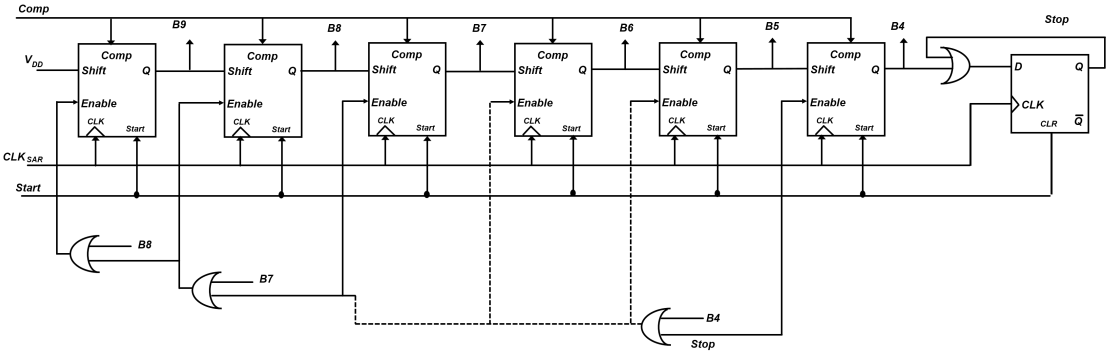


FIGURE 3.12: Six bit conventional SAR circuit

By the help of the multiplexer and decoder as shown in fig 3.13, each flip flop can choose one input among the three inputs possible. The following truth table helps to understand the above mentioned operation well.

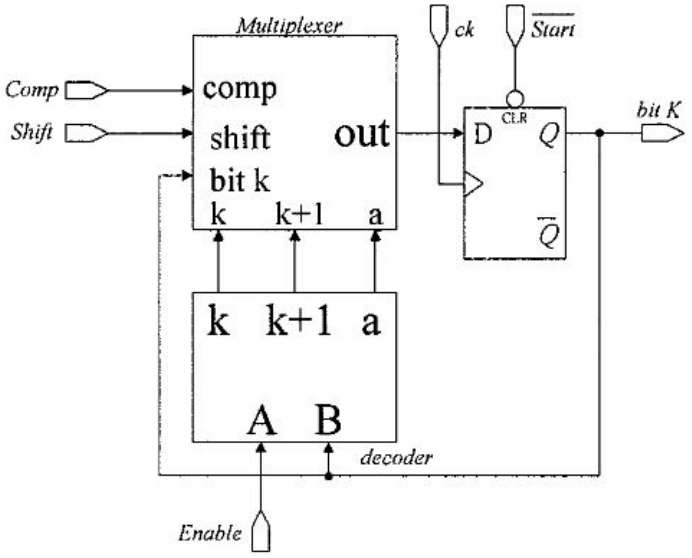


FIGURE 3.13: One bit Shift Register Circuit

TABLE 3.3: Truth Table of SAR

A	B	Operation
0	0	shift right
0	1	data load
1	X	memorization

Figure 3.14 shows the same 1 bit shift register as shown in fig 3.13 in a different manner. The multiplexer is implemented by the help of the AOI (and-or-invert) circuit and decoder part is implemented by the help of NAND gates.

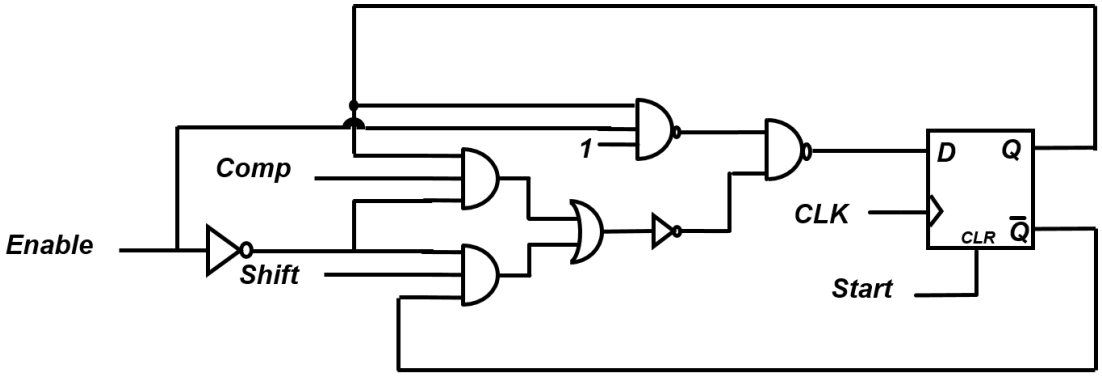


FIGURE 3.14: One bit conventional SAR circuit

Generally, the number of bits of the SAR determines the number of clock cycles needed to lock the circuit. However, for some applications, after receiving the comparison signal result, the system needs some time to set to the value of SAR. This time involves the delay through the delay line, phase detectors, buffers (if any) etc. So, to have a faithful transmission of data, a negative edge triggered divide-by-four counter is required to generate the clock for conventional SAR, to lower down the frequency of operation. This do affect the lock time required for a circuit but for applications, where high frequencies are involved this action is important in order to capture the data faithfully.

3.4.2 Modified Successive Approximation register

The conventional Successive Approximation Register stops working when the binary search is done and a high stop signal is asserted. In this case, the delay locked loop reaches an open-loo configuration and cannot trace any process, voltage, load or temperature variations. So, it is need to modify the conventional SAR in such a manner that even after performing binary operation, it is still able to track any changes and bring the circuit back to locked state, thus forming a closed loop circuit. For this purpose, modified SAR is used. It works both as a SAR and a counter depending upon the inputs U and D from the phase detector. When the signals U *and* D are low, it works in binary search mode, perform SAR's operations, determines the bit D[3:0], helps the circuit to bring the output clock in fine lock with the input clock, generates a signal to enable the phase detector after binary search is done. When phase detector is enabled, based on the values of U *and* D, the MSAR works as a upward or downward binary counter (if U!=D). The figure 3.15 shows the one bit MSAR circuit. It shows the additional circuitry required as compared to 3.14 to implement the counter logic.

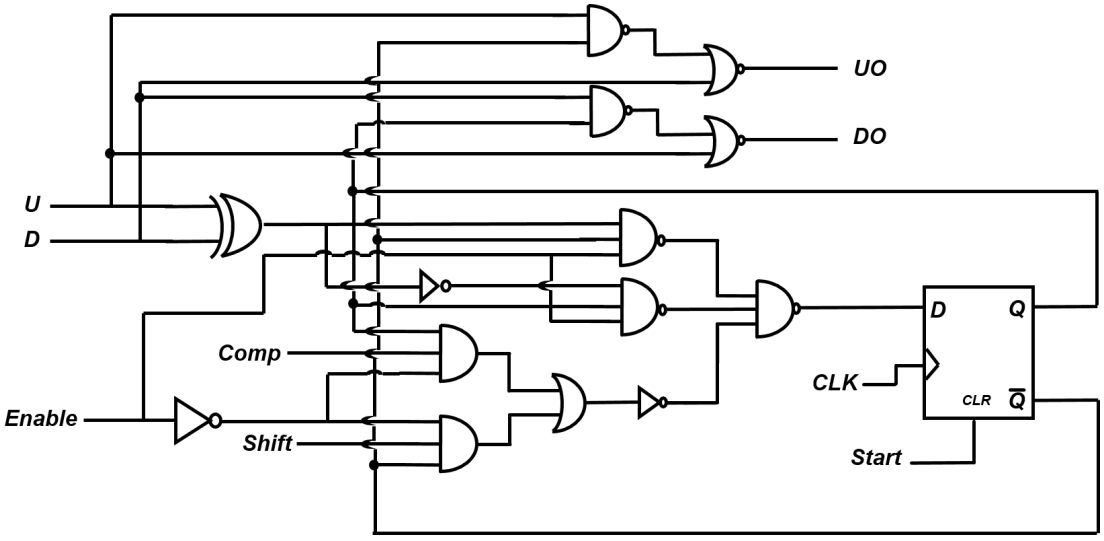


FIGURE 3.15: One bit Modified SAR circuit

The below state diagram explains the operation of the MSAR. The MSAR works identically as conventional SAR when Enable is 0. When Enable = 1, it works as a counter unit and perform the closed-loop operation. Figure 3.17 shows the 4-bit MSAR circuit used in this thesis work. Here, to map the timing delays of clock propagation through various blocks of the circuit, a negative edge triggered divide-by-2 counter is used to lower down the frequency of operation. The bits produced by this circuit is used by fine delay line to adjust the phase error of the circuit. The *Count* signal generated by the circuit helps in accomplishing the closed-loop operation.

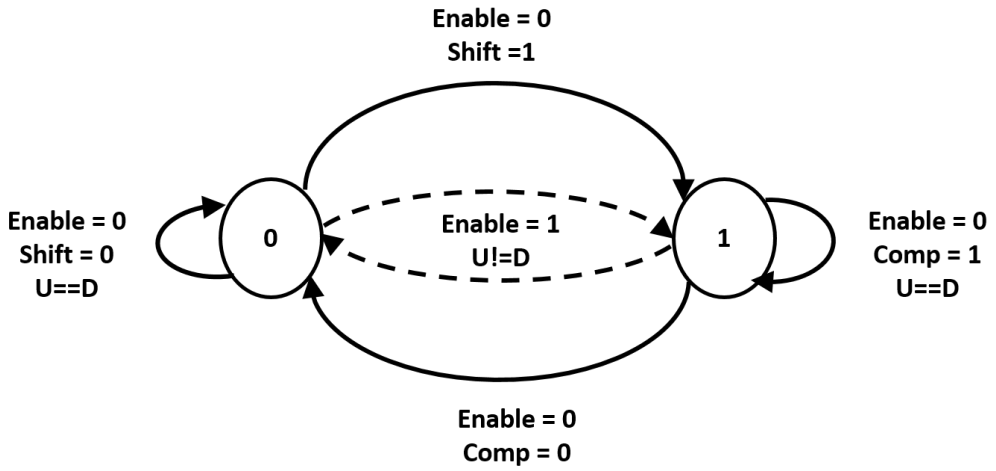


FIGURE 3.16: State Diagram of Modified SAR circuit

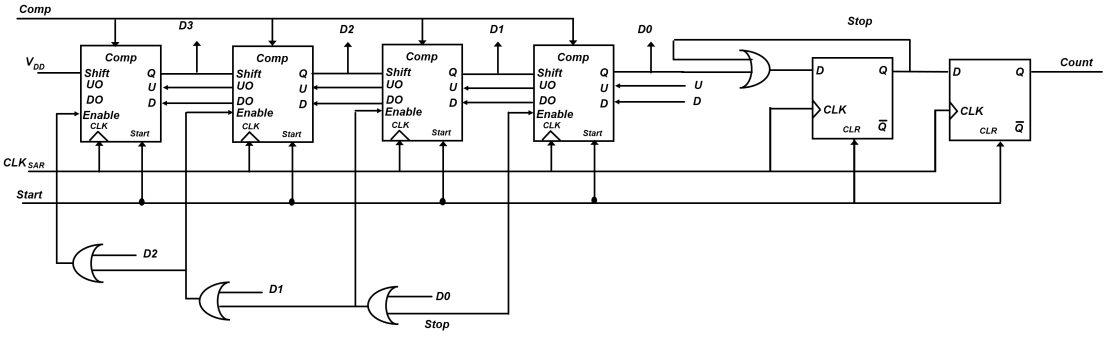


FIGURE 3.17: 4 bit Modified SAR circuit

The following figure shows an example the how the MSAR behaves. First the MSAR performs the binary search operation. D3 bit is set to 1 and other remaining bits are 0. The comparator compares the input and output clock signal and send a high *Comp* signal. Hence, D3 will hold its value of 1. Next, D2 is set high and the same process is repeated. After four clock cycle (clock cycles = no. of bits of SAR), *Stop* is asserted. This stops the binary operation. the final value of code is D[3:0] = 1001. After a clock cycle, the *Conut* signal is asserted which enables the phase detector. After comparing

the phases, we are receiving a high *Down* signal, hence counter will start counting downwards. So the code becomes 1000. In the next clock cycle as well, the *Down* signal is high. So the code becomes 0111. Hence it is working as a downward counter.

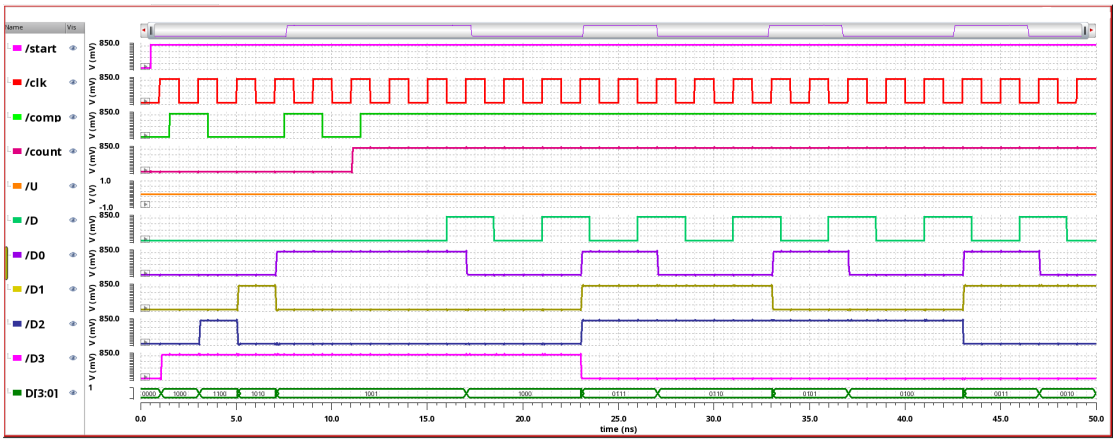


FIGURE 3.18: Output waveforms of MSAR

3.4.3 Timing Controller

This section discusses about the proposed timing controller. This timing controller receives the bits from the conventional SAR and produces output bits which becomes the input of the coarse part of our delay line. The another input to timing controller is *Stop* signal coming from the conventional SAR. Timing controller basically contains a full adder. The inputs to full adder are 1. Latch storing 6 bits information from the conventional SAR, 2. Stop signal 3. Carry bit is tied to zero. The operation of this proposed timing controller is explained in flow chart shown in 3.21.

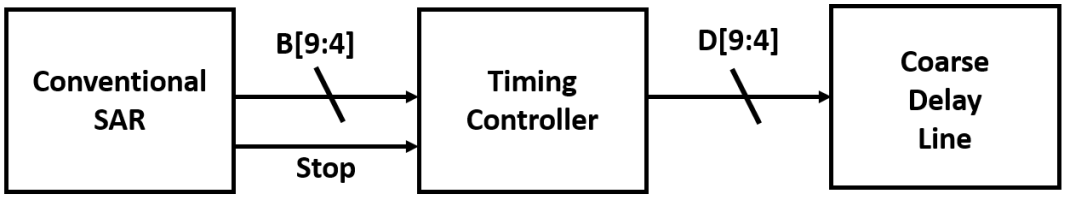


FIGURE 3.19: Timing Controller

The timing controller checks the value of input signal *Stop*. If it is zero, that means SAR is not done with operation yet. So, the bits received from the SAR are passed as it is from the SAR to the coarse line. When the *Stop* signal goes high, the output clock is coarsely locked with the input clock. At this point we add "1" to the bits received. By doing so, we shift the output clock leftwards by decreasing the delay, as there is a monotonically decreasing relation between the delay value and the digital code (will be

explained in next section). So by shifting one bit towards right side, the delay value is decreased by one delay unit value.

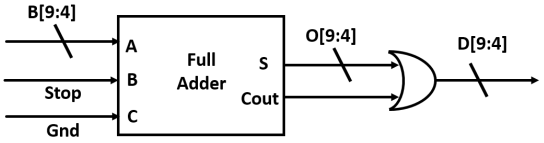


FIGURE 3.20: Inside Timing Controller

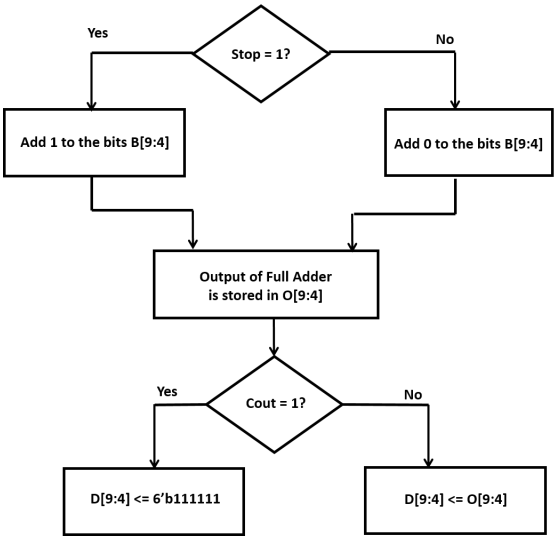


FIGURE 3.21: Flow Chart explaining Timing Controller

When the digital code is already 63 i.e. code is B[9:4] = 6'b111111, then the delay line is already offering minimum delay to the clock signal. Hence, the delay value cant be further reduced. To check this condition, an OR gate is added to the circuit, having one input as Carry out (C_{out}) and other input as six bits binary value. There will be only one case, where the value of C_{out} will be equal to 1, i.e. when the digital code is 6'b111111. In that case, by OR operation we get a 6'b111111 at the output end (in D[9:4]) otherwise, the output register is loaded with bits O[9:4]. In a normal scenario, the worst case delay of fine delay element should be equal to minimum delay offered by coarse delay element, i.e. the 1 LSB delay of coarse delay element. To have a better delay resolution, the minimum delay offered by fine delay element should be as small as possible, such that it can lock to the input signal with a phase error as small as possible. Since, fine delay elements are mostly current controlled inverter. the delay value is controlled by the current flowing in the circuit. In TSMC16nm, to obtain a very small very value of delay we need a large amount of current to flow through the circuit. This increase power dissipation by the circuit. In order to reduce that we have intentionally removed one delay unit value from the clock signal. By doing so, we are allowing our fine delay element to have a range of delay value from twice the value of 1 LSB of coarse delay line to the 1 LSB value. So, if suppose we want our DLL to have a resolution of 5ps, then the minimum delay of fine delay element should be 5ps. But now, instead of settling our fine block at 5ps for minimum delay,w e are allowing it to have a delay of say 50ps (1 LSB of coarse) which can be easily achieved by a small value of current. Hence, the current required in the circuit and the power consumption decreases. Thus,

with the help of this proposed circuit, the accuracy of the circuit is improved, a small value of fine resolution is guaranteed and power consumption also reduces.

3.5 Digital Controlled Delay Line

A digitally controlled delay line mainly has two components namely:

- Coarse Delay Line
- Fine Delay Line

Each of them are described in detail in the sections below.

3.5.1 Coarse Delay Line

As explained in section 2.4.3.1, the coarse delay line is formed by cascading standard CMOS logic gates where intermediate outputs are tapped and routed through high-fan in multiplexers. Another way of realizing a digitally controlled unit is by using two different delay times as input to the multiplexer as shown in figure 3.22. By cascading such units, a large value of delay can be realized and hence the tuning range of the delay line increases. But this also increase the intrinsic delay i.e. minimum delay of the circuit which limits the maximum operating frequency for the DCDL. Hence, there is a trade-off between tuning range and the ability to be used for high-speed applications.

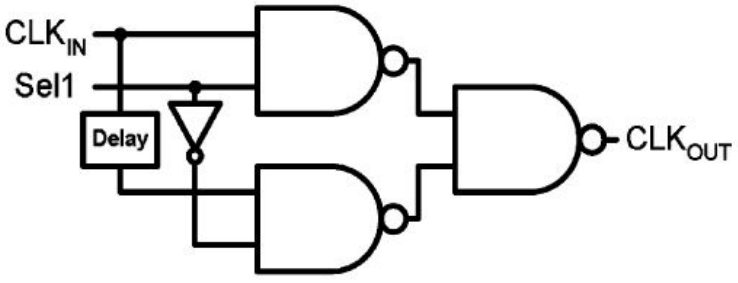


FIGURE 3.22: Conventional digital delay unit [2]

The coarse delay line should be designed in such a manner that it covers the entire clock period in best case. So, the maximum delay for the coarse delay line in the best case is equal to one clock period and this clock period corresponds to minimum operating frequency or maximum time period of our DLL (in our case 2500ps as 400 MHz). In

this thesis work, a configuration shown in 2.30 has been used. In this configuration, the NAND cells are connected in telescopic fashion. We have used a delay line having 64 stages, where each stage has three NAND gates as shown in the box in fig 3.23. Each stage is controlled by the thermometer bits. A 6 bit binary to 64 bit thermometer decoder is used to convert the binary bits to thermometer bits to control the delay line. With the change in one bit of binary input, one delay step will be added to the delay value. To determine the one delay step, we should understand the following example:

- When D[9:4] = 6'b111111, the 64 bit thermometer code will be 64'b1...1. So, the input clock (CLK_{REF}) is routed through $NAND_1$ and $NAND_5$. Hence, the delay will be equal to delay offered by the two NAND gates i.e. delay of $NAND_1$ + $NAND_5$. This defines the 1 LSB delay of the coarse delay line.
- When D[9:4] = 6'b111110, the 64 bit thermometer code will be 64'b01...1. So, the input clock (CLK_{REF}) is now routed through $NAND_2$, $NAND_3$, $NAND_6$ and $NAND_5$. Hence, the delay will be equal to delay offered by the two NAND gates i.e. delay of $NAND_2$ + $NAND_3$ + $NAND_6$ + $NAND_5$.

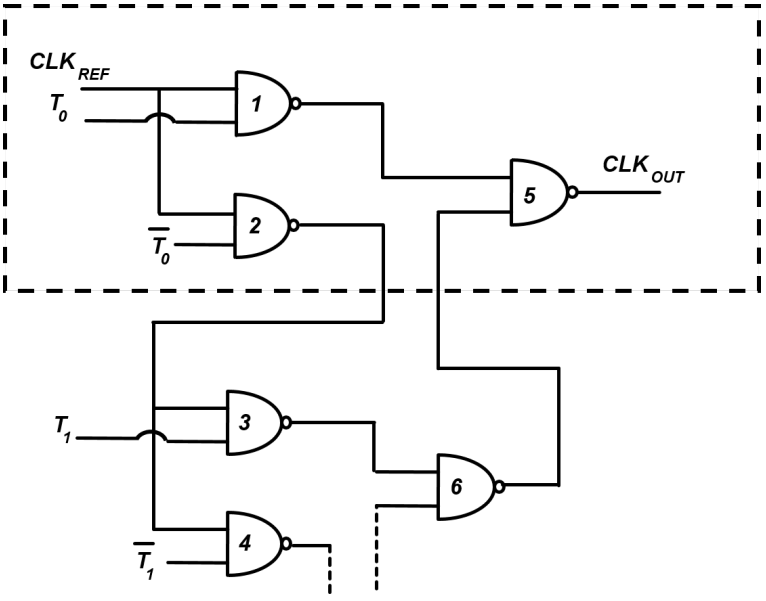


FIGURE 3.23: NAND cells connected in telescopic fashion

Hence, for one bit change in binary input the delay of two NAND gates are getting added. So the delay resolution (d_r) or the delay step for coarse delay line is equal to delay of two NAND gates. The main advantage of this structure is that to support a lower frequency value, we just need to add more stages in the same fashion. Adding more stages will not change the minimal or intrinsic delay of the circuit and hence, the maximum operating frequency will not be affected (unlike the case in which we were

cascading the delay units serially and limiting the maximum operating frequency by increasing intrinsic delay). In such a delay line, both the delay resolution and minimal delay is equal to delay of two NAND gates. The only disadvantage here could be the area required for 6 bit to 64 bit binary to thermometer decoder. As more stages will be added, a large number of bits will be required which will increase the area of the decoder. Fig 3.24 shows the complete circuit used for coarse delay line in this thesis work.

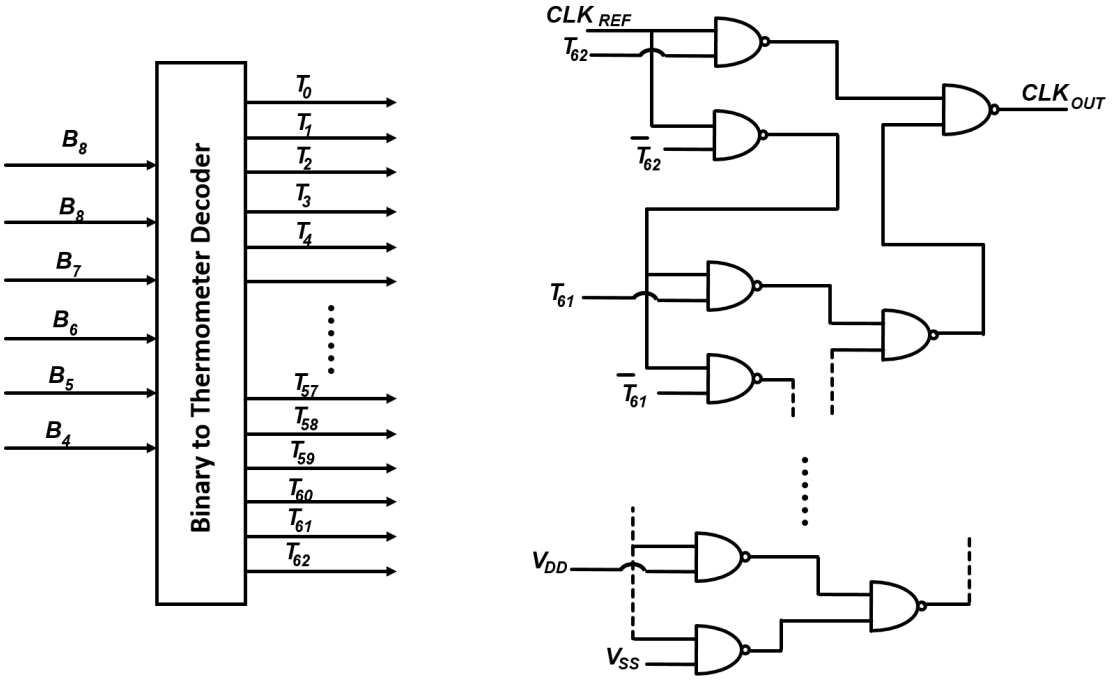


FIGURE 3.24: Coarse Delay Line

3.5.2 Fine Delay Line

The design of fine delay line mainly depends on the current-starved inverter technique. The circuit used for fine delay element in this thesis work is shown in fig 3.25. This proposed fine delay block is a modified version of circuit mentioned in [23]. This circuit has a monotonically decreasing delay behavior with respect to the input digital code. The current is controlled via the bits D[3:0] which fine block receives from the modified SAR.

The working of this circuit is as follows: When the transistor M_2 turns on, the capacitor present at its output node starts to discharge. The current through this capacitor will be controlled by transistor M_8 and M_5 which are acting as a current source. The gate voltage of M_{14} will decide the amount of the current passing through M_5 and M_8. The same current will be mirrored by the pMOS M_6 and the gate voltage of M_6 mirrors the

current in transistor M_7. Thus, the current through M_7 and M_8 controls the discharging rate of the capacitor present at the output node. The gate voltage of M_{14} is decided by the drain current which is controlled by the pMOS transistors M_9- M_13. The digital bits D_0 to D_3 decides whether to switch on or off the devices. The maximum delay will be decided by transistor M_9. It is done by setting the W/L ratio of the pMOS device. At this point, digital code is 4'b0000. The devices M_{10}-M_{13} are sized in the manner described by equation 3.1.

$$\frac{W}{L}_{M_i} = \frac{2^{i-1}}{2^N - 1} . \frac{W}{L}_{M_9} \tag{3.1}$$

where i = 1,2,3, ...N and N = 4.

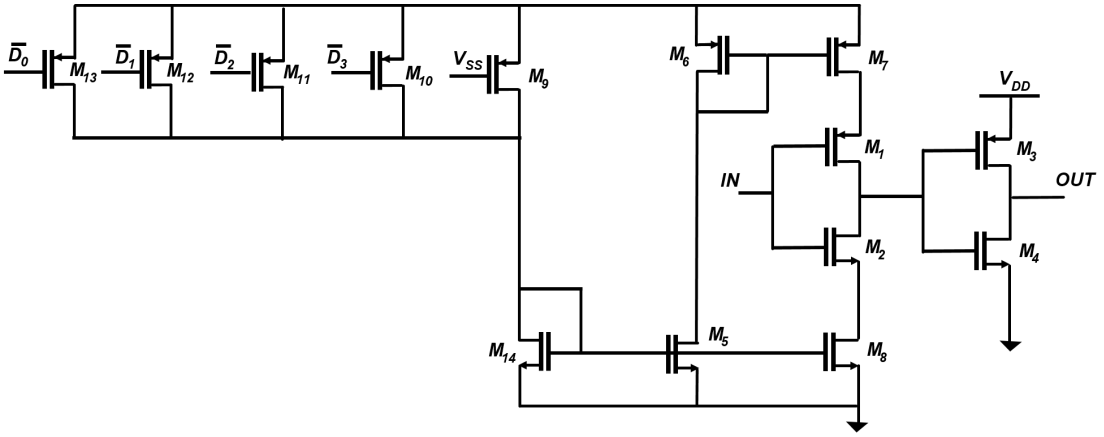


FIGURE 3.25: Fine Delay Element

The delay profile for the above mentioned fine delay line has been shown in result section. This fine delay element is responsible for phase accuracy for a delay locked loop and also defines the delay resolution of the DLL.

3.6 Conclusion

The detailed analysis and design considerations for all the building blocks required for a successive approximation register based DLL were discussed in this chapter. A novel timing controller circuit which helps in improving performance by reducing the current consumption of the circuit has been discussed . This improvement has been achieved with very less area overhead. By adding an additional circuit, which is a simple 6 bit full adder and an or gate, we have reduced the dimensions required by the transistors in the fine delay line by a great extent. This is because, before the full adder circuit, the current consumption by the circuit reached upto a value $200uA$. To support such a high value of current, lowe-resitive devices were needed. This value had been reduced

to $40uA$ by the timing controller block added. Block level and system level simulation results will be shown in the next chapter.

Chapter 4

Results

The simulation results which defines the proposed delay locked loop will be presented in the course of this chapter. For a DLL, the lock time, the jitter performance, the power consumption and the phase error between the input and the output clocks after the locking is achieved are considered as the main parameter for analyzing the performance of a digital DLL. The schematics are designed in Cadence Virtuoso and in TSMC 16nm FinFET technology. The characterization has been done on basis of the variables such as supply voltage (10%) and temperature variation (-40°C to 125°C). Other variations could be due fabrication process variation such as V_{th} and C_{ox} variation. The combination of PMOS and NMOS could be fast fast, fast slow, slow fast and slow slow. All permutations of these variations are considered for the simulation and these sum upto 9 corners. PSS and Pnoise simulations are performed in order to determine the jitter performance of the proposed DLL.

4.1 Linearity of Delay Line

In a digital DLL, the delay line offers the quantized delay steps. A DLL is characterized by the help of the linearity offered by the delay line. Between every two consecutive codes, there should a finite amount of delay present and this number should be same for one corner across all the digital codes given as input to the delay line. The behaviour across corners should be either monotonically decreasing or monotonically increasing for all the digital bits. In this thesis work, a monotonically decreasing delay value with respect to increasing digital codes has been implemented, th input code 0 will be having the maximum delay and the input code $2^N - 1$ will be having minimum delay.

4.1.1 Coarse Delay Line

The figure 4.1 shows the linearity behaviour for the coarse delay line across the best (FF-125), typical (TT-27) and the worst (SS-n40) corners. For a 6 bit input, the input codes will vary from 0 to 63. As explained in previous chapters, a coarse delay line is designed in such a manner that for the best case i.e. FF-125, the coarse delay line should be able to cover the entire clock period. The operating frequency for our system is 400 MHz and hence the time period is 2500ps. It can be seen from the figure 4.1 that in FF125 corner the maximum delay is 2500ps and then linearly decreasing. For FF125, 1 LSB corresponds to $38ps$.

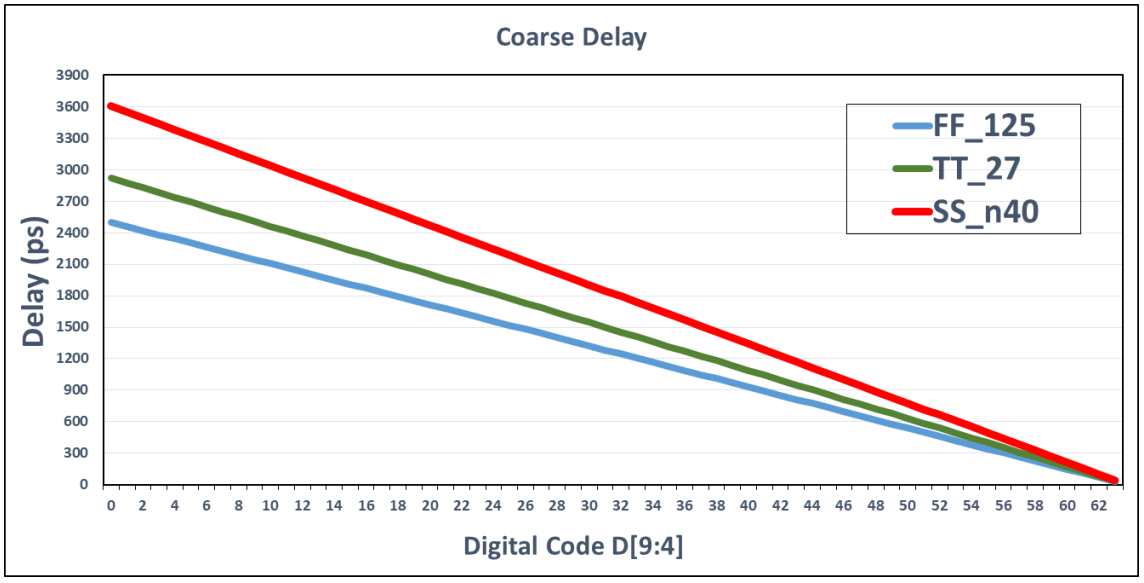


FIGURE 4.1: The delay profile of coarse delay line for proposed DLL across best, typical and worst corners

4.1.2 Fine Delay Line

The figure 4.2 shows the linearity behaviour for the coarse delay line across the best (FF-125), typical (TT-27) and the worst (SS-n40) corners. For a 4 bit input, the input codes will vary from 0 to 15. Conventionally, a fine delay line is designed in such a manner that the worst case delay for FDl i.e. SSn40 with digital code 0 should be equal to 1 LSB of coarse delay line in worst case (i.e. SSn40). From figure 4.1, it can be seen that 1 LSB for SSn40 is 53ps. So, the maximum delay of the fine delay line should be equal to 53ps

The delay resolution for fine delay line is defined by eq. 4.1

$$\Delta\tau = \frac{1LSBofcoarsedelayline}{2^N - 1} \tag{4.1}$$

where, N stands for the number of bits controlling the fine delay line.

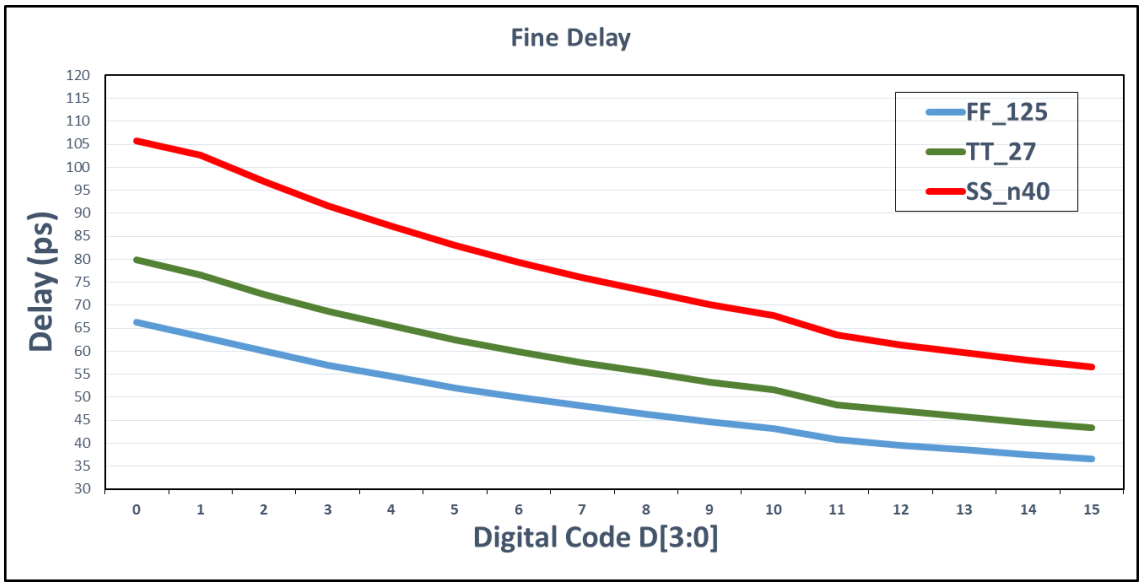


FIGURE 4.2: The delay profile of fine delay line for proposed DLL across best, typical and worst corners

So, the delay resolution for our system should be around 4 ps. But to achieve, such a low value of delay, a large amount of current is required to control the fine delay line. It was seen that in TSMC16nm FinFET even with a very large value of current, a delay value of 4ps could not be achieved. So, a timing controller has been added in order to aid this problem. By the help of timing controller, the FDL now has a maximum delay equal to twice the value of 1 LSB of coarse delay line. The minimum delay, however, should not go below the 1 LSB value. In figure 4.2, it can be seen that for SS-n40, the worst case or maximum delay is around 106ps, which is twice the value of 1 LSB of coarse delay line for SS-n40.

4.2 PSS and Pnoise Analysis

PSS and Pnoise analysis are methods to solve large signal steady state time domain response of a system. For systems having frequency translation effects, these are important to analyse the phase noise of the system. Periodic Steady-State (PSS) Analysis evaluates the periodic steady-state behaviour of a circuit for a defined frequency known as beat frequency. This analysis should be carried first to determine the operating point of the circuit which will be used by Pnoise analysis as the starting point. A desirable value of Pnoise for a circuit is $<$ -70dBc/Hz. As shown in figure 4.3, the designed DLL in this thesis work has a phase noise of $<$ -76.5 dBc/Hz.

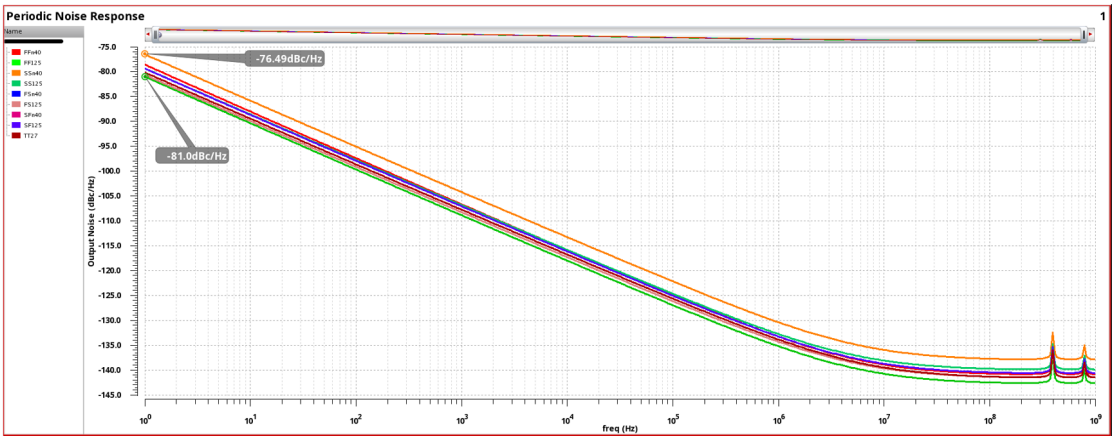


FIGURE 4.3: Phase Noise of DLL across corners for V_{CC}= 0.8V

4.3 Lock Time

One of the most important aspect of any delay locked loop is the lock time or lock cycles delay locked loop takes. The table 4.1 shows the lock time (or lock cycles) required by the delay locked loop across corners.

TABLE 4.1: Lock Time across corners for V_{CC}=0.8V

Corner	Lock time (in ns)	Lock time (in clock cycles)
FF n40	80	32
FF 125	80	32
FS n40	90	36
FS 125	85	34
SF n40	95	38
SF 125	90	36
SS n40	95	40
SS 125	90	36
TT 27	80	32

4.4 Current profile of FDL

The fine delay line controls the delay offered by the help of the current generated in the circuit. Figure 4.4 shows how by using the proposed timing controller circuit, the current required by the fine delay line has reduced. The result shown is performed for V_{CC} = 0.8V, temp = -40 and process corners is slow NMOS and slow PMOS.

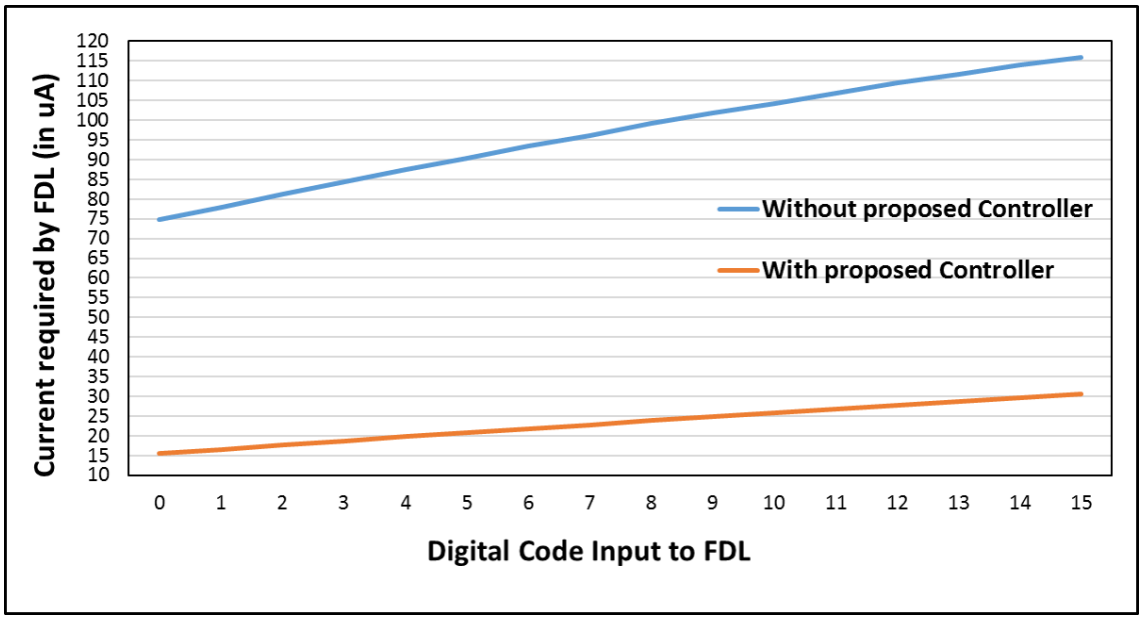


FIGURE 4.4: Current profile of FDL with and without proposed controller for SS -40

4.5 Transient Response

Figure 4.5 shows the *Start* signal showing the beginning of operation, input clock, output clock, binary codes (D[9:0] and B[9:4]), clock output of coarse delay line, *Stop* signal of the MSAR block indicating the end of binary search mode, *Count* signal, indicating the starting of counting or sequential search mode and 2 bit Up and Down signal as UD.

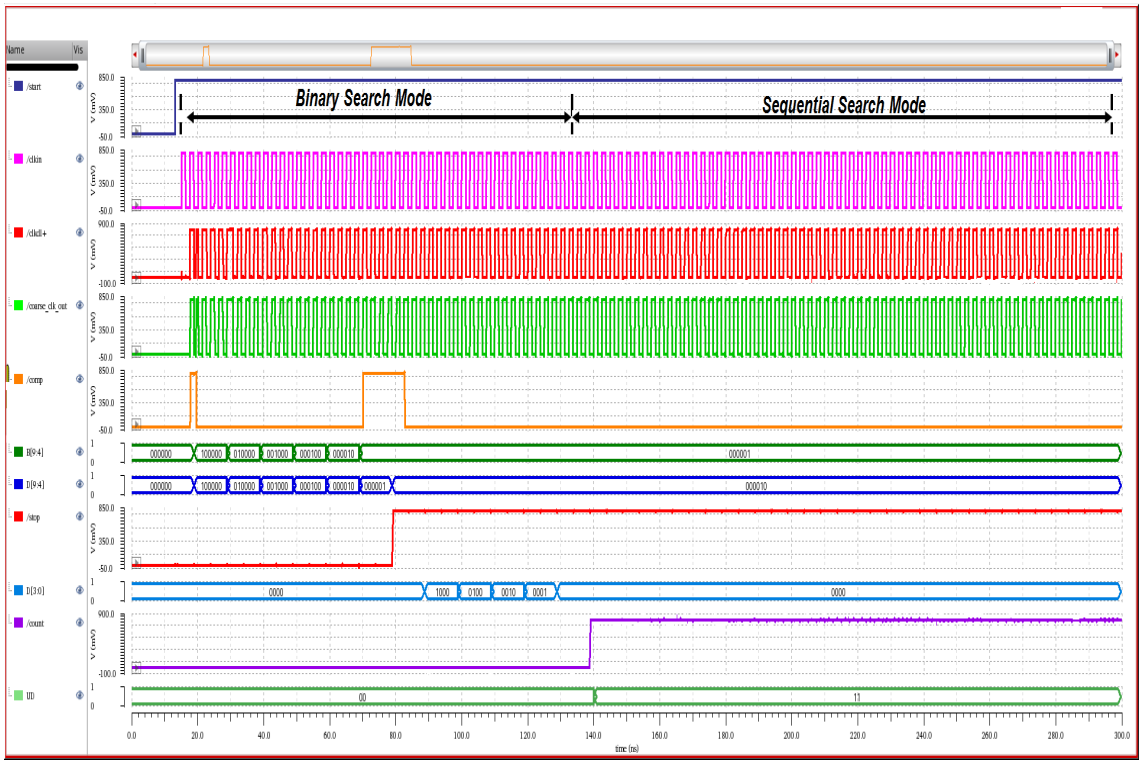


FIGURE 4.5: Transient response of DLL showing locking, locked and sequential search region

Figure 4.6 shows the delay difference between in the input and output clocks for locked condition in corner FF 125 at voltage supply of 0.8V

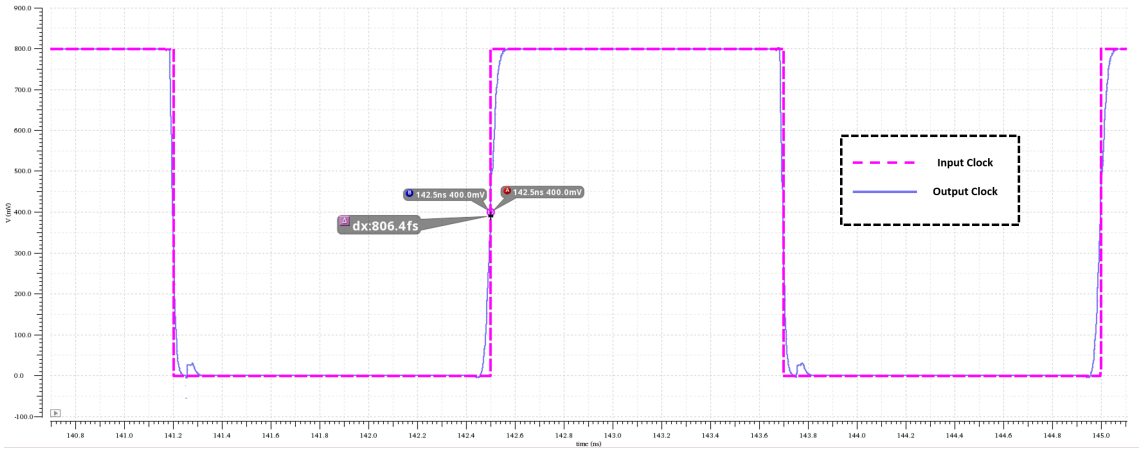


FIGURE 4.6: Delay difference between input and output clocks for locked condition

4.6 Accuracy of DLL

In DLL, another important parameter is the residual phase error between the input and the output clocks after the DLL is in locked condition. This phase error in DLL could

be because of the limited resolution of DLL, the setup and hold times of phase detector, the setup and hold time of comparator circuit used by the SAR etc. This value should be as small as possible as this could be a source of jitter in the output clock. Usually, the DLL is expected to have ±5% accuracy. But in this thesis work, we are able to achieve an accuracy of ±0.5% across corners. The accuracy results shown in table 4.2 were obtained using transient analysis done for a substantial amount of time after the DLL has achieved locked by considering variations in process and temperature.

TABLE 4.2: Residual phase error across corners for V_{CC}=0.8V

Corner	**Residual phase error (in ps)**	**Phase error percentage (in %)**
FF n40	-0.337	0.0135
FF 125	-0.807	0.0323
FS n40	-10.89	0.4356
FS 125	-2.98	0.1192
SF n40	10.3	0.412
SF 125	4.58	0.1832
SS n40	-1.54	0.0616
SS 125	4.28	0.712
TT 27	3.43	0.1372

4.7 Comparison with State-of-the-Art work

The tables below show the comparison of the performance metrics of the proposed SAR based digital DLL with the other State-of-the-Art digital DLLs available.

The tables 4.3 below show the comparison of the performance metrics of the proposed SAR based digital DLL with the other State-of-the-Art analog DLLs available. This table concludes the performance comparison between analog DLLs and digital DLLs. The jitter performance of analog DLLs are better than digital DLLs but the other advantages of digital DLL like flexibility, portability, ease of design, lower power consumption, ability to operate at lower supply voltage etc makes digital DLL an attractive choice for advanced technology nodes.

Table 4.3: Performance Summaries and Comparisons

	[14]	[29]	[30]	This work
Category	Analog	Analog	Analog	Digital
Technology	0.18μm CMOS	0.35μm CMOS	0.18μm CMOS	TSMC 16nm
Supply Voltage	1.8V	3.3V	1.8V	0.8V
Operating Frequency	2 GHz	1.1 GHz	60-760 MHz	0.4-1.3GHz
R.M.S Jitter	1.6 ps	2ps	5ps	6.094ps
Pk-Pk Jitter	13.1 ps	14.6ps	28ps	20.76ps
Power Consumption	12 mW	42.9mW	63mW	240.64uW

The tables 4.4 below show the comparison of the performance metrics of the proposed SAR based digital DLL with the other State-of-the-Art digital DLLs available. With a very low power consumption and reasonable jitter value, a delay resolution of around 5 ps is obtained for a frequency of 400MHz. Usually, for such a low frequency, the delay resolution comes around 10 ps. The performance has been compared with [2], [8], [31] [32].

Table 4.4: Performance Summaries and Comparisons

	[2]	[8]	[31]	[32]	This work
Category	Digital	Digital	Digital	Digital	Digital
Technology	0.18μm CMOS	0.13μm CMOS	65nm CMOS	55nm CMOS	TSMC 16nm
Supply Voltage	1.8V	1.2V	1.1V	1V	0.8V
Operating Frequency	40-550 MHz	1.5-3.3 GHz	400-800 MHz	0.1-2.5GHz MHz	400MHz-1.3 GHz
Delay Resolution	10ps	10ps	X	5ps	5ps
Lock Time	14 cycles	38 cycles	38-41	<24 cycles	40 cycles
R.M.S Jitter	3.96ps	1.96ps	4.8ps	0.24ps	3.094ps
Pk-Pk Jitter	12ps	12-.6ps	26.26ps	3ps	14.6ps
Power Consumption	5.4mW	7mW	3.52mW	1.96mW	240.64uW

Chapter 5

Conclusion

The designs and aberrations of the digital delay locked loop were thoroughly analyzed; different topologies of delay locked loop were studied to obtain a simple and robust circuit which could accomplish all the proposed specification.

The thesis addresses the design considerations, circuit implementations and the detailed analysis of the delay locked loop. Firstly, the basic architecture differences of two types of delay locked loops i.i analog DLL and digital DLL were discussed. A thorough study of various topologies possible to implement the analog DLLs and digital DLLs were presented. The various advantages and disadvantages of both types of DLL were highlighted.

Secondly, the proposed design, the challenges faced while designing and the techniques to overcome those challenges were discussed. Each block of the delay locked loop used in this thesis work was discussed in depth. The concept of dithering phenomenon and how to solve it by designing TSPC type DFF based phase detector was explained. The Successive Approximation Register block has been used as controlling mechanism to reduce the lock time. The digital delay line used has a resolution of around 5ps. Usually, the DLLs operating for a lower frequency has a higher delay resolution due to the minimum delay offered by the coarse delay line. Also, to reduce the amount of current that should be supported the fine delay line to control the delay value has been reduced by the help of a proposed timing controller. This DLL design works on TSMC 16nm on a voltage supply of 0.8V supporting process and temperature variations. The lock time supported by system is around 120 ns (40 clock cycles) where as in state of the art study the counter analog part takes around 5us for the similar frequency. This work has a power consumption as low as 250uW. The digital blocks designed here are simple and can be easily ported to other technologies. This work supports an accuracy

of 95.5 as it has a residual phase error percentage as 0.5% (which comes around 1.5° phase difference) between the input and output clocks after the system is locked.

5.1 Future Work

In the current work, the output frequency is same as the input reference frequency. This circuit can be converted to a multiplying digital DLL by adding an extra circuit which includes multiplexer and a select logic circuit which helps the multiplexer to decide whether the input should be reference clock edge or output of quasi ring oscillator formed by the delay elements. This select logic also helps the phase detector to detect the phase error only between the reference clock edge and the multiplied output from the delay line. With the help of this, the advantages of PLL i.e. frequency synthesization and advantages of DLL i.e no jitter accumulation can be clubbed into one circuit.

Bibliography

[1] Cheng Jia. *A delay-locked loop for multiple clock phases/delays generation*. PhD thesis, Georgia Institute of Technology, 2005.

[2] Rong-Jyi Yang and Shen-Iuan Liu. A 40–550 mhz harmonic-free all-digital delay-locked loop using a variable sar algorithm. *IEEE Journal of solid-state circuits*, 42 (2):361–373, 2007.

[3] Rong-Jyi Yang and Shen-Iuan Liu. A 2.5 ghz all-digital delay-locked loop in 0.13μm cmos technology. *IEEE Journal of Solid-State Circuits*, 42(11):2338–2347, 2007.

[4] Eric R Booth. *Wide Range, Low Jitter Delay-locked Loop Using a Graduated Digital Delay Line and Phase Interpolator*. PhD thesis, Boise State University, 2006.

[5] Thucydides Xanthopoulos. *Clocking in modern VLSI systems*. Springer Science & Business Media, 2009.

[6] Aditya Mittal Deepak Nagariya. Beginners Guide To Clock Data Recovery. `https://protocol-debug.com/2017/02/01/beginners-guide-to-clock-data-recovery/`, 2017. [Online; accessed 26-June-2018].

[7] Ming-ta Hsieh and Gerald E Sobelman. Architectures for multi-gigabit wire-linked clock and data recovery. *IEEE Circuits and systems magazine*, 8(4), 2008.

[8] Erkan Bayram, Ahmed Farouk Aref, Mohamed Saeed, and Renato Negra. 1.5–3.3 ghz, 0.0077 mm2, 7 mw all-digital delay-locked loop with dead-zone free phase detector in 0.13\mu\text {m} *cmos*. *IEEE Transactions on Circuits and Systems I: Regular Papers*, $65(1) : 39 - -50, 2018$.

[9] Jong-Tae Kwak, Chang-Ki Kwon, Kwan-Weon Kim, Seong-Hoon Lee, and Joong-Sik Kih. A low cost high performance register-controlled digital dll for 1 gbps/spl times/32 ddr sdram. In *VLSI Circuits, 2003. Digest of Technical Papers. 2003 Symposium on*, pages 283–284. IEEE, 2003.

[10] Mark G Johnson and Edwin L Hudson. A variable delay line pll for cpu-coprocessor synchronization. *IEEE Journal of Solid-State Circuits*, 23(5):1218–1223, 1988.

[11] Behzad Razavi. *Design of analog CMOS integrated circuits.* , 2001.

[12] Young-Jin Jeon, Joong-Ho Lee, Hyun-Chul Lee, Kyo-Won Jin, Kyeong-Sik Min, Jin-Yong Chung, and H-J Park. A 66-333-mhz 12-mw register-controlled dll with a single delay line and adaptive-duty-cycle clock dividers for production ddr sdrams. *IEEE journal of solid-state circuits*, 39(11):2087–2092, 2004.

[13] Atila Alvandpour, Ram K Krishnamurthy, Daniel Eckerbert, Stuart Apperson, Bradley Bloechel, and Shekhar Borkar. A 3.5 ghz 32mw 150nm multiphase clock generator for high-performance microprocessors. In *Solid-State Circuits Conference, 2003. Digest of Technical Papers. ISSCC. 2003 IEEE International*, pages 112–482. IEEE, 2003.

[14] Koichiro Minami, Masayuki Mizuno, Hiroshi Yamaguchi, Toshihiko Nakano, Yusuke Matsushima, Yoshikazu Sumi, Takanori Sato, Hisashi Yamashida, and Masakazu Yamashina. A 1-ghz portable digital delay-locked loop with infinite phase capture ranges. *IEICE transactions on electronics*, 84(2):220–228, 2001.

[15] Takeshi Hamamoto, Kiyohiro Furutani, Takashi Kubo, Satoshi Kawasaki, Hironori Iga, Takashi Kono, Yasuhiro Konishi, and Tsutomu Yoshihara. A 667-mb/s operating digital dll architecture for 512-mb ddr sdram. *IEEE Journal of Solid-State Circuits*, 39(1):194–206, 2004.

[16] Dan I Porat. Review of sub-nanosecond time-interval measurements. *IEEE Transactions on Nuclear Science*, 20(5):36–51, 1973.

[17] Piotr Dudek, Stanislaw Szczepanski, and John V Hatfield. A high-resolution cmos time-to-digital converter utilizing a vernier delay line. *IEEE Journal of Solid-State Circuits*, 35(2):240–247, 2000.

[18] Osamu Sasaki, T Taniguchi, TK Ohska, H Mori, T Nonaka, K Kaminishi, A Tsukuda, H Nishimura, M Takeda, and Y Kawakami. 1.2 ghz gaas shift register ic for dead-timeless tdc application. *IEEE Transactions on Nuclear Science*, 36(1):512–516, 1989.

[19] Andrew E Stevens, Richard P Van Berg, Jan Van der Spiegel, and Hugh H Williams. A time-to-voltage converter and analog memory for colliding beam detectors. *IEEE Journal of solid-state circuits*, 24(6):1748–1752, 1989.

[20] Elvi Raisanen-Ruotsalainen, Timo Rahkonen, and Juha Kostamovaara. An integrated time-to-digital converter with 30-ps single-shot precision. *IEEE Journal of Solid-State Circuits*, 35(10):1507–1510, 2000.

[21] Atsushi Hatakeyama, Hirohiko Mochizuki, Tadao Aikawa, Masato Takita, Yuki Ishii, Hironobu Tsuboi, Shin-ya Fujioka, Shusaku Yamaguchi, Makoto Koga, Yuji Serizawa, et al. A 256-mb sdram using a register-controlled digital dll. *IEEE Journal of Solid-State Circuits*, 32(11):1728–1734, 1997.

[22] Feng Lin, Jason Miller, Aaron Schoenfeld, Manny Ma, and R Jacob Baker. A register-controlled symmetrical dll for double-data-rate dram. *IEEE Journal of Solid-State Circuits*, 34(4):565–568, 1999.

[23] Mohammad Maymandi-Nejad and Manoj Sachdev. A digitally programmable delay element: design and analysis. *IEEE transactions on very large scale integration (VLSI) systems*, 11(5):871–878, 2003.

[24] Tuvia Liran and Ran Ginosar. All-digital dll architecture and applications. *Technical Report*, 2005.

[25] Dongsuk Shin, Janghoon Song, Hyunsoo Chae, Kwan-Weon Kim, Young Jung Choi, and Chulwoo Kim. A 7ps-jitter 0.053 mm2 fast-lock addll with wide-range and high-resolution all-digital dcc. In *Solid-State Circuits Conference, 2007. ISSCC 2007. Digest of Technical Papers. IEEE International*, pages 184–595. IEEE, 2007.

[26] Shao-Ku Kao, Bo-Jiun Chen, and Shen-Iuan Liu. A 62.5–625-mhz anti-reset all-digital delay-locked loop. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 54 (7):566–570, 2007.

[27] A Rossi and G Fucili. Nonredundant successive approximation register for a/d converters. *Electronics letters*, 32(12):1055–1057, 1996.

[28] Randall L Geiger, Phillip E Allen, and Noel R Strader. *VLSI design techniques for analog and digital circuits*, volume 90. McGraw-Hill New York, 1990.

[29] Chulwoo Kim, In-Chul Hwang, and Sung-Mo Kang. A low-power small-area/spl plusmn/7.28-ps-jitter 1-ghz dll-based clock generator. *IEEE Journal of Solid-State Circuits*, 37(11):1414–1420, 2002.

[30] Seung-Jun Bae, Hyung-Joon Chi, Young-Soo Sohn, and H-J Park. A vcdl-based 60-760-mhz dual-loop dll with infinite phase-shift capability and adaptive-bandwidth scheme. *IEEE journal of solid-state circuits*, 40(5):1119–1129, 2005.

[31] Dong-Hoon Jung, Young-Jae An, Kyungho Ryu, Jung-Hyun Park, and Seong-Ook Jung. All-digital fast-locking delay-locked loop using a cyclic-locking loop for dram. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 62(11):1023–1027, 2015.

[32] Jinn-Shyan Wang and Chun-Yuan Cheng. An all-digital delay-locked loop using an in-time phase maintenance scheme for low-jitter gigahertz operations. *IEEE Trans. on Circuits and Systems*, 62(2):395–404, 2015.