DESIGN OF A CLOCK AND DATA RECOVERY CIRCUIT IN 65 NM TECHNOLOGY

BY

YI REN

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2016

Urbana, Illinois

Adviser:

Professor Jose E. Schutt-Aine

ABSTRACT

As semiconductor fabrication technology develops, the demand for higher transmission data rates constantly increases; thus there is an urgent need for a power-efficient, robust and broad bandwidth chip-to-chip communication method. A lot of work has been done to address this issue as researchers strive for more integrated inter-IC communication technology with CMOS. A high-speed serial link (HSSL) can help meet this goal. The clock and data recovery circuit (CDR) is a critical component of the HSSL.

CDR is built on the receiver end of the link after proper equalization. Its purpose is to extract clock signal which is not transmitted from the driver end and to use the extracted clock signal to sample the incoming data stream with optimal timing.

In this thesis, the working mechanism of the CDR is described. A CDR consists of a phase detector, a charge pump, a loop filter and a voltage-controlled oscillator. This thesis includes an overview of all the building blocks of a PLL-based CDR, derivation of the mathematical formulations of the negative feedback loop, and a report on closed loop behavioral modeling of the entire CDR and implemented CDR building blocks at transistor level with TSMC 65 nm technology PDK with a 6.4 Gbps data rate. Also, this thesis provides a detailed noise analysis of the CDR. Lastly, some future work and possible design improvements are proposed.

ACKNOWLEDGEMENTS

As I finish up with all the courses and research as a graduate student at the University of Illinois, Urbana-Champaign, I think again how many people have helped me and nourished me along this journey.

First of all, I want to thank my graduate advisor, Professor Jose Schutt-Aine. I could not do this without his support and guidance. Whenever I encountered an obstacle, I would always go and talk to him. His experience, kind heart and academic knowledge have been the greatest thing I had throughout my graduate studies. He enlightens me on how much education can change people's lives, and I will always look up to him as I start my professional career.

Secondly, I would like to thank all of my fellow graduate and undergraduate students in the research group. It is a pleasure to know all of them. Thanks to Da Wei and Sabareeshkumar Ravikumar who motivated and guided me in this research process. Thanks to Rushabh Mehta for your constant help on the CDR with your work experience and kindness. Thanks to Xu Chen, Xiao Ma, Ishita Bisht, Maryam Hajimiri, Xinying Wang, Jerry Yang, and Zexian Li for your academic help and for making my graduate student life a memory with a lot of fun.

I thank my family. My parents have always been there for me through thick and thin. Whenever I was confused and did not know where to go, my parents have always been there supporting me and encouraging me. I could not have done this without them.

Lastly, I would like to thank all of my friends, Tianyilin Zhu, Manfei Wu, Zhiyuan Zheng and Siyue Li, for their support and for standing by me throughout my ECE career. They made my decision to attend UIUC the best one in my life.

CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Motivation

The development of CMOS technology demands high-speed communication with higher bandwidth, less power consumption and more accuracy. In this thesis, chip-to-chip signaling and the ways that electrical engineering researchers have come up with to develop and study it will be mainly discussed. In the 1980s, the speed requirement for chip-to-chip signaling was slightly more than 10 Mbs. Therefore, lumped elements such as capacitors could be used as a channel. The transceivers were inverters on both ends. In the 1990s, the speed requirement increased to more than 100 Mbs. That was when using a transmission line became a pervasive method of chip-to-chip communication. Ever since 2000, at least 1 Gbs speed has been necessary. At this high frequency, the transmission line has become very noisy due to several factors of the channel such as attenuation and dispersion. The signaling went from parallel, to serial, to point-to-point serial streams with adaptive equalization, and the transmission line is equipped with low-power clock and alternate channel materials [1]. Figure 1.1 shows the chip-to-chip signal trends.

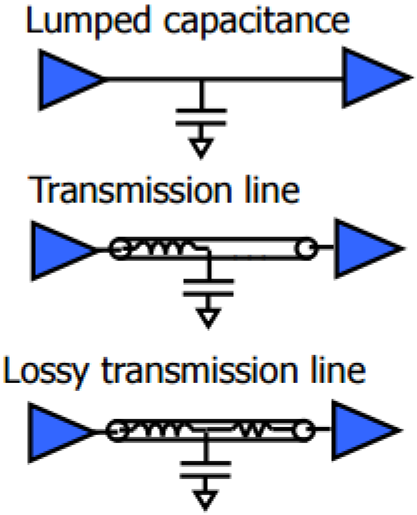


Figure 1.1: Chip-to-chip signal trends

For serial links, there are different types of applications. For example, for a processor-to-peripheral application, people normally use PCIe whose speed can be 2.5, 5 or 8 Gbps. For storage purposes, there is SATA bus which is targeted at a speed of 6 Gbps [1]. Here, I will mainly talk about chip-to-chip serial links.

As mentioned earlier, communication speeds of all types of applications are exploding. Figure 1.2 shows growth in average internet connectivity speeds of end users in the United States. Figure 1.3 shows the trends in data rate scaling of high-speed I/O signaling links as predicted by the International Solid State Conference. While data rate is booming, the need for clock frequencies in the multi-GHz demands the usage of a SerDes circuit in the serial communication. It offers a wide range of functionality. A phase-locked-loop (PLL) which is contained in the SerDes block can generate high-frequency, low-jitter clocks with minimal timing skew. Moreover, PLL can also be used to implement a CDR circuit.

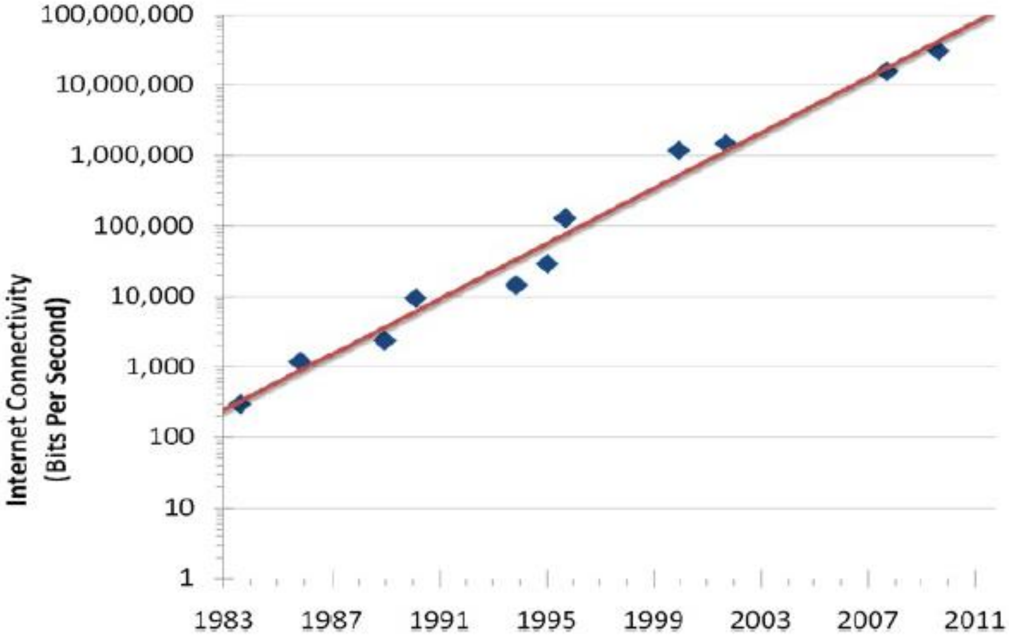


Figure 1.2: Growth in average Internet connectivity

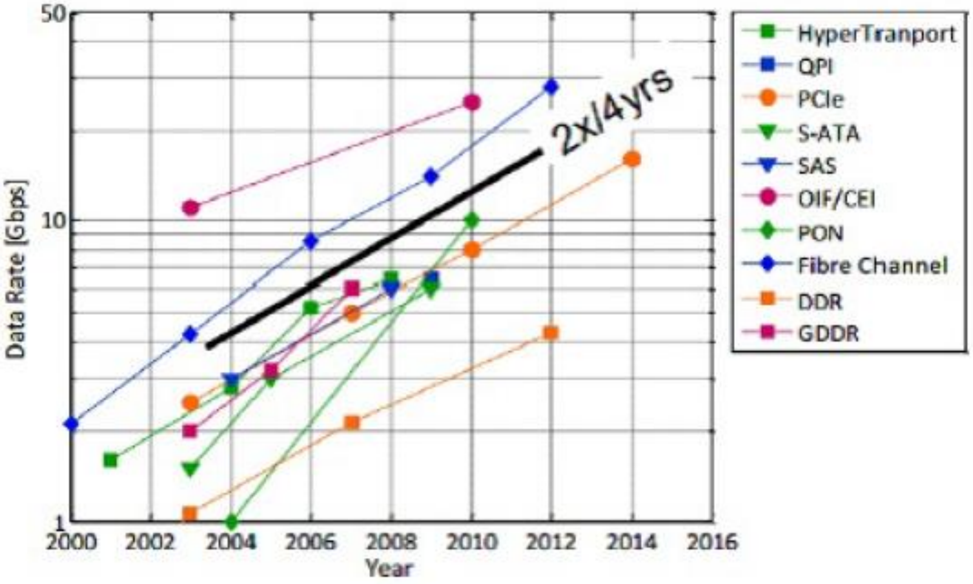


Figure 1.3: Predicted date rate trend in high-speed I/O

1.2 Thesis Outline

The goal of this thesis is to provide a design of a CDR circuit with 65 nm technology aiming at 6.4 Gbps data rate. In addition to that, the thesis can be used as a reference manual for the designing process which includes behavioral modeling with Verilog-AMS and Cadence Virtuoso. The following chapters are organized as follows:

Chapter 2 provides a high-speed serial link overview with its building blocks and the HSSL's non-idealities.

Chapter 3 provides a more detailed description of each building block in the CDR circuitry and the topologies of each of them. It also gives a mathematical overview of the negative feedback loop in the CDR, the CDR loop design procedure and noise analysis.

Chapter 4 describes how to perform behavioral modeling of all the building blocks of the CDR with a negative feedback loop.

Chapter 5 describes the transistor-level implementation of some of the CDR building blocks and their results.

Chapter 6 summarizes the thesis with the work completed and suggests future work to improve the design.

CHAPTER 2

HIGH-SPEED SERIAL LINK OVERVIEW

Before serial links were introduced, traditional I/O buses were mostly parallel buses such as SATA, PCIe and RDRAM. They all transmit data at their designed data rate. These interfaces require one conductor for each bit of the transmitted data word. As a result, a bus that only transmits one word would require a lot of wire to transmit. It is very space-consuming. As shown in Figure 2.1, there are many IO pins for a PCIe bus.

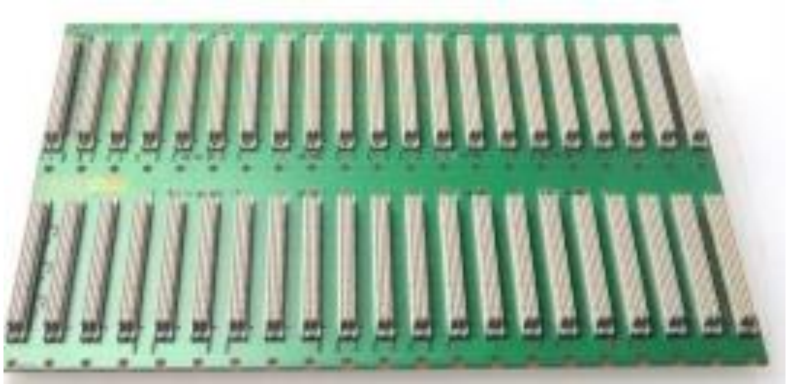


Figure 2.1 PICe I/O pins

Another drawback of parallel data transmission is the synchronization of the signals. Since we have all the parallel signals coming in at the receiver end, we need to sample the data at the same time. However, due to the imperfections in the fabrication process and length of transmission, skewing makes the sampling very difficult. Another issue with a parallel bus is power consumption. As the scale of circuits increases, the power consumed in the transmission is also growing rapidly.

This is where serial links come in handy. They do not have the disadvantages of parallel links. They save area and power across several CMOS process nodes. As Dobkin et al. argue, high-speed serial links (HSSL) are more practical for current applications [2]. They conducted experiments to study in what situation we should choose to implement parallel transmission or serial. The results are shown in Figure 2.2.

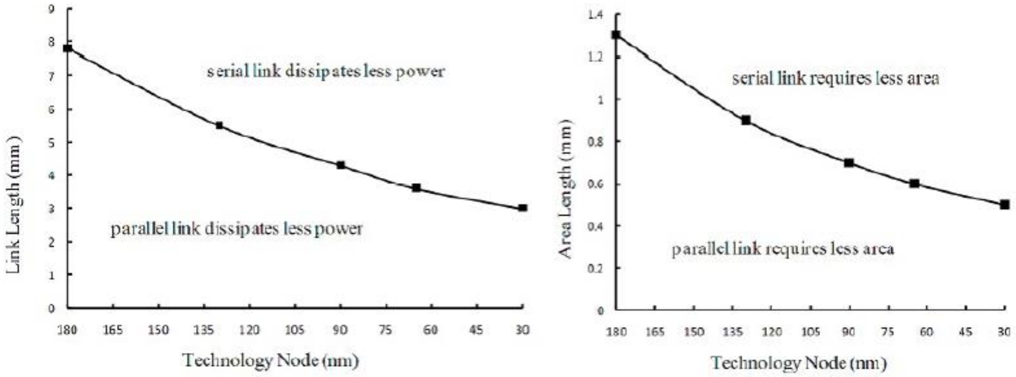


Figure 2.2: Comparison of serial and parallel communication

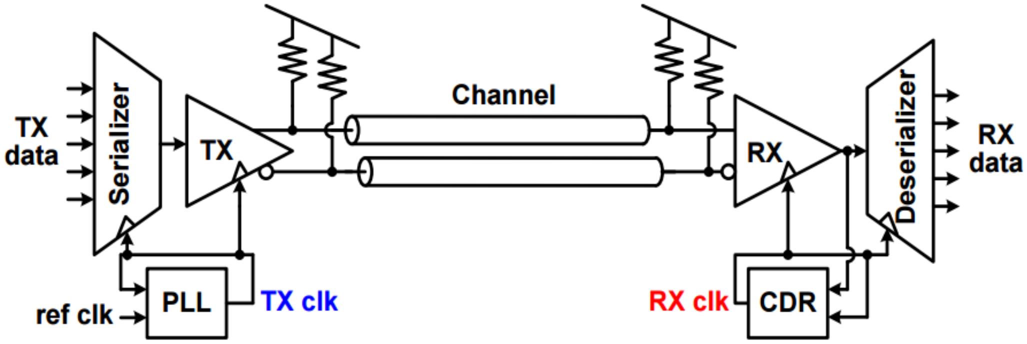


Figure 2.3: General model of HSSL

Figure 2.3 shows the general model for a HSSL. Parallel data coming from the chip serves as the serializer input, which is converted into serial data in a specific outgoing order. The PLL has two inputs, which are the reference clock and the feedback clock. The feedback clock can be used as both serializer clock and the transmitter clock. With proper termination, the data gets transferred onto the channel, which we typically model as a transmission line. The clock signal will not be transmitted. Figure 2.4 shows how the clock data is transmitted at the transmitter and recovered at the receiver together with the data.

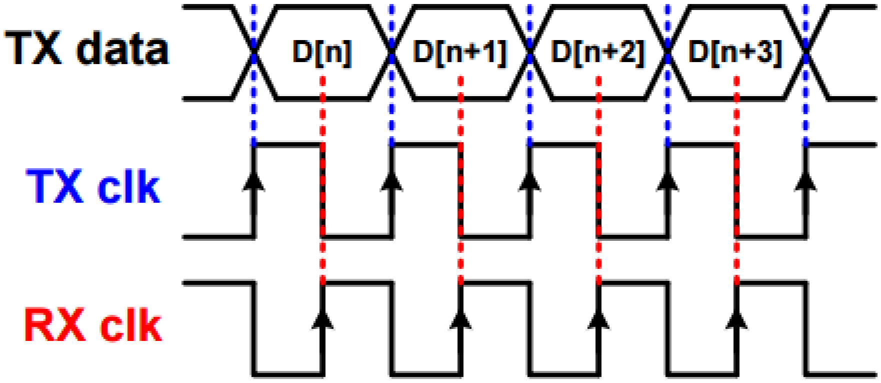


Figure 2.4: TX data with TX and RX clock edge

The channel can be as short as an inch or as long as 20 in. Several phenomena happen on the channel during signal transmission such as reflection, attenuation and dispersion. These all cause the signal to be very difficult to read and sample at the receiver end. Therefore, equalization is necessary. Typical equalization methods such as FIR, DEF and CTLE can be implemented. Figure 2.5 shows the SerDes link with equalization. The FIR equalization is on the transmitter side while the CTLE and DFE are on the receiver side.

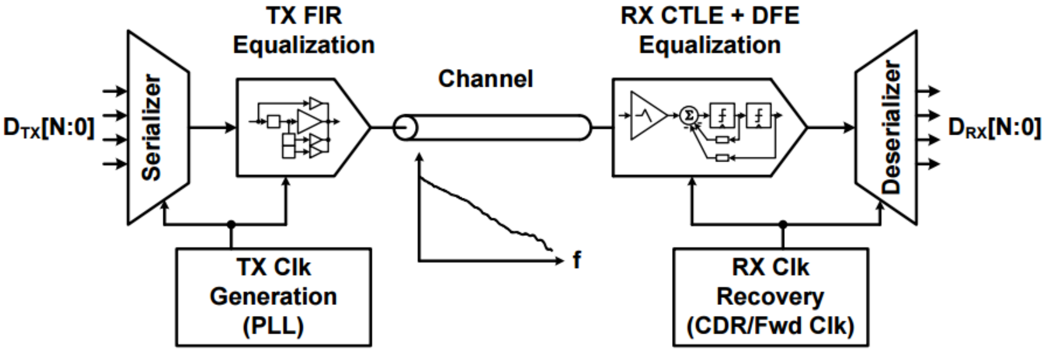


Figure 2.5: SerDes link with equalization

As mentioned earlier, the required data rate is increasing rapidly. As shown in Figure 2.6, channel performance varies with data rate. At higher data rate, the eye opening of the eye diagram is smaller, or the unit eye height has reduced. This illustrates why equalization is needed for HSSL.

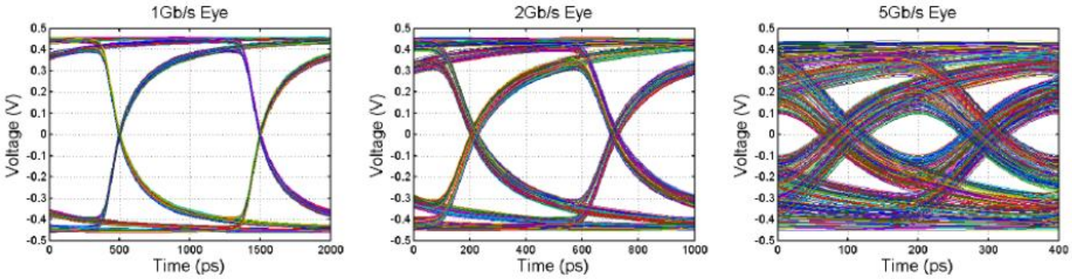


Figure 2.6: Channel performance at difference data rate

After the channel and receiver, the data goes to the deserializer which converts the serial bit stream back to parallel data. The architecture of the deserializer is normally reciprocal to the serializer while the serializer uses Mux and the deserializer uses DEMUX.

CDR provides clock signal for both the receiver and the deserializer since the clock data was not transmitted together with the bit stream. A CDR schematic is very similar to that of a PLL with some minor modification. Recovering clock signal is not the only functionality of a CDR; it can also recover data with a data sampling circuit. We want the CDR to sample the data at the optimal position of the eye diagram, which adds complexity to the circuit design.

CDR is the main topic of this thesis. It consists of several building blocks which will be discussed in more detail in the following chapters. The CDR in this thesis is a PLL based CDR, meaning it has very similar structure as the PLL with a few changes in detail and building components. The CDR needs to extract the clock signal from the transmitted signal, and the extracted clock signal can be used in the receiver and the decision circuit, which samples the incoming data stream. The block diagram of a CDR is shown in Figure 2.7. It is made up of a phase detector (PD), a charge pump (CP), a low-pass filter (LPF) and a voltage-controlled oscillator (VCO). These components have different building topologies and some of the architectures will be discussed later in detail with respect to advantages, disadvantages and applications.

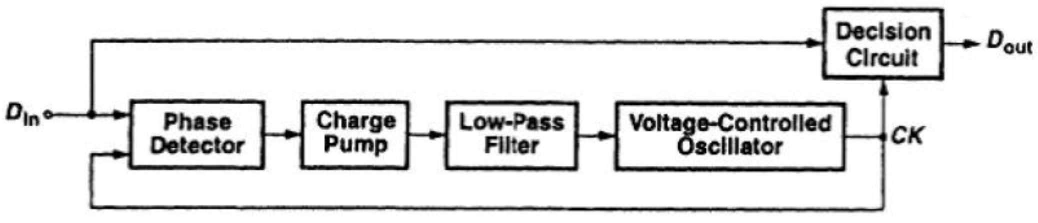


Figure 2.7: Block diagram of CDR

To characterize the circuit, one needs a few figures or measurements. The first and the most important is timing jitter, which is the time-domain variation in the clock signal. Figure 2.8 shows the definition of jitter in a graph. Jitter in the clock signal is often determined by power supply noise or substrate noise. These all come with the fabrication process and cannot be changed with the modification of the circuit design. Therefore, as data rates increase, the frequency of the bit stream increases to giga-bit scale. Jitter can become a deterministic factor in the design process.

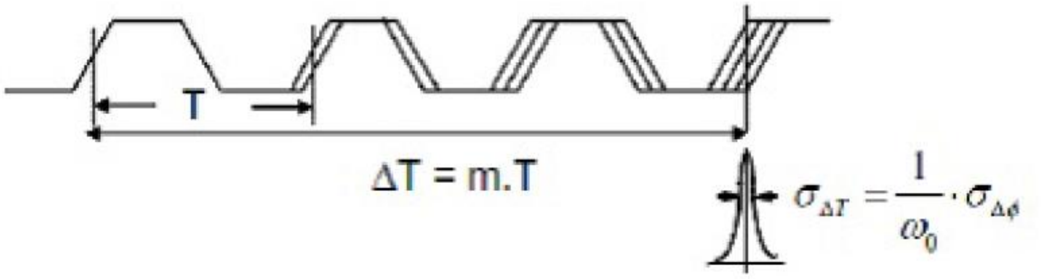

T
ΔT = m.T
$\sigma_{\Delta T} = \frac{1}{\omega_0} \cdot \sigma_{\Delta\phi}$

Figure 2.8: Definition of Jitter

Figure 2.9 shows the general categories of timing jitter [1].

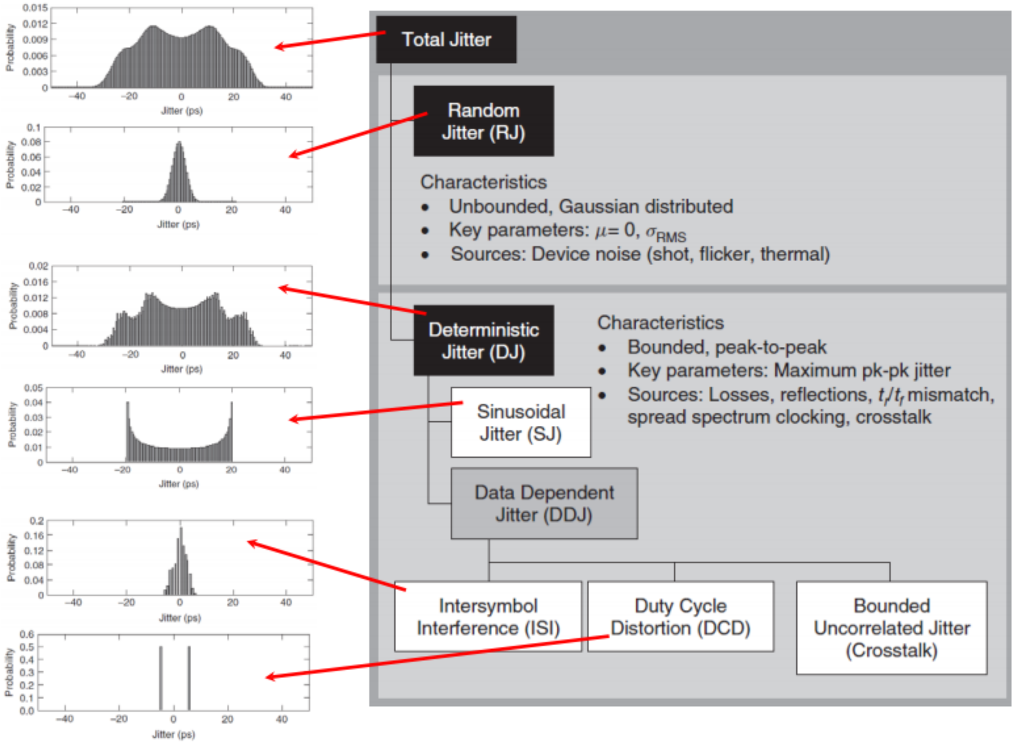

Total Jitter
Random Jitter (RJ)
Characteristics
Unbounded, Gaussian distributed
Key parameters: $\mu = 0$, σ_{RMS}
Sources: Device noise (shot, flicker, thermal)
Deterministic Jitter (DJ)
Characteristics
Bounded, peak-to-peak
Key parameters: Maximum pk-pk jitter
Sources: Losses, reflections, t_r/t_f mismatch, spread spectrum clocking, crosstalk
Sinusoidal Jitter (SJ)
Data Dependent Jitter (DDJ)
Intersymbol Interference (ISI)
Duty Cycle Distortion (DCD)
Bounded Uncorrelated Jitter (Crosstalk)
Probability
Jitter (ps)
−40
−20
0
20
40

Figure 2.9: General categories of timing jitter

CHAPTER 3

CDR BUILDING BLOCKS AND ANALYSIS

The clock and data recovery (CDR) circuit discussed in this thesis is a phase-locked loop (PLL) based circuit. The CDR is on the receiver's end and its function is to extract the clock signal from the incoming bit stream and sample the incoming data at a correct time to obtain data recovery. Moreover, the recovered clock is also used as clock to the deserializer. Therefore, CDR is very critical to the entire HSSL system. Its performance is one of the key factors limiting SerDes links. The CDR circuit at the receiver end is very similar to the PLL that is on the transmitter end but with some difference. They are both negative feedback loops which keep the link in a more stable working state even if there are unpredictable elements that can interfere with it. In this chapter, we will take a look at each of the building blocks of the CDR circuit and analyze the loop dynamics and noise performance.

3.1 Basic CDR Building Blocks

The basic building blocks of a CDR are shown in Figure 3.1 with the connections between blocks.

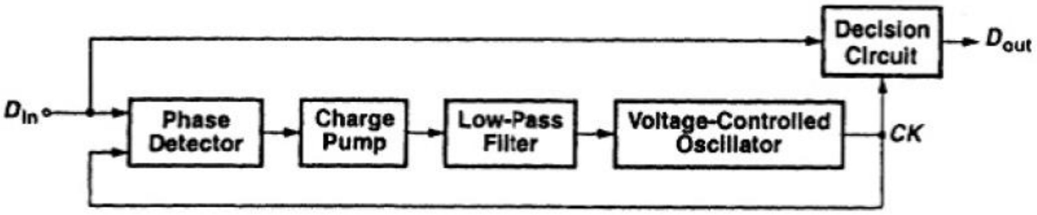


Figure 3.1: Building blocks of CDR

The CDR consists of a phase detector, a charge pump, a low-pass filter and a voltage-controller oscillator. D_{in} represents the incoming bit stream and the output of the VCO serves as the other input to the phase detector which forms a negative feedback loop. The CDR compares the phase of the data input and the phase of the generated clock and adjusts the output frequency accordingly. Unlike a lot of other circuitry in which we focus on the voltages and currents of each stage, in the CDR we also need to pay attention to the phase performance in order to have functioning loop dynamics.

3.1.1 Phase Detector

The phase detector is the first component of the CDR. It takes the incoming bit stream and generated clock signal from the VCO as inputs and compares the phase difference between them. The output of the phase detector serves as the input of the second component, which is the charge pump. The phase detector converts the incoming phase difference into voltage. When the CDR is locked, the phase difference between the reference clock and feedback clock should remain a constant value. An ideal linear phase detector produces an output signal whose DC value is linearly proportional to the phase difference. There is also another type of phase detector called binary PD which produces an error signal whose value depends only on the sign of phase error. Figure 3.2 shows the phase transfer functions of linear and nonlinear phase detectors. If we denote the phase error as $\Delta\phi$ and the gain of the phase detector as K_{PD}, then the input output relationship of a PD is [3]:

$$V_e = K_{PD} \times \Delta\phi$$

where $K_{PD} = \frac{TD}{\pi}$, where TD is the transition density.

In this thesis, we will focus on a linear phase detector, which has wider frequency acquisition range and enables loop parameter calculation [4]. One of the structures of a linear PD is called the Hogge phase detector, the block diagram of which is given in Figure 3.3. It consists of a positive edge triggered D flip-flop, a negative edge triggered D flip-flop and two XOR gates. Path UP produces proportional pulses in relation to phase difference while path DOWN produces half-clock-period-wide reference pulses. Under locked condition, UP and DOWN show pulses with equal width [4].

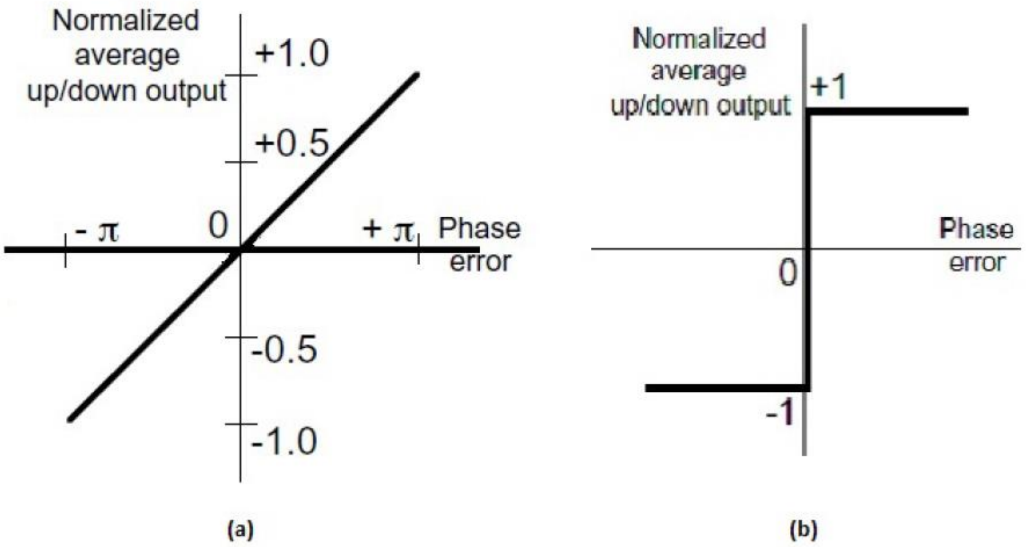


Figure 3.2: Linear and nonlinear phase detectors

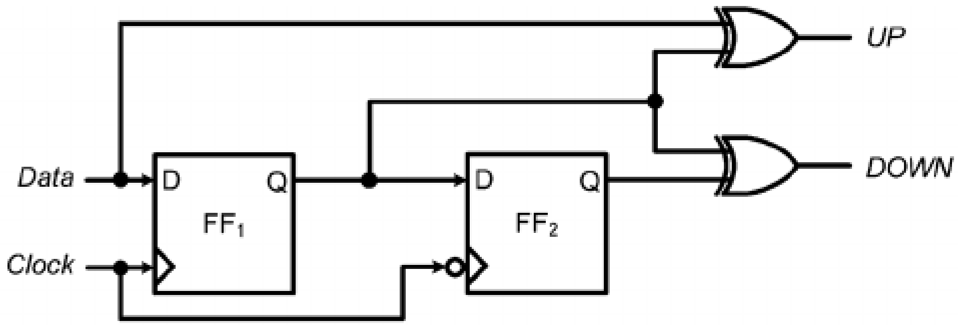


Figure 3.3: Block diagram of a Hogge phase detector

3.1.2 Charge Pump

The charge pump is the second stage of a CDR. Some researchers include the charge pump as a part of the phase detector. A charge pump takes the output of the phase detector, a voltage signal, and transforms it into a current signal. Since the voltage-controlled oscillator needs a stable voltage signal to generate stable frequency signal, a large-valued capacitor is necessary. The general diagram of a charge pump is shown in Figure 3.4.

The output of the charge pump is connected to a large-valued capacitor as mentioned before. The upper and lower switches can be turned on and off according to the value of the phase detector output. When the UP signal is high, the upper switch is closed and the charges will be pumped into the capacitor. When the DOWN signal is high, the lower switch will be closed and charges will be drained from the capacitor. Also, when UP and DOWN are the same, meaning they are both high or low, no charge will be pumped or drained from the capacitor.

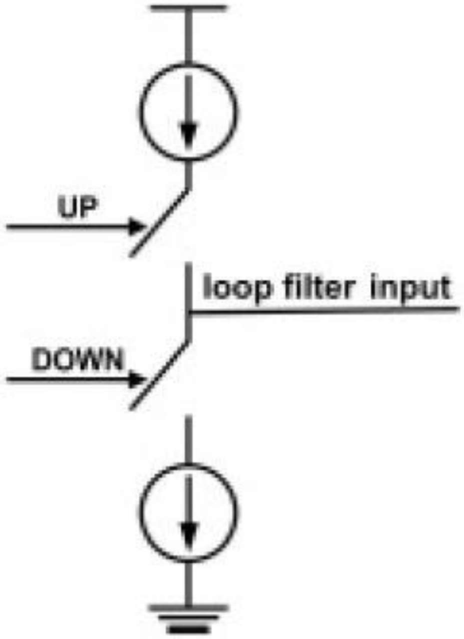


Figure 3.4: General diagram of a charge pump

The transfer function of the phase detector and charge pump together is

$$PD(s) = K_{PD} = \frac{i_{cp}}{2\pi} \tag{1}$$

3.1.3 Low-pass Filter

The output of the phase detector typically has a lot of high-frequency noise. Therefore, the LPF is to eliminate the high-frequency noise. Moreover, we need a charge storage device to maintain a stable input voltage signal to the voltage-controlled oscillator.

We will consider a passive loop filter in this case because it offers greater noise and power rejection performance and it is simpler to implement. It consists of a resister in series with a capacitor and they are in parallel with another capacitor. The loop filter diagram is shown in Figure 3.5

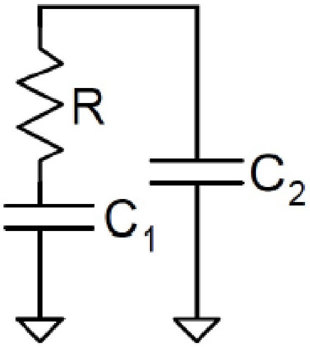


Figure 3.5: Schematic of a loop filter

The transfer function of the loop filter is:

$$LF(s) = \frac{V_{ctrl}(s)}{i_{cp}} = \frac{s + \frac{1}{RC_1}}{C_2 s(s + \frac{C_1 + C_2}{RC_1 C_2})} \tag{2}$$

3.1.4 Voltage-Controlled Oscillator

A voltage-controlled oscillator (VCO) is a device that can take in a control voltage and generate an output at a specific frequency. The output frequency ideally should be proportional to the input control voltage. There are two types of oscillators, ring oscillator and LC-tank oscillator. The ring oscillator is a digital circuit which has an odd number of inverters, with the last inverter output connected as the input to the first inverter. By utilizing the fact that the delay of each inverter depends on the amount of current it can sink in, the frequency of oscillation can be controlled. This is the type of VCO that we will mainly discuss in this thesis. The Laplace transform function of the VCO is derived as follows:

$$\omega_{out}(t) = K_{VCO}v_{ctrl}(t) \tag{3}$$

$$\mathcal{L}[\omega_{out}(t)] = \omega_{out}(s) = K_{VCO}v_{ctrl}(s) \tag{4}$$

$$\phi_{out}(t) = \int_0^t \omega_{out}(s)\,d\tau = \int_0^t K_{VCO}\,v_{ctrl}(s) \tag{5}$$

$$\mathcal{L}[\phi_{out}(t)] = \phi_{out}(s) = \frac{\omega_{out}(s)}{s} = \frac{K_{VCO}v_{ctrl}(s)}{s} \tag{6}$$

$$H_{vco}(s) = \frac{\phi_{out}(s)}{v_{ctrl}(s)} = \frac{K_{VCO}}{s} \tag{7}$$

where K_{VCO} is the gain of VCO.

Figure 3.6 shows the typical diagram of a ring oscillator.

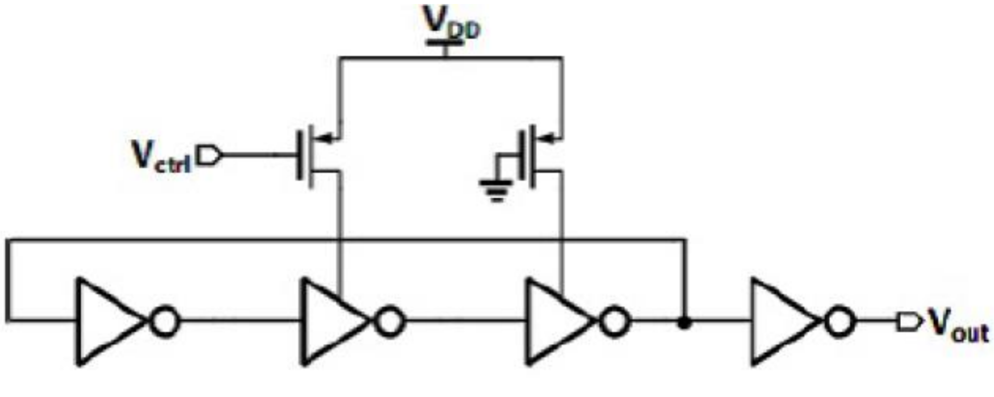


Figure 3.6: Ring oscillator diagram

3.2 CDR Loop Dynamics in Lock State

3.2.1 CDR Loop Analysis with a Simple LF

The transient response of phase-locked loops is generally a nonlinear process that cannot be mathematically derived easily as it is a negative feedback system [3]. We can study the PLL in a locked state to gain some intuition. Our ultimate purpose is to find a transfer function of the closed loop with respect with phase, namely $\phi_{out}(s)/\phi_{in}(s)$. Assuming the LPF has a transfer function $G_{LPF}(s)$, the open loop transfer function is:

$$H_O(s) = K_{PD} G_{LPF} \frac{K_{VCO}}{s} \tag{8}$$

And from this we know that the closed-loop transfer function is:

$$\begin{aligned} H(s) &= \frac{\phi_{out}(s)}{\phi_{in}(s)} \\ &= \frac{K_{PD} G_{LPF}(s) K_{VCO}}{s + K_{PD} G_{LPF}(s) K_{VCO}} \end{aligned} \tag{9}$$

If we only consider the simplest form of low-pass filter, then its transfer function is:

$$G_{LPF}(s) = \frac{1}{1 + \frac{s}{\omega_{LPF}}} \tag{10}$$

where $\omega_{LPF} = 1/(RC)$.

The simplify the closed-loop function, we will have

$$H(s) = \frac{K_{PD} K_{VCO}}{\frac{s^2}{\omega_{LPF}} + s + K_{PD} K_{VCO}} \tag{11}$$

indicating that the system is of second order with one pole contributed by the VCO and another by LPF. Here, loop gain in defined as $K = K_{PD} K_{VCO}$. Figure 3.7 shows the closed-loop transfer function in a diagram.

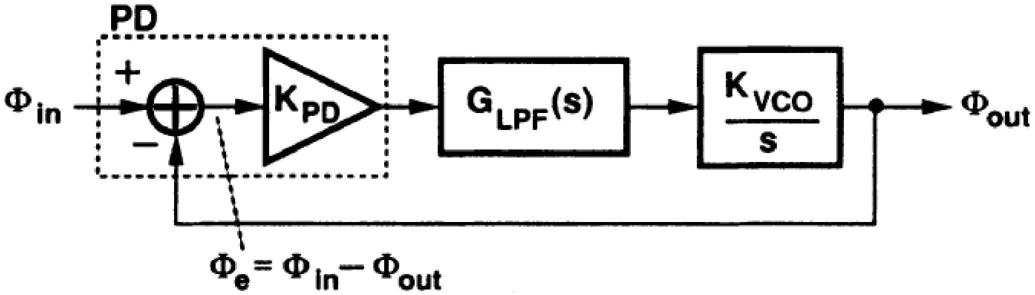


Figure 3.7: Loop transfer function

To understand the loop behavior better, we need to convert the transfer function into a more generic form as in control theory, $s^2 + 2\zeta\omega_n s + \omega_n^2$, where ζ is the damping factor and ω_n is the natural frequency of the system. Then we will have

$$H(s) = \frac{\omega_n}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{12}$$

where

$$\omega_n = \sqrt{\omega_{LPF} K} \tag{13}$$

$$\zeta = \frac{1}{2}\sqrt{\frac{\omega_{LPF}}{K}} \tag{14}$$

Here, ω_n is the geometric mean of the -3 dB bandwidth of the LPF and the loop gain. Also, the damping factor is inversely proportional to the loop gain, an important and often undesirable trade-off.

In a well-designed second-order system, the damping factor is usually greater that 0.5 and preferably equal to $\sqrt{2}/2$, so that the frequency response can be flatter. Therefore, K and ω_{LPF} cannot be chosen independently. When choosing these values, remember that noise suppression issues typically impose an upper bound on ω_{LPF} and hence K. These limitations translate to significant phase error between the input and output as well as a narrow capture range.

Reading from $H(s)$ shown above, as $s \to 0, H(s) \to 1$, meaning that a static phase shift at the input is transferred to the output unchanged. This is because for phase quantities, the presence of integration in the VCO makes the open-loop gain approach infinity as $s \to 0$ [4]. Now we can examine the phase error transfer function, defined as $H_e(s) = \Phi_e(s)/\Phi_{in}(s)$, which is

$$H_e(s) = 1 - H(s) \\ = \frac{s^2 + 2\zeta\omega_n s}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{15}$$

which drops to zero as $s \to 0$.

3.2.2 CDR Loop Analysis with a Second-Order LF

In section 3.2.1, we analyzed a CDR loop with a simple loop filter. In this section, we are going to analyze a loop with a second-order loop filter which is also the type of filter that is designed in this thesis.

The open-loop transfer function is

$$H_O(s) = K_{PD} G_{LPF}(s) \frac{K_{VCO}}{s} \tag{16}$$

With that given, the closed-loop transfer function is

$$H(s) = \frac{\Phi_e(s)}{\Phi_{in}(s)} = \frac{K_{PD} G_{LPF}(s) K_{VCO}}{s + K_{PD} G_{LPF}(s) K_{VCO}} \tag{17}$$

The transfer function of the second-order LPF is given by:

$$G_{LPF} = \frac{s + \frac{1}{RC_1}}{C_2 s(s + \omega_{LPF})} \tag{18}$$

where $\omega_{LPF} = \frac{1}{RC_{eq}}$ is the -3 dB bandwidth of the LPF and $C_{eq} = \frac{C_1 C_2}{C_1 + C_2}$.

Therefore, the closed-loop transfer function is:

$$H(s) = \frac{K(s + \frac{1}{RC_1})}{C_2 s^3 + \omega_{LPF} C_2 s^2 + Ks + \frac{K}{RC_1}} \tag{19}$$

where $K = K_{PD}K_{VCO}$ is the loop gain.

Since normally C_2 is in the order of 10E-12, the cubic term of s can be neglected. Hence, the remaining transfer function is:

$$H(s) = \frac{\frac{\omega_a^2}{\alpha}(s+\alpha)}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{20}$$

where $\omega_n = \sqrt{\frac{K}{C_1+C_2}}\alpha = \frac{1}{RC_1}$ and $\zeta = \frac{1}{2}\frac{\omega_n}{2}$.

To find the phase error transfer function, we define $H_3(s) = 1 - H(s) = \frac{1}{1+H_O(s)}$.

Applying the final value theorem, the steady state phase error is:

$$\begin{aligned} \phi_{ss}^{F_{step}} &= \lim_{s\to 0} s.H_e(s).\phi_{in}(s) \\ &= \lim_{s\to 0} s.\frac{1}{1+H_O(s)}.\frac{\Delta\omega}{s^2} \\ &= \lim_{s\to 0} \frac{[RC_1C_2s^2 + (C_1+C_2)s]\Delta\omega}{RC_1C_2S^3 + (C_1+C_2)s^2 + Ks + 1} \\ &= \frac{0}{1} \\ &= 0 \end{aligned} \tag{21}$$

This proves that the CDR with a second-order LPF can track step changes in the input frequency and establish a relock with zero steady state phase error which is not possible with the first order LPF.

3.3 Loop Design Procedure

To design a CDR that can lock, there are a few steps to follow in order to have the desired bandwidth and phase margin [5].

1. Choose the designed value of unity gain bandwidth ω_{ugb} and desired phase margin Φ_M.

2. Calculate the ratio between the two capacitors defined as $K_c = \frac{C_1}{C_2}$:

$$K_c = 2(tan^2\Phi_M + tan\Phi_M\sqrt{tan^2\Phi_M + 1}) \qquad (22)$$

3. Calculate the frequency of the zero ω_Z:

$$\omega_Z = \frac{\omega_{ugb}}{\sqrt{\frac{C_1}{C_2} + 1}} \qquad (23)$$

4. Choose R for low noise and calculate C_1 and C_2

$$C_1 = \frac{1}{\omega_2 R} \qquad (24)$$

$$C_2 = \frac{C_1}{K_C} \qquad (25)$$

5. Calculate the third pole with the values from step 4.

$$\omega_{p3} = \frac{1}{R\frac{C_1 C_2}{C_1 + C_2}} \qquad (26)$$

6. Calculated the charge pump current:

$$I_{CP} = \frac{2\pi C_2}{K_{VCO}} \cdot \omega_{ugb}^2 \cdot \sqrt{\frac{\omega_{p3}^2 + \omega_{ugb}^2}{\omega_Z^2 + \omega_{ugb}^2}} \qquad (27)$$

The values of unity gain bandwidth, phase margin and K_{VCO} are determined by the limitation of the circuit and design specifications.

Since the calculation process is a bit tedious, we can implement this function in MATLab to automate this process.

There are other procedures for the design process as mentioned in other research papers. We will continue on with the procedure described above.

The design procedure of a CDR is illustrated in Figure 3.8.

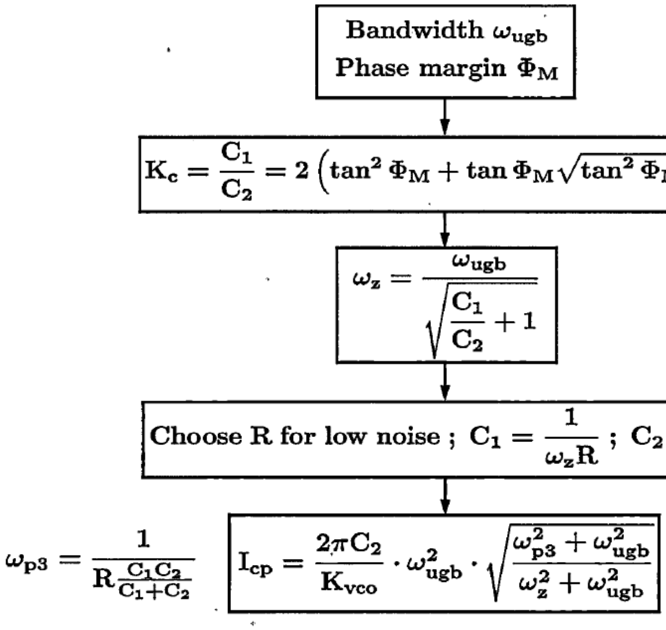

Bandwidth ω_{ugb}
Phase margin Φ_M
$K_c = \frac{C_1}{C_2} = 2\left(\tan^2\Phi_M + \tan\Phi_M\sqrt{\tan^2\Phi_M + 1}\right)$
$\omega_z = \frac{\omega_{ugb}}{\sqrt{\frac{C_1}{C_2} + 1}}$
Choose R for low noise ; $C_1 = \frac{1}{\omega_z R}$; $C_2 = \frac{C_1}{K_c}$
$\omega_{p3} = \frac{1}{R\frac{C_1C_2}{C_1+C_2}}$
$I_{cp} = \frac{2\pi C_2}{K_{vco}} \cdot \omega_{ugb}^2 \cdot \sqrt{\frac{\omega_{p3}^2 + \omega_{ugb}^2}{\omega_z^2 + \omega_{ugb}^2}}$

Figure 3.8: CDR design procedure

3.4 CDR Noise Analysis

Noise has always been an unavoidable issue in circuit design. It is the same with the design of a CDR circuit. Each building block in the CDR will generate noise and in order to optimize the circuit to the best performance, we need to study the noise characteristics of the CDR circuit. Figure 3.9 shows the noise injection at each stage of the circuit.

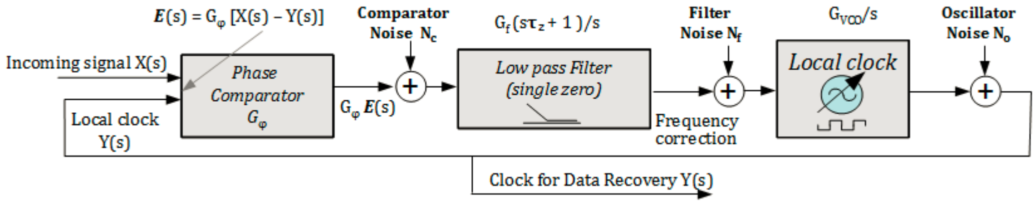


Figure 3.9: CDR noise injection

The output referred noise level and noise transfer function (NTF) can be calculated as:

$S_{\Phi_{IN}}$: Reference clock noise PSD

$S_{\Phi_{CP}}$: PFD/CP noise PSD (CP noise dominates)

S_{v_R}: Loop filter resistor noise PSD

$S_{\Phi_{VCO}}$: VCO phase noise PSD

$$NTF_{IN}(s) = \frac{\Phi_{OUT}(s)}{\Phi_{IN}(s)} = \frac{LG(s)}{1 + LG(s)} \tag{28}$$

$$NTF_{CP}(s) = \frac{\Phi_{OUT}(s)}{\Phi_{CP}(s)} = \frac{2\pi}{I_{CP}} \cdot NTF_{IN}(s) \tag{29}$$

$$NTF_R(s) = \frac{\Phi_{OUT}(s)}{\mathrm{v}_R(s)} = \frac{\frac{K_{VCO}}{s}}{1 + LG(s)} \tag{30}$$

$$NTF_R(s) = \frac{\Phi_{OUT}(s)}{\mathrm{v}_R(s)} = \frac{\frac{K_{VCO}}{s}}{1 + LG(s)} \tag{31}$$

$$NTF_{VCO}(s) = \frac{\Phi_{OUT}(s)}{\Phi_{VCO}(s)} = \frac{1}{1 + LG(s)} \tag{32}$$

From here, we can calculate:

$$S_{\Phi_{OUT}}^{\Phi_{IN}} = S_{\Phi_{IN}} \cdot |NTF_{IN}(s)|^2 \tag{33}$$

$$S_{\Phi_{OUT}}^{\Phi_{CP}} = S_{\Phi_{CP}} \cdot |NTF_{CP}(s)|^2 \tag{34}$$

$$S_{\Phi_{OUT}}^{\mathrm{v}_R} = S_{\mathrm{v}_R} \cdot |NTF_R(s)|^2 \tag{35}$$

$$S_{\Phi_{OUT}}^{\Phi_{VCO}} = S_{\Phi_{VCO}} \cdot |NTF_{VCO}(s)|^2 \tag{36}$$

Therefore,

$$S_{\Phi_{OUT}}^{TOTAL} = S_{\Phi_{OUT}}^{\Phi_{IN}} + S_{\Phi_{OUT}}^{\Phi_{CP}} + S_{\Phi_{OUT}}^{\mathrm{v}_R} + S_{\Phi_{OUT}}^{\Phi_{VCO}} \tag{37}$$

CHAPTER 4

BEHAVIORAL MODELING OF CDR

As mentioned the previous chapter, designing a CDR circuit is quite a complicated process. There are a lot of factors that circuit designers need to take into consideration. As a result, once the math has been derived for the circuit, that is when behavioral modeling starts. In this chapter, behavioral modeling will be discussed.

For mixed signal circuit design such as a CDR, behavioral modeling is normally where the circuit designers will start, given the design specifications. It gives a better understanding of the design process. For example, if we change a parameter in the circuit such as the bandwidth, we can predict how the entire system will react with behavioral modeling. Another reason why behavioral modeling is needed is that transistor-level simulation takes much more time than behavioral. For the entire CDR circuit, it can take up to hours for a transient response. As a comparison, behavioral modeling is more efficient as it only takes a couple of minutes. Behavioral modeling affects power techniques for system level design and they can easily interface with SPICE. We represent each building block in a script, provide connections between them as is, and simulate. We utilize Verilog-AMS for this purpose, in which AMS stands for analog mixed-signal. Figure 4.1 shows a typical digital circuit design flow.

4.1 Introduction to Verilog-AMS

Verilog-AMS hardware description language (HDL) is used for mixed-signal behavioral modeling and it is derived from IEEE Std 1364-2005. According to Kundert, Verilog-AMS allows the circuit designer to create and use building components which encapsulate high-level behavioral description and structural description [6]. There is one script file for each module and in each file, input/output ports, signal flow and circuit behavior are described.

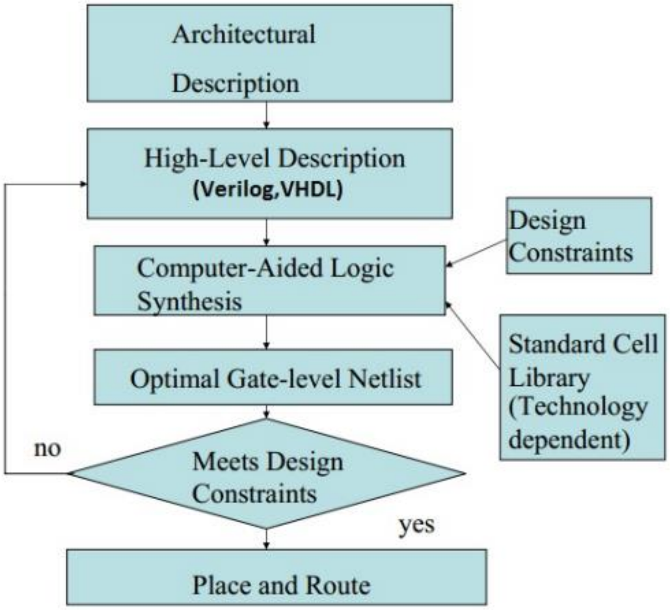

Architectural Description
High-Level Description (Verilog,VHDL)
Computer-Aided Logic Synthesis
Design Constraints
Standard Cell Library (Technology dependent)
Optimal Gate-level Netlist
no
Meets Design Constraints
yes
Place and Route

Figure 4.1: Behavioral modeling procedure

Verilog-AMS HDL extends the features of the digital modeling language to provide a single unified language with both analog and digital semantics with backward compatibility. Following are some features of Verilog-AMS

1. It can describe both analog and digital signals in the same module.
2. *Initial*, *always*, and *analog* procedural blocks can appear in the same module.
3. Both analog and digital signal values can be read from any context in the same module.
4. Digital signal values can be written from any context outside of an *analog* process.
5. Analog potentials and flows can only receive contributions from inside an *analog* procedural block.
6. The semantics of the *initial* and *always* blocks remain the same as in IEE Std 1364-2005 Verilog HDL; the semantics for the *analog* block are described in the designers' guide [6].
7. The *discipline* declaration is extended to digital signals.
8. A new construct, *connect* statement, is added to facilitate auto-insertion of user-defined connection models between the analog and digital domains.

9. When hierarchical connections are of mixed type, user-defined connection modules are automatically inserted to perform signal value conversion.

4.2 PLL Simulation in AMS Using Cadence Virtuoso

4.2.1 Phase Detector

A Hogge phase detector is implemented with AMS. It consists of two D flip-flops with opposite edge trigger and two XOR gates. It is a digital circuit, so it can be modelled with traditional Verilog. Figure 4.2 shows the block diagram of the phase detector.

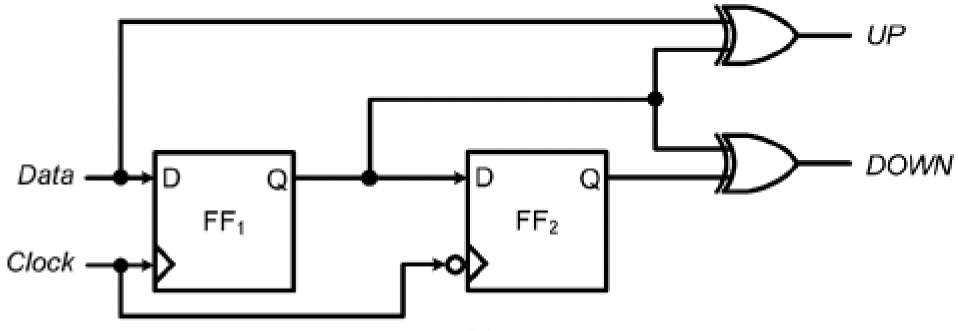
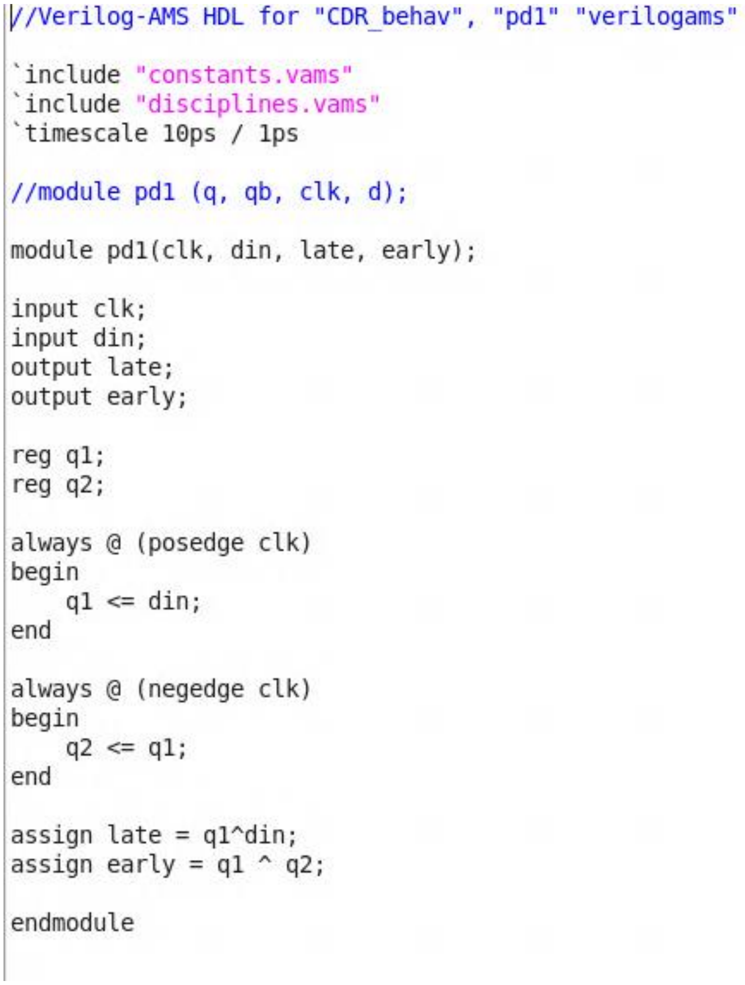


Figure 4.2: Phase detector block diagram

The Verilog-AMS code is shown in Figure 4.3.

The output waveform with two inputs and UP, DN outputs is shown in Figure 4.4. The width of the UP pulse reflects the phase relation between data and the clock and the DN pulse has a constant width and is used a reference.



```
//Verilog-AMS HDL for "CDR_behav", "pd1" "verilogams"

`include "constants.vams"
`include "disciplines.vams"
`timescale 10ps / 1ps

//module pd1 (q, qb, clk, d);

module pd1(clk, din, late, early);

input clk;
input din;
output late;
output early;

reg q1;
reg q2;

always @ (posedge clk)
begin
    q1 <= din;
end

always @ (negedge clk)
begin
    q2 <= q1;
end

assign late = q1^din;
assign early = q1 ^ q2;

endmodule
```

Figure 4.3: Phase detector Verilog code

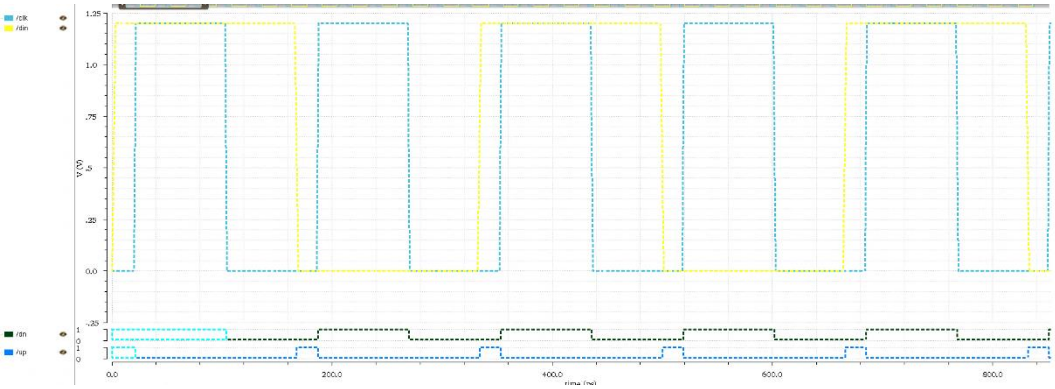
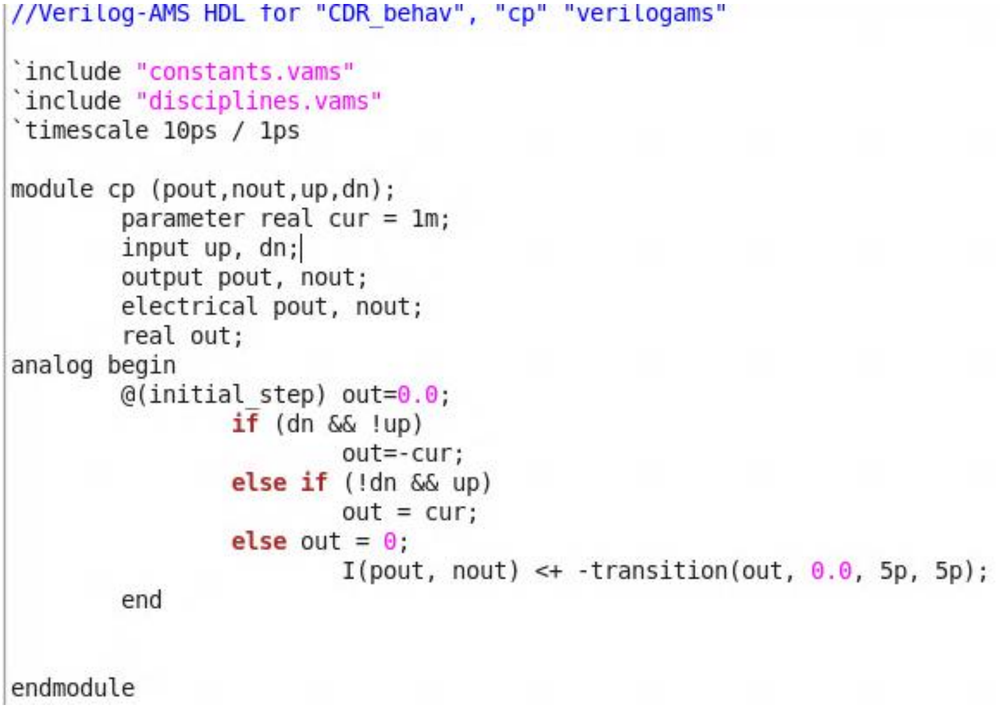


Figure 4.4: Phase detector output

4.2.2 Charge Pump

The charge pump is a mixed-signal circuit. When the Up signal goes high, the charge pump will sink charges into the loop filter, and when DN signal is high, charges will be drained from the loop filter so that the voltage that the loop filter provides can be constant to an extent.

The Verilog-AMS script is shown in Figure 4.5 and the output wave form is shown in Figure 4.6. The simulation is done together with the phase detector. The test bench is shown in Figure 4.7.



```
//Verilog-AMS HDL for "CDR_behav", "cp" "verilogams"

`include "constants.vams"
`include "disciplines.vams"
`timescale 10ps / 1ps

module cp (pout,nout,up,dn);
        parameter real cur = 1m;
        input up, dn;
        output pout, nout;
        electrical pout, nout;
        real out;
analog begin
        @(initial_step) out=0.0;
                if (dn && !up)
                        out=-cur;
                else if (!dn && up)
                        out = cur;
                else out = 0;
                        I(pout, nout) <+ -transition(out, 0.0, 5p, 5p);
        end


endmodule
```

Figure 4.5: Charge pump Verilog code

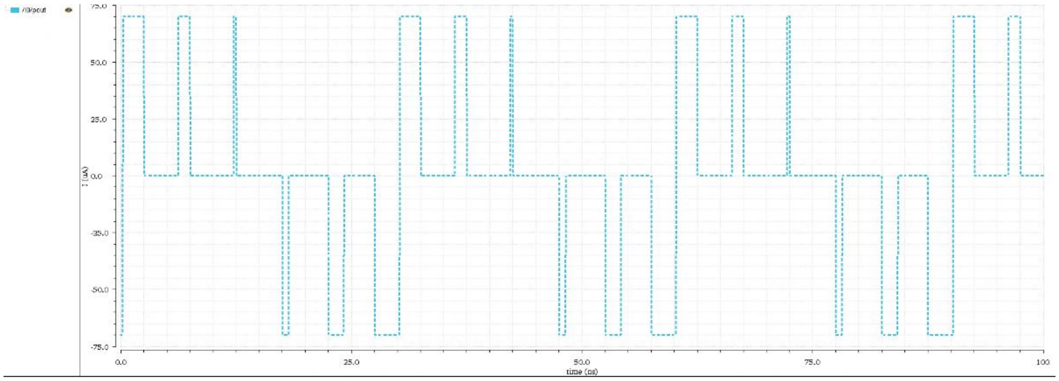


Figure 4.6: Charge pump output

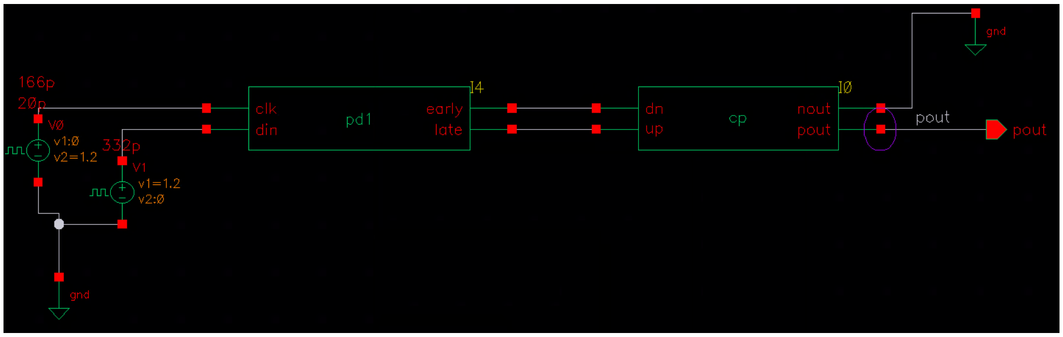


Figure 4.7: Test bench for phase detector and charge pump

4.2.3 Loop Filter

The loop filter is entirely analog with a resistor and two capacitors. From the design steps described in the previous chapter, we can calculate the resistor and capacitor values after a proper bandwidth and phase margin are chosen. In my design, R_1 = 1.276 KΩ, C_1 = 155.16 pF and C_2 = 12.002 pF. The schematic of the loop filter is shown in Figure 4.8.

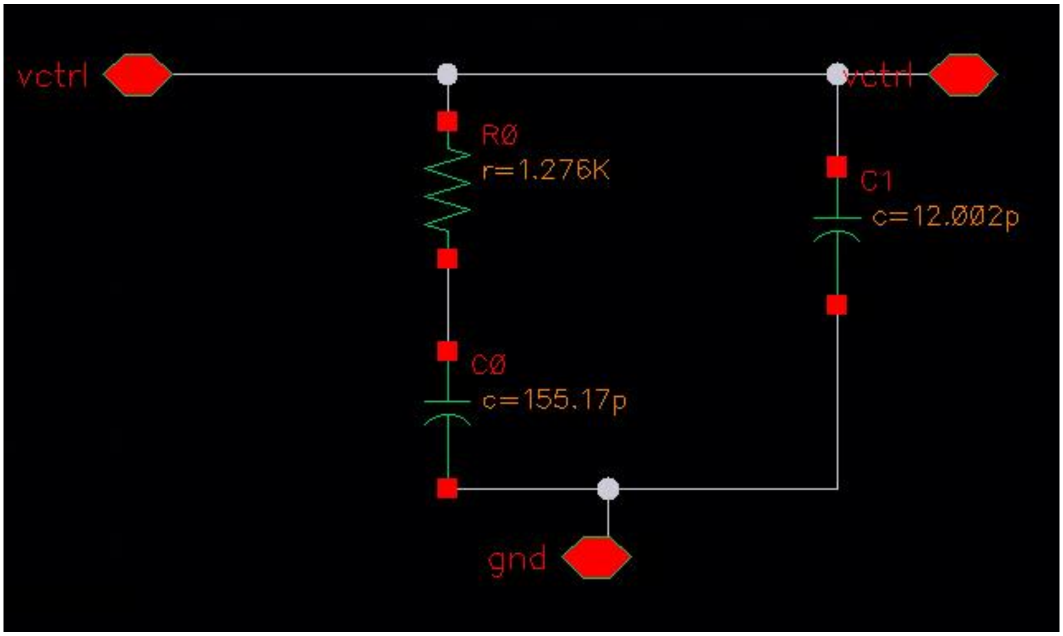
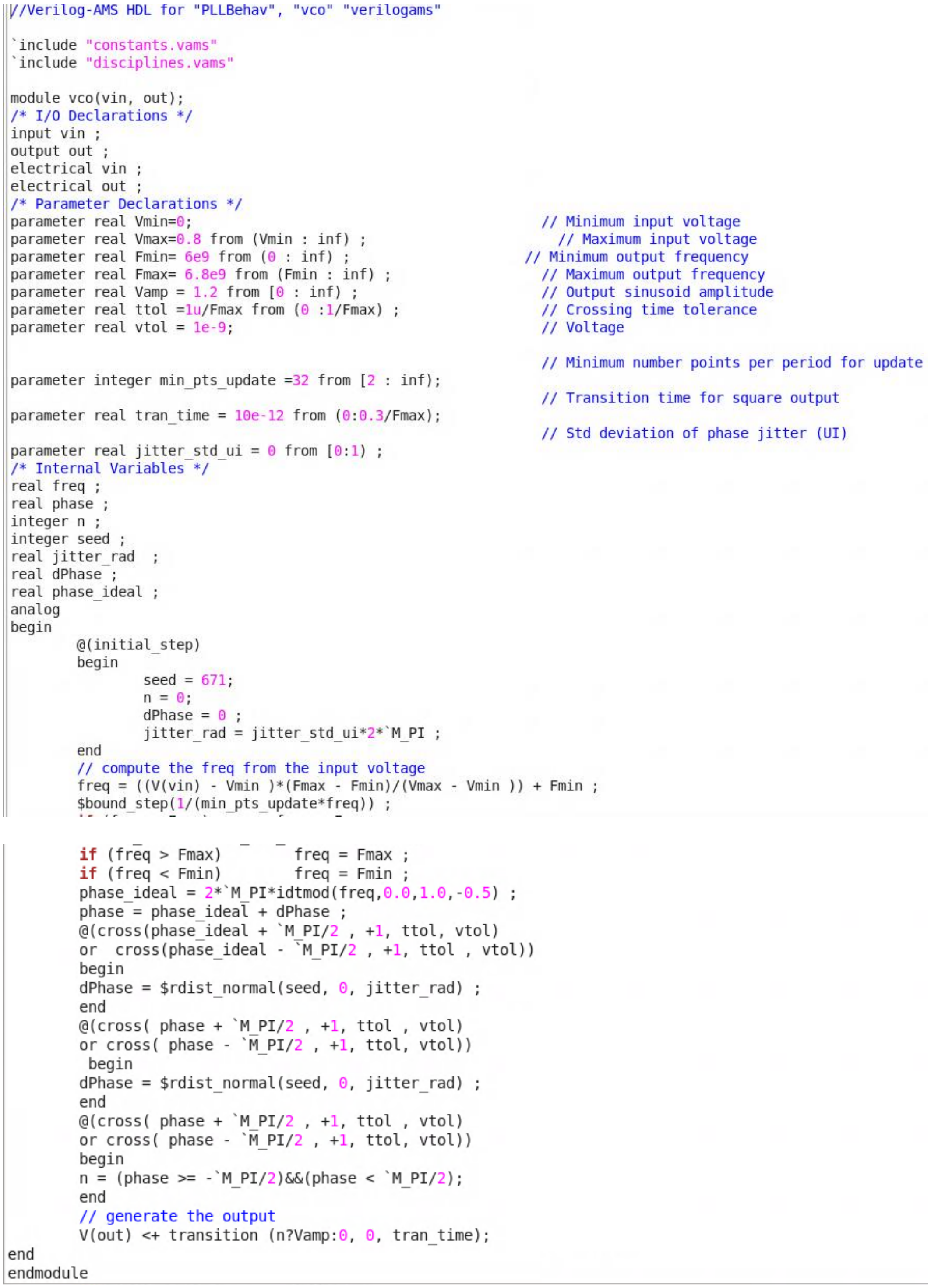


Figure 4.8: Loop filter schematic

4.2.4 Voltage-Controlled Oscillator

The code of the VCO is shown in Figure 4.9. The test bench is shown in Figure 10. The outcome waveform is shown in Figure 11. V_{ctrl} is set to be 400 mV and the K_{VCO} is calculated. The output frequency, shown in Figure 12, is set on 6 GHz.



```
//Verilog-AMS HDL for "PLLBehav", "vco" "verilogams"

`include "constants.vams"
`include "disciplines.vams"

module vco(vin, out);
/* I/O Declarations */
input vin ;
output out ;
electrical vin ;
electrical out ;
/* Parameter Declarations */
parameter real Vmin=0;                                          // Minimum input voltage
parameter real Vmax=0.8 from (Vmin : inf) ;                       // Maximum input voltage
parameter real Fmin= 6e9 from (0 : inf) ;                     // Minimum output frequency
parameter real Fmax= 6.8e9 from (Fmin : inf) ;                  // Maximum output frequency
parameter real Vamp = 1.2 from [0 : inf) ;                      // Output sinusoid amplitude
parameter real ttol =1u/Fmax from (0 :1/Fmax) ;                 // Crossing time tolerance
parameter real vtol = 1e-9;                                     // Voltage

                                                                // Minimum number points per period for update
parameter integer min_pts_update =32 from [2 : inf);

                                                                // Transition time for square output
parameter real tran_time = 10e-12 from (0:0.3/Fmax);

                                                                // Std deviation of phase jitter (UI)
parameter real jitter_std_ui = 0 from [0:1) ;
/* Internal Variables */
real freq ;
real phase ;
integer n ;
integer seed ;
real jitter_rad  ;
real dPhase ;
real phase_ideal ;
analog
begin
        @(initial_step)
        begin
                seed = 671;
                n = 0;
                dPhase = 0 ;
                jitter_rad = jitter_std_ui*2*`M_PI ;
        end
        // compute the freq from the input voltage
        freq = ((V(vin) - Vmin )*(Fmax - Fmin)/(Vmax - Vmin )) + Fmin ;
        $bound_step(1/(min_pts_update*freq)) ;

        if (freq > Fmax)            freq = Fmax ;
        if (freq < Fmin)            freq = Fmin ;
        phase_ideal = 2*`M_PI*idtmod(freq,0.0,1.0,-0.5) ;
        phase = phase_ideal + dPhase ;
        @(cross(phase_ideal + `M_PI/2 , +1, ttol, vtol)
        or  cross(phase_ideal - `M_PI/2 , +1, ttol , vtol))
        begin
        dPhase = $rdist_normal(seed, 0, jitter_rad) ;
        end
        @(cross( phase + `M_PI/2 , +1, ttol , vtol)
        or cross( phase - `M_PI/2 , +1, ttol, vtol))
         begin
        dPhase = $rdist_normal(seed, 0, jitter_rad) ;
        end
        @(cross( phase + `M_PI/2 , +1, ttol , vtol)
        or cross( phase - `M_PI/2 , +1, ttol, vtol))
        begin
        n = (phase >= -`M_PI/2)&&(phase < `M_PI/2);
        end
        // generate the output
        V(out) <+ transition (n?Vamp:0, 0, tran_time);
end
endmodule
```

Figure 4.9 VCO Verilog code

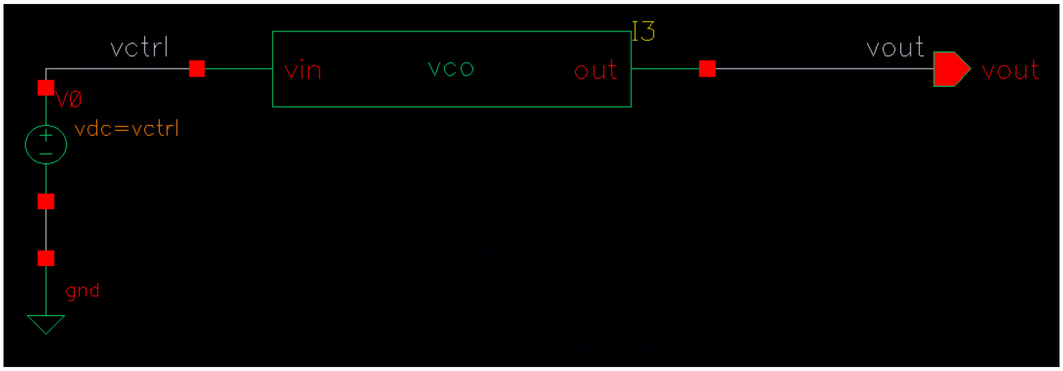


Figure 4.10: VCO testbench

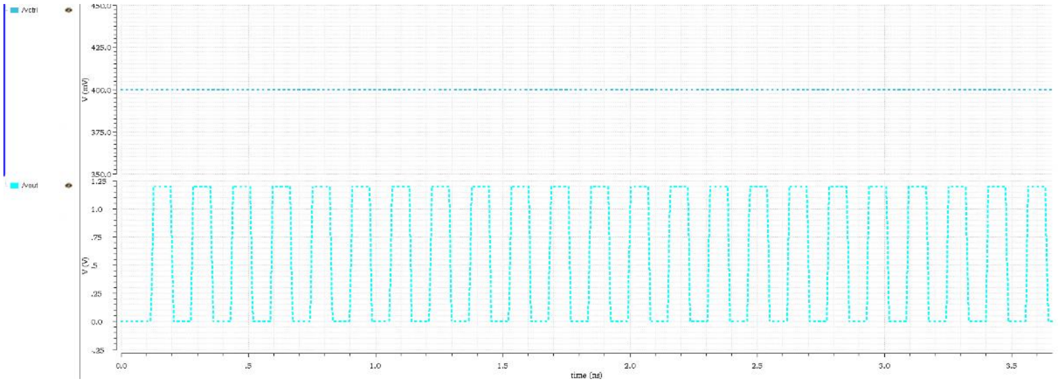


Figure 4.11: VCO output waveform

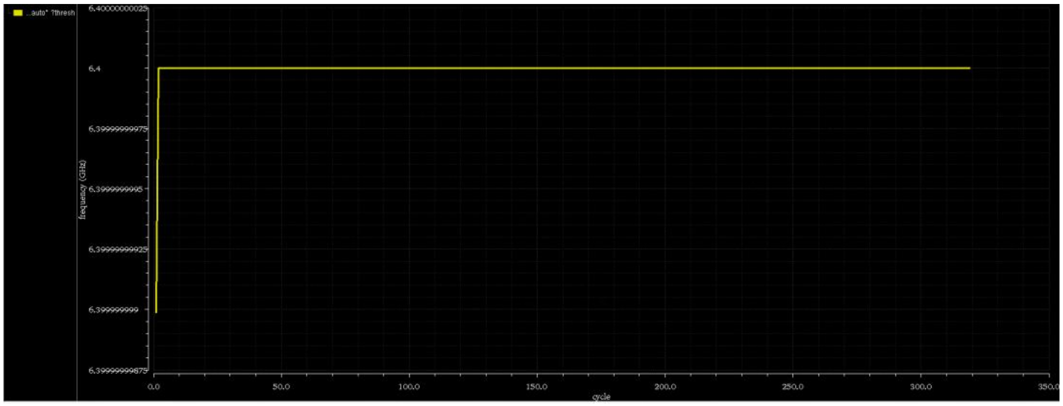


Figure 4.12: VCO output frequency

4.2.5 Entire CDR Modeling

When all the blocks are functioning correctly, it is time to connect all the modules together. The top-level simulation in Cadence Virtuoso is shown in Figure 4.13.

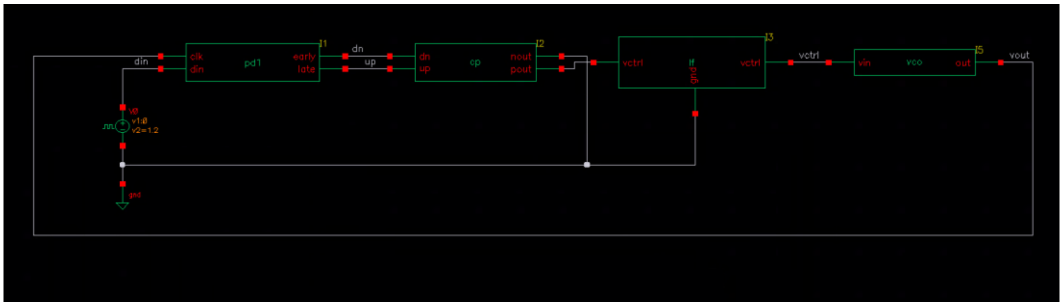


Figure 4.13: Top-level CDR simulation

The output signals along with the output frequency calculated from the output are shown in Figure 4.14 and Figure 4.15. We can see that the CDR starts to lock at about 37.2 μs.

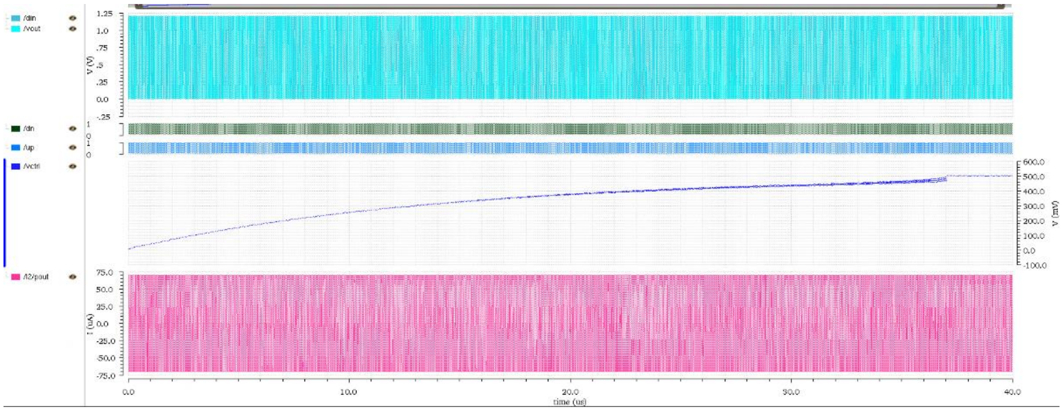


Figure 4.14: CDR output waveform

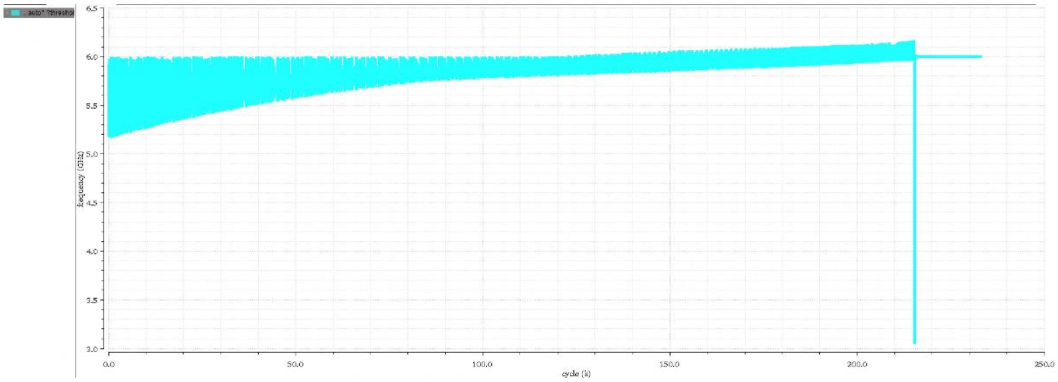


Figure 4.15: Output wave frequency interpolated

CHAPTER 5

CDR IMPLEMENTATION AT TRANSISTOR LEVEL

After completing behavioral modeling of the CDR circuit, we can move on to the simulation on the transistor level. There is more intricacy and complexity in the transistor-level design. We have to take more factors into consideration and there are more unpredictable variations.

In this chapter, we will discuss the transistor-level implementation of the building blocks of the CDR and a noise analysis of VCO will be included. The targeted data rate is 6.4 Gbps and the implementation is done in TSMC's 65 nm technology PDK. All the circuit designs are done in Cadence Virtuoso Spectre. The block diagram of the CDR is shown in Figure 5.1

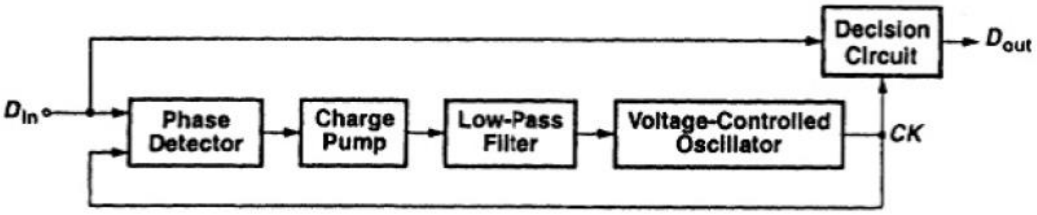


Figure 5.1: CDR block diagram

5.1 Phase Detector

The phase detector is the first component of the CDR. It takes incoming bit stream and generated clock signal from the VCO as inputs and compares the phase difference between them. The phase detector converts the incoming phase difference into voltage. When the CDR is locked, the phase difference between the reference clock and feedback clock should remain a constant value.

A linear phase detector, Hogge PD, is designed. The block diagram of the Hogge PD is shown in Figure 5.2. It has a positive edge triggered D flip-flop, a negative edge triggered D filp-flop, and two XOR gates. The flip-flops are in TSPC topology. Figure 5.3 and Figure 5.4 show the transistor-level design of the positive edge triggered D flip-flop and negative edge triggered D flip-flop. Figure 5.5 shows the waveform of the phase detector output.

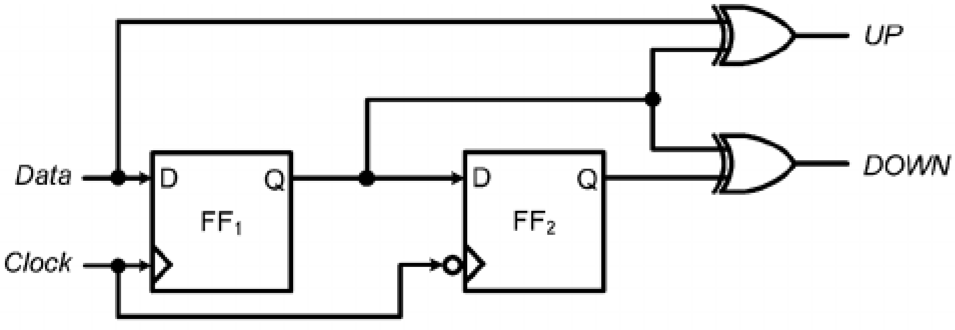


Figure 5.2: Phase detector block diagram

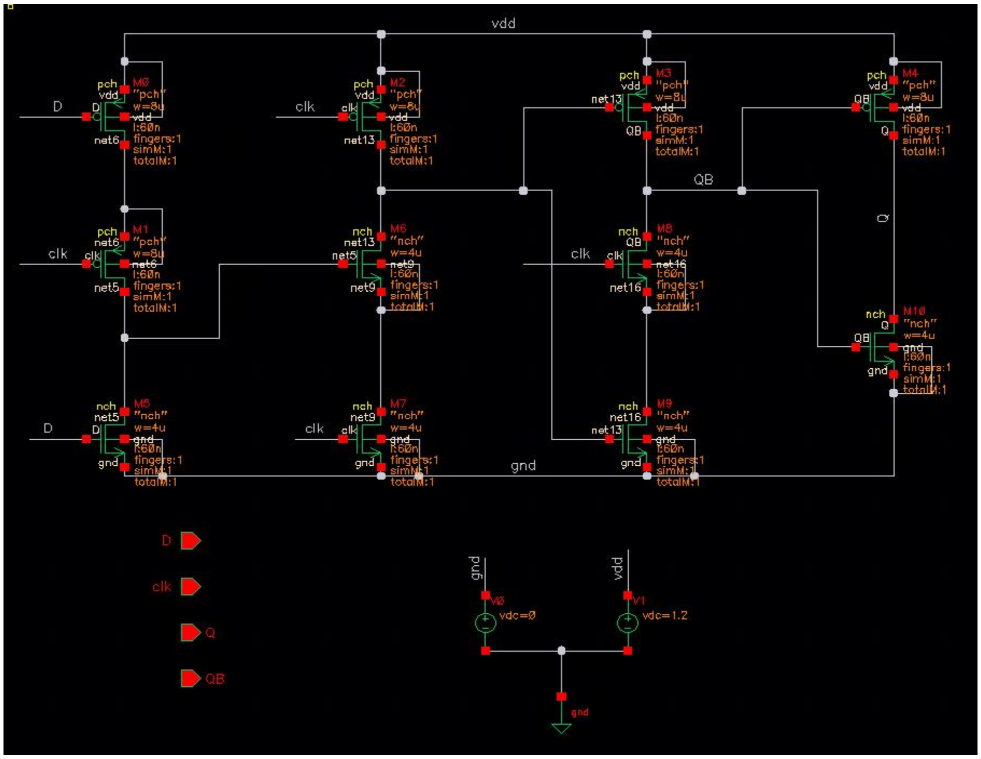


Figure 5.3: Transistor-level design of a positive edge DFF

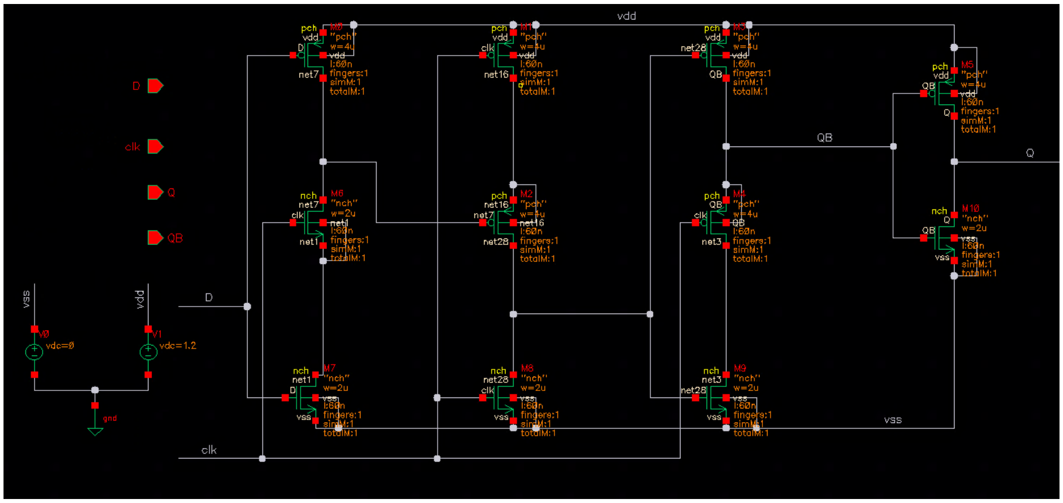


Figure 5.4: Transistor-level design of a negative edge DFF

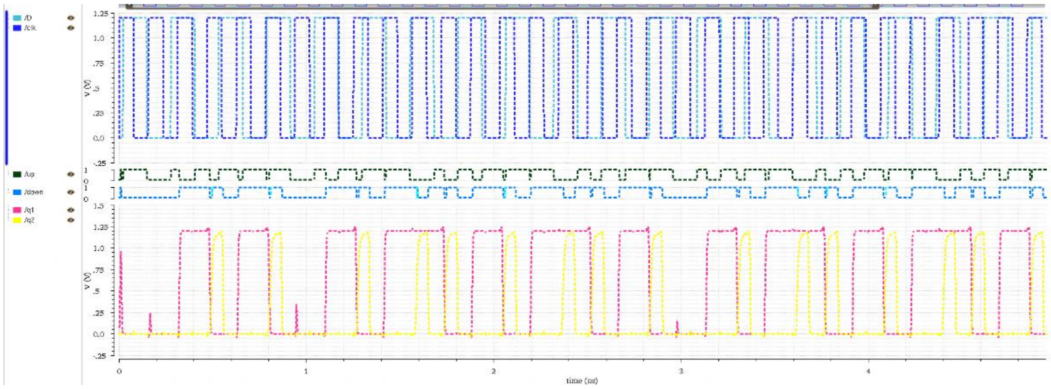


Figure 5.5: Phase detector output

There are times when the inputs to the CDR are not rail to rail. The voltage swing can be too low to drive a TSPC. This is when a sense amplifier flip-flop (SAFF) enters the picture. SAFF is a sense amplifier based flip-flop which can detect voltage difference. The implementation of a SAFF is shown in Figure 5.6. The transistor-level design of the SA latch is shown in Figure 5.7.

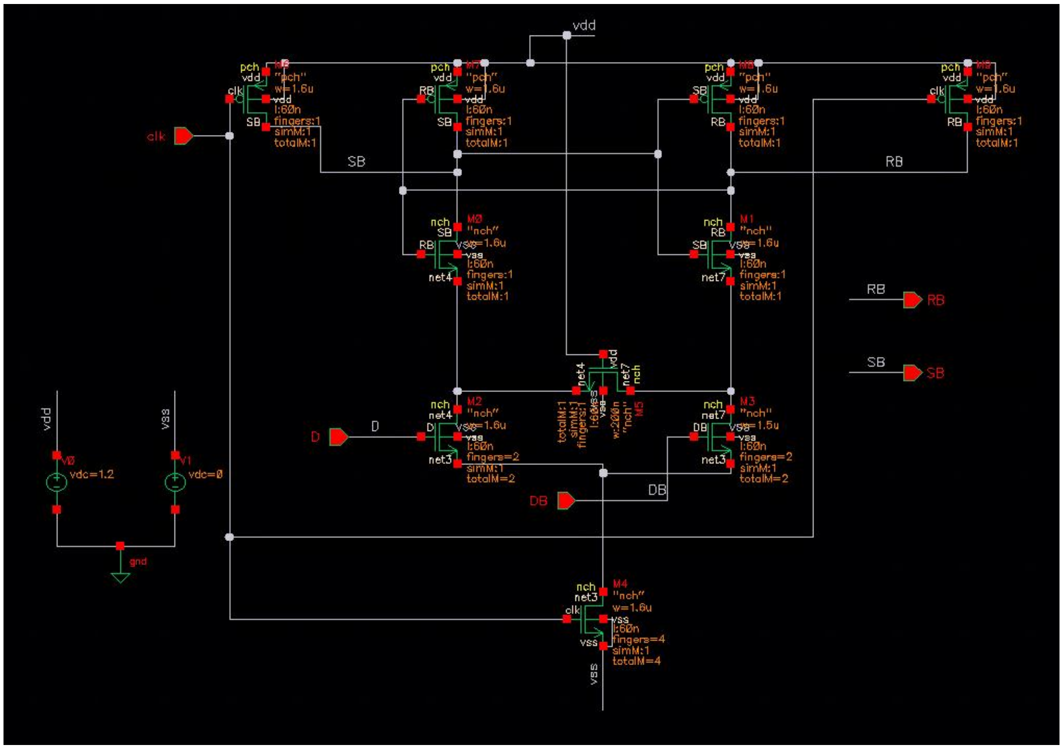


Figure 5.6 Transistor-level design of SAFF

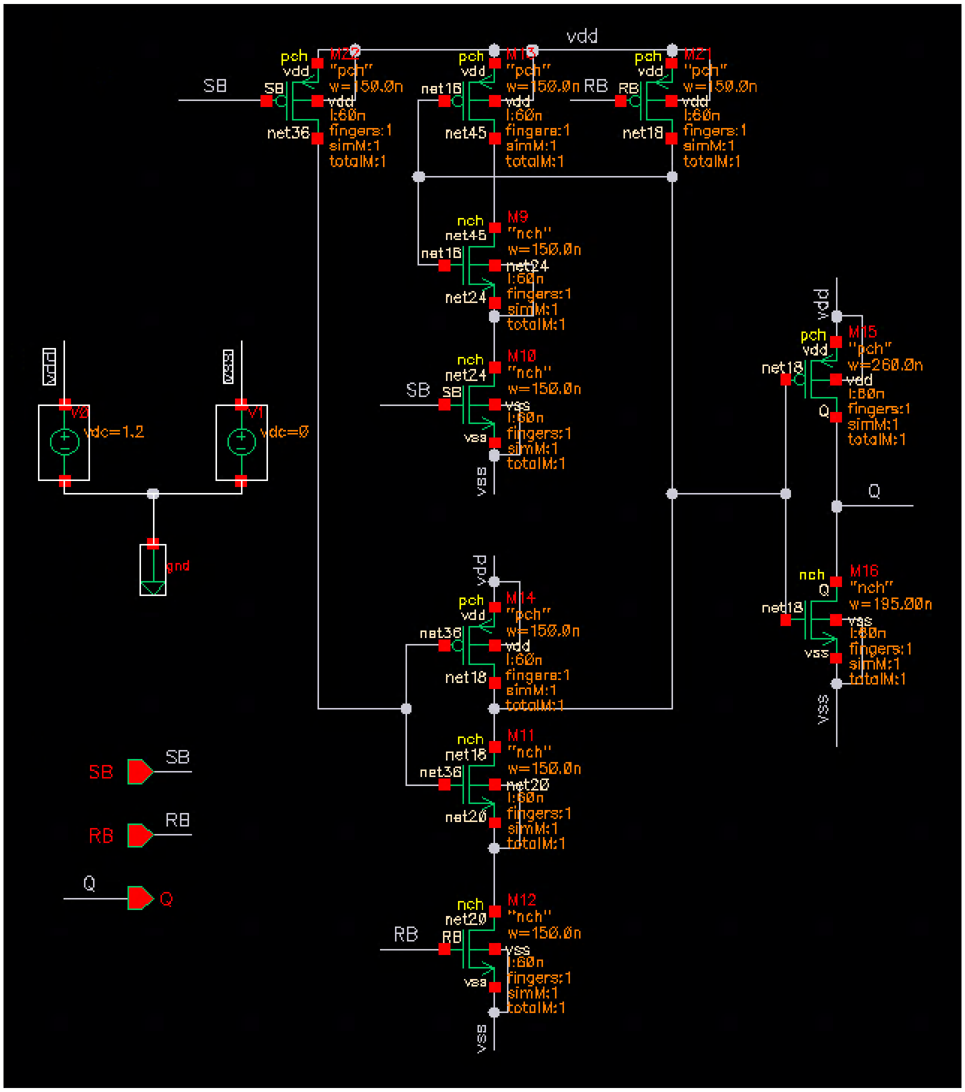


Figure 5.7: SA latch

5.2 Loop Filter

The output of the phase detector typically has a lot of high-frequency noise. Therefore, the LPF is to eliminate the high-frequency noise. Moreover, we need a charge storage device to maintain a stable input voltage signal to the voltage-controlled oscillator. Figure 5.8 shows the design of the loop filter.

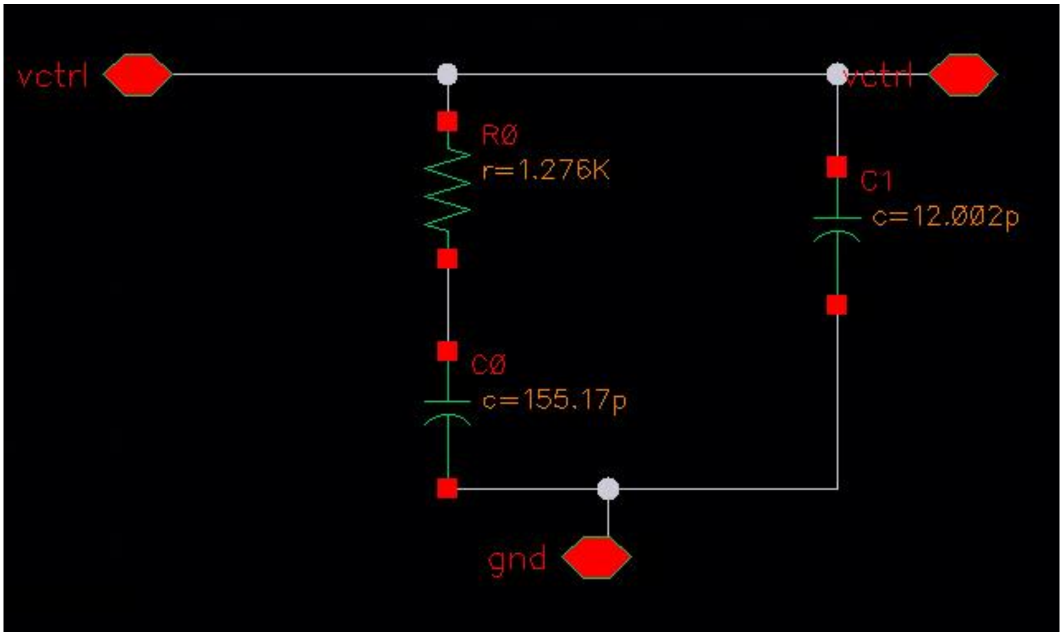


Figure 5.8: Loop filter schematic

5.3 Voltage-Controlled Oscillator

The voltage-controlled oscillator (VCO) is a device that can take in a control voltage and generate an output at a specific frequency. The output frequency ideally should be proportional to the input control voltage. A ring oscillator is designed for this CDR. The ring oscillator is a digital circuit which has an odd number of inverters and the last inverter output connected as the input to the first inverter. By utilizing the fact that the delay of each inverter depends on the amount of current it can sink in, the frequency of oscillation can be controlled. Figure 5.9 shows the transistor level VCO design. Figure 5.10 shows the output waveform of the VCO. Figure 5.11 shows the output frequency with respect to V_{ctrl}. Figure 5.12 shows the phase noise of the VCO.

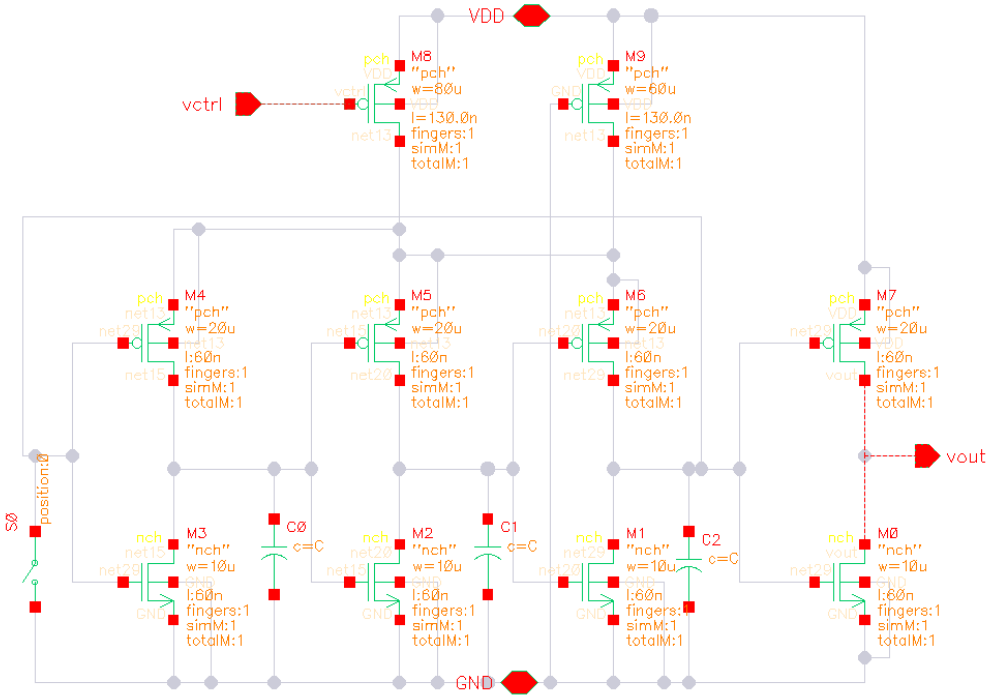


Figure 5.9: Transistor-level VCO design

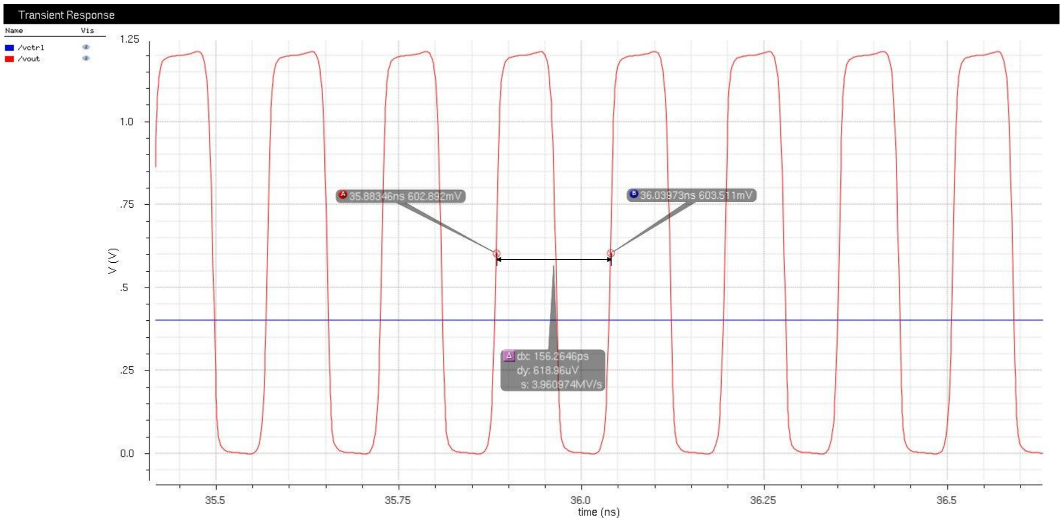


Figure 5.10: VCO output waveform

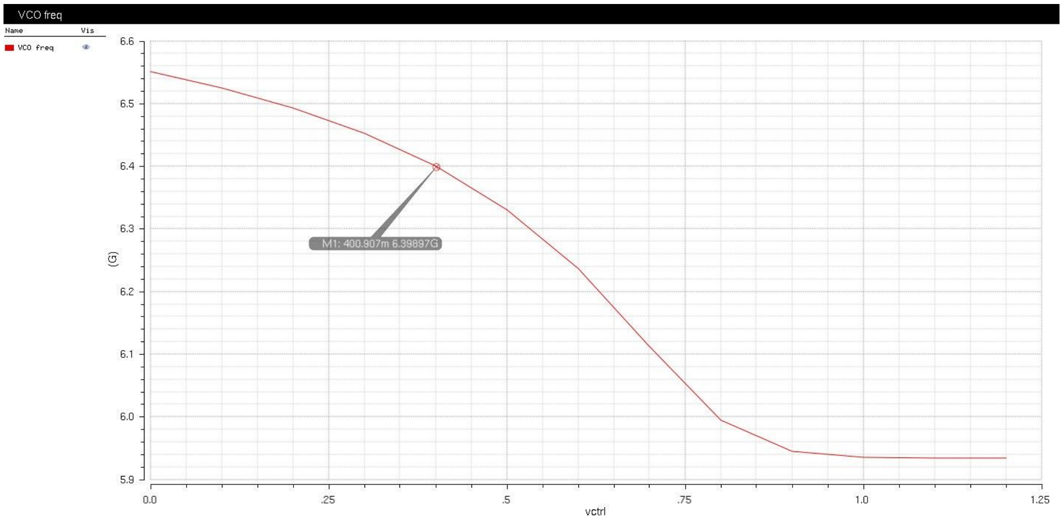


Figure 5.11: Output frequency vs. V_{ctrl}

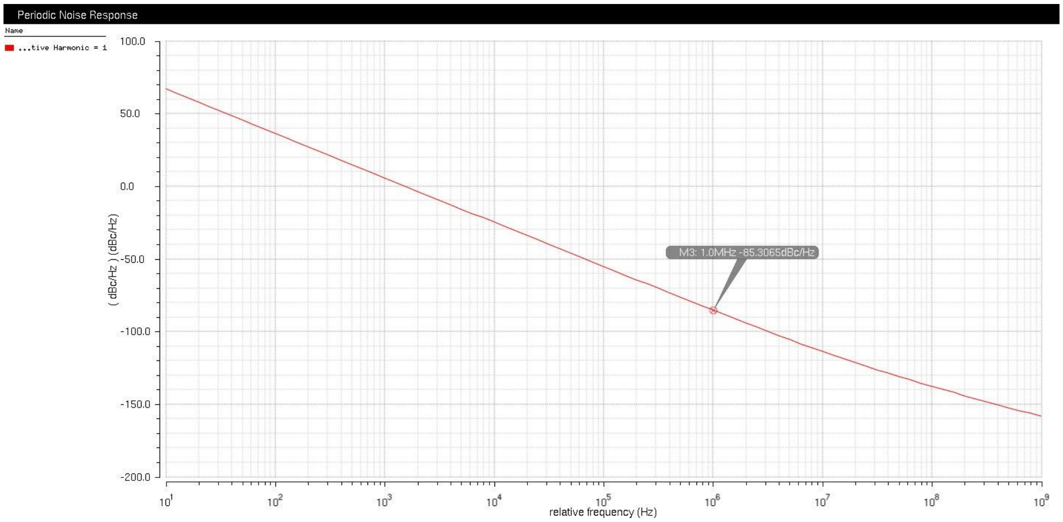


Figure 5.12: Phase noise of VCO

CHAPTER 6

CONCLUSION

In this thesis, the working mechanism of the CDR is described. A CDR consists of a phase detector, a charge pump, a loop filter and a voltage-controlled oscillator. This thesis provided an overview of all the building blocks of a PLL-based CDR, worked out the mathematical formulations of the negative feedback loop, and reported on a closed-loop behavioral modeling of the entire CDR and implementation of building blocks on the transistor level with TSMC 65 nm technology PDK with a 6.4 Gbps data rate. Also, this thesis provides a detailed noise analysis of the CDR.

More work can be done on this single-ended CDR design to gain a locked state at the targeted data rate to achieve stability on the transistor level. The designed CDR in this thesis is a single-loop PLL-based CDR. However, other topologies of CDR can also be implemented to improve the performance and stability. For example, there can be a dual-loop CDR with a PLL or DLL and phase interpolators, or a phase-rotator PLL [1]. Moreover, the design of each building block can be different such that the design of a phase detector can be different from a Hogge linear phase detector. Hopefully, there will be an entire link integration after the completion of all other HSSL components.

REFERENCES

[1] S. Palermo, “High-Speed Links Circuits and Systems,” class notes for ECEN 720, Department of Electrical and Computer Engineering, Texas A&M University, Jan. 2015.

[2] R. R. Dobkin, A. Morgenshtein, A. Kolodny, and R. Ginosar, “Parallel vs. serial on-chip communication,” in Proceedings of the 2008 International Workshop on System Level Interconnect Prediction, New York, NY, USA: ACM, 2008. [Online]. Available: http://doi.acm.org/10.1145/1353610.1353620 pp. 43–50.

[3] B. Razavi, “Clock and Data Recovery Circuits,” in *Monolithic Phase-Locked Loops and Clock Recovery Circuits: Theory and Design*, Wiley-IEEE Press, 1996, pp.381-381. doi: 10.1109/9780470545331.part5.

[4] M. Forcan, “Concept Creation and Design of a Parameterizable, fast-locking 65 nm CMOS CDR-PLL for Gigabit,” Available: http://hft.uni-duisburg-essen.de/arbeiten/Vortrag_Forcan_Milan.pdf

[5] P. Hanumolu, “ECE599: Phase-Locked Loops –II,” class notes for ECE 599, Department of Electrical and Computer Engineering, Oregon State University, Sep. 2012.

[6] K. Kundert and O. Zinke, *Designer’s Guide to Verilog-AMS*. Boston, MA: Kluwer-Academic Publishers, 2004.