

To the Graduate Council:

I am submitting herewith a dissertation written by James D. Vandersand Jr. entitled "An Analog Multiphase Self-Calibrating DLL to Minimize the Effects of Process, Supply Voltage, and Temperature Variations". I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Electrical Engineering.

Benjamin J. Blalock, Major Professor

We have read this dissertation
and recommend its acceptance:

M. Nance Ericson

Charles L. Britton, Jr.

Ethan Farquhar

Syed K. Islam

Vasilios Alexiades

Accepted for the Council:

Carolyn R. Hodges

Vice Provost and Dean of the
Graduate School

Original signatures are on file with official student records

An Analog Multiphase Self-Calibrating DLL to Minimize the Effects of Process, Supply Voltage, and Temperature Variations

A Dissertation
Presented for the
Doctor of Philosophy
Degree
The University of Tennessee, Knoxville

James D. Vandersand Jr.
May 2008

Dedication

This dissertation is dedicated to all that have helped me get to where I am today.

Acknowledgments

I would like to give a special thanks to Dr. Benjamin Blalock for giving me this opportunity to complete my Ph.D., the financial support, and allowing me to also train for hours a day in kung fu. In addition, I would like to thank my committee members Dr. Nance Ericson, Dr. Charles Britton, Dr. Vasilios Alexiades, Dr. Ethan Farquhar, and Dr. Syed Islam for serving on my dissertation committee. Also, thanks for Dr. Ericson for allowing me to have access to equipment needed for testing, and providing me with necessary feedback to make this dissertation possible.

Without Boeing allowing me to have access to silicon space on one of their projects, this dissertation would not have been possible.

To my parents and the rest of my family, thank you for providing encouragement along the way, even though you had no technical understanding of what it was that I was doing.

Thanks to all the members of the ICASL team for having to put up with me for all these years. Chandra Durisetty, Stephen Terry, Mark Hale, and Suheng Chen have always been helpful through numerous projects, and of course, Robert Greenwell always made sure that the PDK was running.

I am extremely thankful to Rajagopal Vijayaraghavan for helping acquire a job, which ultimately drove me to finish my dissertation.

To my sifu Leroy Kautz, and the rest of my kung fu family, thanks for always being there, and keeping me focused on my physical and mental training.

Abstract

Delay locked loops have been found to be useful tools in such applications as computing, TDCs, and communications. These system can be found in space exploration vehicles and satellites, which operate in extreme environments. Unfortunately, in these environments supply voltage and temperature will not be constant, therefore they must be under consideration when designing a DLL. Furthermore, solar radiation in conjunction with the varying environmental aspects, could cause the delay locked loop to lose it locked state.

Delay locked loops are inherently good at tracking these environmental aspects, but in order to do so, the voltage controlled delay line must exhibit a very large gain, which translates to a large capture range. Assuming charged particles hit a key node in the DLL (e.g. the control voltage), the DLL would lose lock and would have to recapture it. Depending on the severity of the fluctuation, this relocking process could easily take on the order of many microseconds assuming the bandwidth was kept low to minimize jitter. To date, no delay locked loops have been published for extreme environment applications.

In many other extreme environment circuits, calibration techniques have been applied to minimize the environmental effects. Whereas there have been multiple calibration methods

published related to delay locked loops, none of them were intended for extreme environments. Furthermore, none of these methods are directly suitable for an analog multiphase delay locked loop.

The self-calibrating DLL in this work includes an all digital calibration circuit, as well as a system transient monitor. The coarse calibration helps minimize global process, voltage, and temperature errors for an analog multiphase DLL. The system monitor is used to detect any transients that might cause the DLL to unlock, which could be used to allow the DLL to be recalibrated to the new environmental conditions. The presented measurement results will demonstrate that the DLL can be used in extreme environments such as space, or other extreme environment applications.

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Research Goals	3
1.3	Overview of the Dissertation	4
2	Delay Locked Loop Fundamentals	6
2.1	Introduction	6
2.2	DLL System Fundamentals	6
2.2.1	DLL Linearized Model	8
2.2.2	DLL Noise Issues	10
2.3	Types of DLLs	16
2.3.1	Digital Delay Locked Loops	16
2.3.2	Analog Delay Locked Loops	18

2.4	Summary	27
3	CMOS Scaling and Extreme Environment Operation and their Implications on DLLs	28
3.1	Introduction	28
3.2	CMOS Process, Voltage, and Temperature Variations Associated with the DLL	30
3.2.1	Process Variations	30
3.2.2	Supply Voltage Variations	33
3.2.3	Temperature Effects	35
3.2.4	Combined Effects	37
3.3	DLL Calibration Techniques	38
3.3.1	Global (Coarse) PVT Calibration Prior Art	39
3.4	Summary	50
4	The Self-Calibrating DLL	52
4.1	Self-Calibrating DLL System Architecture	53
4.1.1	DLL Specifications	53
4.1.2	Voltage Control Delay Line	55
4.1.3	Charge Pump and Loop Filter	59
4.1.4	Phase Detector	61

4.2	Coarse Calibration	63
4.2.1	Digitally Controllable VCDL	64
4.2.2	Digital Coarse Calibration Circuit	68
4.2.3	Post Coarse Calibration	75
4.3	System Transient Monitor	80
4.4	DLL Layout	81
4.5	Full DLL Simulations	83
4.6	Summary	86
5	Self-Calibrating DLL Test Results	88
5.1	Introduction	88
5.2	Test Setup	88
5.2.1	Coarse Calibration Test	89
5.2.2	Transient Monitor Test	89
5.2.3	Test Boards and Equipment	90
5.3	Test Results	96
5.3.1	Coarse Calibrations Test	96
5.3.2	System Transient Monitor Test Results	112
5.3.3	Comparison to Prior Art	114

5.4	Summary	114
6	Future Work	116
6.1	Introduction	116
6.2	Alternate DLL Components	116
6.3	Alternate System Transient Monitor	119
6.4	Further Testing	121
6.5	Fine Calibration	121
6.6	Summary	122
7	Conclusions	123
	Bibliography	124
	Appendix	130
	Vita	148

List of Figures

2.1	Basic PLL architecture.	7
2.2	Basic DLL architecture.	7
2.3	DLL used in clock recovery applications.	8
2.4	DLL linearized model.	9
2.5	DLL Z-domain model.	13
2.6	Example jitter transfer function for a type-I DLL	14
2.7	Jitter peaking timing diagram.	14
2.8	Typical all digital DLL block diagram.	17
2.9	XOR type phase detector.	19
2.10	Common phase-frequency detector.	20
2.11	Dynamic phase detector.	21
2.12	Simplified charge pump.	22
2.13	Basic charge pump schematic.	23

2.14	Block diagrams of single-ended and differential VCDLs.	25
2.15	Current starved inverter delay cell.	25
2.16	Inverter delay cell with an RC load.	26
2.17	Differential delay cells with diode loads (left) and triode loads (right). . . .	27
3.1	Simple current starved inverter delay cell.	32
3.2	Monte Carlo results for the current starved inverter delay cell of Figure 3.1.	32
3.3	Simulation results at various V_{DD} levels for the current starved inverter delay cell of Figure 3.1.	34
3.4	Simulation results at different temperatures for the current starved inverter delay cell of Figure 3.1.	37
3.5	Simulation results at for various PVT shifts for the current starved inverter delay cell of Figure 3.1.	38
3.6	Calibrating PLL via an auto-lock circuit	40
3.7	Calibrating PLL via a digital temperature compensation system	41
3.8	Self-calibrating digital DLL using an all digital calibration technique	42
3.9	Digitally tunable VCO with transfer characteristics	44
3.10	PLL with coarse calibration using a VCO with digital tuning	44
3.11	Temperature compensated VCO using a DLL	45
3.12	False lock prevention circuit for DLL	46

3.13 Self-calibrating DLL	46
3.14 Edge detection circuit	47
3.15 Self-calibrating DLL using a TDC based approach	48
3.16 Calibration based TDC with encoder	49
3.17 Self-calibrating PLL presented by Aktas	49
3.18 Digitally controllable LC-VCO	50
4.1 System diagram of the self-calibrating DLL.	54
4.2 Simplified delay cell.	56
4.3 Voltage-to-current converter.	57
4.4 Charge pump schematic.	60
4.5 Main DLL phase detector.	62
4.6 Main DLL phase detector simulation with small phase error at the inputs. .	62
4.7 Charge pump transfer plot.	63
4.8 Delay cell schematic with complementary MOSCAPs.	64
4.9 Simulated VCDL delay versus control voltage for all digital calibration codes.	66
4.10 Simulated VCDL delay versus control voltage zoomed to the lower gain range.	66
4.11 Simulated VCDL delay versus control voltage across temperature and supply voltages.	68

4.12 Coarse calibration algorithm flow chart.	69
4.13 Edge detection circuit used for the coarse calibration block.	72
4.14 Example of VCDL outputs for delay being to long.	72
4.15 Clock edge generation circuit used for the coarse calibration circuit.	73
4.16 Circuit used to initially charge loop filter.	74
4.17 Phase detector enable issue.	76
4.18 Circuit used to enable the phase detector.	76
4.19 Example of the process used to enable the main phase detector.	78
4.20 Schematic of the DLL transient monitor.	81
4.21 Waveforms of the system transient monitor where (a) is the slow case, and (b) is the fast case.	82
4.22 Layout of the self-calibrating DLL.	82
4.23 Simulation of the control voltage and coarse calibration bits for an input clock of 62.5 MHz.	84
4.24 Plot of all of the reference clock and all of the DLL output clocks for an input clock of 62.5 MHz.	84
4.25 Closeup of the locked input and output clocks for an input clock of 62.5 MHz.	85
4.26 Simulation of the control voltage, calibration done, and calibration bits for an input clock of 125 MHz.	85
4.27 Closeup of the locked input and output clocks for an input clock of 125 MHz.	86

5.1	Circuit used to test DLL transient monitor.	90
5.2	Block diagram of the test setup.	91
5.3	Simplified schematic of the bias board.	92
5.4	PCB layout of the bias board.	93
5.5	Simplified schematic of the DLL board.	93
5.6	PCB layout of the DLL board.	94
5.7	Actual bias board.	94
5.8	Actual DLL board.	95
5.9	Full characterization setup.	95
5.10	Self-calibrating DLL control voltage for an input clock of 110 MHz.	97
5.11	DLL locked to 75 MHz reference clock.	98
5.12	Coarse calibration ranges for Chip 1 with $V_{LOWER} = 0.9$ V and $V_{UPPER} = 1.2$ V at room temperature.	99
5.13	Coarse calibration ranges for Chip 1 with $V_{LOWER} = 1.1$ V and $V_{UPPER} = 1.35$ V at room temperature.	99
5.14	Coarse calibration ranges for Chip 2 with $V_{LOWER} = 0.9$ V and $V_{UPPER} = 1.2$ V at room temperature.	100
5.15	Coarse calibration ranges for Chip 2 with $V_{LOWER} = 1.1$ V and $V_{UPPER} = 1.35$ V at room temperature.	100
5.16	Measured detailed curve for digital code 11111 (all capacitors switched in) for Chip 2 with $V_{LOWER} = 1.1$ V and $V_{UPPER} = 1.35$ V at room temperature. 101	

5.17	Measured temperature data for chip 1 at 1.7 V.	104
5.18	Measured temperature data for chip 1 at 1.8 V.	105
5.19	Measured temperature data for chip 1 at 1.9 V.	105
5.20	Measured temperature data for chip 1 at 2.0 V.	106
5.21	Measured temperature data for chip 1 at 2.1 V.	106
5.22	Measured temperature data for chip 2 at 1.7 V.	107
5.23	Measured temperature data for chip 2 at 1.8 V.	107
5.24	Measured temperature data for chip 2 at 1.9 V.	108
5.25	Measured temperature data for chip 2 at 2.0 V.	108
5.26	Measured temperature data for chip 2 at 2.1 V.	109
5.27	Cumulative temperature and voltage measured data for chip 1.	109
5.28	Cumulative temperature and voltage measured data for chip 2.	110
5.29	Locked DLL waveform at 51 MHz for chip 1 with a 2.1 V supply at 175°C.	110
5.30	Locked DLL waveform at 111 MHz for chip 1 with a 2.1 V supply at 175°C.	111
5.31	Lock error test where the VCDL became too fast	113
5.32	Lock error test where the VCDL became too slow	113
6.1	Common differential VCDL architecture.	117
6.2	Differential pair delay cell with digital calibration.	118

6.3	a) Bias generator with calibration, and b) Inverter based differential pair delay cell with digital calibration.	119
6.4	Alternative solution to the system transient monitor.	120
6.5	Proposed fine calibration system.	122
1	Duty cycle error for VCDL too fast case.	136
2	Self-calibrating DLL with coarse and fine calibration capabilities.	138
3	Fine calibration algorithm.	142
4	Fine calibration block diagram.	143
5	Delay cell with fine calibration digital control.	143
6	Proposed pseudo-digital phase detector used for the fine calibration system.	146
7	Simulation of the modified phase detector where slave leads the reference clock.	147

Chapter 1

Introduction

1.1 Motivation

Applications such as wireless communications, data communications, time-to-digital converters (TDC), and computing systems all require some form of precise clocking. In wireless communications, a frequency synthesizer can be used for the local oscillator block of the transceiver/receiver system. In data communication systems, wireless or wired, data needs to be extracted from the data stream at the correct time, otherwise data errors will occur. Furthermore, in many data communication systems it is necessary to extract the clock from the incoming data. TDCs need a precise clock as a means to properly convert some time step to a digital word. In computing systems similar reference clocks are needed, where clock distribution to processors and memory require clocking from multiple phases of a system clock to process data at high frequencies.

There have been numerous systems designed, both board level and integrated, to address the needs of the applications mentioned above. One of the most common circuits that has been used in communications systems is the Phase-Locked Loop, or PLL. The PLL has been widely used for frequency synthesis and clock-and-data recovery circuits [1–4]. The

PLL in its general form is a second order feedback loop whose system dynamics have been the focus of many books and journal articles. A variation of the PLL has been used as an alternative clock generation circuit, available in both a digital [5] and analog [6] forms, which simplifies the feedback loop to a single-order system, thereby inherently stable, by replacing the voltage-controlled oscillator with a voltage-controlled delay line. This system has been come to known as the Delay-Locked Loop, or DLL. The DLL has been used as a possible solution for all of the previously mentioned applications [7–12]. Even though the PLL and DLL can be implemented with discrete components, this research will focus solely on the integrated circuit version.

Unfortunately most these systems will not be operating under ideal conditions. Whether it be in the depths of space or the controlled lab bench environment, fluctuations in temperature and supply voltage are inevitable. In space and planets other than earth, temperatures can exceed the military specified range (-55°C to 125°C) for which many electronics are designed. The DLL can be found in a spaced based TDC presented by Karadamoglou [13]. Systems such as NASA’s RAD6000 microprocessor [14] could easily use a DLL for memory clocking or clock deskewing, though it is not mentioned in the literature.

Integrated circuits are fabricated on some type of semiconductor material, where silicon (Si) is the most common type. The MOSFET, being the most prevalent semiconductor device used, has led to decades of research in an attempt to enhance its performance. For integrated circuit designers, including both analog and digital, performance enhancement includes making circuits faster, consume less power, and use less silicon area. The semiconductor foundries have responded by creating CMOS processes that have continually reduced the gate feature size as well as the voltage supply. In doing so, devices with higher cutoff frequencies are produced allowing for faster circuits [15,16]. Unfortunately there are negative side-effects that stem from this process scaling. One of the most obvious effects is that device mismatch becomes much more pronounced as the processes reach deep sub-micron technologies [16]. This leads to problems such as mismatched currents, switching points,

and transconductances just to name a few. Also, it is difficult to generate a perfectly steady DC power supply due to every supply exhibiting some finite output impedance. With DC fluctuations on the power rails, the operating point of the devices will change too, thereby altering the circuit's performance. Beyond process mismatch and voltage supply variations, temperature change also causes problems in MOSFETs. The threshold voltage and mobility are functions of temperature; thereby the circuit's performance will be affected [16]. In space applications, the extreme temperature range could very easily cause a circuit to fail to meet its designed specifications, or even the possibility of not working at all. Process, voltage, and temperature variations are typically referred to as PVT variations in much of the literature. These problems may not be of much consequence in simple circuits or systems, but as the system begins to rely on precise or accurate values of some timing, voltage reference, or gain specification of a sub-circuit, proper operation of the system may be compromised.

In the case of the DLL, multiple issues can arise due to the PVT variations. For one, it is possible that the variations could push the actual locking range out of the designed specification. If the DLL is unable to achieve a locked condition, every single one of the previously mentioned applications will fail. The frequency synthesizer would generate an invalid frequency and the TDC, clock distribution, and data communication applications would cease to operate correctly due to an excessive number of errors. Assume that the DLL is in a locked state for the current environmental and electrical conditions. If either the temperature or supply voltage changes (process parameters might change due to a radiation event or device aging), it is possible for the DLL to lose lock.

1.2 Research Goals

This research focuses on the following:

- investigate the issues that are the main sources of lock errors in a DLL,
- design a global calibration scheme that will ensure that an analog multiphase DLL is able to lock over a wide process, supply voltage, and temperature range,
- implement a system monitor that will generate an error in the event that the DLL loses lock,
- prototype a DLL that demonstrates the proposed calibration scheme and system monitor including supply and temperature measurements that exceed the military specified range.

These goals will be accomplished by studying the literature involving both PLL and DLL circuits to determine the requirements for locking and to identify major sources of timing error, which can include process, voltage, and temperature shifts. This prior art investigation will include any techniques that can be used to minimize or calibrate the listed sources of error. After completing the literature review, the calibration schemes and system monitor will be developed at a system level, and then be designed and implemented at the circuit level. The implemented calibration schemes will be digital in nature, but the fine resolution calibration technique required the design of a new pseudo-digital phase detector architecture for the system to be realized. This DLL will then be implemented in a 1.8-V 150-nm partially depleted silicon-on-insulator (PDSOI) CMOS technology, which will help prove its viability in deep sub-micron technologies. The prototype DLL's results will be compared to the prior art in locking range, chip area, jitter, and effectiveness of the calibration.

1.3 Overview of the Dissertation

This dissertation is broken up into seven chapters. Chapter 2 presents a brief overview of the PLL and DLL system dynamics, including the components associated with them. In

Chapter 3, an overview of CMOS scaling trends and temperature effects, and the problems associated with scaling DLL circuits, as well as a literature review of calibration techniques applied to DLL and some applicable PLL architectures. Chapter 4 will present the proposed self-calibrating DLL, including the DLL system design, component design, and design of the calibration systems. The test results will be presented in Chapter 5. Chapter 6 will present future additions to the self-calibrating DLL. Finally, conclusions will be given in Chapter 7.

Chapter 2

Delay Locked Loop Fundamentals

2.1 Introduction

The main purpose of Chapter 2 is to provide the fundamentals of the operation and design of the delay locked loop as a means to understand the issues needed to design a method to calibrate out PVT induced errors. The DLL was derived from the PLL, which is why its architecture and loop dynamics are very similar. In Section 2.2 DLL fundamentals will be presented, which includes a discussion of the loop dynamics, different types of DLL architectures, and the major DLL components.

2.2 DLL System Fundamentals

The basic PLL architecture is shown in Figure 2.1. The DLL differs slightly from a PLL in that both systems share all of the same components with the exception that the DLL uses a voltage-controlled delay line, VCDL, as opposed to a VCO, and it does not use a feedback divider as a gain element, with the exception of recirculating DLL clock multipliers [17]. There are two classifications of DLLs as shown in Figure 2.2 and Figure 2.3. The most

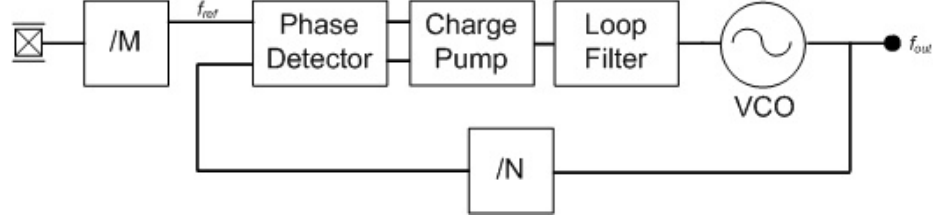


Figure 2.1: Basic PLL architecture.

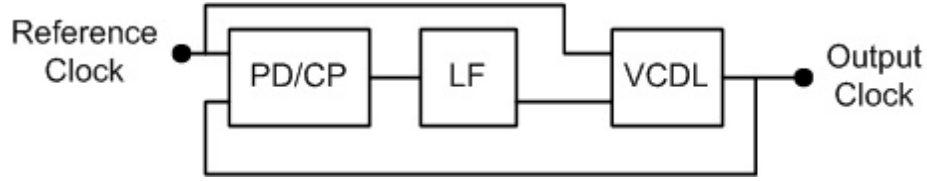


Figure 2.2: Basic DLL architecture.

common type of DLL, shown in Figure 2.2, can be used for applications such as clock generation, frequency synthesis, and clock synchronization. This DLL architecture has only one reference clock input that is used in both the phase detector and the VCDL. The second type of DLL, shown in Figure 2.3, is commonly used for clock and data recovery circuits. This DLL architecture consists of two input clocks, a reference clock that is connected to the phase detector, and an uncorrelated source clock that is connected to the VCDL. This work will focus on the first type of DLL shown in Figure 2.2.

The negative feedback operation of the DLL is similar to that of the PLL in that the output of the VCDL is compared to a reference clock, just like the output of the VCO is in the PLL. Referring to Figure 2.2, at some initial condition, the VCDL is biased such that its input, in this case the reference clock, is delayed by a certain amount of time. When the delayed clock edge reaches the phase detector, it is compared to the edge of the reference clock. For now it is assumed that the rising edge is compared, but it is just as easy to change the architecture of the phase detector to compare the falling edges. If the VCDL output

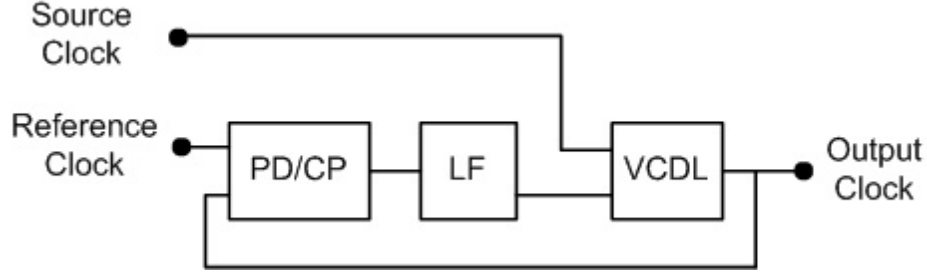


Figure 2.3: DLL used in clock recovery applications.

edge is leading the reference clock, it means that the VCDL's delay is too fast and the delay must be reduced. The phase detector will output a signal to the charge pump, which entails that the voltage at the loop filter is lowered. The amount that the bias voltage is decreased is based upon the magnitude of the phase difference between the input and output signals. The reverse operation would occur in the case where the VCDL output edge was lagging that of the reference clock. This operation continues until the output and reference clock edges are aligned.

2.2.1 DLL Linearized Model

To analyze the DLL, the first assumption that must be made is that it is a linear system. This is true when the DLL is close to lock as with the PLL [18]. The linearized DLL model is shown in Figure 2.4. The phase detector is described by the summing node and the gain factor K_{PD} , the charge pump by I_{CP}/T_{REF} , the loop filter $F_{LF(s)}$, and the VCDL by the gain factor K_{VCDL} .

Performing a loop analysis on Figure 2.4, the closed loop phase transfer equation is found to be

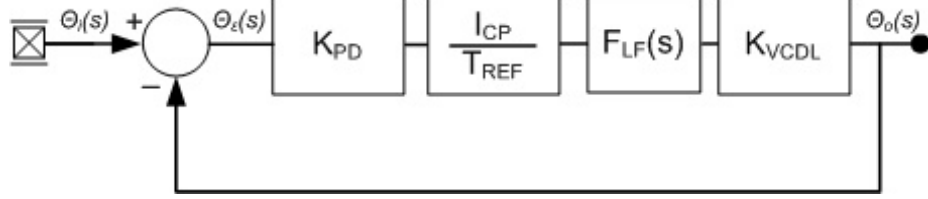


Figure 2.4: DLL linearized model.

$$\frac{\Theta_o(s)}{\Theta_i(s)} = \frac{1}{1 + \frac{s}{\omega_N}}, \quad (2.1)$$

where the a loop bandwidth is

$$\omega_N = \frac{K_{PD}K_{VCDL}I_{CP}}{2\pi C_{LF}}\omega_{REF}. \quad (2.2)$$

This analysis is provided in detail in Appendix A.

It is obvious from (2.1) that the DLL is a single pole system that is unconditionally stable, assuming that none of the terms in (2.2) have any reactive component associated with them. This aspect makes the DLL more desirable to the multiple pole PLL for some applications.

Equation (2.2) demonstrates that the DLL's bandwidth will changes as the input clock period changes assuming that the other terms remain constant. This means that as the input frequency increases, the loop bandwidth increases, which allows for a faster locking process. This same increase in bandwidth also means that more noise seen at the input will

be propagated through to the output due to the low pass nature of the loop. According to [19], a rule of thumb used for designing both PLLs and DLLs is

$$\frac{\omega_N}{\omega_{REF}} = \frac{K_{PD}K_{VCDL}I_{CP}}{2\pi C_{LF}} \leq \frac{1}{10}. \quad (2.3)$$

Another point that should be noted is that in the above analysis, the VCDL transfer characteristic should contain a delay term since its output should ideally delay the input signal by one period. This was analyzed in [20] by adding the s-domain equivalent to a time delay and showing that the loop was still stable.

2.2.2 DLL Noise Issues

Noise consideration is an important design aspect in DLL design. The standard specification for noise in DLLs is referred to as jitter. Jitter is the random variations of a signal in the time domain due to environmental and circuit noise. Ideally a clock has a period given as T_{nom} , but the actual clock period is defined as $T_n = t_{n+1} - t_n$ at some n th crossing. This crossing could be defined as the positive or negative going edge, but for this work the positive edge will be considered as the reference point. Jitter is defined as the difference between the actual clock period and the nominal clock period, or $\delta T_n = T_n - T_{nom}$.

There are three standard definitions for jitter as described by [21]. The first type of jitter describes the total accumulated jitter from the first cycle until N clock cycles, which is known as absolute or long term jitter. Absolute jitter is written as

$$\Delta T_{abs}(N) = \sum_{n=1}^N \Delta T_n. \quad (2.4)$$

The second type of jitter, cycle jitter, is a long term average, or RMS, of the clock cycle variations ΔT_n . Cycle jitter is defined as

$$\Delta T_c = \lim_{N \rightarrow \infty} \sqrt{\frac{1}{N} \sum_{n=1}^N \Delta T_n^2}. \quad (2.5)$$

Cycle-to-cycle jitter is the third type described in [21], which is the RMS of the difference between two consecutive clock cycles. This is written as

$$\Delta T_{CC} = \lim_{N \rightarrow \infty} \sqrt{\frac{1}{N} \sum_{n=1}^N (T_{n+1} - T_n)^2}. \quad (2.6)$$

It is mentioned in [21] that absolute jitter is useful for describing PLLs due to the accumulating nature of the system. However, for DLLs and other timing circuits, the other parameters are used most of the time.

To be able to quantify DLL jitter, the sources of jitter must first be determined. A thorough analysis of DLL jitter was performed in [22], and it is shown that deterministic and random jitter are the two major components of the total DLL jitter. The deterministic type of jitter is mainly caused by device mismatches and non-ideal input clock effects such as jitter and duty cycle errors. Random jitter is predominately caused by thermal, substrate, and power supply noise.

Normally, a low jitter DLL is desirable to minimize timing errors in the desired application. In order to design a low jitter DLL, it is important to understand the main components of deterministic jitter, which is also known as systematic jitter.

The main source of DLL systematic jitter is due to the input clock. Looking at the transfer function (2.1), it is obvious that any phase error occurring at the input, be it from the clock or another source, will go through a low pass filter. This would mean that any noise above the DLL loop bandwidth would be filtered out. Unfortunately, that only explains part of the problem. It is important to consider the time-domain, or even better, the Z-domain transfer functions since the DLL is a sampling process. A type-I DLL is shown in Figure 2.5 with its Z-domain equivalent functional blocks. In this block diagram, a z^{-1} block is used in the feedback path to model that the output clock is delayed by one clock period. For the single capacitor loop filter, the product of K_{CP} and $F_{LF}(z)$ is given as

$$K_{CP}F_{LF}(z) = \frac{I_{CP}T_{ref}}{2\pi C_{LF}} \left(\frac{z}{z-1} \right). \quad (2.7)$$

Performing the loop analysis on Z-domain DLL yields

$$\frac{\Theta_o(s)}{\Theta_i(s)} = \frac{(\alpha + 1)z - 1}{z - (1 - \alpha)}, \quad (2.8)$$

where α is

$$\alpha = \frac{K_{PD}K_{VCDL}I_{CP}T_{ref}}{2\pi C_{LF}}. \quad (2.9)$$

The result of 2.8 matches with that by Lee [22].

It is apparent that a pole exists at $(1-\alpha)$ and zero at $1/(\alpha+1)$. By using acceptable values for α , it is possible to create the jitter transfer plot in Figure 2.6 [22]. It can be seen that the jitter peaks as it approaches half of the DLL sampling frequency. As pointed out in [22], the jitter above half of the sampling frequency is aliased down due to the discrete-time nature of the DLL.

To qualitatively understand this jitter peaking process, it is appropriate to demonstrate output shifts due to input clock jitter [22]. To illustrate the jitter peaking process, Figure 2.7 demonstrates a possible scenario where the input clock deviates from its ideal state. To start, the DLL's output is locked with the input clock. In the next cycle, the reference clock's rising edge is delayed by some time $\delta 1$, causing the phase detector to see the output signal as leading the input signal. This action will attempt to reduce the VCDL delay by lowering the control voltage. The input clock jump would not be seen at the output of the VCDL until one clock period after it occurs, which is the third rising edge in Figure 2.7. Due to the immediate change in the control voltage and the shift in the input clock, there is a double hit in terms of timing error, which is denoted by $\delta 1 + \delta 2$. It is desirable to have a fast locking and tracking DLL, which requires a large loop gain, but in order to reduce jitter peaking, it is necessary to keep a low loop gain. As pointed out in [22], it is possible to keep the steady state gain down by allowing for a startup circuit that puts the

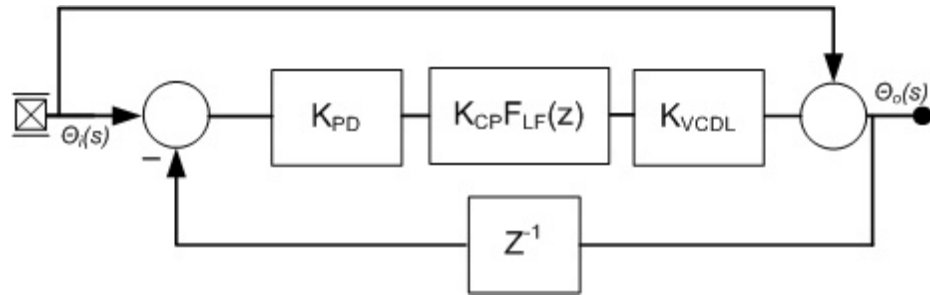


Figure 2.5: DLL Z-domain model.

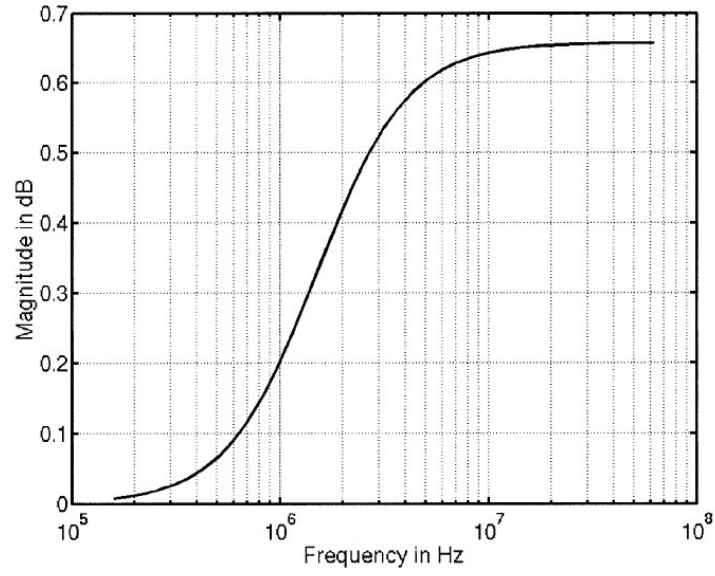


Figure 2.6: Example jitter transfer function for a type-I DLL [22].

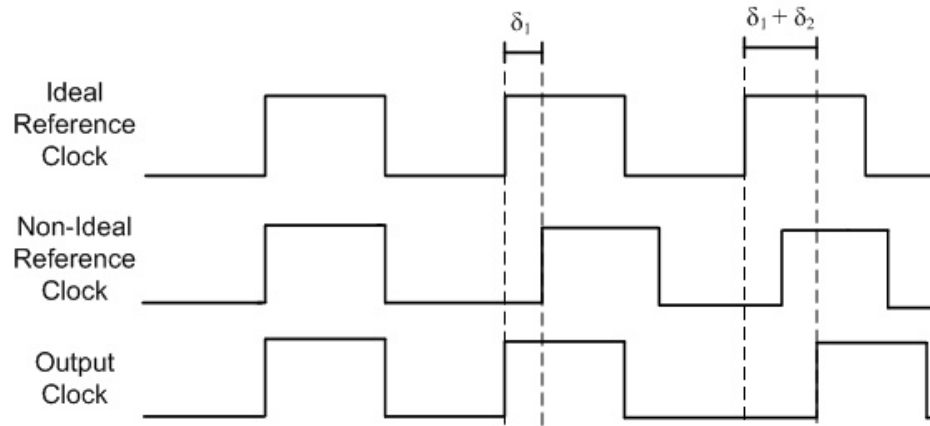


Figure 2.7: Jitter peaking timing diagram.

DLL in a high gain mode, and then putting it back to a low gain mode after its close to a lock condition.

Considering random jitter sources in the DLL, the devices in the VCDL play a major role in the total jitter. The VCDL will be discussed in more detail later in this chapter, but for now consider it to be a chain of N delay elements such as inverters. Each of these elements exhibits some random noise characteristics, which will translate into jitter delay variations from delay cell to delay cell. Assume that the VCDL has N delay elements each with an ideal delay of t_d . The first delay cell sees the reference clock, therefore its output timing error would be determined only by the random variations of the circuit element, or the clock would be delayed by $t_d + \delta t_{d1}$, where δt_{d1} is random noise associated with the first delay cell, which would be from substrate, thermal, and power supply sources. Mismatch would produce a static offset error. For the second delay cell, the total delay at its output would be $2t_d + \delta t_{d1} + \delta t_{d2}$. Continuing through the VCDL, the final output delay would be given as

$$T_{d.VCDL} = Nt_d + \sum_{n=1}^N (\delta t_{dn}). \quad (2.10)$$

Other sources of random noise timing errors in the DLL exist in the phase detector and charge pump. If the phase detector and charge pump noise contributions are referred back to the input, the sources are low pass filtered by the loop. This helps minimize the noise contributions. In a locked condition, the output of the phase detector produces up and down signals that effectively cancel each other out, thereby not allowing the charge pump to change the control voltage. Assume that some noise source causes a slight variation in the duration of one of the charging signals, thereby causing a slight change in the control voltage. This could be minimized by lowering the gain of the charge pump, but this would slow the tracking speed of the DLL. Random noise in the charge pump devices is directly

placed on the loop filter, creating a slight variation in the control voltage. This can be minimized by increasing the loop filter, but at the cost of silicon area and tracking speed.

The power supply noise can be associated to static variations (i.e. voltage drops due to loading) and dynamic variations (i.e. temporary changes due to environmental noise) [23]. The delay variations due to static power supply shifts can be minimized by using cascoded devices or regulating the power supply [23]. One option to help minimize the dynamic noise due to power supplies, or even common mode noise seen at the inputs, is to use differential configurations [23]. This may not be applicable depending on the application. Thermal noise and flicker noise of the devices can be minimized by adjusting the g_m of the devices used to minimize their input referred noise, but first a circuit configuration must be chosen before a noise analysis can be performed.

2.3 Types of DLLs

DLLs can be classified as one of two major types: a digital or an analog DLL. Digital DLLs are solely made from digital components, making them robust to process variations, require lower power supply voltages, scale easily with process, and can be synthesized. However, analog DLLs tend to have better substrate noise rejection, require less area and power, and have better phase resolution capabilities. The following section will discuss the two types of architectures, but with a heavier focus on the analog DLL.

2.3.1 Digital Delay Locked Loops

The most basic form of an all digital DLL is shown in Figure 2.8. This all digital DLL architecture was first presented by Efendovich [5].

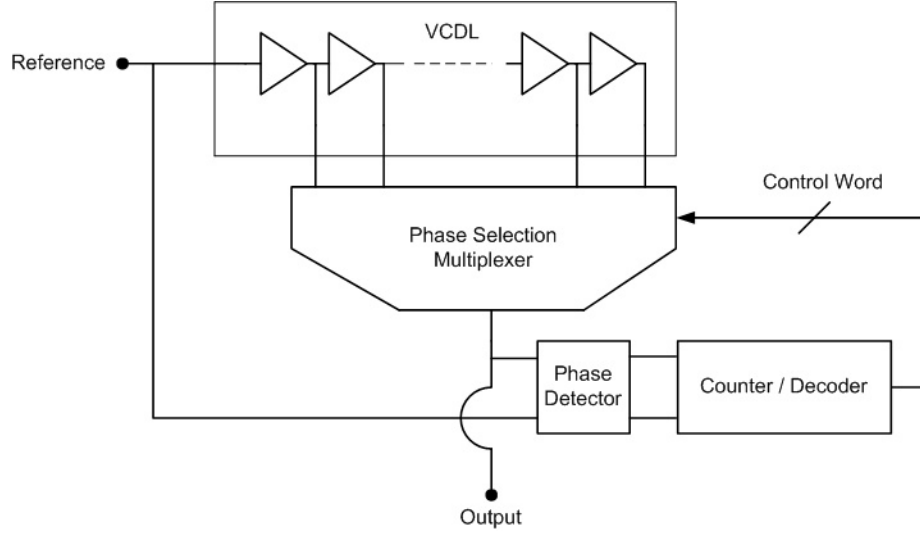


Figure 2.8: Typical all digital DLL block diagram.

The basic digital DLL architecture is similar to the analog DLL discussed at the beginning of Section 2.2, but in place of a loop filter a counter and decoder are used to change the VCDL delay. The digital DLL phase detector comparator will either increment or decrement the phase selection counter, which selects another tap in the VCDL, until the VCDL delay is equal to that of the reference period.

The digital DLL scales easily as process feature sizes decrease due to all of the components being of a digital nature. Consisting solely of digital components will increase the chances of a successful circuit on the first silicon pass compared to the analog DLL. This also means that the digital DLL works better with lower voltage supplies.

Unfortunately, the digital DLL delay step resolution is limited by the delay of two inverters (need 360° phase shift per stage). This means that the output would not necessarily be exactly in phase with the reference clock, which requires the phase detector to have some mechanism built-in, such as intentionally keeping a dead zone, that keeps the counter from flipping up and down when the DLL is close to lock. This was done by making the PD with

SR flip-flops [5] that will output zero for the up and down signal when the reference and output clocks are within some window.

There have been techniques such as phase blending [24] that been used to increase the timing resolution of the digital DLL, but even this technique still uses discrete timing steps. Also, the output of the digital DLL may not be from the last tap of the VCDL. This would mean that it is difficult to obtain multiple phases without increasing the number of stages as done by Chung [11].

2.3.2 Analog Delay Locked Loops

Where as digital DLLs exhibit some unknown static phase offset, analog DLLs are able to minimize this parameter. This is most valuable for applications where the timing requirements need to be exact, such as RAM clocking or DLL frequency multipliers. The simplest form of the DLL was shown in Figure 2.2. The basic operation of DLL was described earlier in the section, but the components of the DLL have not been discussed. In this section the phase detector, charge pump, and VCDL will be presented. The loop filter will be assumed to be a simple single capacitor to ground, but other types of loop filters can be found in the literature.

Phase Detector

As discussed earlier, the phase detector is needed to determine the relative phase position of the input reference to the output of the VCDL. In essence, it subtracts the VCDL output phase from the input phase, and outputs a value that is related to the difference.

One of the earlier phase detectors presented was a simple XOR gate as shown in Figure 2.9. It should be noted that in PLLs a multiplier based phase detector is used, however these are not traditionally seen in DLLs, and hence will not be discussed. The XOR PD inputs



Figure 2.9: XOR type phase detector.

are the input reference and VCDL output, and its output would be directly connected to the loop filter, which is a low pass filter.

The operation of the XOR gate will produce a '0' at the output if both inputs are the same, and will output a '1' if the inputs are different. Assume that the input reference and VCDL output exhibit equal duty cycles. If the two inputs were in perfect phase, PD_{OUT} would be '0', whereas if they were 180 degrees apart, PD_{OUT} would be '1'. This would correspond to average values of VSS and V_{DD} respectively. If the VCDL output phase differs between 0 - 180 degrees from the reference, PD_{OUT} will begin to pulse for a time that linearly corresponds to the phase difference, and the loop filter will produce an average value to the VCDL. When the two clocks are separated by 90 degrees, the high and low pulses are equal in duration; hence the average value is zero. This means that the DLL will attempt to lock when the reference clock and VCDL output are in quadrature.

Unfortunately, despite it being extremely simple, there are many disadvantages to the XOR PD. It is only useful for quadrature locking, which is normally not the case for DLLs. It is also sensitive to the duty cycle of the two inputs. Also, the transfer characteristic of the XOR PD repeats every 180 degrees, which means that the DLL could lock to 90 degree phase separation.

To overcome the issues involved with the XOR PD, Brown [25] introduced a more robust phase detector that could discern both phase and frequency, which was known as the phase-frequency detector (PFD). Later Gardner [18] applied this PFD with a charge

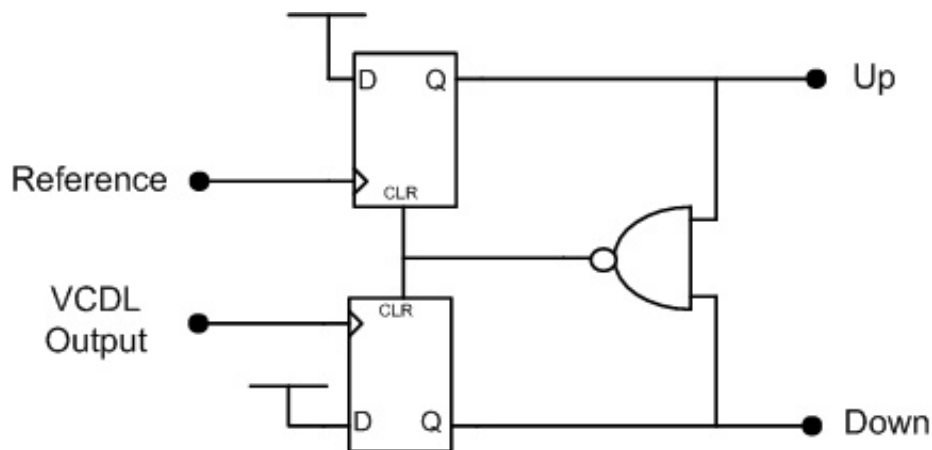


Figure 2.10: Common phase-frequency detector.

pump to introduce the first charge pump PLL, which is still applied to many of today's PLLs and DLLs. Figure 2.10 is a schematic of one of the most common forms of modern PFD.

Assume that both flip-flops' outputs are reset to '0'. On the first rising edge of either the reference or VCDL output will cause the corresponding output, Up or Down, to go high. This signal will stay high until the other input goes high, in which the NAND gate will reset the flip-flops. In a charge pump PLL/DLL it is a requirement that the transfer characteristic of the phase detector not exhibit a flat region, or dead zone, where the input and output clocks are close together. This would result in a phase offset error where the loop would be unable to correct the offset. Many times extra delay is added after the NAND gate as a means to eliminate the dead zone.

The basic PFD is fine for lower speed applications where the input clock's frequency is less than the setup, hold, and reset times of the D flip-flops. To combat this issue a dynamic phase detector was described by Hatoni [26]. This dynamic PD is shown in Figure 2.11. Using the principles of precharging, the dynamic PD has extremely fast transition times.

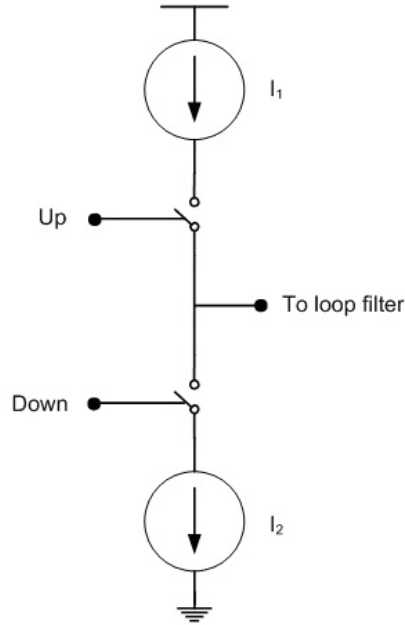


Figure 2.12: Simplified charge pump.

dead zone, and the two currents must cancel each other out as to not (dis)charge the loop filter. When the two switches are open, the loop filter holds its charge as the charge pump output is a high impedance. Since this node could potentially float in the high impedance condition, it is desirable to have the PFD refresh the loop filter every cycle.

There are various circuit implementation of this simple charge pump. The most straight forward approach is shown in Figure 2.13. The switches have been replaced by NMOS and PMOS devices, and the currents I_1 and I_2 are mirrored into the switches. The current mirrors can be cascoded, but at the cost of voltage headroom.

There are some short comings with the particular charge pump implementation. First, even if I_1 and I_2 are equal, mismatch in the devices between the two current mirrors could lead to unequal charging and discharging currents. In order for the DLL to lock, one of

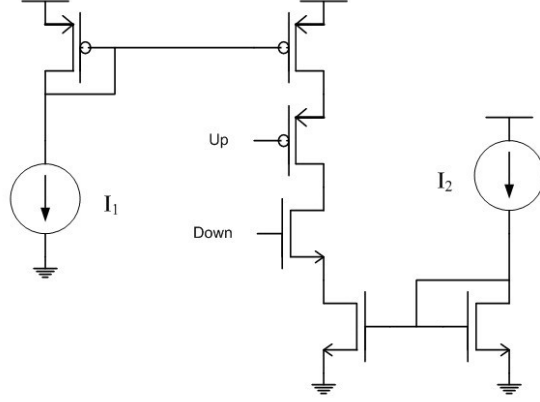


Figure 2.13: Basic charge pump schematic.

the PD control signals will have to be on for a longer period of time to compensate for the current mismatch. This will produce a static phase offset between the reference clock and VCDL output. This phase offset was described by Rhee [27] as

$$\phi_{offset} = 2\pi \frac{\Delta t_{on}}{T_{ref}} \frac{\Delta i}{I_{CP}}, \quad (2.11)$$

where Δt_{on} difference in pulse widths of the up and down signals, Δi is the difference between the charging and discharging currents, T_{ref} is the reference clock period, and I_{CP} is the ideal (dis)charging current. Also, the fact that the standard PFD outputs pulses that go from low to high requires that the up signal be inverted. Plus, the switching times of the PMOS and NMOS switches are different. Both of these issues attribute to the static phase offset.

Two other single-ended charge pumps that were presented by Rhee [27] are similar to that of Figure 2.13, but the switches were alternatively placed at the gates or the sources of the current mirrors. Rhee also presented that the option with the switches in the sources

dissipated less power and switched faster. To eliminate the need for the up signal inversion and switching mismatches, an all NMOS switch charge pump can be designed [27]; however this architecture requires more devices and still has a slow branch through the sourcing PMOS device.

Another issue of concern with the single-ended charge pump is low supply and substrate noise rejection. A differential topology can be implemented that will exhibit better immunity to supply and other common-mode noise sources. Unfortunately, these differential topologies require more devices, hence higher power, and also need two loop filters, which will require more chip area. In many cases a single-ended charge pump is chosen for these reasons.

Voltage Controlled Delay Line

The VCDL is a vital component as the DLL would be unable to delay the reference clock without it. The VCDL design will play a major role on the loop stability and the amount of jitter seen at the DLL output.

A VCDL consists of N equal adjustable delay blocks, where N does not need to be odd like the single-ended VCO. The VCDL can be either single-ended or differential as demonstrated in Figure 2.14. In most cases with a DLL, the goal is to delay the input reference clock by one period, or 360° . Therefore each of the delay cells will exhibit a delay of T_{ref}/N seconds, or a phase shift of $360^\circ/N$. The maximum and minimum delay of the VCDL is related to the number and the transfer characteristic of the delay cells.

One of the most basic single-ended delay cells is the current starved inverter as shown in Figure 2.15. This is similar to the basic delay element of the digital DLL, but by limiting the available current to the inverter via the control voltage, the rise and fall times can be

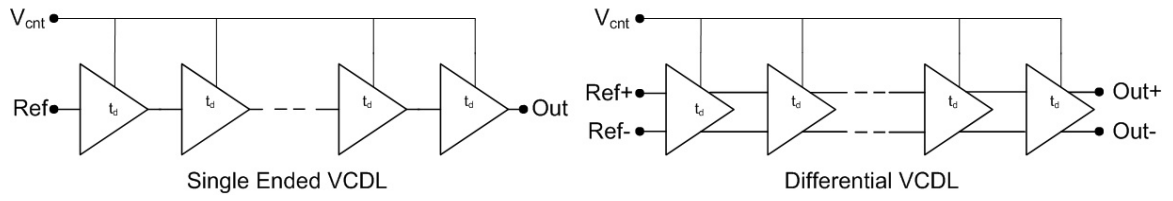


Figure 2.14: Block diagrams of single-ended and differential VCDLs.

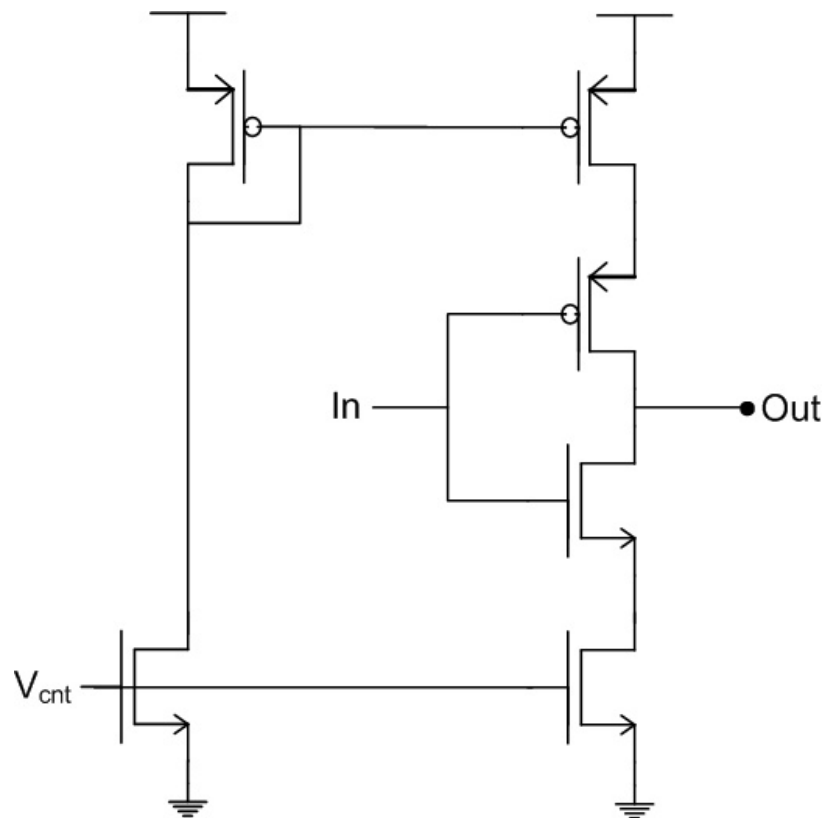


Figure 2.15: Current starved inverter delay cell.

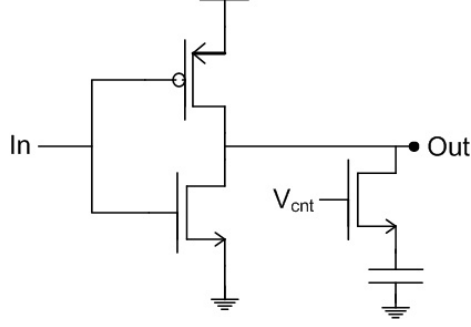


Figure 2.16: Inverter delay cell with an RC load.

controlled in an analog manner. Another common type of single-ended delay cell shown in Figure 2.16 is the inverter with an adjustable RC load. By adjusting the control voltage at the gate of the NMOS, the effective channel resistance can be changed, thereby adjusting the RC time constant. Many times the capacitor is replaced with a MOSCAP to save area. The current delay cells' delays are related to the available sourcing and sinking currents and the loads. This is discussed later in Chapter 4.

Despite the simplicity of the single-ended delay cells presented above, they are susceptible to supply and substrate noise. This could be reduced by cascading the current sources on the current starved inverter, but that will come at a cost of signal swing.

The differential delay cell is able to suppress the common-mode noise much better than its single-ended counterparts. There are two major types of differential delay cells discussed in the literature which are based on a source connected differential pair, as drawn in Figure 2.17. The delay of the cell is related to the small signal output resistance and load capacitance seen at the output nodes. The small signal resistance can be adjusted by changing the tail current, which is accomplished by changing the control voltage at the gate of the NMOS current source. This is discussed in more detail by Maneatis [6].

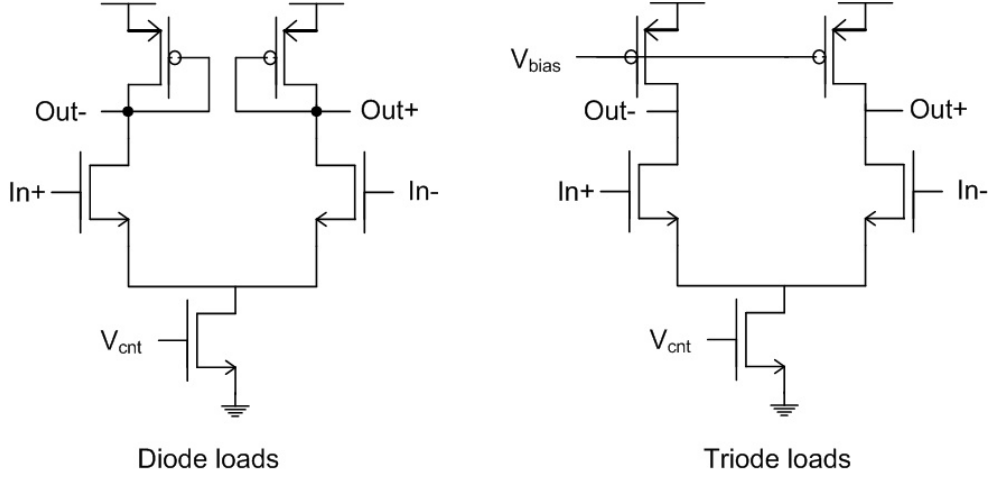


Figure 2.17: Differential delay cells with diode loads (left) and triode loads (right).

The diode connected PMOS load delay does not require any load biasing circuit as needed for the triode load delay cell, but unfortunately the PMOS diodes require more voltage headroom, thereby limiting the signal swing. Also, the common mode DC level will change as the available current changes. The triode load delay cell exhibits a much larger signal swing, but it is much more difficult to bias the loads such that they operate in triode across the tuning range. Maneatis [6] was able to overcome this triode biasing issue by using a replica bias cell in conjunction with a symmetric load configuration.

2.4 Summary

The operation of the DLL is similar to that of the PLL, but there are key differences that must be considered when designing the systems. Digital and analog DLLs both have advantages, but their limitations will ultimately determine which one must be used in the desired application. Also, it is important to consider the advantages and disadvantages of the different types of phase detector, charge pump, and VCDL architectures in conjunction with the target application design requirements.

Chapter 3

CMOS Scaling and Extreme Environment Operation and their Implications on DLLs

3.1 Introduction

From the birth of the integrated circuit, IC development has focused on reducing the feature size of transistors, or increasing the density of IC's, which in turn would make the products cheaper and perform better. A figure of merit known to most IC designers is Moore's Law, which quantifies the rate of semiconductor process scaling. In a 1965 publication [28], Gordon Moore proposed that IC density would approximately double every year for the next decade, but he was not able to prove his theory for another decade [29], where he stated that if IC density would continue to increase at an exponential rate for decades to come, however at a slower rate than originally predicted. As demonstrated in [30] twenty two years after Moore's quantification, the exponential growth rate had stayed consistent with that of Moore's predication. The ramifications of this process scaling can be seen

as positive in modern times, but unfortunately for the IC designers there comes a cost of increased complexity in their designs. This is especially true for the analog designer, but as processes approach the sub-50 nm feature size, digital designers must even begin to consider the non-idealities associated with these small feature sizes.

To further complicate IC design, many applications require that these circuits operate in extreme environments, which means that the temperatures can exceed the mil-spec standard. CMOS devices, both active and passive, exhibit some form of temperature characteristics that causes their device parameters to fluctuate as the temperature changes. These parameter fluctuations can easily cause an IC, be it digital or analog, to be unable to operate at the designed specification, or potentially stop working all together. As the ICs would be operated in a self-sufficient system, for example a satellite, these systems will most likely be powered via some battery, or other regenerative source, in conjunction with a regulator. Both of the components would also have some temperature coefficient associated with them, therefore it would be expected that the supply voltage would shift.

If a DLL would be needed in such an extreme environment, it would be important that these issues be considered during its design process. This chapter will highlight what the problems are associated with process scaling, voltage variations, and temperature on DLL performance, and methods which can be used to suppress these issues.

In Section 3.2, the trends associated with PVT effects will on DLLs discussed, with focus on the issues associated with DLL sub-circuits. This will relate issue with scaling trends for both the system and circuit level of DLLs. In Section 3.3, a review of the literature will be presented regarding calibration techniques applied to DLLs and some applicable PLLs that help minimize the PVT induced errors. Finally, Section 3.4 will conclude this discussion.

3.2 CMOS Process, Voltage, and Temperature Variations Associated with the DLL

PVT variations can affect all types of CMOS ICs, including DLLs. The following section will provide some insight into these effects. Each of the non-idealities will be discussed separately, and then a simple delay cell simulation will be provided to demonstrate the combined effects.

3.2.1 Process Variations

Matching devices has always been a concern in analog IC design. Be it a current mirror, input pair, or diodes in a voltage reference, it is important that these paired devices need to have the same electrical characteristics to ensure proper operation. As process feature sizes have continued to decrease, device mismatch has become even a larger problem. In most analog circuits, like opamps, an easy way to minimize this mismatch is to use longer gate lengths, effectively making the mismatch ratio between the matched devices smaller. However for high speed applications, like DLLs and PLLs, it is a necessity to keep the gate lengths to a minimum as to minimize the effective node capacitance, thereby making mismatch between devices much more pronounced.

Standard techniques for minimizing process mismatch include techniques like common centroid layout [16]. This technique can be applied to circuits that can be placed in close proximity to each other, but in larger systems like DLLs, the sub-circuits can not usually be placed close together. The delay cells ideally would exhibit the same delay, however with process mismatch, their delay will differ. Plus most of the structures of the delay cells prohibit them from being laid out using common centroid techniques.

Consider the current starved inverter of Figure 2.15, including the voltage-to-current converter. Many delay cells used two cascaded current starved inverters to form one delay

cell (a simple inverter can be used in the place of the second current starved inverter), and the cascade however many delay cells needed to form the VCDL. This could easily cover hundreds of micrometers of silicon for a VCDL with many tap points. Due to processing uncertainty, it can be easily visualized that the drawn device dimensions and implant dopings will vary between the taps. This localized mismatch will lead to both mismatch in physical size and threshold voltages, therefore the delay cells will exhibit different delays.

This concept also can be expanded to wafer to wafer process mismatch. Using corner models (fast, typical, and slow) can demonstrate how much deviation could occur across DLL's from different lots.

Consider the simple current starved inverter based delay cell of Figure 3.1. Applying Monte Carlo simulations on this circuit in Spectre simulator with models that support statistical simulations can provide useful information regarding delay cell delay deviation. The Monte Carlo results from a 150-nm partially depleted silicon-on-insulator process yield the scatter plot of Figure 3.2. The standard deviation for this delay cell is less than 1 ps, but the maximum deviation is 4.6 ps.

For a VCDL with multiple taps using this delay cell, it would take about 32 delay cells for a 1 GHz input clock. It may not always be desirable to have 32 taps such as applications where only quadrature signals are needed. For this, one effective delay cell would now consist of 8 cascaded of the delay cells. This could mean that the effective standard deviation between delays cells is now 6.88 ps (0.688% of the input clock), or a max range of 36.8 ps (3.68% of the clock). This may or may not be acceptable for the application at hand.

The previous simulation was performed with the control voltage set in the middle of the tuning range. The tuning range must be large enough to be able to correct for the process induced timing errors. For this simple delay cell, the tuning range is approximately 6 ps,

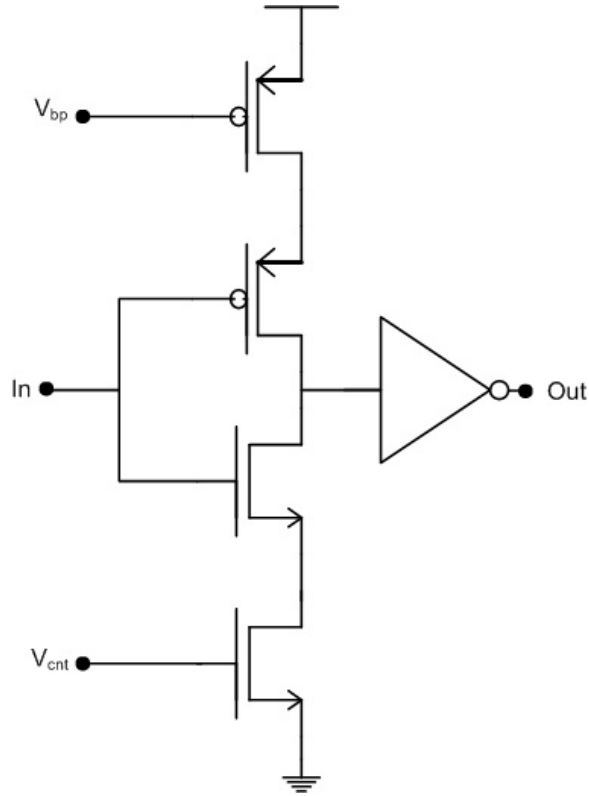


Figure 3.1: Simple current starved inverter delay cell.

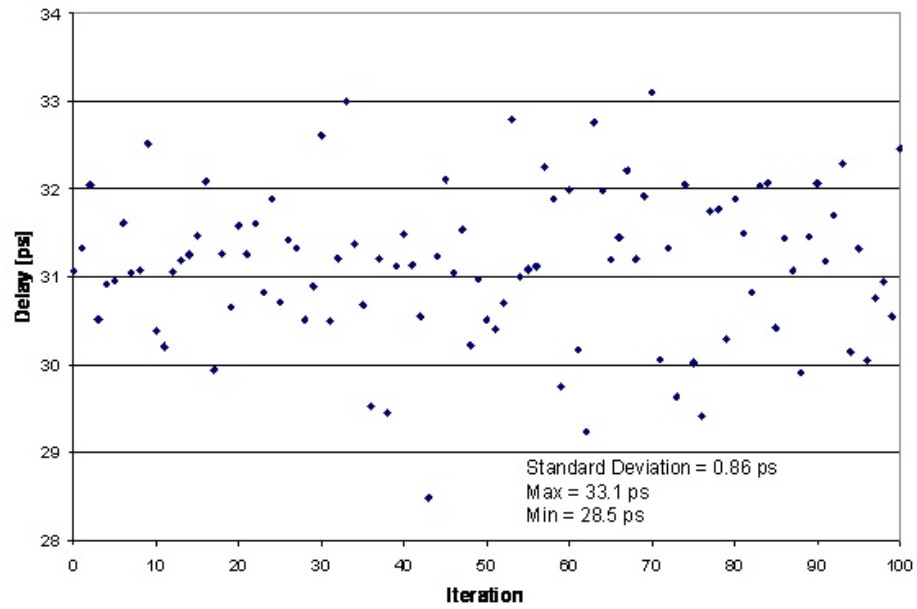


Figure 3.2: Monte Carlo results for the current starved inverter delay cell of Figure 3.1.

or 3 ps from the center tuning point. In this case the tuning range is more than sufficient to tune out the timing errors, but this may not always be the case as must be considered.

The phase detector, charge pump, and loop filter are also subject to process mismatch. Consider the common PFD of Figure 2.10. Any mismatch in the D flip-flops, or input-to-output delays of the logic can lead to static errors in the up and down signal turn on and off times. This will lead to a static offset in the reference and DLL output clocks. The charge pump current mismatches have already been discussed in Chapter 2. Variations in the loop filter capacitor will lead to a change in the loop dynamics as described in Chapter 2. By using the process design rules, the capacitor size can be chosen such that the deviation in the range of loop bandwidth is acceptable.

3.2.2 Supply Voltage Variations

All CMOS processes have an ideal value at which they operate (e.g. 1.8 V for a 150 nm process). Unfortunately power supplies (regulators and batteries too) will not always produce the desired value for numerous reasons, including temperature effects, finite output impedance, and regulator dropout. When this DC value (AC noise is a different consideration) varies from the ideal, circuits will operate differently due to a change in the bias conditions.

Once again, consider the current starved inverter of Figure 3.1. The voltage-to-current converter's bias points are dependent on the control voltage and V_{DD} . As V_{DD} decreases, the overdrive voltages decrease, thereby reducing the available current in the current starved inverter. The delay of the current starved inverter is based on the relationship between the available current and load capacitance; therefore the delay will increase as the supply decreases. Of course this change is more pronounced when the upper and lower devices are not saturated.

The same current starved cell described by Figure 3.1 is simulated at different supply voltages. The results are shown in Figure 3.3. As the supply voltage decrease from 1.8 V to 1.4 V, the delay tuning range decreased by 45%. Depending on the required VCDL tuning this could cause the DLL to be unable to lock.

The phase detector will still operate properly under reduced supply voltage (assuming it is sufficiently larger than the threshold voltages), but its highest detection speed will be reduced. The charge pump will be limited by the point at which all devices stay in saturation. For the simple charge pump of Figure 2.13, this will be

$$V_{DD,min} = V_{TH} + V_{DSsat}, \quad (3.1)$$

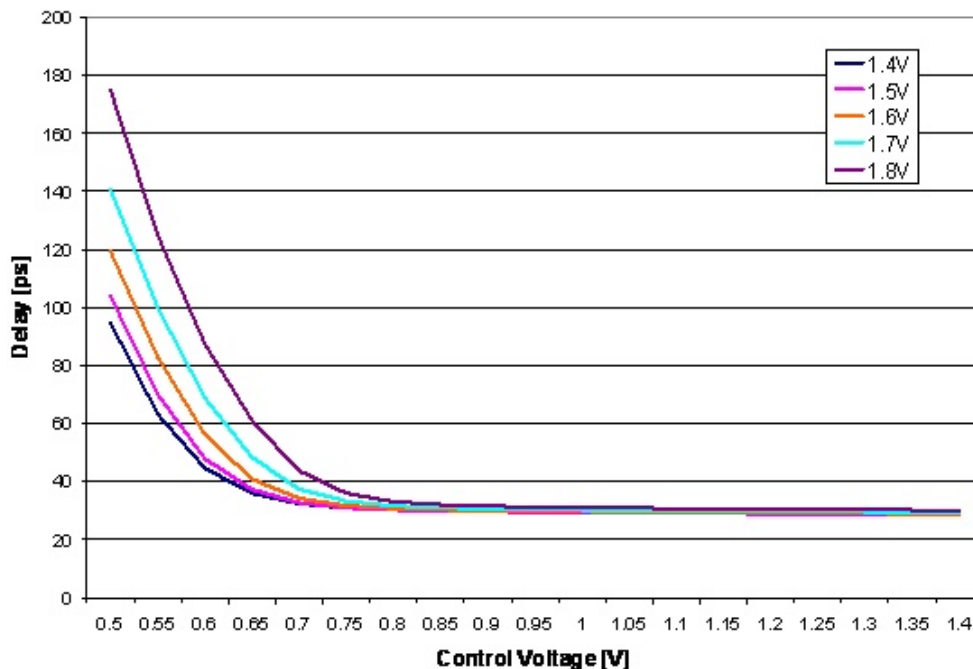


Figure 3.3: Simulation results at various V_{DD} levels for the current starved inverter delay cell of Figure 3.1.

where V_{TH} is the larger threshold voltage of either the NMOS or PMOS, and V_{DSsat} is the saturation voltage of the current sources. As the devices are cascoded, the minimum supply voltage will increase. Also, this directly correlates to highest level at which the control voltage can be charged.

3.2.3 Temperature Effects

In extreme environment applications, the temperature will extend beyond the military specification (-55°C – 125°). Temperature effects of the CMOS devices are related to the mobility and threshold shifts. The mobility temperature coefficient is given as

$$V_{TH}(T) = V_{TH}(T_0)[1 + TCV_{TH}(T - T_0)], \quad (3.2)$$

where $V_{TH}(T_0)$ is the threshold at an initial temperature T_0 , and TCV_{TH} is the threshold temperature coefficient [16]. The device mobility temperature characteristic is given as

$$\mu(T) = \mu(T_0) \left(\frac{T}{T_0} \right)^{-1.5}, \quad (3.3)$$

where $\mu(T_0)$ is the initial temperature mobility [16].

Depending on whether the device is be operated in a digital or analog manner will decide on which parameter will be the dominate temperature error source. In general digital circuits' temperature characteristics are dominated by shifts in mobility [16]. Analog applications will require a circuit dependent analysis.

The temperature effects of the current starved inverter of Figure 2.15 will now be considered. In general the current starved inverter acts in a digital manner which means that the temperature coefficient will be dominated by the mobility (decreases with temperature). This means that the available current decreases as temperature increases. However the voltage-to-current converter is an analog circuit where the PMOS bias voltage temperature coefficient is given as

$$TC(V_{bp}) = \left(\frac{1}{V_{bp}} \right) \left(\frac{\partial V_{bp}}{\partial T} \right) = -\frac{1}{V_{bp}} \left(\frac{\partial V_{THP}}{\partial T} + \sqrt{\frac{B_n}{B_p}} \frac{\partial V_{THN}}{\partial T} \right). \quad (3.4)$$

In general this value will exhibit a positive temperature coefficient, which means that as temperature increases, V_{bp} increases, thereby decreasing the available overdrive to the current starved inverter. The cumulative results are that the delay will increase as temperature increases. Simulating the delay cell of Figure 3.1 across temperature demonstrates this via the plot of Figure 3.4. Also, it can be seen that the delay range decreases as temperature increases. The models were only validated over the mil-spec range.

The phase detector's speed will decrease as temperature increases, where the charge pump has some other temperature dependent concerns. As temperature increases, sub-threshold and gate current increases [31]. However, gate oxide leakage tends to be an issue when oxides are thinner than 20 Å [31], so this will not be directly considered. Sub-threshold leakage will exist in both the sourcing (PMOS) and sinking (NMOS) legs of the charge pump, so some of it will be canceled, but if one of the other exhibits a higher leakage current, net charge will be added or removed from the loop filter. The loop filter is updated every clock cycle, but depending on the rate of this (dis)charging due to leakage, the control voltage can vary sufficiently such that the VCDL can change during each cycle, which is directly related to deterministic jitter. This must be considered for high temperature operation.

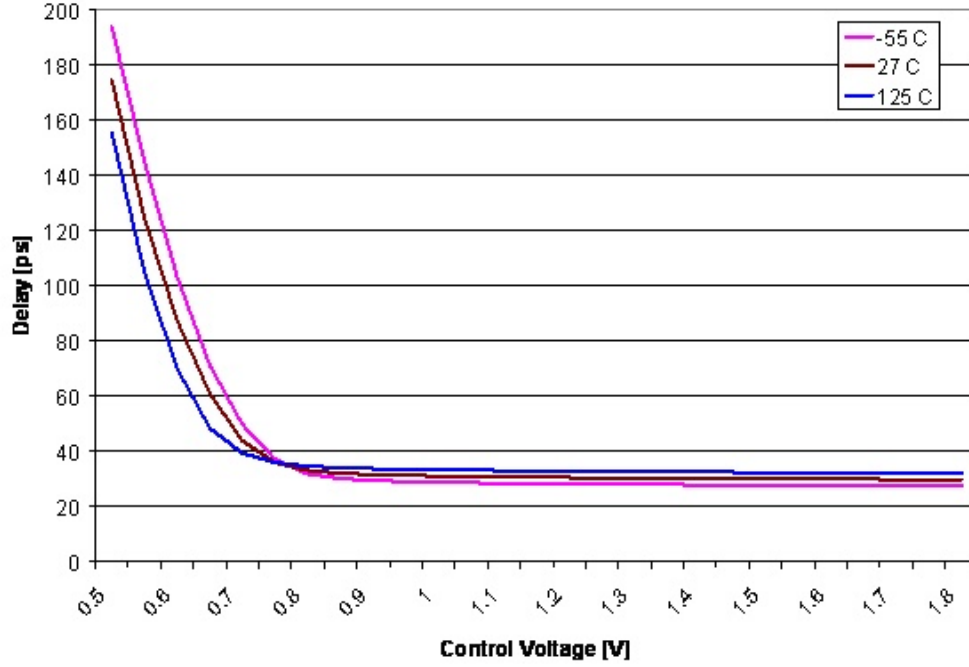


Figure 3.4: Simulation results at different temperatures for the current starved inverter delay cell of Figure 3.1.

3.2.4 Combined Effects

Now the combined effects of PVT variations will be considered. Simulating the delay cell of Figure 3.1 across various model corners, temperature, and supply voltages, the plot of Figure 3.5 was produced. It is apparent that the tuning range of the delay cell varies largely across PVT variations. Not surprisingly, the largest tuning range occurs for the 1.8 V supply for fast corner models at -55°C , where the smallest occurs for the 1.4 V supply for slow corner models at 125°C . Using the room temperature simulation at a V_{DD} of 1.8 V using typical model simulation as a baseline, the largest tuning is 1.3 times larger, and the smallest is 7.8 times smaller. If the DLL is to operate at high temperature with a reduce supply rail, the DLL may not be able to operate properly across the desired tuning range, unless the DLL is only specified to work at the smallest tuning range.

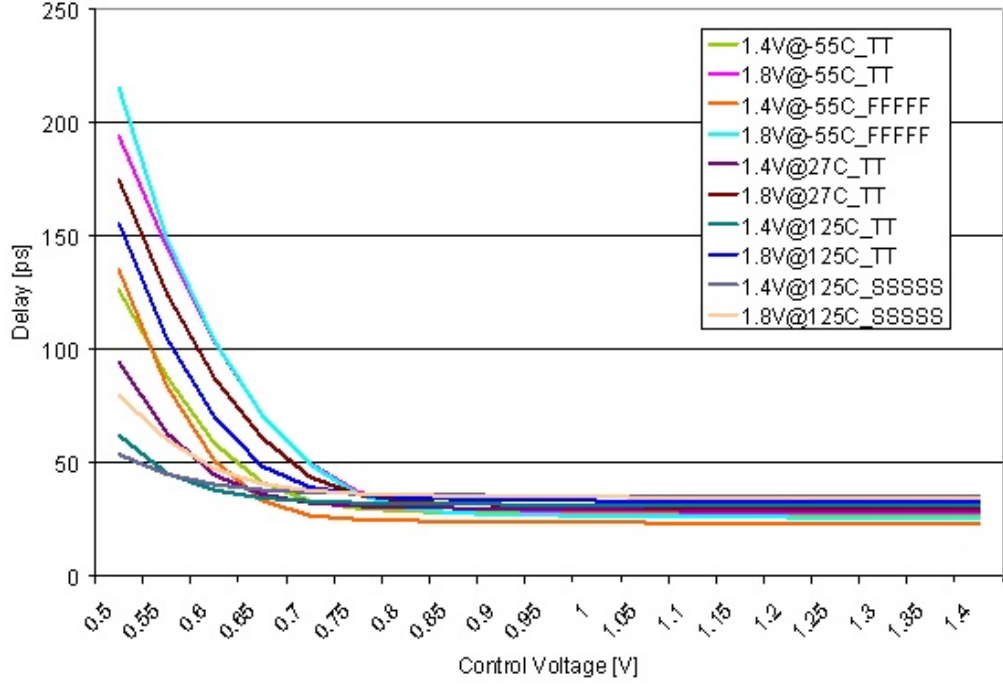


Figure 3.5: Simulation results at for various PVT shifts for the current starved inverter delay cell of Figure 3.1.

3.3 DLL Calibration Techniques

With the PVT issues discussed in the last section, it is obvious that many things must be considered for a DLL to operate in extreme environments. The most important design concern for an extreme environment DLL will be the tuning range of the VCDL over the entire PVT range.

It is possible to minimize each of the PVT effects separately in the DLL by some type of compensation circuit (such as many analog techniques); however this would require many systems to work at the same time to be useful for extreme environments. This section will serve as a review of the literature involving calibration techniques to minimize DLL PVT effects.

Process error can be classified as either global (process mismatch from die-to-die) or localized (process mismatch between devices) PVT errors, but only the global process error will be reviewed here. This literature review will be presented in chronological order.

One important note is that of the presented DLL calibration prior art, many of them will mention the techniques useful for helping with PVT, but none provide any data confirming the viability of the calibration technique over temperature and voltage, let alone extreme temperatures. This is probably because the DLL is inherently good at suppressing PVT via the negative feedback action, and is many times used for compensating other circuits [8,32]. However, as shown from the previous discussion the VCDL tuning range may become small enough across temperature such that will be unable to cancel out the effects of PVT.

3.3.1 Global (Coarse) PVT Calibration Prior Art

This subsection will present relevant DLL, as well as applicable PLL, prior art related to digital coarse calibration techniques for minimizing PVT.

One of the earliest described PLL digital calibration techniques was presented in 1996 [33]. This technique uses an auto-lock circuit that detects if the PLL is in lock by using a lock detector (like a phase detector) and if the control voltage is close to a limit in the tuning range of the oscillator. If the conditions show that the PLL may not lock, a selector is incremented, and another oscillator is selected from a group of three oscillators that have been designed to cover multiple ranges. This is shown in Figure 3.6 [33]. The major downfall of the technique is that it requires multiple oscillators, hence requires more silicon area. Also, the designer must be concerned with how to power off and on the unused oscillators, or how to keep power and supply and substrate noise low if they are to be continually operating.

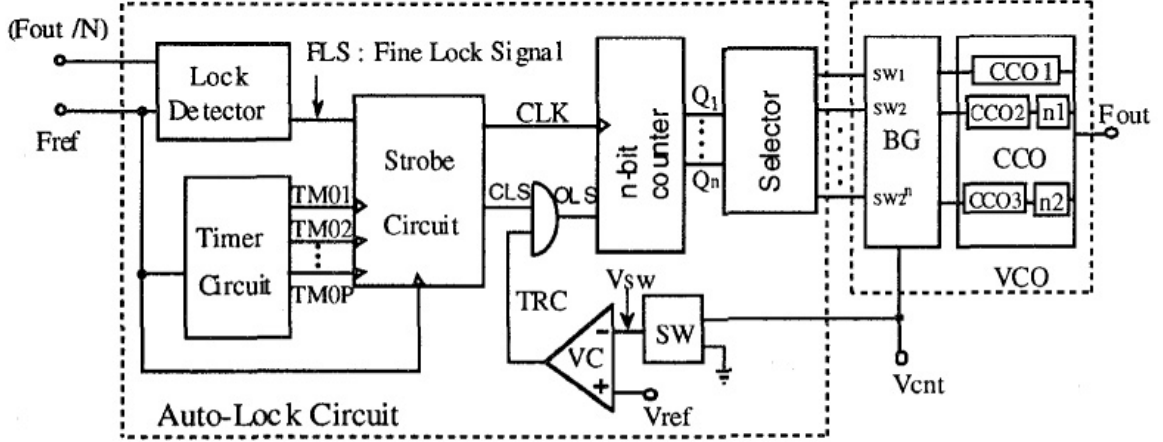


Figure 3.6: Calibrating PLL via an auto-lock circuit [33].

The second PLL coarse calibration technique involves using a digital temperature compensation sub-system to cancel the temperature effects on a PLL [34]. The digital compensation system consists of a temperature sensor, ADC, EEPROM, and a DAC. The ADC converts the temperature sensor voltage level to a digital code, which is used as the address of the EEPROM. The EEPROM has been programmed such that stored word consists of a digital code that will be sent to a DAC, which then produces an analog compensation voltage for the VCO. This compensation voltage is summed to the control voltage. This is shown in Figure 3.7.

There are many problems with this approach. First, it requires many analog components (ADC and DAC) in the subsystem which would make extreme environment operation difficult. Also, the PLL and compensation system must be characterized so that EEPROM may be programmed with the proper code. This technique would require a substantial amount of silicon due to the subsystem high resolution data converters.

The first self-calibrating DLL was presented by Hatakeyama *et al.* [35]. Their self-calibrating DLL is demonstrated in Figure 3.8 [35]. This digital DLL uses a simple digital

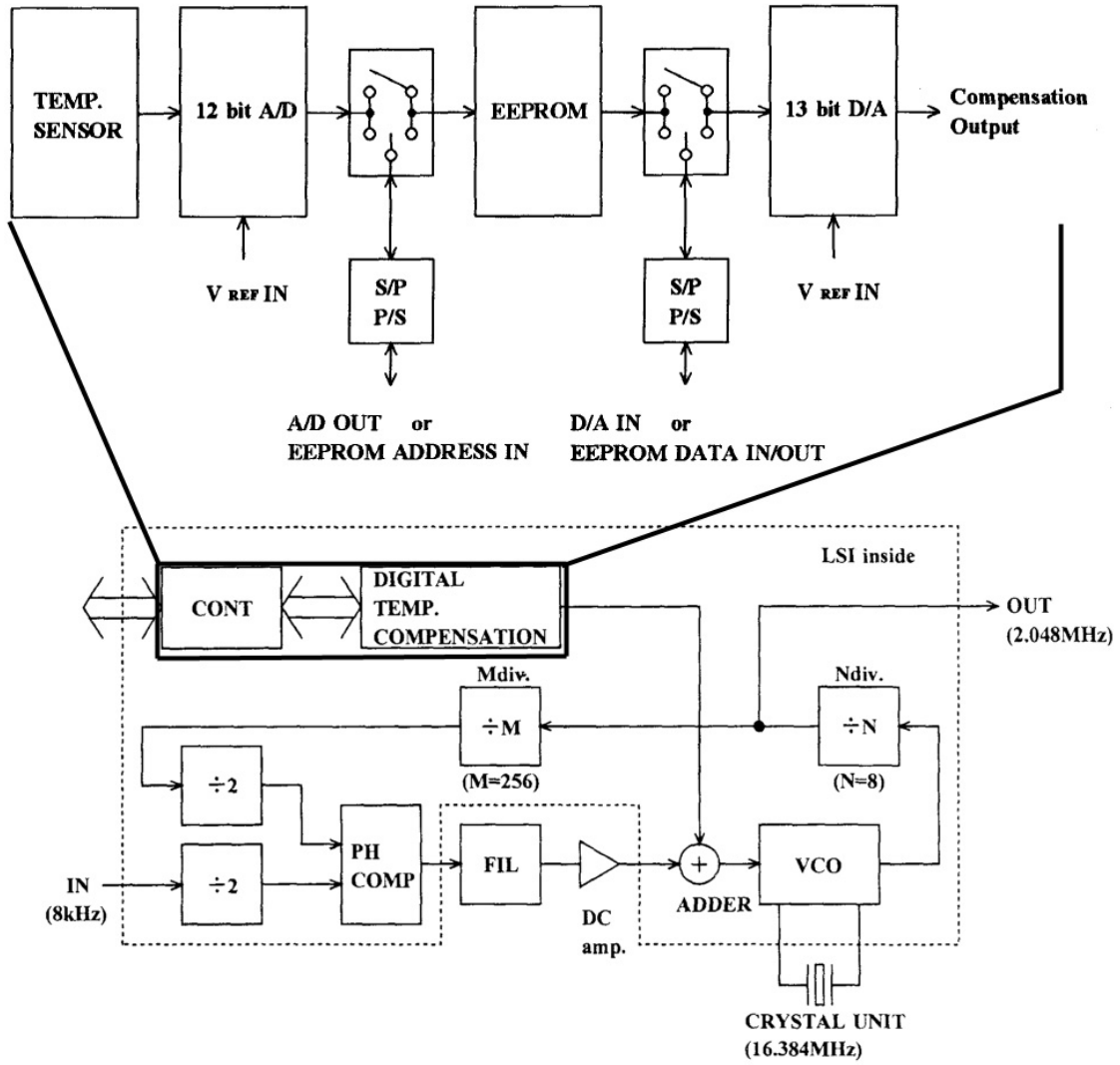


Figure 3.7: Calibrating PLL via a digital temperature compensation system [34].

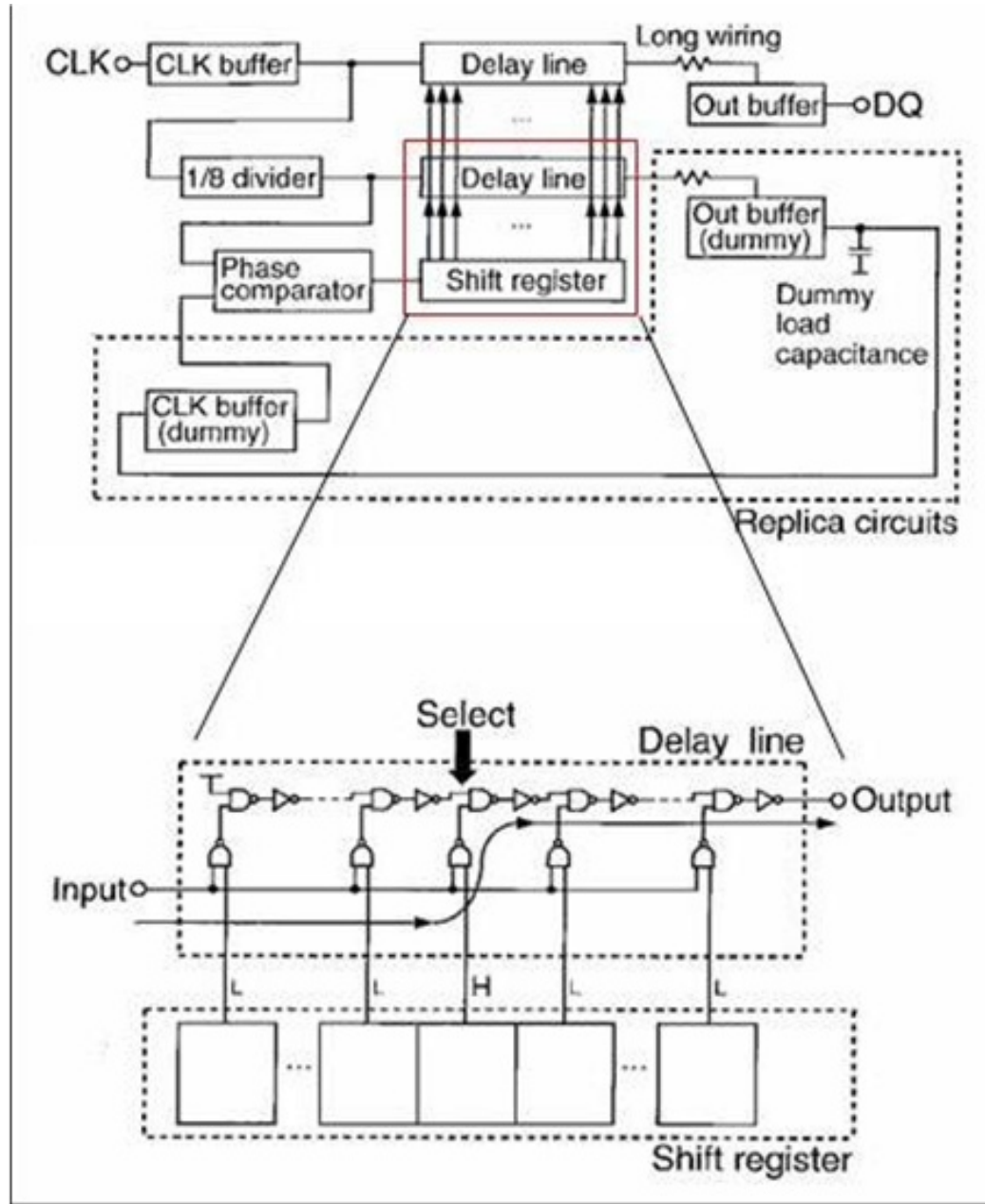


Figure 3.8: Self-calibrating digital DLL using an all digital calibration technique [35].

calibration technique to properly select the appropriate VCDL delay. The phase detectors up and down signals are used to increment/decrement a shift register, which is used to the proper delay value. It should be noted that this technique is not directly meant as a PVT calibration technique, but the concept is still applicable.

The all digital nature of this DLL architecture makes it suitable for extreme environments, however no temperature data was provided. The downfall of the approach is that there the DLL is digital and does not exhibit multi phases, so this limits its applications. Also, the DLL does not directly prevent false lock conditions, but by the general architecture, the chance of false lock is low.

The next calibrating PLL uses a different approach from the previous PLL work. Wilson and Moon *et al.* [36] decided to make the VCO exhibit discrete tuning steps as a means to make the PLL more robust to PVT. This digitally tunable VCO architecture is described in Figure 3.9 [36]. By setting a control word to the voltage-to-current converter and current multiplier, the current that biases the current controlled oscillator can be selected in discrete steps. The steps exhibit minimal overlap as to increase the tuning range.

The actual calibration system uses a frequency detector, digital accumulator, and a state machine. This is shown in Figure 3.10 [36]. The frequency detector is designed such that its resolution will increment or decrement an accumulator, thereby digitally adjusting the VCO until it is in the proper frequency range.

No PVT data was reported in [36], but this technique lends itself to PVT error suppression. This technique could easily be adapted to work with DLLs.

The next DLL article is not directly a calibration technique, but the work deserves mention due to its false lock method and temperature data. The temperature compensated VCO uses a DLL to compensate its VCO. The VCO is a replica of the DLL's VCDL

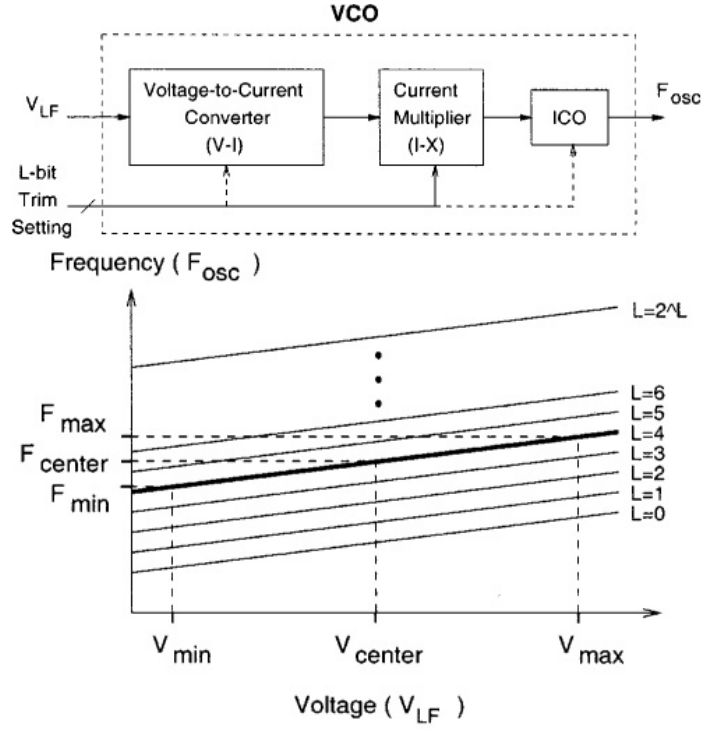


Figure 3.9: Digitally tunable VCO with transfer characteristics [36].

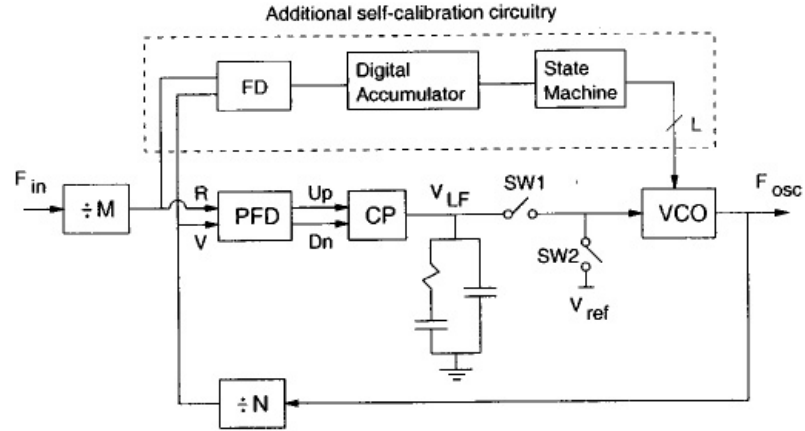


Figure 3.10: PLL with coarse calibration using a VCO with digital tuning [36].

architecture, but its outputs have been tied back to its inputs. Since they are replicas they will exhibit the same temperature characteristics, therefore as long as the VCDL tuning range is valid for a given temperature (i.e. can lock), the VCO can be coarsely adjusted. This architecture is shown in Figure 3.11 [8]. The temperature results demonstrated that the temperature compensated VCO varied 1.8% over a 0°C - 85°C range.

Foley and Flynn [8] reported a simple digital technique to prevent false lock in DLLs. The circuit is shown in Figure 3.12. The main problem with this approach is that this circuit is particular to a DLL with nine phases, and would have to be redesigned to work with a DLL with a different number of phases.

The next digital calibration scheme for DLLs was presented by Chang *et al.* [37]. This DLL is analog without multiple phases as shown in Figure 3.13. First the loop filter is charged up until a set point, and then the outputs of the VCDL are monitored with edge detector circuit of Figure 3.14. The edge detection circuit determines which phase of the

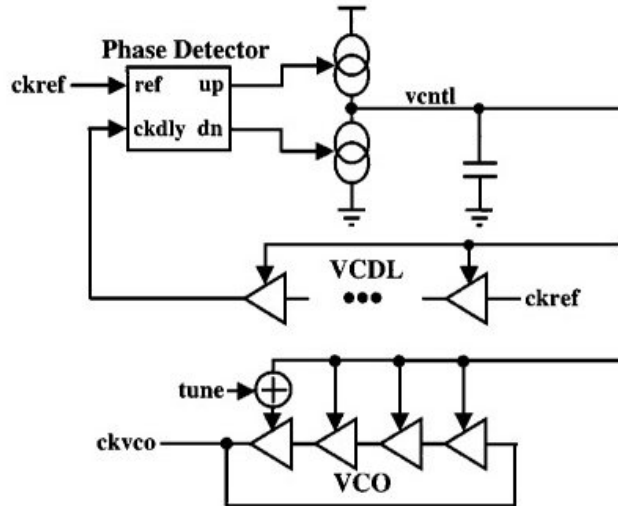


Figure 3.11: Temperature compensated VCO using a DLL [8].

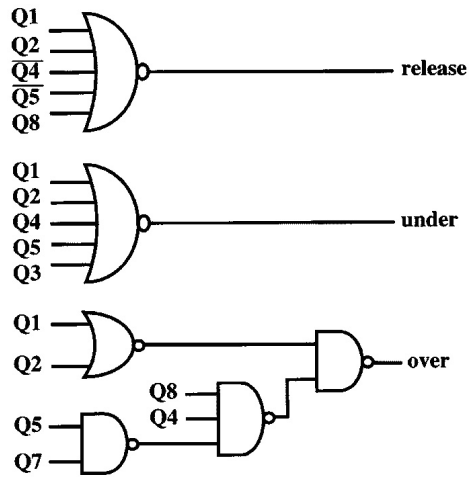


Figure 3.12: False lock prevention circuit for DLL [8].

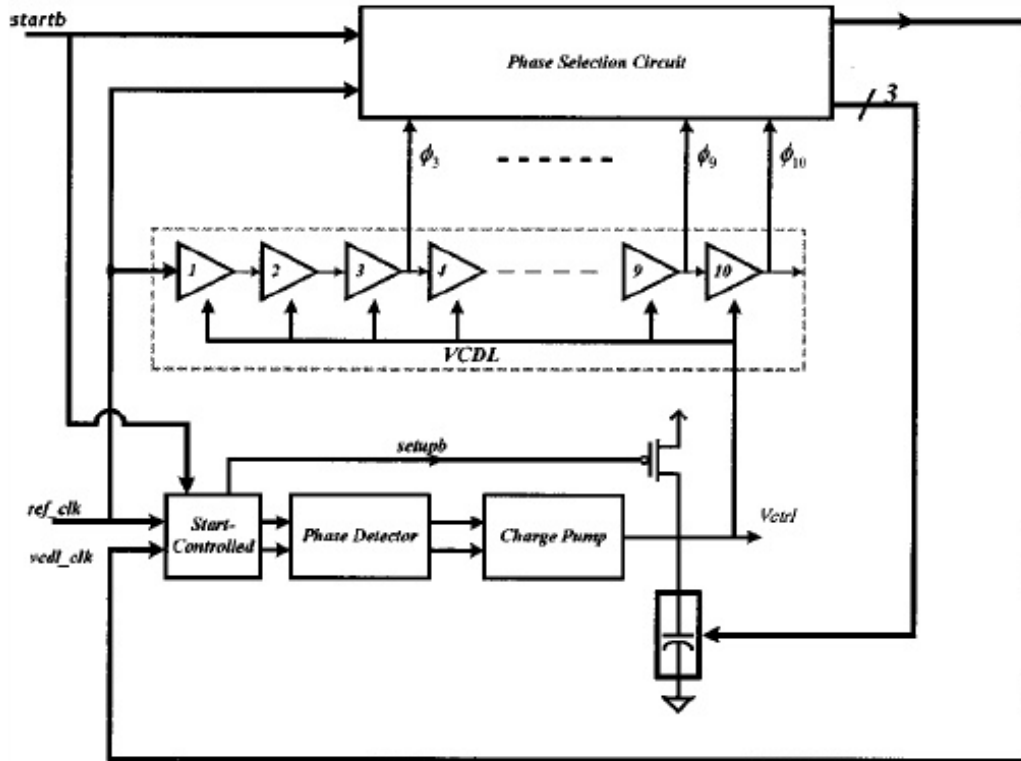


Figure 3.13: Self-calibrating DLL [37].

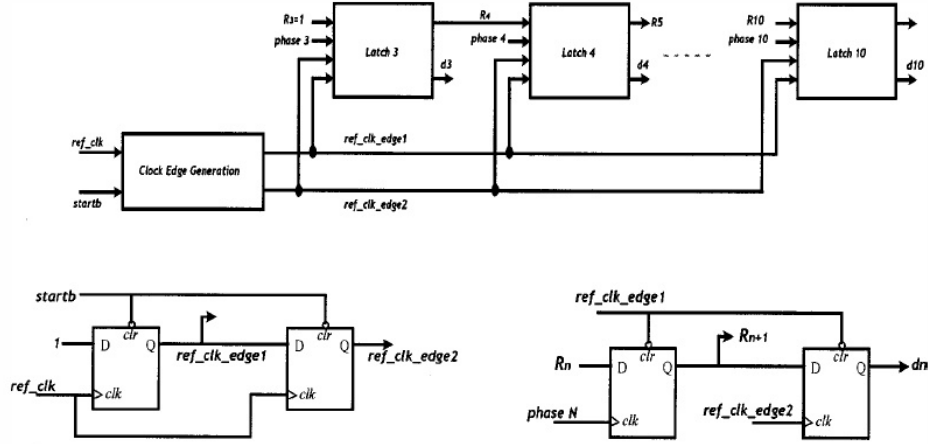


Figure 3.14: Edge detection circuit [37].

VCDL is the closest to one period delay. This is accomplished by using a clock generator to create two pulses that occur at two concurrent reference clock edges as a means to give an estimation of one clock period. The first pulse starts a chain of D flip-flops that will go high as the successive VCDL phase clocks it high. The second pulse that signifies that one clock period has occurred, and to use the point at which the D flip-flop chain outputs transitions from a '1' to a '0'. This point is used to select the VCDL's output since its delay is approximately one clock period. Then the loops analog control finishes the locking process.

The technique prevents false lock due to the edge detection circuit requiring that the output of the VCDL be approximately one clock period. Unfortunately since this analog DLL does not have multiple phase outputs, it is limited in its applications. Also, no PVT data was reported.

The only other digital calibration technique reported for DLLs was presented by Chung *et al.* [11]. This paper presents a digital DLL with multiphase outputs as presented in

Figure 3.15. The calibration circuit consists of a simple TDC with a digital controller. The TDC consists of register delay units (RDU), whose output are registered by in reference clock as shown in Figure 3.16. The output of the TDC relates to how long the reference clock period is, and then that code is used to place a digital VCDL in its proper range.

No temperature data was reported for this, even though the authors mention this technique being good for PVT error suppression. The TDC consists of digital components, so the validity of the TDC may be questionable in extreme environments. This DLL is a digital type, which means that it may not be applicable in some applications.

The final coarse calibration technique to be mentioned was a frequency synthesizing PLL presented by Aktas [38]. This approach is similar to that of Wilson [36], but the VCO is a LC oscillator as opposed to a ring oscillator. The PLL system diagram is shown in Figure 3.17. The digitally tunable LC-VCO, Figure 3.17, uses a capacitor array that can be digitally switched in and out to produce a transfer characteristic similar to that of Figure 3.9.

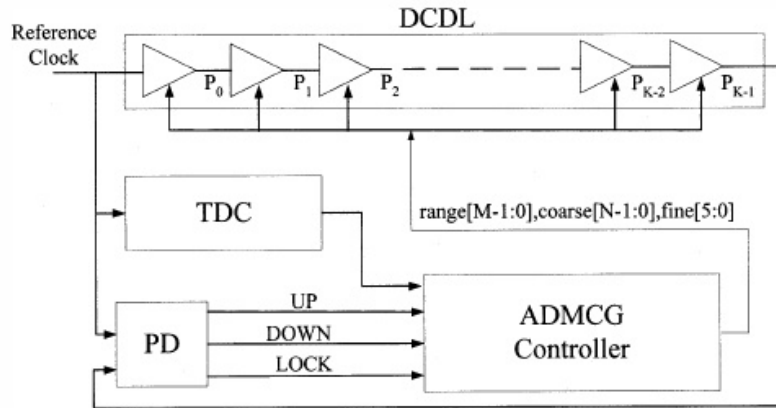


Figure 3.15: Self-calibrating DLL using a TDC based approach [11].



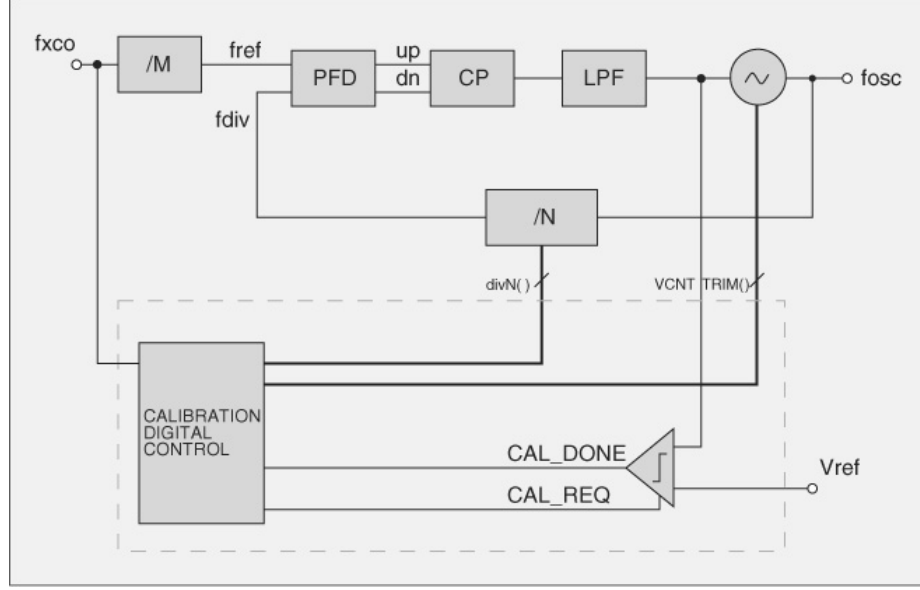


Figure 3.18: Digitally controllable LC-VCO [38].

The proposed calibration technique lends itself well to extreme environments. This digitally tunable capacitor array could be applied to VCDLs with some modification to would with a inverter based delay cells.

3.4 Summary

When designing DLLs for extreme environment or applications with strict timing issues, PVT variations, both global and local must be considered. One of the most common ways of dealing with these issues are calibration schemes.

Many coarse calibration schemes have been presented in the literature, however two major issues remain that need addressing:

1. None of the digital coarse calibration techniques presented any temperature (especially not extreme environment) or supply voltage measurement data demonstrating their PVT capabilities, and
2. none of the coarse calibration schemes worked for an analog multiphase DLL.

One other concern is that none of the discussed systems have any system monitor that watches for an event that would cause the DLL to lose lock. This would be necessary for robust extreme environment operation.

Overall, this review of the literature demonstrates that there is a need for new calibration techniques for minimizing the errors associated with PVT errors, especially for extreme environments. Also, PVT measurement data should be provided demonstrating the viability of the calibration schemes for extreme environment applications.

Chapter 4

The Self-Calibrating DLL

There are two main issues that were considered in designing the DLL. The first issue was how to ensure that the DLL will lock in the presence of PVT deviations. As discussed in Chapter 3, these global circuit issues could cause the DLL to be unable to lock. A coarse calibration block will be introduced to the DLL that will help ensure that a locked state is achieved across a wide range of PVT variations, in particular extreme environments. The final issue deals with the event that the DLL loses lock due to some external factor. This could be due to a temperature change, power supply drift, or in the case of space applications, a radiation strike. A DLL system transient monitor has been implemented that will watch the DLL outputs, and in the event that the DLL has lost lock, the system will flag an error.

In Section 4.1, an overview of the self-calibrating DLL will be presented, which will include the main DLL components that are not part of the calibration subsystems. Sections 4.2 and 4.3 will present the coarse calibration and system transient monitor, respectively. The full system layout will be presented in Section 4.4, followed up with extracted DLL simulations in Section 4.5. The chapter is concluded with a summary in Section 4.6.

4.1 Self-Calibrating DLL System Architecture

The basic DLL was discussed in Chapter 2, with the most basic architecture described in Figure 2.2. This work follows the same general DLL system, but incorporates the calibrating subsystems. The first consideration discussed will be the desired specifications for the DLL, which will include design of the main loop components such as the phase detector, charge pump, and loop filter. The VCDL will be discussed in basic terms, but the design will be built upon in Sections 4.2 to meet the needs of the calibration algorithms. The chapter will end with a discussion of the layout and full DLL simulations.

4.1.1 DLL Specifications

This research is meant to demonstrate a general purpose DLL that will be robust to PVT variations, in particular extreme environments. Even though it is possible to design a DLL that can operate at high frequencies, applications such as frequency synthesis by means of DLL tap multiplying or high-speed data acquisition by means of using the different phases of the input clock do not require a high frequency reference clock. These applications do not necessarily need the DLL to have a large locking range, but it is still important to have a margin of error built into the system to allow for the negative feedback loop to correct for PVT variations and allow the user to make changes to the input clock. As a demonstration of the calibration techniques, a VCDL with a center locking frequency of 115 MHz is chosen, with a lock range of 90 MHz to 140 MHz to allow for some tunability. These values are not specific to an application, so they could easily be adjusted to fit the desired needs. The DLL will be designed in a 150 nm partially depleted silicon-on-insulator (PDSOI) CMOS technology, which uses a nominal supply voltage of 1.8 V. To allow for a large margin of error in the supply voltage drift, the self-calibrating DLL will be designed to operate with a power supply range of 1.6 V to 1.8 V. As a demonstration of the extreme environment capabilities of the self-calibrating DLL, the DLL will be designed to operate across the temperature range of -125°C to 175°C .

A block diagram of the self-calibrating DLL is shown in Figure 4.1. The main loop of the DLL is shown in the bottom part of Figure 4.1 consisting of the phase detector (PD), charge pump (CP), loop filter (C_{Filter}), and the VCDL. The NMOS and PMOS devices connected to the node labeled V_{cnt} are part of the coarse calibration block. The coarse calibration block (Coarse Cal), phase detector enable circuit, and the transient monitor are also shown in Figure 4.1. These blocks will be discussed in detail in the proceeding sections. The label Ref CK is the input for the reference clock that is applied externally. Signals V_{UPPER} and V_{LOWER} are externally set reference voltages used by the coarse calibration block (discussed in Section 4.2). RST is an externally set reset that is used as global reset. RST is also connected to all of the subsystems, but is not shown in Figure 4.1. All other labeled signals are internal to the DLL, and will be discussed later.

The basic operation of the self calibrating DLL will now be described. First a global reset is applied to the DLL to ensure that all of the digital logic in the calibration block

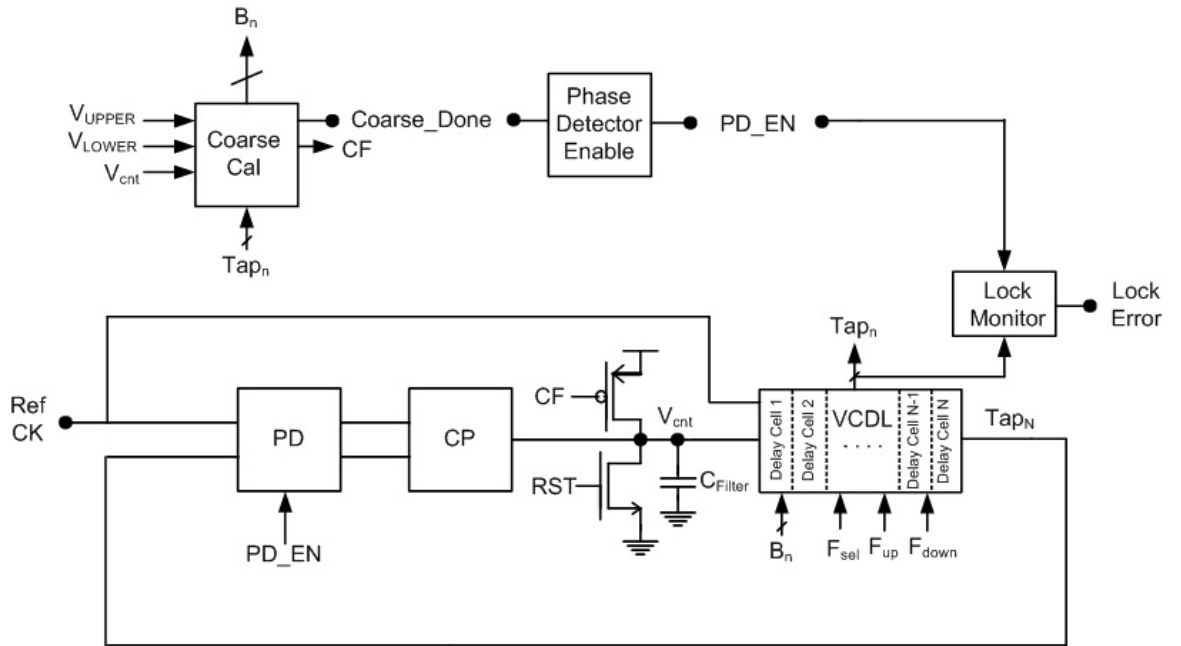


Figure 4.1: System diagram of the self-calibrating DLL.

is set to a known state, as well as to discharge the loop filter to ground as a means to create a known starting condition. After the reset is released, the coarse calibration block begins. At this time the main phase detector is disabled, leaving the charge pump in a high impedance state, effectively leaving the main DLL loop open. The signal CF (charge filter) is set low, thereby turning on the PMOS at the V_{cnt} node, which begins charging C_{Filter} up towards V_{DD} . This charging continues until V_{cnt} has reached a minimum of V_{LOWER} (discussed in Section 4.2). This PMOS is sized such that the charging rate of the of the loop control voltage ($dV/dt = I_{CF}/C_{Filter}$ is at least 10 times longer than the slowest allowed input reference clock period (17 ns). This is performed to ensure that the coarse calibration loop, which operates off of the reference clock, has at least 10 clock periods to respond to certain conditions inside the coarse calibration block. The coarse calibration will then digitally adjust the VCDL (last tap) delay is approximately one reference clock period. Upon completion of the coarse calibration, the phase detector is enabled, thereby closing the feedback loop of the DLL which allow the DLL to lock in the standard manner. The transient monitor is also enabled at the completion of the fine calibration. In the event that the DLL loses lock, the transient monitor will produce an error, which would signify that the system may need to be recalibrated.

4.1.2 Voltage Control Delay Line

The first consideration that needed to be considered when designing the main DLL is the VCDL architecture. Even though differential VCDLs provide higher PSRR, the simplicity of a single-ended delay cell was chosen for demonstration purposes of the digital calibration technique. It is possible to apply the presented calibration techniques in a differential manner, which can be added future revisions of the self-calibrating DLL. The delay cell without the calibration components is shown in Figure 4.2. This delay cell is based off of the work presented by Baronti [39]; however it was modified to use a current starved inverter for analog delay control. In this delay cell, the first stage is a current starved inverter, which is controlled by a voltage-to-current converter from the main loop delay

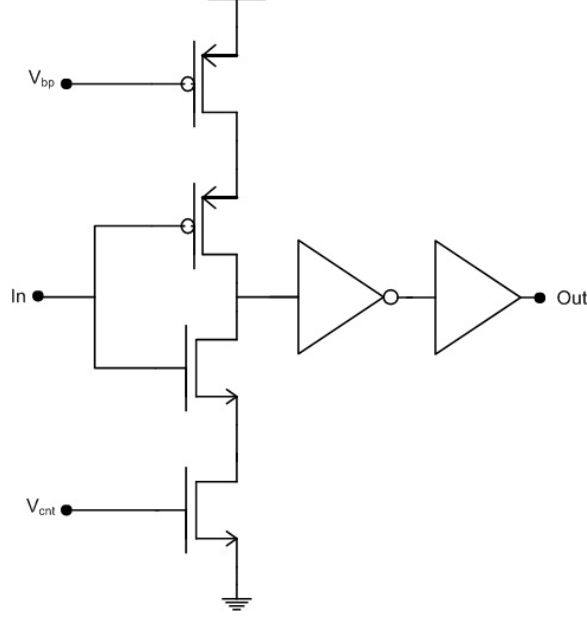


Figure 4.2: Simplified delay cell.

control. The following stages consist of a simple inverter, and a buffer used to drive long traces. The voltage-to-current (V-to-I) converter is shown in Figure 4.3, and simply is a replica of the current starved inverter in first stage of the delay cell, except the PMOS is diode connected and the middle PMOS and NMOS devices are connected to the rails such that they are both fully inverted. The generated bias voltages V_{cnt} and V_{bp} are used as the bias voltages in all of the VCDL's delay cells.

Referring to Figure 4.2, the delay of a single delay cell is given as

$$T_{delay_{cell}} = T_{d1} + T_{d2} + T_{d3}, \quad (4.1)$$

where T_{d1} is the delay of the current starved inverter, T_{d2} is the delay of the second inverter, and T_{d3} is the delay of the buffer. T_{d1} is a dependent upon the state of the output of the

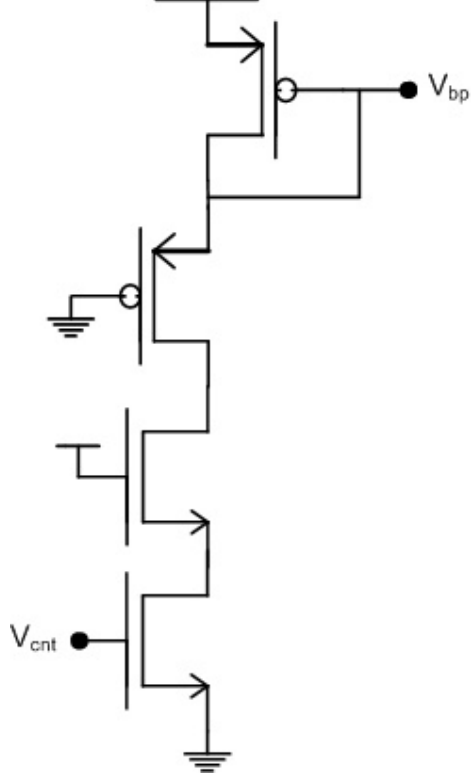


Figure 4.3: Voltage-to-current converter.

current starved inverter, or whether the output is going from V_{DD} to V_{SS} or vice versa. Ideally the current that the top PMOS sources should be equal to that the bottom inverter sinks. The rate at which the output of the current starved inverter switches is given by

$$\frac{dV_{o1}}{dt} = \frac{i_{dcnt}}{C_{L1}}, \quad (4.2)$$

where V_{o1} is the output of the current starved inverter, i_{dcnt} is the current level of the PMOS source and NMOS sink (assuming they are in saturation), and C_{L1} is the equivalent capacitance seen at the output of the current starved inverter. T_{d1} would approximately be

the time that it takes for the current starved inverter output to reach the switching point of the next inverter, V_{SW2} . By integrating 4.3, T_{d1} is given as

$$T_{d1} = \frac{V_{SW2}C_{L1}}{i_{dcnt}}. \quad (4.3)$$

This equation does not take into consideration the effects of the on resistance of the middle MOS devices. Also, this value would be different for positive and negative going input edges if the switching point is not at the mid supply. The second inverter delay is given by the standard inverter delay equation

$$T_{d2} = \frac{V_{SW3}C_{L2}}{i_{dn,p}}, \quad (4.4)$$

where V_{SW3} is the switching point of the buffer, $i_{dn,p}$ are the currents sourced by the inverter MOS devices, and C_{L2} is the equivalent capacitance seen at the output of the second inverter. The buffer, which contains two inverters, would simply be the sum of two inverter delays as described by 4.4.

It should be pointed out that the actual delays will differ from the previous equations since the rate at which the previous inverter stage impacts the switching speed. Consider the devices of the second inverter as the current starved inverter output switches from V_{DD} to V_{SS} . The output of the current starved inverter will change relatively slow compared a normal inverter, deviating from an ideal pulse. If the input of the second inverter is at V_{SW2} , the NMOS and PMOS devices are both on, therefore the output will not change keeping it at V_{SS} . Even as the input switches below V_{SW} , the NMOS will still sink some current, even though it is less than the PMOS sources. The output will begin to change at this time, but at a much slower rate than described by 4.4. Knowing this, it is important to

design the inverters (excluding the current starved inverter) with a high gain as to minimize the second order effect.

As mentioned, equation 4.3 neglects the effect of that the on resistance of the middle MOS devices. It is important to ensure this resistance is low enough such that in voltage drops across the MOS devices will not cause the current source/sink devices to go out of saturation. If either the sink or source was to go out of saturation, the charging current would change from the nominal values. This would limit the range of operation of the delay cell.

As a demonstration, the VCDL contains 10 delay cells. The delay cell and VCDL will be discussed more in Section 4.2.

4.1.3 Charge Pump and Loop Filter

The implemented VCDL only requires that the charge pump be single-ended. Even though there is variety of charge pumps discussed in Chapter 2, a cascoded single branch architecture was chosen for this work based on the circuit shown in Figure 2.12.

The implemented charge pump is shown in Figure 4.4. This architecture was chosen based upon the design of the phase detector, which generates small charging pulses in the near lock state. The cascode devices are biased by a Minch bias cell [40] as a means to operate the cascode devices at the edge of saturation such that it will operate properly at lower voltages as required by the design specifications. This biasing technique was also done a means to allow the loop filter to be charged within two V_{DSSAT} of V_{DD} , even though it is possible for the loop filter to be charged above that level if the cascode devices were to go out of saturation. The NMOS and PMOS devices in the middle with their sources and drains connected are used as a means to help minimize charge injection from the switches.

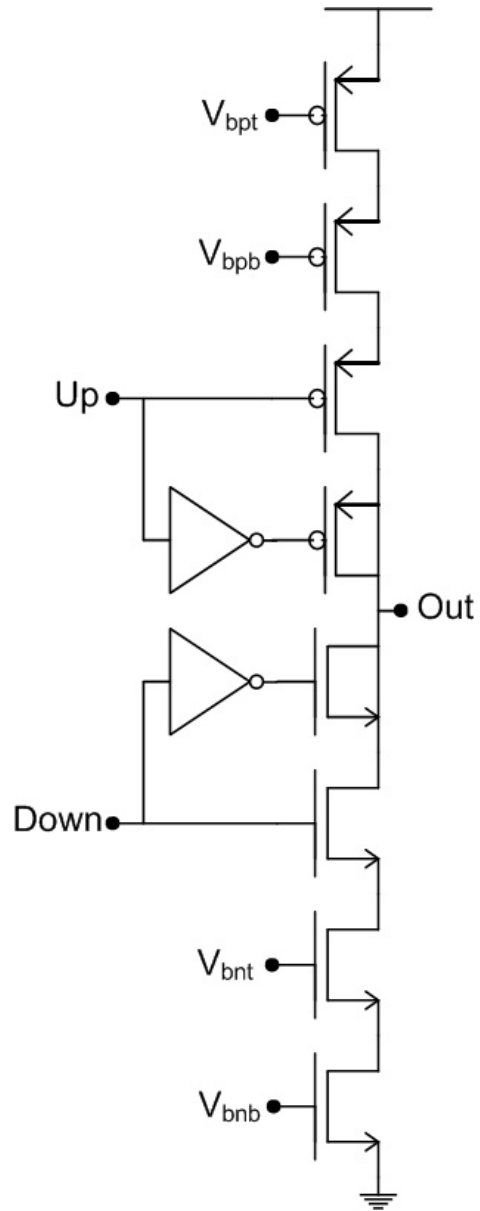


Figure 4.4: Charge pump schematic.

For this work, the single capacitor architecture was chosen for the loop filter. The size of loop filter and charging current are chosen based on the DLL input range specifications. Using the general rule of thumb shown in gives that the DLL bandwidth should be on the order of 6 MHz based on the lower limit specification. To keep noise (jitter) low, the actual bandwidth of this DLL was chosen to be much lower. This normally would result in a long locking time, but the coarse calibration speeds the locking process up. For this work, the loop filter was chosen to be 16 pF MIMCAP as a tradeoff between area and loop gain. The charging current is nominally chosen as 10 μA , but this current will be set off-chip to allow for extra control of the loop bandwidth by the user.

4.1.4 Phase Detector

The phase detector of Figure 4.5 is based on a standard phase-frequency detector and the pulse resaper presented by Kim *et al.* [41]. The addition of an enable function was added to the circuit for control by the calibration circuit. The operation is the same as the standard PFD, but with the addition of the pulse shaper, it is possible to have better control of the dead-zone. Assume that the charge pump source and sink currents are unmatched. Using a standard PFD, there is a large overlap between the Up and Down signals to eliminate the dead zone issue. In the charge pump with mismatched currents, the static phase offset of the DLL output with respect to the reference clock would become larger. This is discussed in more detail by Kim *et al.* [41]. With proper sizing of the long L inverters in the pulse shaper, it is possible to minimize the static phase offset. Figure 4.6 shows the main phase detector simulation for reference and DLL output signals being close together (i.e. small phase error). As can be seen, there is a soft transition section when the reference and DLL output are 40 ps apart from each other. During this time, the Up and Down signals do not

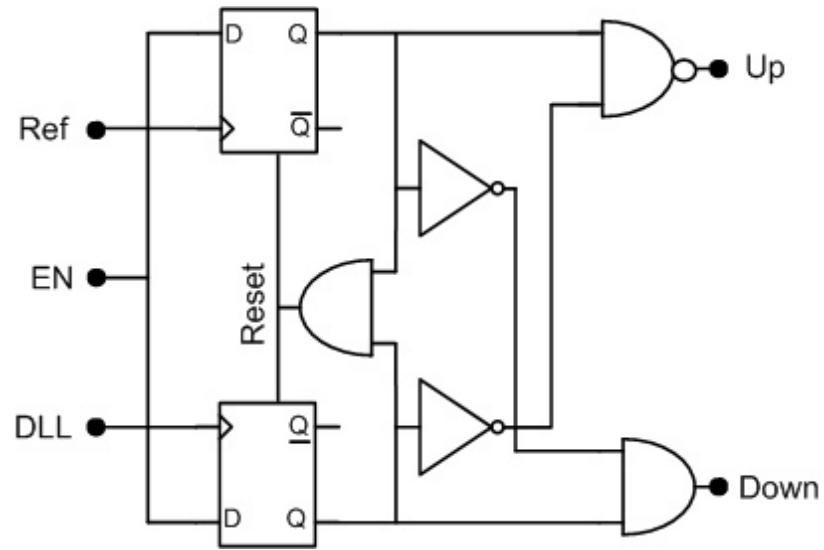


Figure 4.5: Main DLL phase detector.

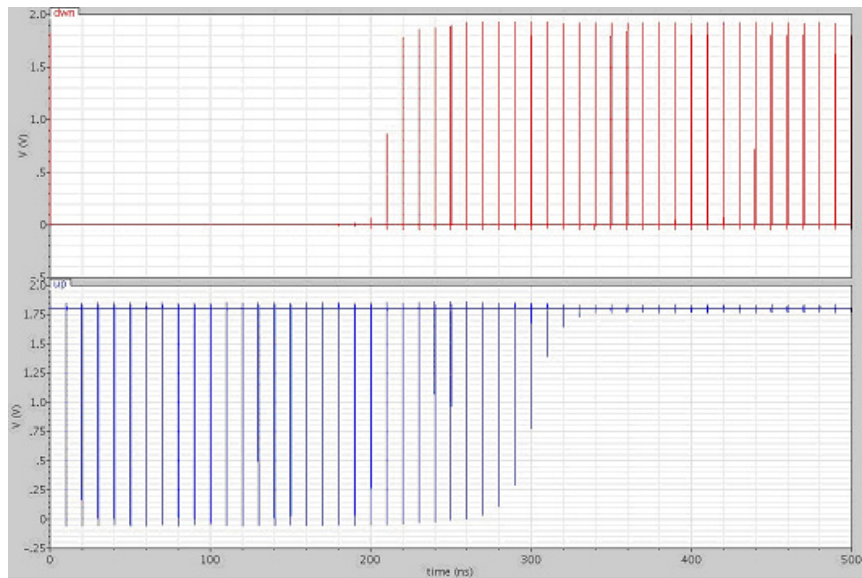


Figure 4.6: Main DLL phase detector simulation with small phase error at the inputs.

reach the corresponding rails at the same time. Figure 4.7 is a generalized transfer plot of the phase detector. The soft dead zone concept is demonstrated in the plot.

4.2 Coarse Calibration

The main purpose of the coarse calibration system is to place the VCDL in a desired delay range for the input clock regardless of any PVT non-idealities. To ensure that the calibration block is insensitive to PVT variations, an all digital implementation is used. This section will introduce the digital controllable aspects of the VCDL, including the coarse calibration algorithm and circuit implementation. After calibration, the DLL main loop must be closed

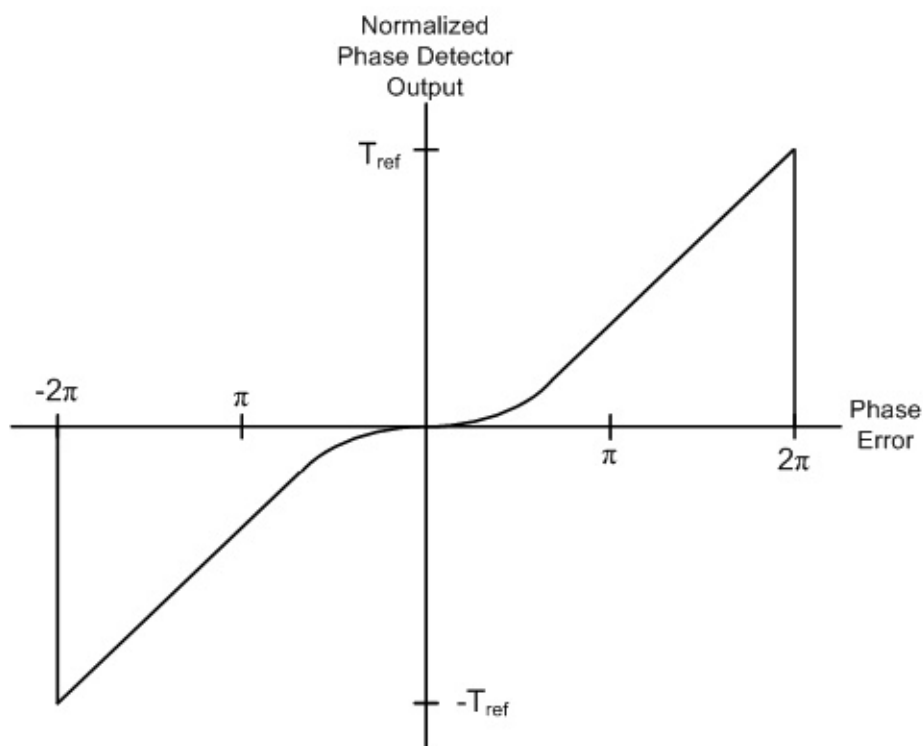


Figure 4.7: Charge pump transfer plot.

at the correct time, therefore a circuit is used to ensure that the phase detector is enabled at the correct time, which will be discussed at the end of this section.

4.2.1 Digitally Controllable VCDL

The first aspect of the coarse calibration is how to change the VCDL's delay in a digital manner. By building on the delay cell of Figure 4.2, digitally controlled MOSCAPs are added to the output of the simple inverter, in a similar to what was done by Baronti *et al.* [39]. This is shown in Figure 4.8. For this design five binary weighted capacitors are switched in and out via transmission gates as a means to digitally adjust the delay cell's delay. This is effectively changing C_{L2} in equation 4.4 in steps of C_C . The tunable capacitance range goes from $31C_C$ Farads down to no added capacitance.

As discussed in Chapter 3, the delay cell used in [39] uses two cascaded inverters with equal NMOS capacitor arrays at each of the inverter outputs to ensure that the output pulse does not shrink. An alternative to this approach is to use complementary MOS capacitors (nMOS and pMOS) at a single node as shown in Figure 4.8. Consider when the second

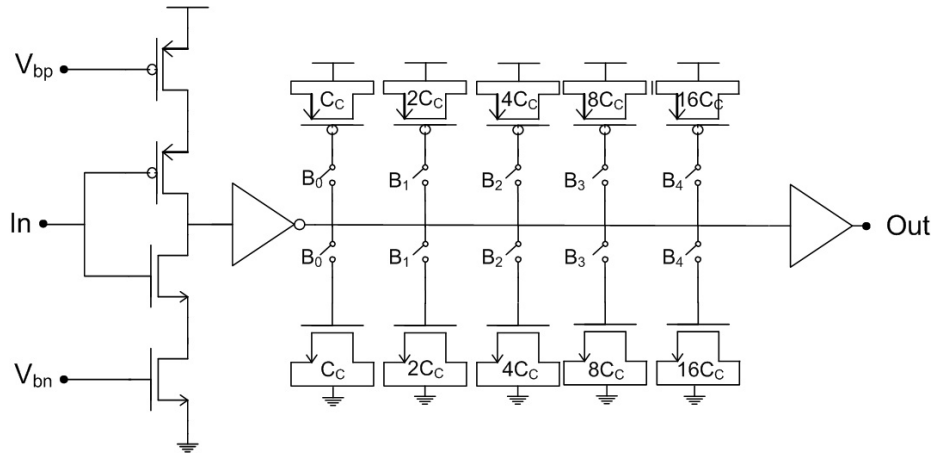


Figure 4.8: Delay cell schematic with complementary MOSCAPs.

inverter (the one connected to the current starved inverter) is at V_{SS} and switching to V_{DD} . The nMOSCAP has 0 V V_{GS} and the pMOSCAP has V_{DD} volts across its V_{GS} . The reverse would occur for a falling edge. Assuming that the nMOS and pMOS capacitors are sized to be equal in magnitude while they are in depletion mode ($V_{GS,D}$ equal to the rail voltage), the capacitance seen at that node at the beginning of the switching condition will be the same. Even though MOSCAPs exhibit some off-state capacitance, it is many orders of magnitude smaller than the depletion capacitance. This means that the off-state MOSCAP will add a negligible amount of capacitance to the second inverter output.

The VCDL was simulated using Spectre to determine the effectiveness of the described topology. The simulation results are shown in Figure 4.9. The limits of the digitally tunable delay range are 3.8 ns up to 25 ns with the bottom curve being for the condition with no capacitance added and the top curve having all 31 capacitors added. When the control voltage is between 0.55 V to 0.8 V, the delay transfer characteristic is very large, as compared to the higher control voltages. Consider any noise or other system fluctuation (e.g. charge injection from the charge pump or kickback from the VCDL) that might exist on the control voltage node. If these fluctuations occur in this large gain region, the shift in the VCDL delay will be much larger than for that in the lower gain linear operating region. For this reason, the proposed coarse calibration will calibrate the VCDL in the lower gain region as shown in transfer plots of Figure 4.10. By limiting the control voltage to this low gain range, the VCDL tuning range is reduced to 4 ns to 14.5 ns. One thing that can be noticed from Figure 4.10 is that the tuning steps overlap each of the successive codes. The large amount of overlap was intentionally designed to ensure that there would be no dead zones across PVT variations, which are delay ranges where the VCDL could not be tuned to via the control voltage. This topic was discussed by Aktas [38].

From the design specifications, the total VCDL delay needs to be in the range of 7.14 ns to 11.11 ns. This range is encapsulated by the simulated results in Figure 4.10. However the effects of temperature and voltage have yet to be considered.

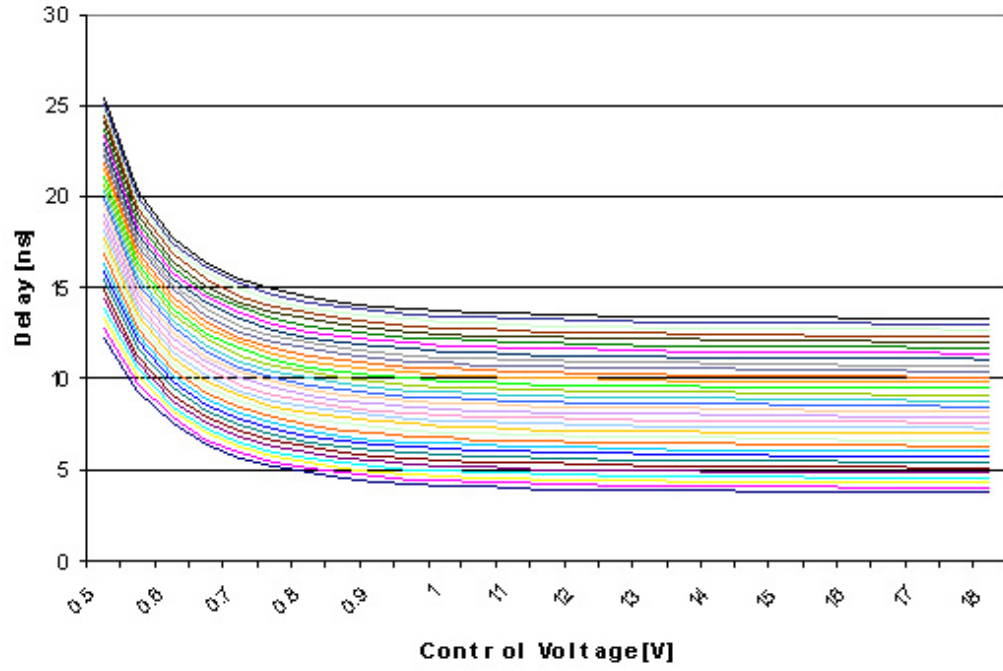


Figure 4.9: Simulated VCDL delay versus control voltage for all digital calibration codes.

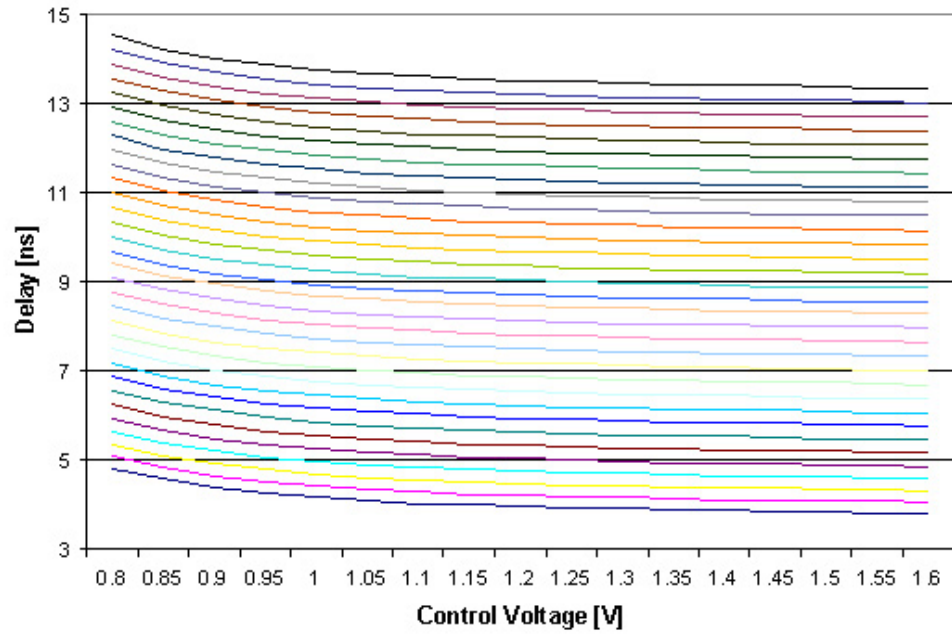


Figure 4.10: Simulated VCDL delay versus control voltage zoomed to the lower gain range.

The temperature effects of the current starved inverter were mentioned in Chapter 3. For the individual delay terms of the delay cell described in 4.3 and 4.3, it can be seen that the switch points of the load inverters are an important component part of the equation. The general equation for an inverter switch point is given as

$$V_{SW} = \frac{\sqrt{\frac{\beta_n}{\beta_p}} V_{THN} + (V_{DD} - V_{THP})}{1 + \sqrt{\frac{\beta_n}{\beta_p}}}. \quad (4.5)$$

Knowing that speed of the current starved inverter decreases with temperature, and that the load capacitance term changes in 4.3 and 4.3 are small, it can be inferred that the delay terms will become longer (slower) as temperature increases.

Based on the discussion of the current starved inverter in Chapter 3, it is also apparent that the VCDL delay will decrease with reduced supply voltages.

Considering the effects of temperature and supply voltage, the VCDL delay range needs to be large enough to encompass the combined delay shifts. The VCDL was simulated with various temperatures and supply voltages, and the results are shown in Figure 4.11. It is obvious that the tunable VCDL delay range has been reduced. The worst case maximum input clock across the PVT specifications is now 140 MHz which occurs when the VCDL is digitally tuned to use no capacitors at -100°C with a supply of 1.4 V. A supply of 1.4 V was used to allow for margin of error with noise issues. The minimum input clock is 88 MHz, which occurs when the digital control word selects all of 31 capacitors at 100°C with a supply of 1.8 V. This range is within the desired specification of 90 MHz to 140 MHz. It should be noted that these minimum and maximum values occur at the mid control voltage point which means that there is some flexibility to the values. Also, the models may not be valid at -100°C , but they at least give a general idea of the PVT trends.

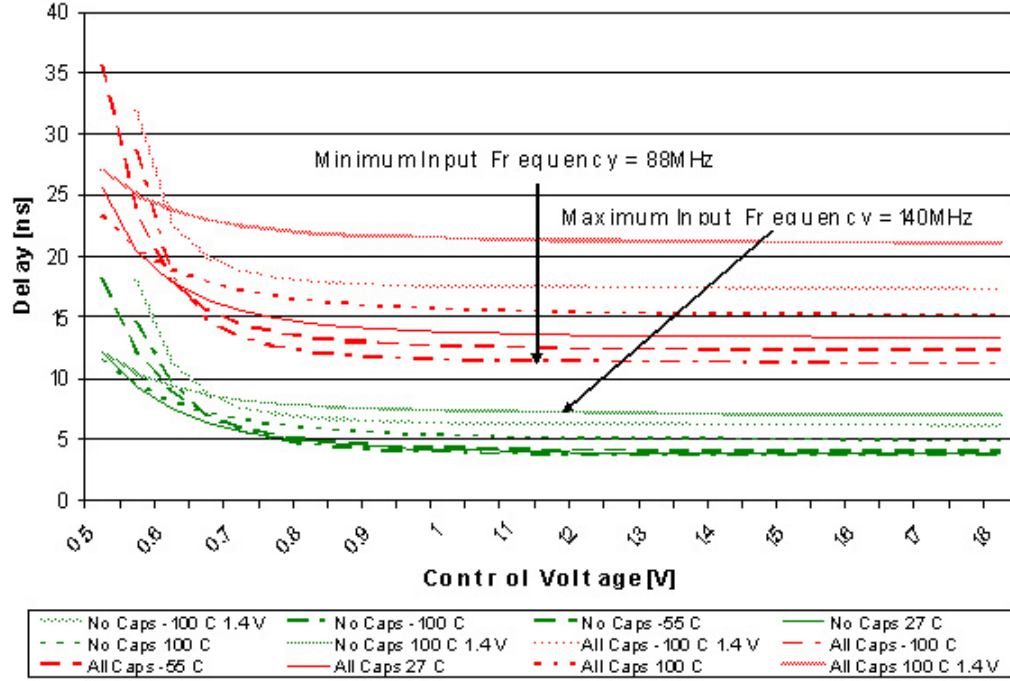


Figure 4.11: Simulated VCDL delay versus control voltage across temperature and supply voltages.

4.2.2 Digital Coarse Calibration Circuit

Various forms of DLL coarse calibration techniques were discussed in Chapter 3. Of all the prior DLL digital calibration techniques discussed, none worked for an analog multi-phase DLL. Whereas all-digital DLLs have many benefits, there are still applications that require analog controlled DLLs, especially if static phase offset may be an issue. Also, no temperature and supply voltage data was presented for these architectures.

The digital coarse calibration technique serves three purposes. The first is to ensure that the does not succumb to the false lock condition. The second is to calibrate the VCDL to minimize the effects of temperature and supply voltage. Finally, the coarse calibration will aid in speeding up the locking process. A flow diagram of the coarse calibration algorithm is shown in Figure 4.12.

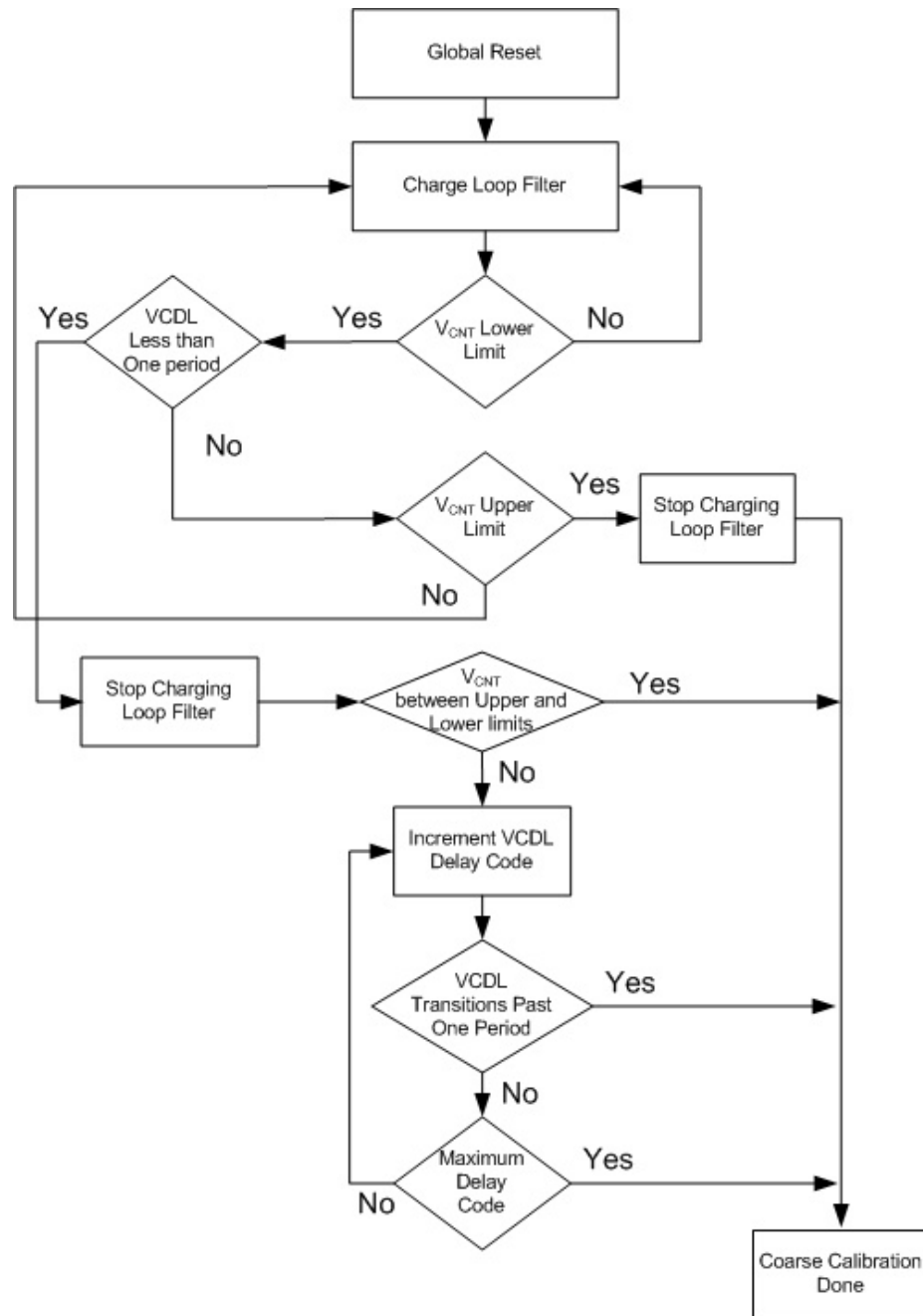


Figure 4.12: Coarse calibration algorithm flow chart.

Initially a global reset is applied to the DLL, which resets all of the digital logic and registers to a known state, as well as pulls the control voltage to ground. Once the reset signal is released, the loop filter begins to charge towards V_{DD} . This is done by control signal CF in Figure 4.1. During this state, the phase detector is disabled via the PD_EN signal as shown in Figure 4.1 to ensure that the DLL feedback loop is opened. This keeps the DLL from trying to lock, while the calibration logic is trying to calibrate the VCDL. While the loop filter is charging, the coarse calibration logic is watching for one of two comparators to trip. These comparators are monitoring the loop control voltage and comparing them to upper and lower reference voltages used to define the low gain range operation for the VCDL. If this lower limit has not been reached, the loop filter continues to charge, otherwise the coarse calibration test logic checks to see if the VCDL delay is less than one reference clock period. If the VCDL is less than one reference clock period, the loop filter stops charging. At that point the VCDL control voltage would be set at the lower limit, and the delay would be less than the reference clock period. In other words, the VCDL delay is too short and will need to be increased. To do this, the coarse capacitor delay word is increased until the VCDL delay transitions from less than one reference clock period to just greater than one reference clock period. If the upper limit of the VCDL capacitor control word is reached, the loop filter stops charging, and the coarse calibration is done. In this case the DLL is at the upper limit of its capture range, and will be unable to lock. However, if the transition occurs, the coarse calibration algorithm is now done, and the VCDL is close to a locked condition, or one reference clock period delay, which allows for faster locking once the DLL feedback loop is closed. In the case where the max capacitor delay word is reached, the coarse calibration is now done, but will have to lock out of the low gain range, or not even lock at all. If the VCDL delay is greater than one reference clock period, and the lower limit has been reached, the upper limit of the delay range is tested. If the upper limit has not been reached, the loop filter continues to charge until either delay of the VCDL becomes less than one reference clock period or V_{CNT} has reached the upper limit.

The coarse calibration method proposed for this work is based on the technique presented in Chang *et al.* [37]; however the method proposed here has access to a fixed number of multiple taps like the digital DLL of Chung and Lee [11]. The edge detection circuit in Figure 4.13 is identical to that presented by Chang, but in order to allow for multiple taps, the calibration algorithm (described above) had to be altered. All of the D flip-flops are reset by the CK1.bar signal (inverse of CK1), and once CK1 goes high, as described later, the bottom row of D flip-flops begin to sample the VCDL taps. For simplicity, only some of the D flip-flops are shown. Once the third tap of the VCDL goes high, the output of that D flip-flop goes high, which is the input to the D flip-flop that will sample the fourth VCDL tap. This process ripples down through the registers until the last VCDL tap is reached. After approximately one clock period has occurred, the top row of D flip-flops registers the state of the sampling D flip-flops. For example, consider the waveforms shown in Figure 4.14 where the VCDL delay is greater than one clock period. At the time where CK2 goes high, dn3 through dn8 will be high, and dn9 and dnout will be low. For the VCDL delay to be less than one reference clock period, dnout must be high. This means that this circuit is able to discern whether the VCDL output is greater or less than one reference clock period. The third tap was chosen as the initial sample point to save hardware, since there would be no condition where the VCDL would be that much longer than the reference clock.

The control signals, also shown in Figure 4.13, are generated by the circuit shown in Figure 4.15. The clock edge generation circuit is based on the circuit presented by Chang *et al.* [37]; however their technique only works for one conversion. The proposed edge generation circuit has extra stages added, plus a fed back reset signal, to allow for continuous operation, and allow time for switching in the capacitors of the VCDL. The circuit consists of cascaded D flip-flops that are triggered of the system reference clock. Assuming the system is reset, all of the D flip-flops' outputs will be low. The first rising clock edge causes signal CK1 to go high, and then after the second reference rising clock edge signal CK2 goes high, while CK1 remains high. This process continues for a total of five clock edges, which is demonstrated in the waveform of Figure 4.15. The signal CK5 is

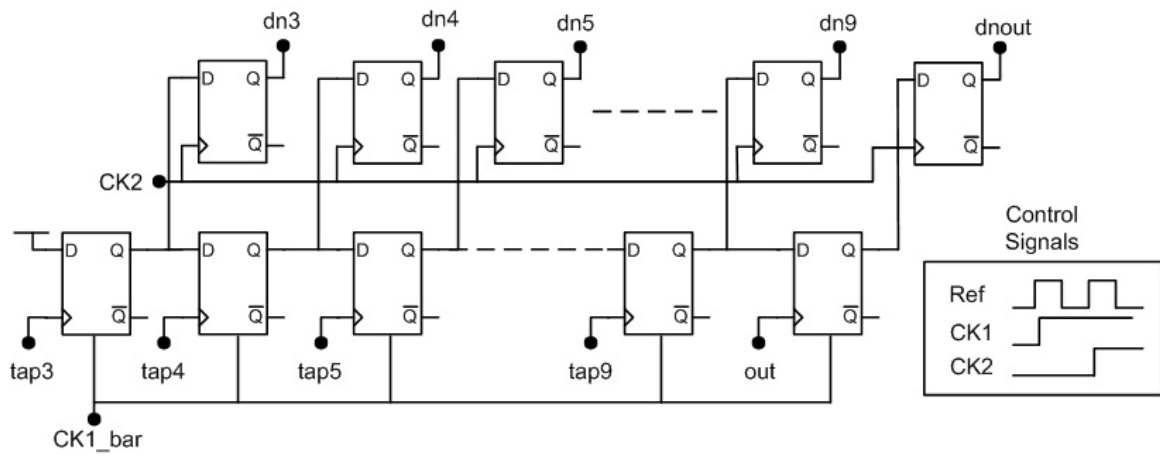


Figure 4.13: Edge detection circuit used for the coarse calibration block.

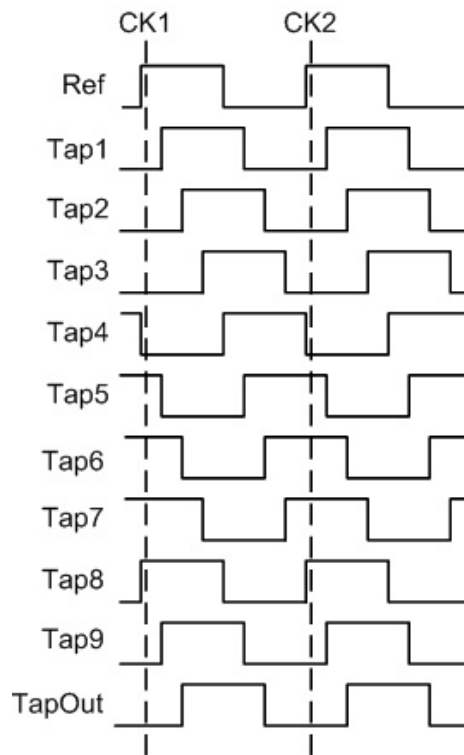


Figure 4.14: Example of VCDL outputs for delay being too long.

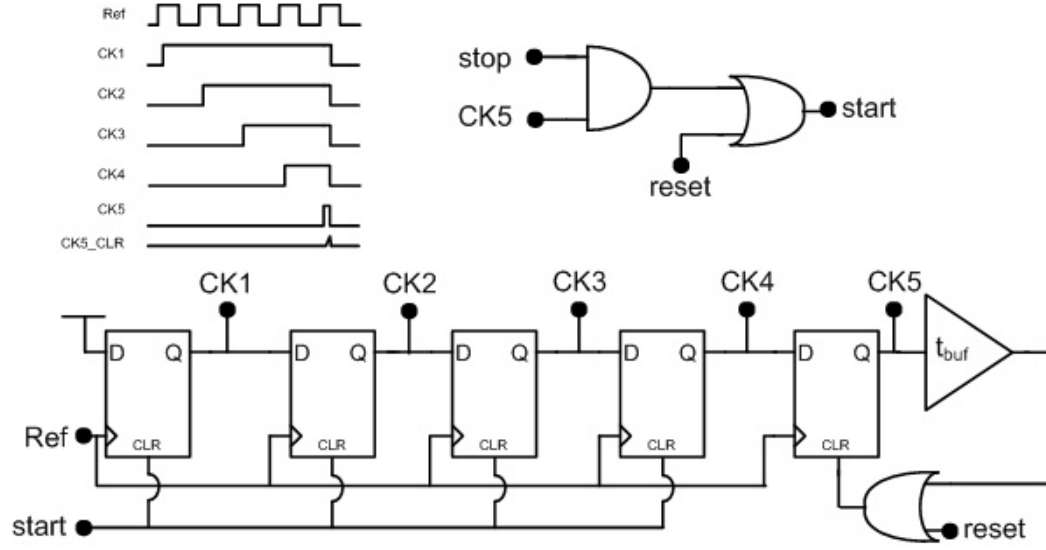


Figure 4.15: Clock edge generation circuit used for the coarse calibration circuit.

used to reset the first four D flip-flops (signals CK1 - CK4), and the signal CK5 is delay via a buffer as means to reset the CK5 register and allow a longer reset pulse for the first four registers. The CK5 register reset signal is labeled as CK5_CLR in the waveforms of Figure 4.15. This long reset pulse (start) is used in other places throughout the coarse calibration circuit, and it is essential that these other circuits are given enough time to reset properly. The duration of the start (reset) signal is set by the delay of the buffer and the connecting logic. This delay must be less than the maximum input reference clock period. Even though CK1 and CK2 have been used thus far, CK3 and CK4 are used during the VCDL capacitor switching portion of the coarse calibration.

After a reset, the loop filter is pulled to ground as shown in Figure 4.1. After the reset goes, the CF signal is set high to charge the loop filter to some value. The circuit shown in Figure 4.16 is used to determine when to stop charging the loop filter. It consists of two voltage comparators, input reference voltages, and some glue logic. The continuous time comparators are based on the ones described in [16], but have D flip-flops at their

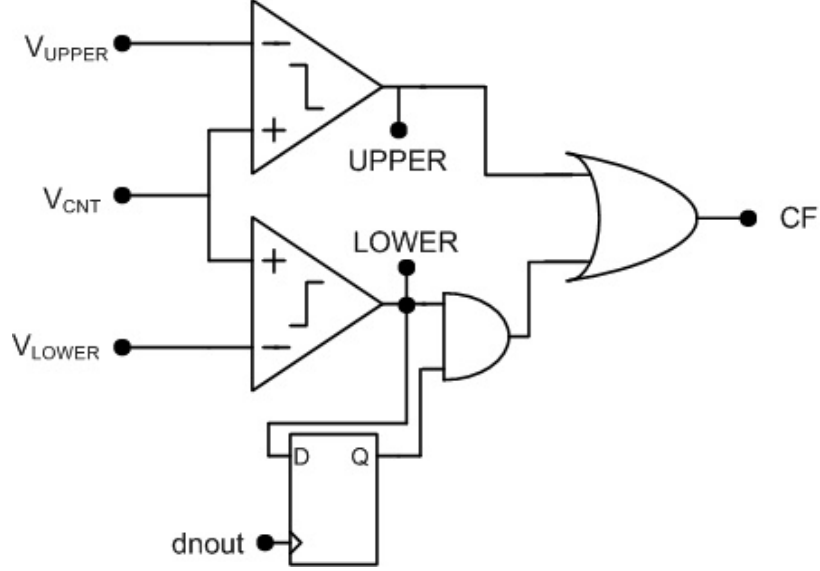


Figure 4.16: Circuit used to initially charge loop filter.

outputs (not shown) to register the triggered event. This is to ensure that the outputs of the comparator outputs do not oscillate in the presence of a noisy control voltage causing an incorrect. This could be further improved by adding a power down feature to the comparator. The lower flip flop triggers whenever dnout (refer to Figure 4.13) triggers indicating that the DLL delay is shorter than T_{REF} . If this flip-flop and the comparator connected to the lower voltage reference goes high, the loop filter stops charging, since the loop filter has been placed in the low gain range, and the VCDL's delay is too fast, therefore must be slowed down. If the upper comparator triggers, this means that the input reference is too fast and is out of range of the DLL. Either one of these events cause the DLL loop filter to stop charging, and allow the coarse calibration to continue into the next step.

At this point the VCDL would be out of range, which stops the calibration algorithm, or it is too fast. The edge detection circuit is now used to analyze the outputs of the DLL in the manner described above. A J-K flip flop up only counter is incremented after each test cycle, which will switch in the MOSCAPs of the delay cell. If the counter reaches the

max count, the calibration algorithm ends, and the loop closes allowing the DLL to try to lock to the signal, but this would be outside of the low gain range.

4.2.3 Post Coarse Calibration

After fine calibration has finished, the DLL loop is still open until PD_EN goes high. However, PD_EN can only go high under certain conditions, otherwise there is a chance the phase detector will start comparing on the wrong edge. This is shown in Figure 4.17. For example, if the phase detector is enabled in between the reference and VCDL output signals (red dashed line), the phase detector will indicate the incorrect action to be taken by the charge pump (in this case the charge pump will continuously lower thereby making the VCDL even longer). The correct case is shown in the lower box of Figure 4.23. The phase detector only detects rising edges, so it is desirable to enable the phase detector sometime before both rising edges. A simple solution that would the majority of the time would be to enable the phase detector while both clocks are low. This could be accomplished by applying both signals to a NOR gate, and using that signal to register the calibration done signal. This technique would work for almost every case of the proposed self-calibrating DLL, since the two clocks should be close to lock upon completion of the calibration. However if the two clocks are 180° out of phase, the output of NOR would not go high. Even though this case is most likely never to occur, a more robust solution was designed to ensure that the phase detector is enabled at the proper time.

The concept employed to enable the phase detector is based on edge detection circuit in Figure 4.13, and is shown in Figure 4.18. Assume all of the registers have been reset on start up, making all of their outputs low. After all of the calibration algorithms have finished (Calibration_Done goes high), and Calibration_Done is registered after the next rising reference clock edge, the even ordered taps are monitored. The registered Calibration_Done signal is propagated through the lower chain of D flip-flops as the delayed initial reference clock (the one that clocked in the Calibration_Done signal) is used to register the other D

flip-flops. Also the same registered Calibration_Done signal is registered by the upper D flip-flop after the next successive reference clock, which is approximately one clock period delay. The Calibration_Done signal delayed by one clock period is used to register the last VCDL registered Calibration_Done signal. If the VCDL delay is less than one clock period, then the D flip-flop clocked by tapOut (last delay cell) will be high before the upper D flip-flop goes high, which means that the Q of the D flip-flop after the two Calibration_Done paths will go high. This causes the upper D flip-flop after the upper AND gate (output labeled Lead) goes high, which signifies that the VCDL is faster than one clock period. The output of the OR gate goes high, thereby registering PD_EN to go high. This signal is connected to the EN pin of Figure 4.1, which are the D terminals of the edge triggered D flip-flops of the main DLL phase detector. Since the VCDL delay was less than one clock period, tapOut was high before next successive Ref clock goes high, and PD_EN goes high after the lagging signal to ensure that the first edge that the main phase detector sees is the leading edge. The waveforms of this example are demonstrated in Figure 4.19. The reverse, which will be the normal case setup by the coarse calibration, occurs whenever the VCDL delay is greater than one reference clock period. In the case where the Ref and tapOut signals are almost in lock (perfectly lined up), the Lead or Lag signals could go high, but PD_EN would still go high after both edges due to the logic delay of the Lead/Lag D flip-flops plus the OR gate plus the PD_EN D flip-flop.

It can be noticed that only the even ordered taps are used in the circuit shown in Figure 4.18. This was only done as a means to reduce the amount of hardware. For this implementation, there is no way with the given tuning range of each delay cell, that the delay step size between two delay cells is large enough to cause an error in the ripple circuit.

After the phase detector has been enabled the DLL will begin to pull-in to a locked state. The amount of time this takes will be dependent on multiple things. Assuming that the coarse calibration has worked properly, and the VCDL was initially faster than the

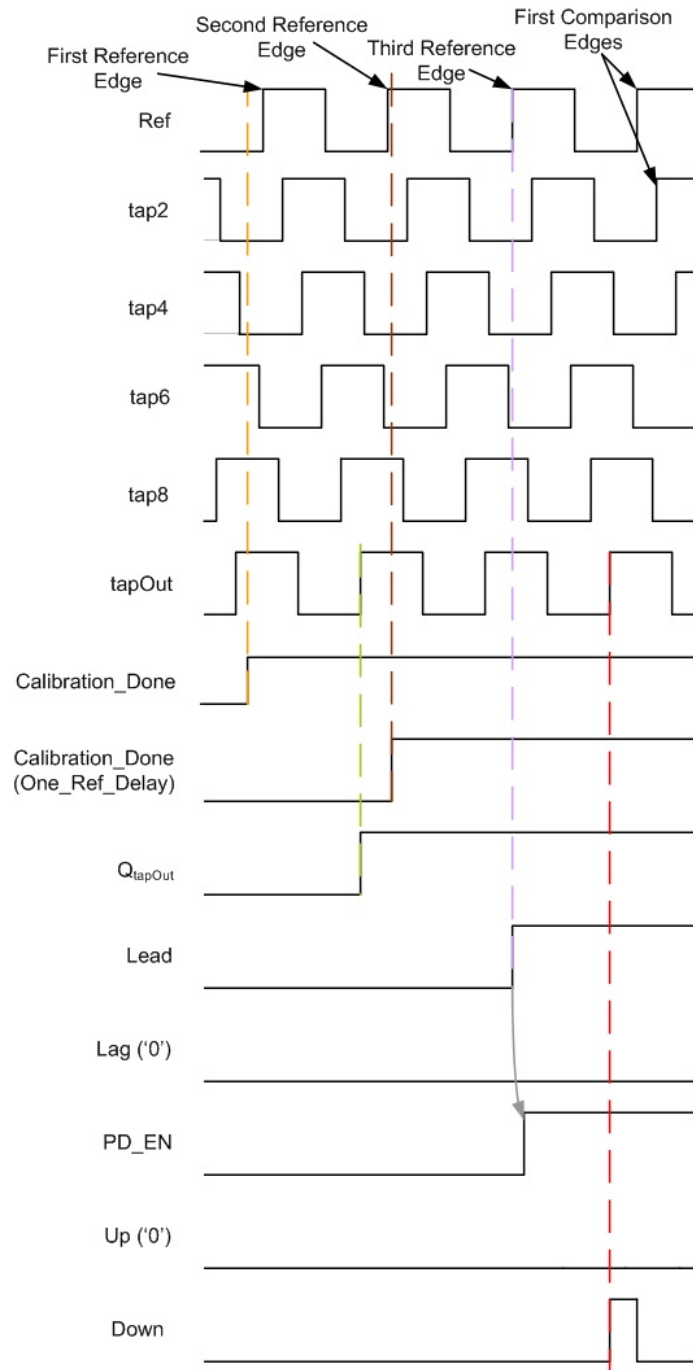


Figure 4.19: Example of the process used to enable the main phase detector.

clock period (meaning coarse calibration capacitors had to be inserted), the output of the VCDL would be within one bit coarse calibration step delay from the reference period. The coarse calibration running time would consist of t_{Charge} , the time required charge the filter to V_{Lower} , and how many capacitors that must be added. The main loop locking time is a function of the input clock frequency (consists of 4 clock periods per test, reference and VCDL output phase offsets, and the charge pump charging current/loop filter size. When the two clocks are further apart, the phase detector gain is larger, hence it sends larger up and down signals to the charge pump. This allows the loop filter to (dis)charge at a rate of $(I_{CP}/C_{Filter}) t_{on}$, where I_{CP} is the charge pump current, C_{Filter} is the loop filter size, and t_{on} is given by $K_{PD} \times \phi_e$ K_{PD} is the phase detector gain and ϕ_e is the phase difference between the reference and VCDL signals. The loop filter gets updated once every clock period until the DLL is locked, which is a function of the reference clock frequency (i.e. the loop filter charges more often with the input clock is faster). Also ϕ_e becomes smaller as the signals become closer to lock, which slows down the close in locking time. It would be difficult to give the exact charging time due to the non-linear nature of the implemented phase detector as it approaches small phase errors, unlike standard phase detectors. Simulations with a 100 MHz clock have shown a lock time of approximately 2 μs , which consisted of 320 ns for the coarse calibration to finish, and over 1.6 μs for the main loop to lock to an initial time difference between the output of the DLL and reference clock of approximately 310 ps. For a DLL with the same loop dynamics, it would take approximately 40 μs .

The lock time can be written as

$$T_{lock} = T_{charge} + 4 \sum_{n=1}^N T_{REF} + T_{DLL}, \quad (4.6)$$

where T_{charge} is the charging time of the loop filter, the summation is the time it takes for the coarse calibration to complete, and T_{DLL} is the time it takes for the DLL to finish the locking process.

4.3 System Transient Monitor

One of the target applications of this DLL is for extreme environment operation, it is important to ensure that the DLL does not lose lock in case some external event, be it temperature shift, supply change, radiation, or some other phenomena causes the DLL output to shift in some transient manner.

After the phase detector is enabled, the DLL is usually close to lock (less than 1 ns for the worst case), which is less than 9% of the clock period if the input clock is at the center of locking frequency of 115 MHz. At this time, since the DLL is close to lock, a system transient monitor will begin to watch the VCDL for any major changes, and will report an error if the VCDL exceeds an acceptable range. This range would ensure that the main DLL loop would be able correctly be able to lock. Assume that the control voltage is close to the center of the VCDL tuning range, and some event causes the DLL output to shift (e.g. the input clock frequency changes or the ambient temperature has changed), it would be possible that the DLL would be unable to relock to the input clock. This would occur if the output of the DLL shift exceeds the 1 ns capture range.

The circuit shown in Figure 4.20 is the schematic of the system transient monitor. The concept of the system transient monitor uses the same ripple concept used in the coarse calibration and phase detector enabler. The implemented system transient monitor actually uses the phase detector enable circuit in Figure 4.18, but has been redrawn for clarity below. Also, the implemented version error range is larger than as is shown in Figure 4.20, but principle is the same as what is presented below. The main purpose is to watch

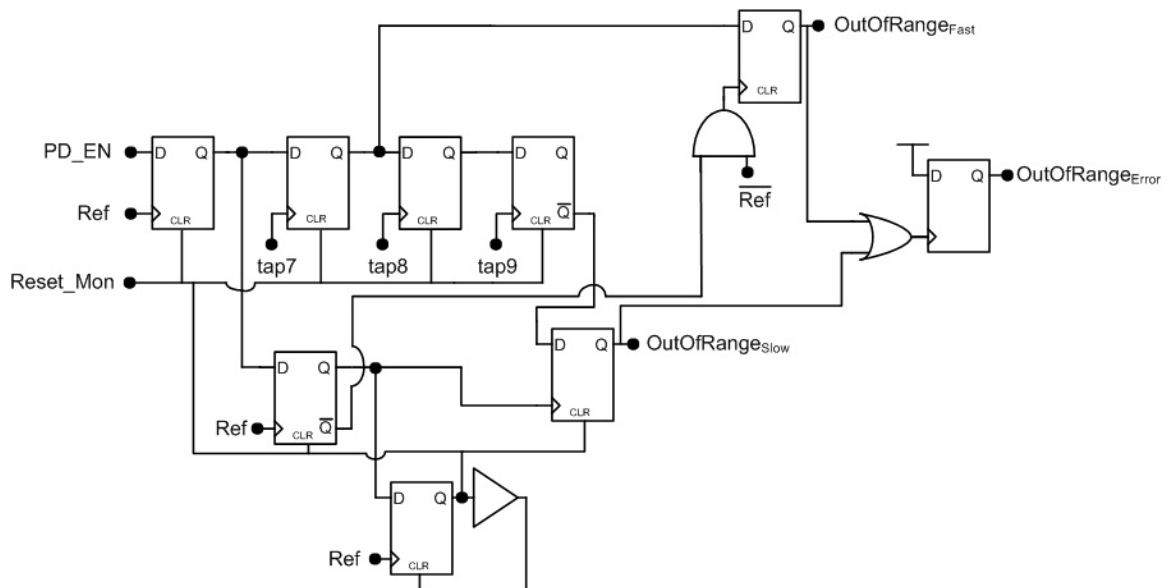


Figure 4.20: Schematic of the DLL transient monitor.

certain VCDL taps, and compare them to the reference clock. If the VCDL delay changes instantaneously by 10%, then an error is flagged indicating that something has caused the DLL to go out of specification. The operation for the system transient monitor (Figure 4.20) is described by the waveforms of Figure 4.21, and is discussed further in Appendix B.

4.4 DLL Layout

The DLL was implemented in a 150-nm partially depleted silicon-on-insulator (PDSOI) Honewell process. All of the cells were laid out by hand, including the digital. The layout is shown in Figure 4.22. The complete dimensions of the DLL are 2 mm $\times$ 0.6 mm.

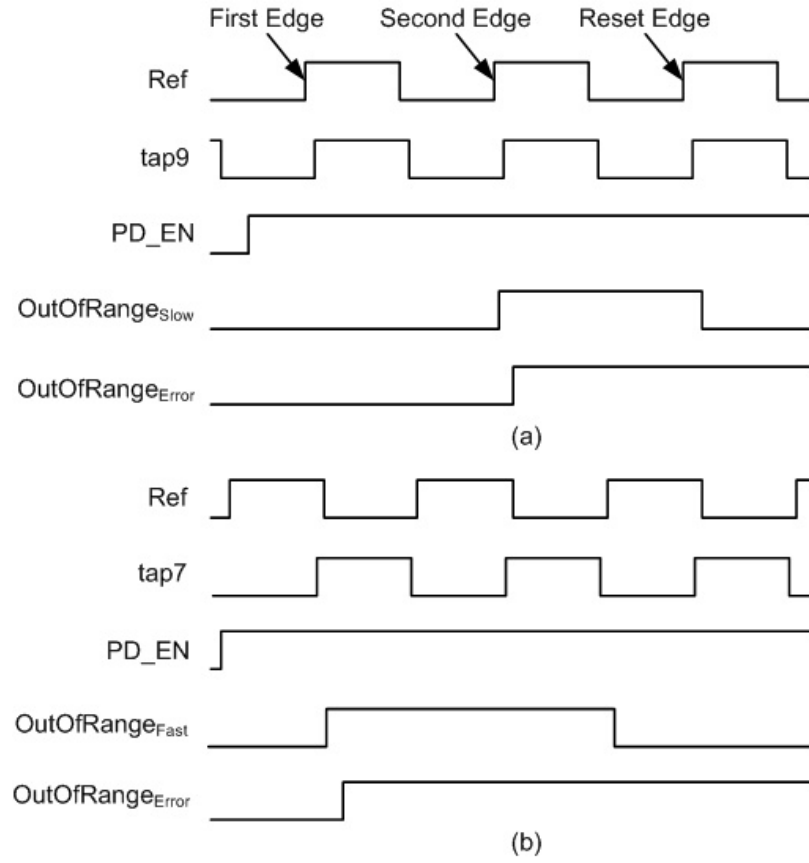


Figure 4.21: Waveforms of the system transient monitor where (a) is the slow case, and (b) is the fast case.

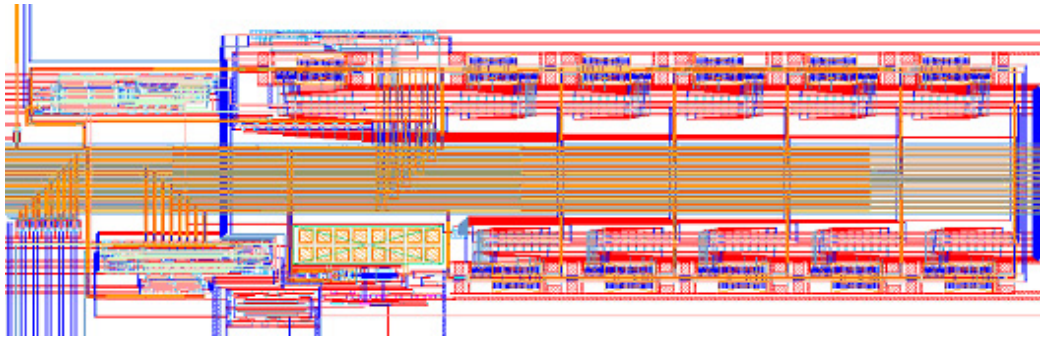


Figure 4.22: Layout of the self-calibrating DLL.

4.5 Full DLL Simulations

Simulations have already been shown for many of the DLL sub-components throughout this chapter, but the complete self-calibrating DLL needed to be simulated too.

Each simulation shown below takes many days to complete, so it was impractical to simulate across all corners and temperatures, which is why the delay cell and VCDL were thoroughly simulated. For the shown simulations, the supply was left at the nominal 1.8 V, the current reference was set at 10 μA , and the lower reference voltage was set at 1.1 V.

Figure 4.23 is the simulation plot of the control voltage and coarse calibration bits for an input clock of 62.5 MHz. As after an initial reset, the control voltage begins to charges up to the lower reference voltage, at which point the coarse calibration circuit takes control. The calibration capacitors are added until the VCDL has become just slower than the reference clock, which for this case is a calibration word of 10100. Then the main DLL loop in closed, and the DLL locks just after 4 μs . Figure 4.24 shows all of the output clocks of the as well as the reference clock. It can be seen that the DLL did not go into a false lock state since the reference clock was delayed one clock period. It can be seen that there is a small amount of pulse shrinking of the output clock, but it was less that 6%. A closeup of these locked input and output clocks are shown in Figure 4.25.

The simulations in Figures 4.26 and 4.27 show the DLL locking to a 125 MHz reference clock. Figure 4.26 shows the control voltage, calibration bits, and the calibration done

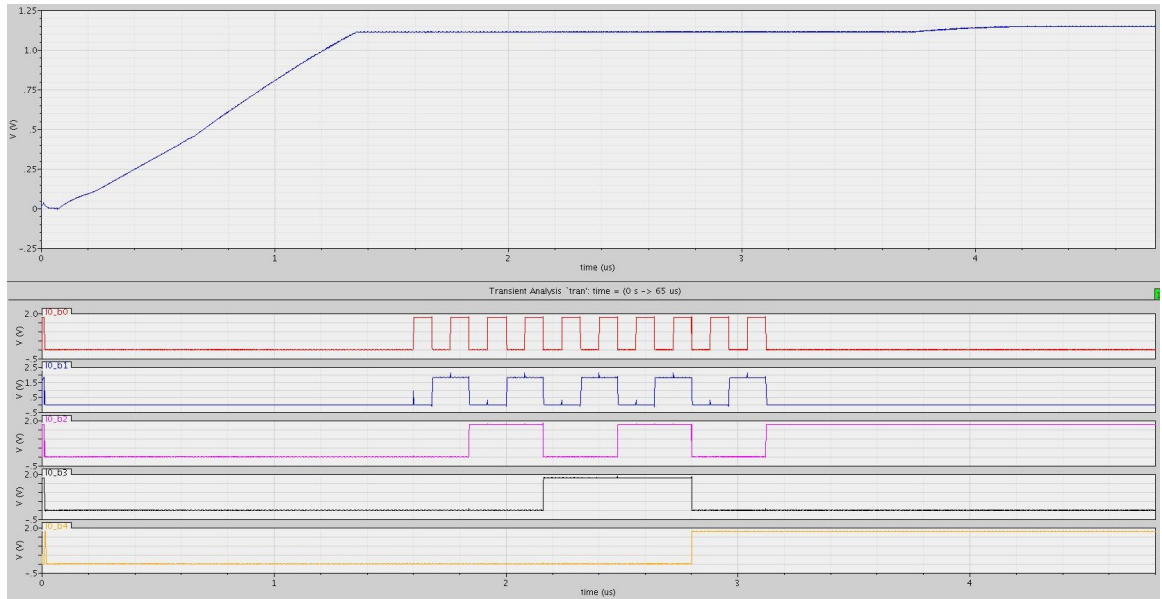


Figure 4.23: Simulation of the control voltage and coarse calibration bits for an input clock of 62.5 MHz.

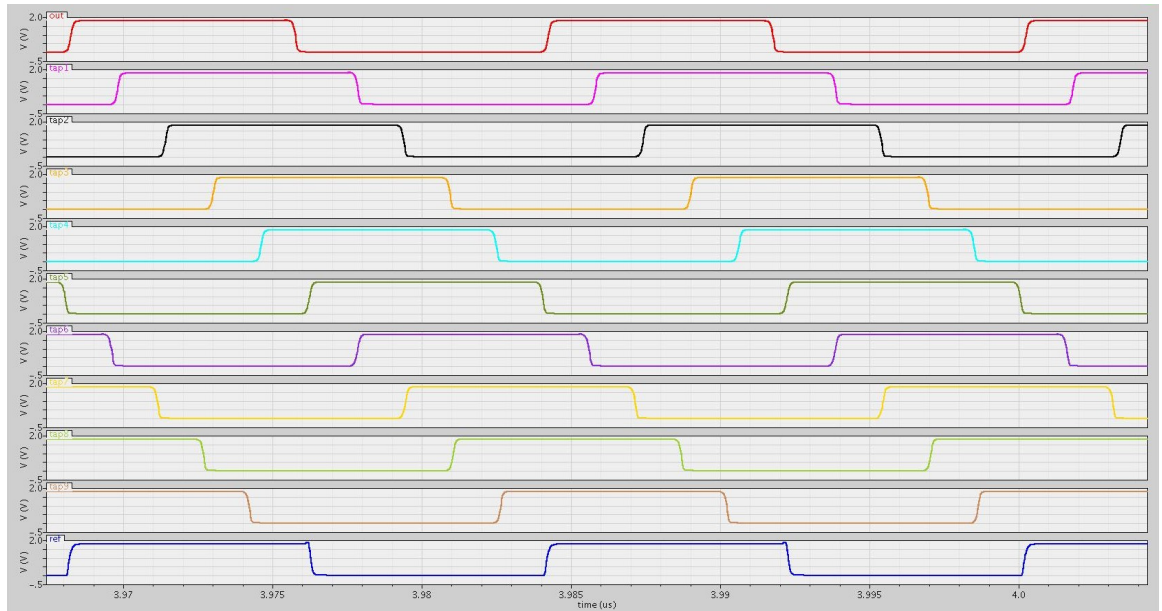


Figure 4.24: Plot of all of the reference clock and all of the DLL output clocks for an input clock of 62.5 MHz.

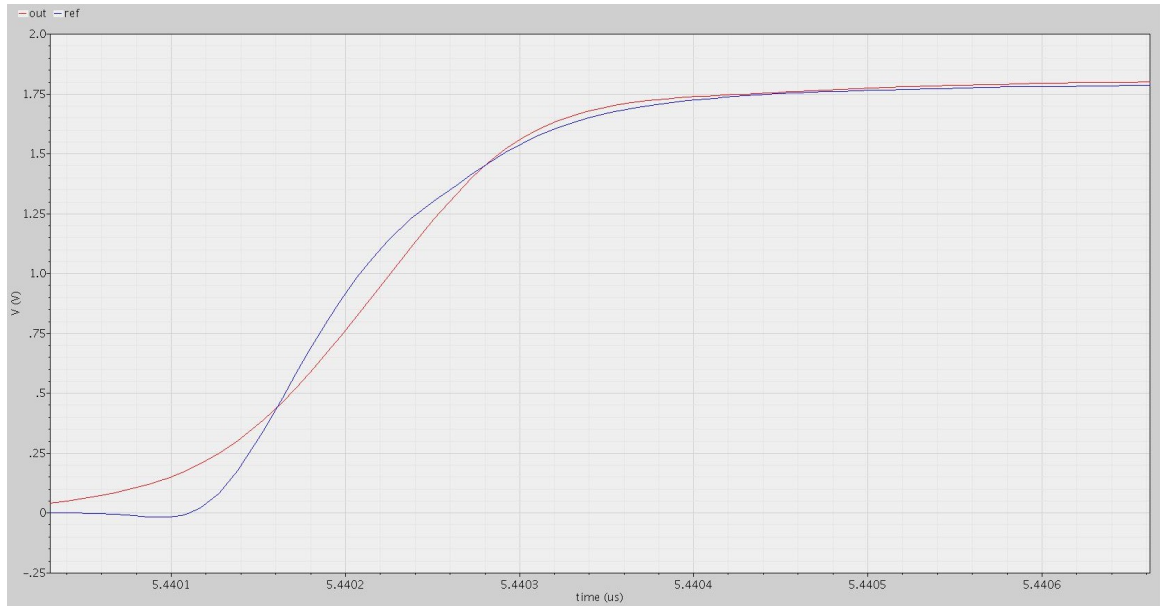


Figure 4.25: Closeup of the locked input and output clocks for an input clock of 62.5 MHz.

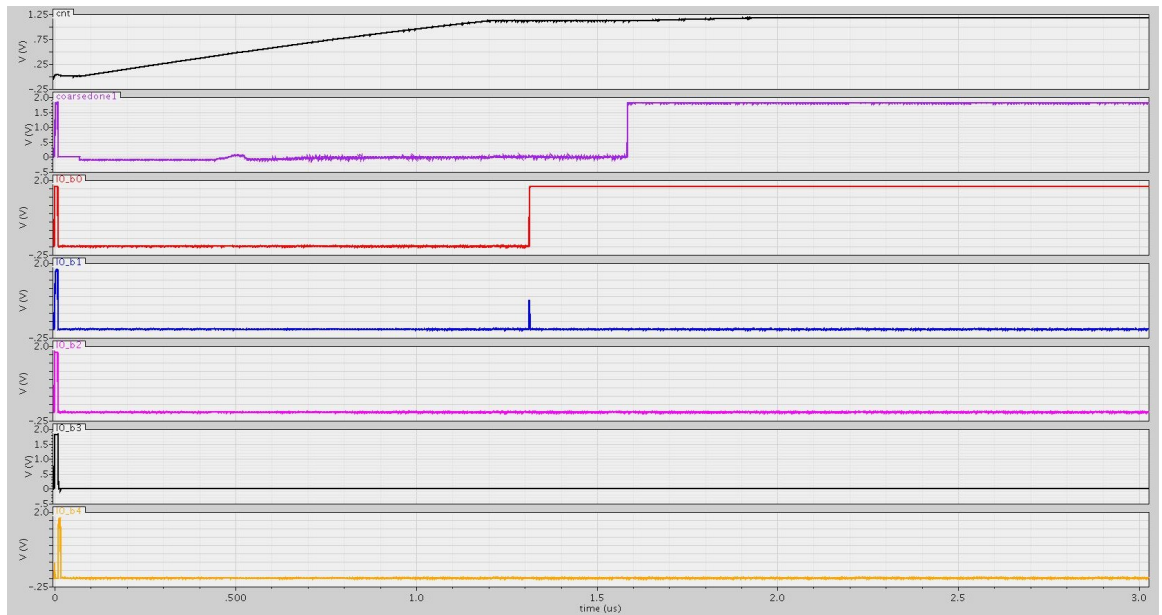


Figure 4.26: Simulation of the control voltage, calibration done, and calibration bits for an input clock of 125 MHz.

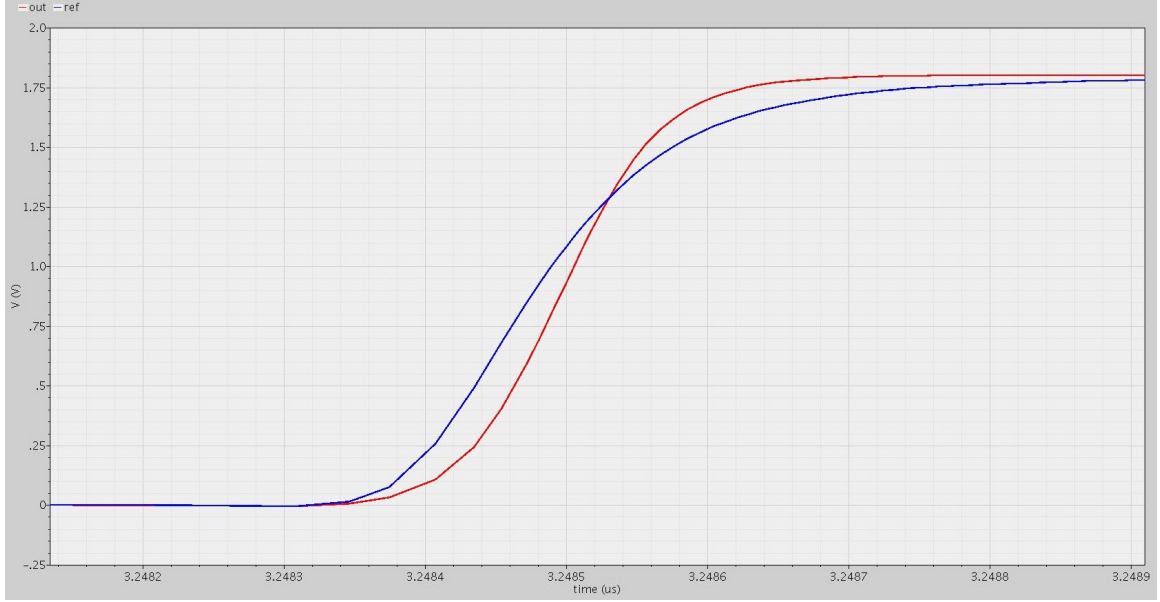


Figure 4.27: Closeup of the locked input and output clocks for an input clock of 125 MHz.

signal (not shown in Figure 4.23). The plot follows the same trend as Figure 4.23, except now the calibration word is 00001. The calibration done bit is also included for completeness. Figure 4.27 is the closeup of the locked reference and output clocks.

4.6 Summary

A self-calibrating DLL has been designed that includes two major blocks: a coarse calibration block and a system transient monitor. These proposed systems create a robust DLL that can be used in extreme environments.

The coarse calibration allows a multiphase analog DLL to lock in the low gain region for a temperature range of -125°C to 125°C , supply range of 1.4 V to 1.8 V, and a lock range of 90 MHz to 140 MHz. This digital calibration technique lends itself to be easily ported across different processes, while minimizing the effects of PVT. For extreme environment

applications, the coarse calibration will help ensure that the DLL will be able to lock under a variety of non-ideal conditions. This method also allows the DLL to lock faster compared to the standard phase detector locking mechanism.

The fine calibration is able to calibrate the individual delay cells within 12 ps from one another. Whereas the proposed technique may not produce lowest tap-to-tap delay error compared to the prior art, but the technique provides a simple technique that uses minimal area, does not require any complex computations, and can be easily ported across technology nodes.

The proposed system transient monitor will produce an error in the event that some environmental factor (change in the reference clock frequency, single-event transient, etc) has caused the DLL output to go out of lock. This digital monitor can be used to recalibrate the slow locking DLL (due to low loop gain) to lock faster by means of the proposed coarse calibration.

Chapter 5

Self-Calibrating DLL Test Results

5.1 Introduction

To verify the operation of the multiple parts of the described self-calibrating DLL, it was fabricated in a commercial 150 nm partially depleted silicon-on-insulator process. This chapter will provide details of the test setup, the results of these tests, including a discussion and explanation of these results.

5.2 Test Setup

The two main systems of the DLL, the coarse calibration and transient monitor, were verified via two separate tests. This section will describe the tests, and will provide details to the test setup including equipment used and test boards.

5.2.1 Coarse Calibration Test

The main function of the coarse calibration is to minimize the effects of global process, voltage, and temperature (PVT) shifts, thereby tailoring the DLL for extreme environment applications. Only a limited number of samples will be available, so it was not possible to verify global process shifts, however the voltage and temperature effects were thoroughly tested. The following describes how the coarse calibration system was tested:

- The DLL will be tested across a temperature range of -125°C - 175°C at 25°C steps.
- At each temperature step, the DLL supply voltage will be incremented from 1.6 V - 2.1 V in 0.1 V steps, where the nominal supply is 1.8 V.
- At each temperature and voltage step, the lock range of the DLL will be verified.
- The DLL operation will be verified above 125°C to test the extreme high temperature capabilities, however due to testing limitations this will only serve to demonstrate the potential of the DLL.
- Using a 1.8 V supply at room temperature, the input reference will be incremented in small steps to uncover any frequencies where the proposed technique may have difficulty locking, thereby giving detailed information about the coarse calibration.

5.2.2 Transient Monitor Test

This system's purpose is to watch for any event that might cause the DLL output to shift by some large amount such that it would take a long time for the analog loop mechanisms to relock the DLL. One thing that would cause a large transient in the DLL would be if the DLL control voltage had a sudden drop or increase such as a radiation strike. This was implemented by using two BJTs that will either pulse the control voltage high or low, depending upon which device was switched into the control voltage. The lock signal should

go high once the conditions described in Section 4.3 are met. Normally the lock signal would have been used as the trigger signal, but there was a design issue that would always cause the output to latch after some period of time that will be discussed later. Instead a delayed version of the system reset signal, which has been delayed long enough to ensure that the DLL has been locked, is used to trigger the oscilloscopes. The basic transient monitor test circuit is shown in Figure 5.1.

5.2.3 Test Boards and Equipment

To facilitate these tests, the self-calibrating DLL needed the following features:

- Tunable regulated supply voltages
- Two reference voltages
- Reset signal
- Tunable input reference clock

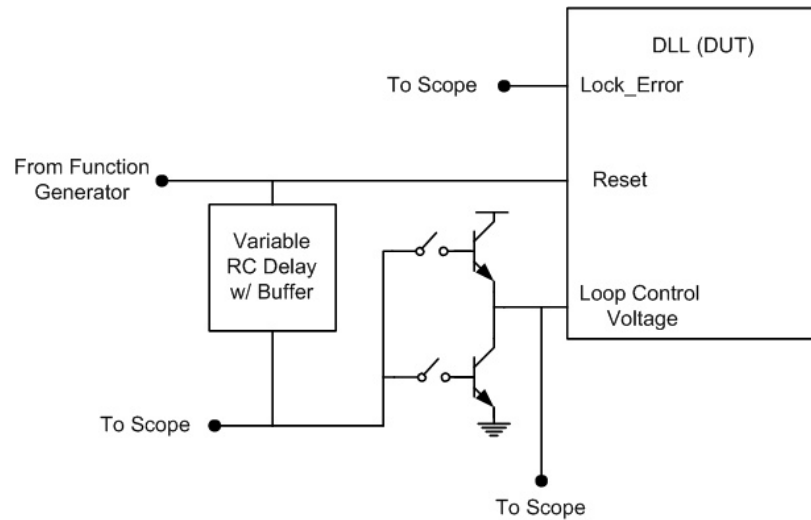


Figure 5.1: Circuit used to test DLL transient monitor.

- Reference current
- Oscilloscope trigger

Figure 5.2 is a block diagram of the test setup. The DLL consisted of four separate supply rails and grounds (phase detector, VCDL, digital logic, and analog cells), which were biased with LM317 regulators that had a potentiometer in the feedback path to allow for supply voltage adjustments. The reference voltages were generated using an Agilent triple output DC power supply. The current reference came from a Keithley 2400 series current/voltage source meter. All of the supplies and references were filtered with multiple bypass capacitors. The reset signal was produced using the pulse function of an Agilent function generator. The input clock was generated using a Lecroy 9211 pulse generator. The oscilloscope trigger was generated by delaying the reference clock with a simple RC delay with a buffer to reconstruct the signal. Actually the inverse of the reset was delayed as the DLL will not start operating until after the reset is deasserted.

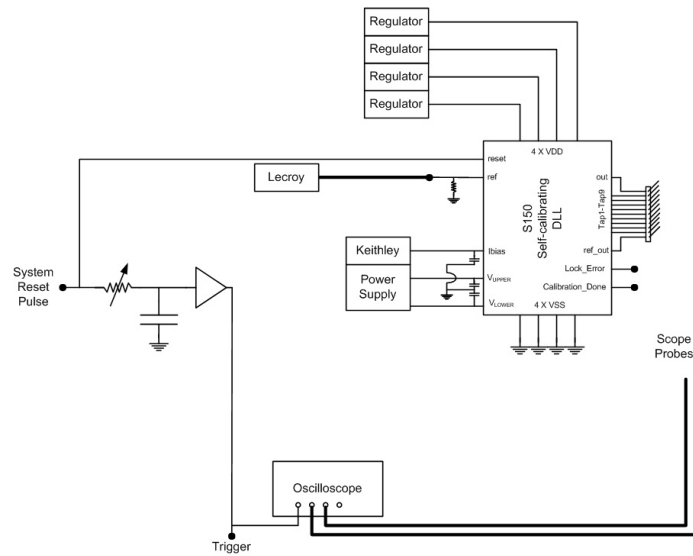


Figure 5.2: Block diagram of the test setup.

To carry out the temperature tests, a Delta Design 9023 environmental chamber was used. However, this chamber is closed during operation, which would make it difficult tune the supply voltages during extreme temperatures if the DLL and the regulators were on the same PCB in the chamber. Therefore a two PCB design was implemented. One board contained all of the supply, reference, and control circuits. The second PCB had the DLL with SMA connectors for the clock signals. The DLL was packaged in a 130 pin CQFP (ceramic quad flat pack) package, and was surface mount soldered to the second board. Each reference and supply pin had one surface mount bypass ceramic capacitor, and one leaded ceramic capacitor soldered (which was needed for operation) close to the package pin. All of the DLL taps were bonded out to pin headers for monitoring purposes. The two boards were connected together using right angle male and female pin headers. The simplified schematics of the test boards are shown in Figures 5.3 and 5.5, and the PCB layouts are shown in Figures 5.4 and 5.6. These are FR4 four layer boards with the the inner layers for supply and ground planes. The actual populated boards are shown in Figures 5.7 and 5.8. The full test setup is shown in Figure 5.9.

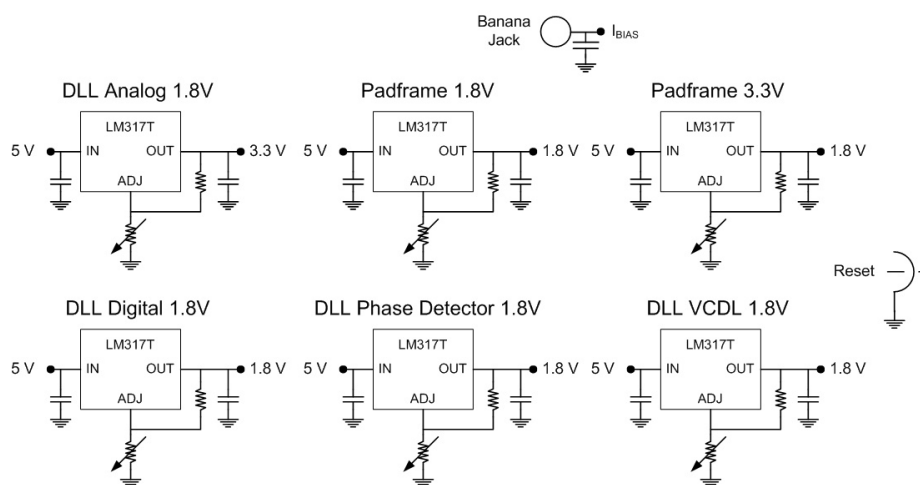


Figure 5.3: Simplified schematic of the bias board.

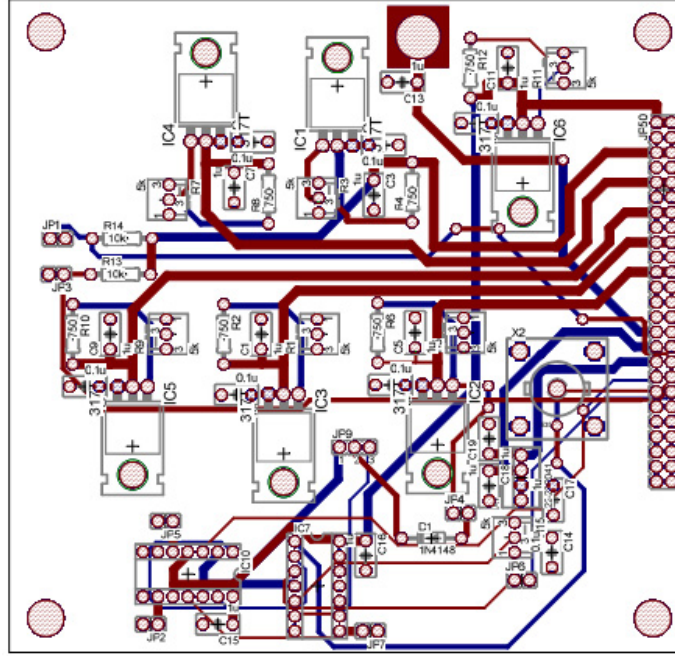


Figure 5.4: PCB layout of the bias board.

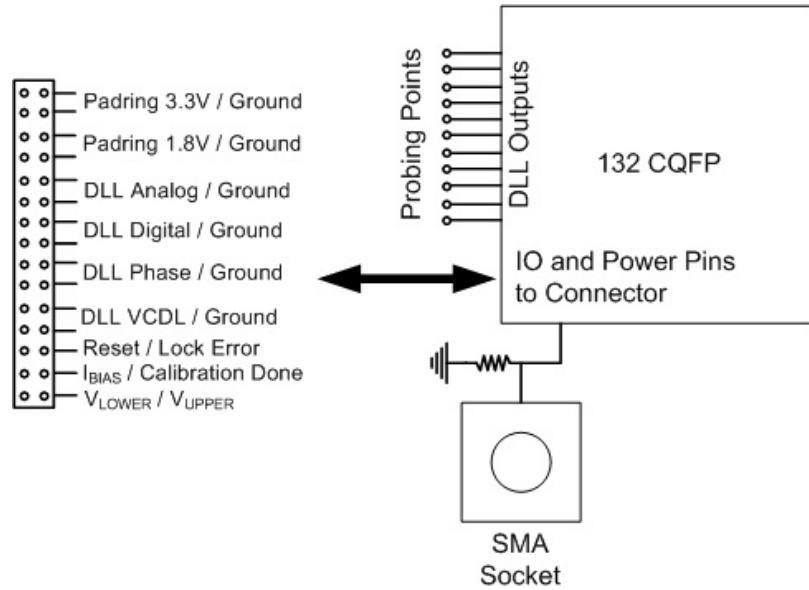


Figure 5.5: Simplified schematic of the DLL board.

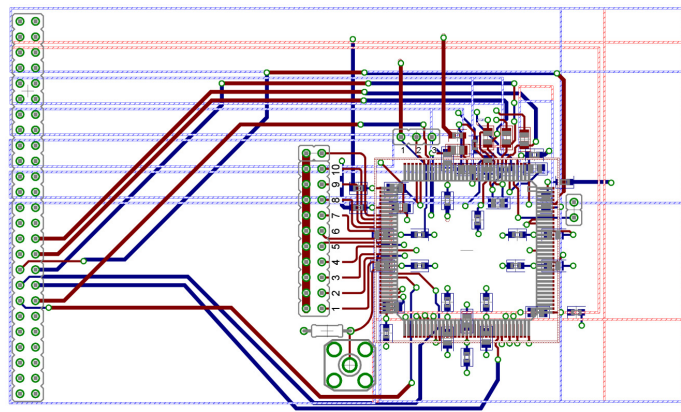


Figure 5.6: PCB layout of the DLL board.

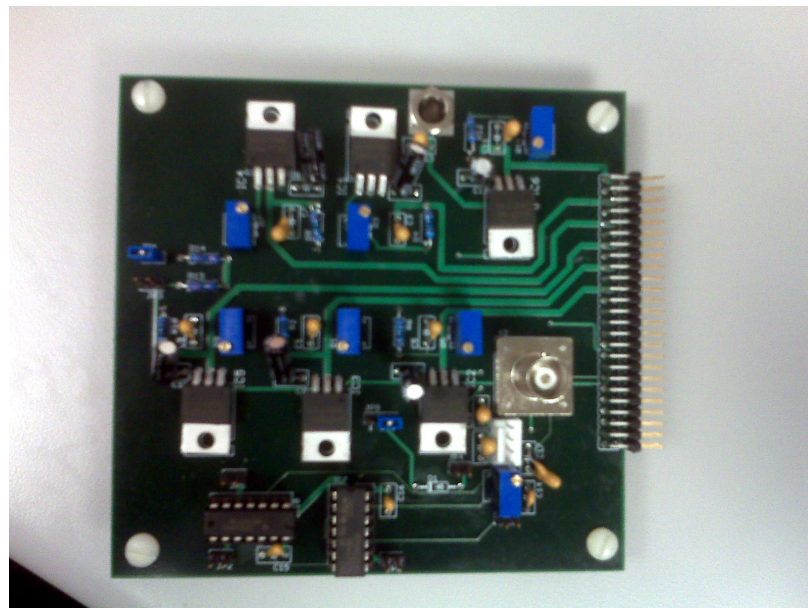


Figure 5.7: Actual bias board.

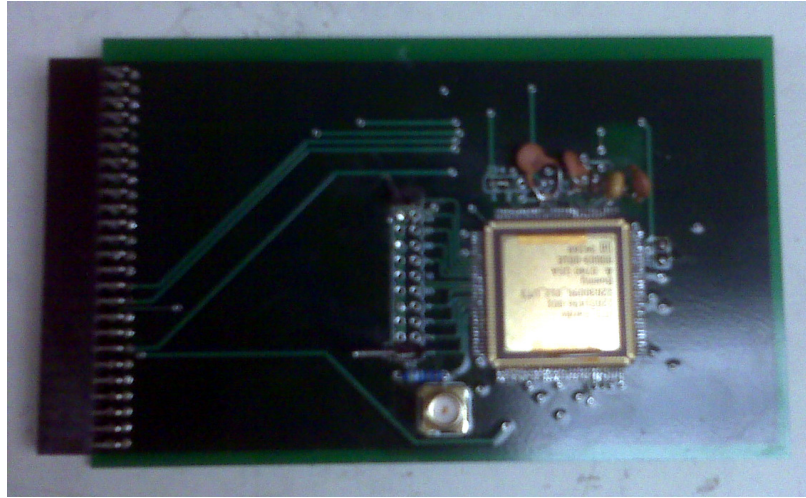


Figure 5.8: Actual DLL board.

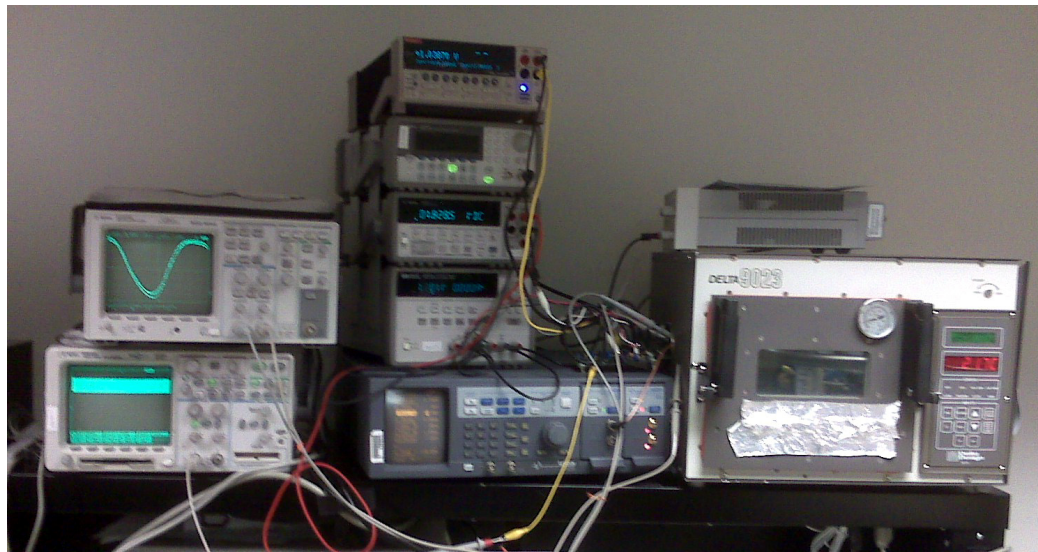


Figure 5.9: Full characterization setup.

5.3 Test Results

Using the above test description, both blocks of the self-calibrating DLL were characterized. This section will present these results, and will discuss the results as well as any discrepancies when compared to simulations.

5.3.1 Coarse Calibrations Test

The self-calibrating DLL operation was verified by applying varying input clock frequencies and monitoring the output as well as the control voltage. At first no results were obtained due to poor bypass filtering. Even after applying generous filtering, the regulated supplies varied 300 mVpp (average of 180 mV). Being that this DLL employed single-ended rail-to-rail delay cells, the supply rejection was far from desired. This also applies to the charge pump. This means that the jitter at the output of the would be higher that would be expected. However, even with the less than acceptable supply noise, the self-calibrating DLL was still able to lock. Figure 5.10 demonstrates the locking process of the self-calibrating DLL. This control voltage was for a 110 MHz input clock. The oscilloscope had averaging set to 256 samples to produce a cleaner waveform, otherwise fluctuations of 250 mVpp can be seen. With lower reference voltage set to 1.0 V, notice that after the reset goes low, the control ramps up close to the reference value. At that point, the phase detector is disabled to keep the loop in an open state, which causes the control voltage to hold until the coarse calibration finishes. Upon completion, the phase detector is enabled to allow the loop to finish locking in an analog manner. This was repeated multiple times to verify that the phase detector enable circuit of Section 4.2.3 worked properly. The time for the coarse calibration to complete and the DLL to lock was approximately 12.5 s.

As a demonstration of the locked signal, Figure 5.11 shows the DLL locked to a 75 MHz signal at room temperature. An accurate jitter measurement was not obtained as the

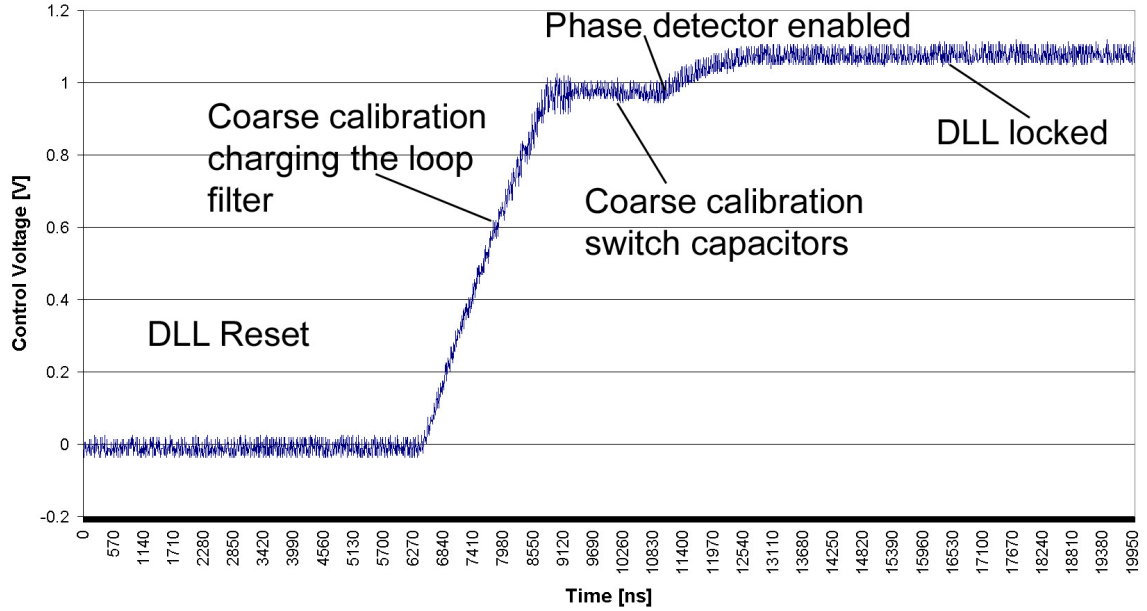


Figure 5.10: Self-calibrating DLL control voltage for an input clock of 110 MHz.

equipment did not have that capability, but it was determined less than 100 ps peak-to-peak. Considering that the Lecroy 9211 adds approximately 40 ps rms jitter and the presence of substantial supply noise, the jitter is less than the 100 ps result, which is to be expected as the VCDL was designed to exhibit low gain.

After verifying that the DLL was operating, the coarse calibration operation was verified. First the lower input range of the DLL was found by reducing the input clock frequency, applying a reset, and when the DLL was no longer able to lock, the lowest frequency was found. At this point the coarse calibration will have switched in all of the calibration capacitors. Now the reference frequency was increased in steps of 0.1 MHz until the DLL begins to lose lock. This is the upper range for that particular calibration code. When taking the data, if the was either unable to maintain lock, or was unusable, the DLL was classified to be out of lock. The DLL would now be reset at this upper frequency, and the upper and lower frequencies would would again be found. At some data points, the reset was

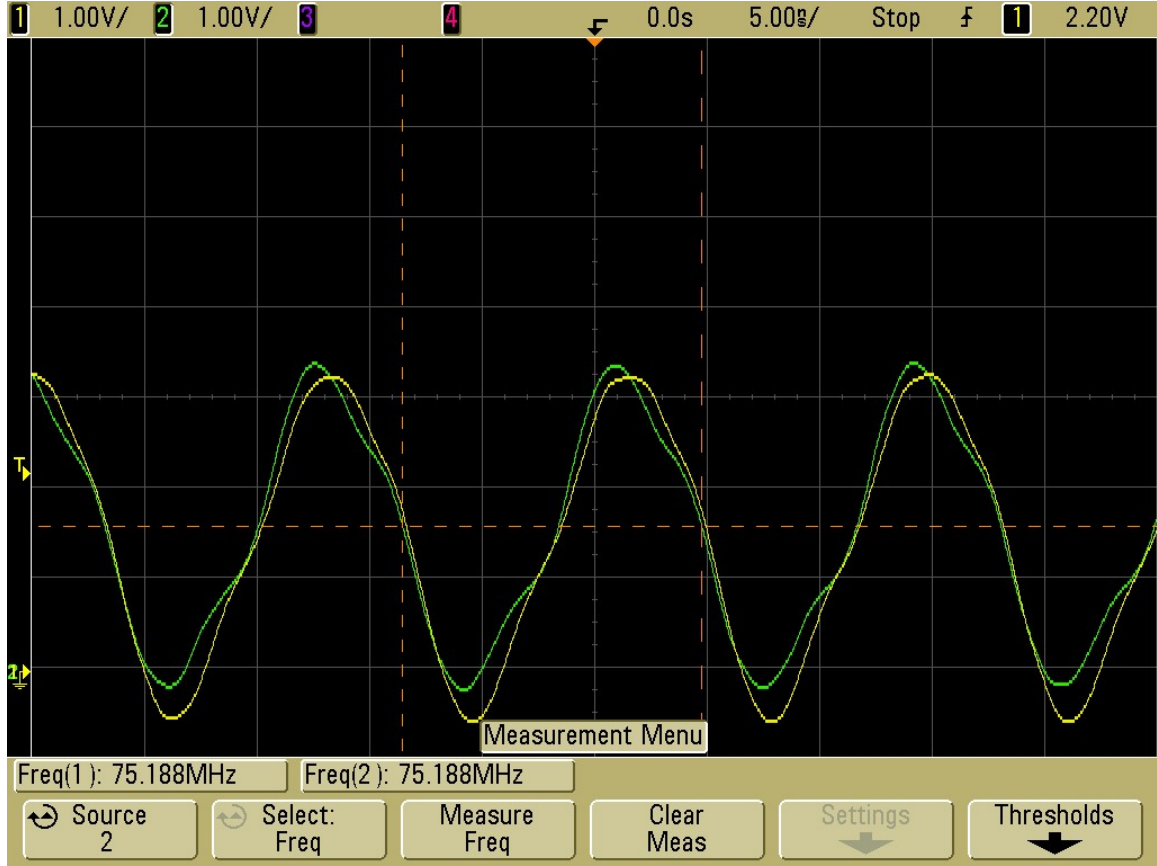


Figure 5.11: DLL locked to 75 MHz reference clock.

performed at a higher frequency if the it was found to be relocking to the same calibration code, or was at an unlockable frequency. This process was repeated until the DLL was at the upper end of its capture range, which would be where no capacitors were switched into the delay cell. The control voltage, which was also padded out, was monitored during all tests. The control voltage value was recorded for the upper and lower frequency for every code tested. This was tested on two samples and at two different sets of reference voltages (V_{LOWER} =[0.9 V, 1.1 V] and V_{UPPER} =[1.2 V, 1.35 V]). The data for Chip 1 is shown in Figures 5.12 and 5.13, and the data fore Chip 2 is shown in Figures 5.14 and 5.15. Keep in mind that the plots only show the upper and lower tuning frequencies of each code, and

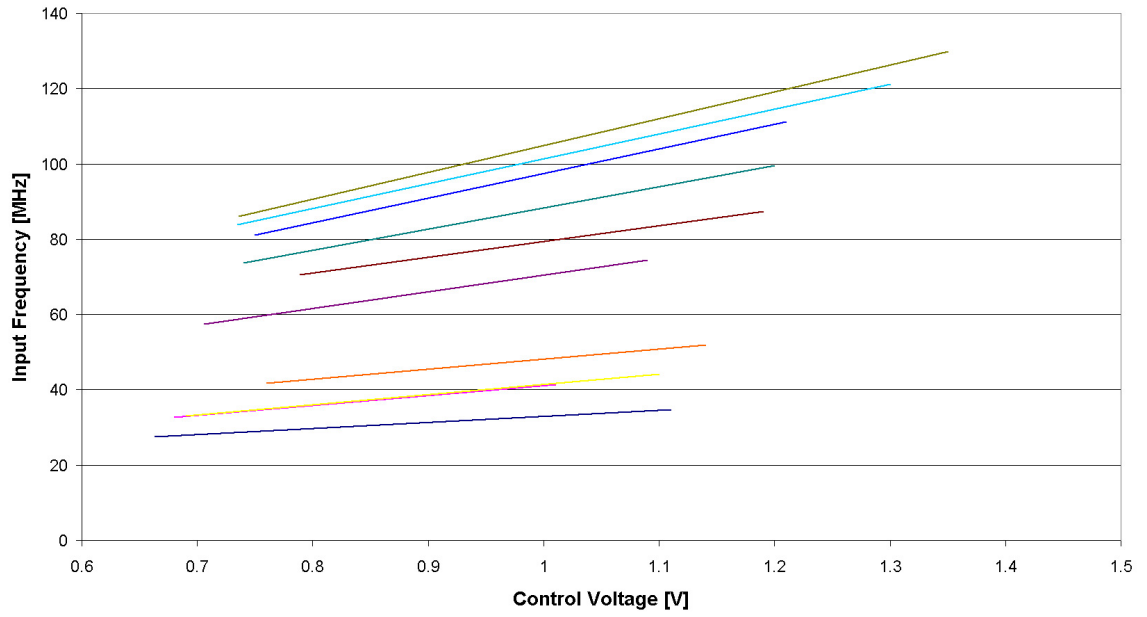


Figure 5.12: Coarse calibration ranges for Chip 1 with $V_{LOWER} = 0.9$ V and $V_{UPPER} = 1.2$ V at room temperature.

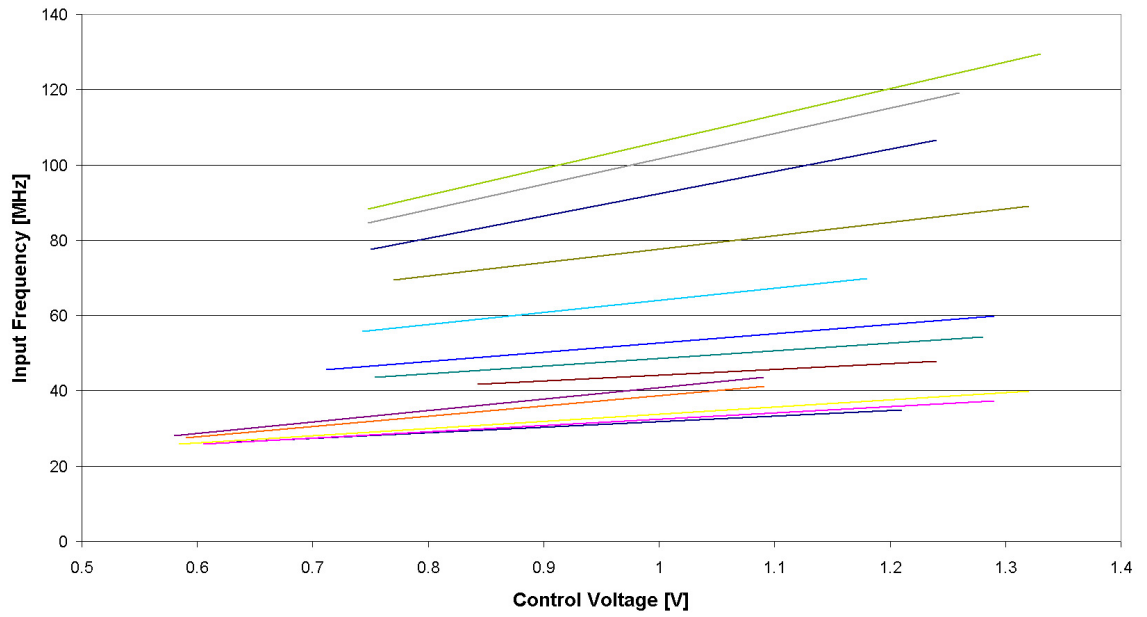


Figure 5.13: Coarse calibration ranges for Chip 1 with $V_{LOWER} = 1.1$ V and $V_{UPPER} = 1.35$ V at room temperature.

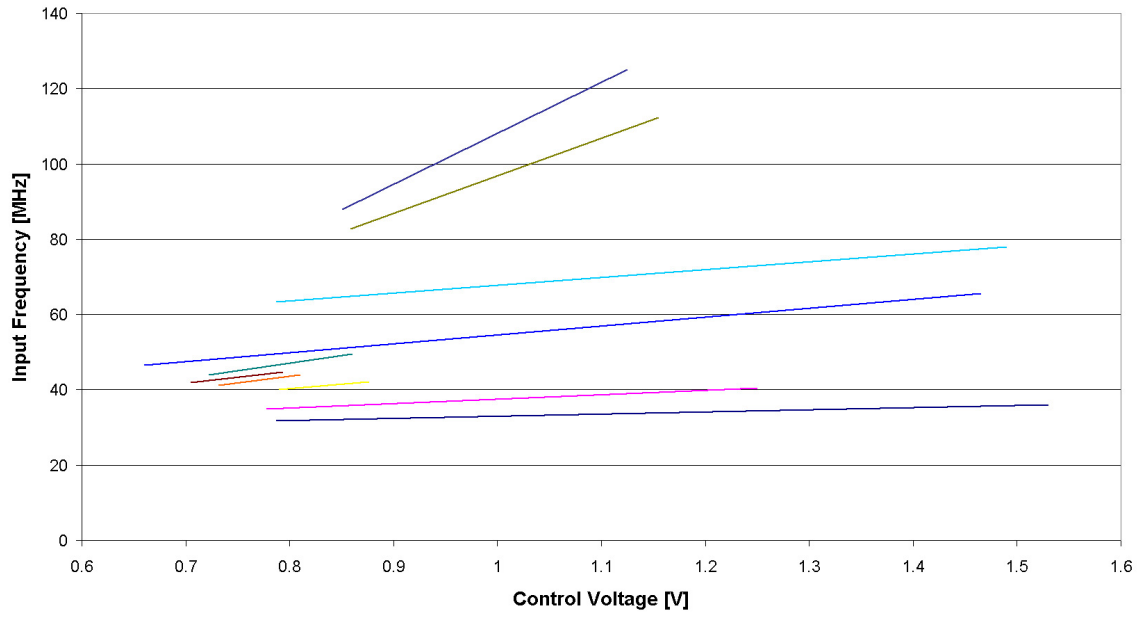


Figure 5.14: Coarse calibration ranges for Chip 2 with $V_{LOWER} = 0.9$ V and $V_{UPPER} = 1.2$ V at room temperature.

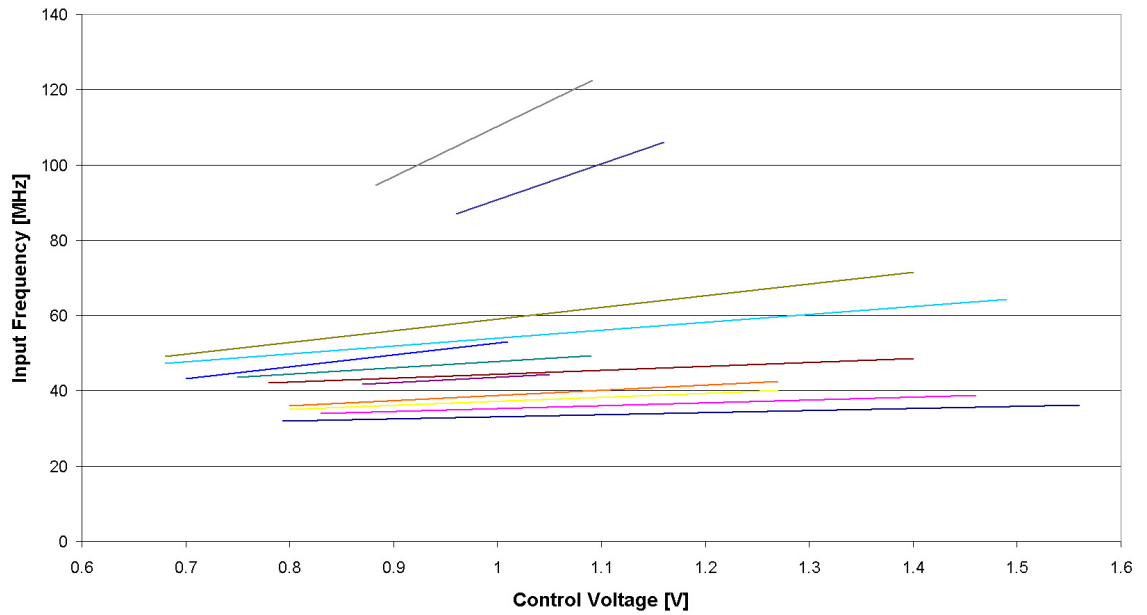


Figure 5.15: Coarse calibration ranges for Chip 2 with $V_{LOWER} = 1.1$ V and $V_{UPPER} = 1.35$ V at room temperature.

that a straight line was drawn between them only to provide a visual aid, which was found to be a reasonable assumption as demonstrated by the detailed plot (control voltage data taken at 0.1 MHz increments) in Figure 5.16. The result is similar for the other chip too.

The data provides promising findings as well as some problems. One problem is that not all of the lines are equal in length indicating that the VCDL tuning range has a discontinuity. However, this is not necessarily completely the problem. It was discovered that by adjusting the input duty cycle of the input clock, that the DLL would be able to lock again. This means that there is a problem with the implemented delay cell architecture. The complementary devices were added at the same node to mitigate the pulse shrinking issue associated with the non-linear capacitance of the MOSCAP. Simulations (both schematic and extracted) did not show any issues with the pulse size varying by a large amount (less than 10% across the entire tuning range), which could mean there is a possible modeling

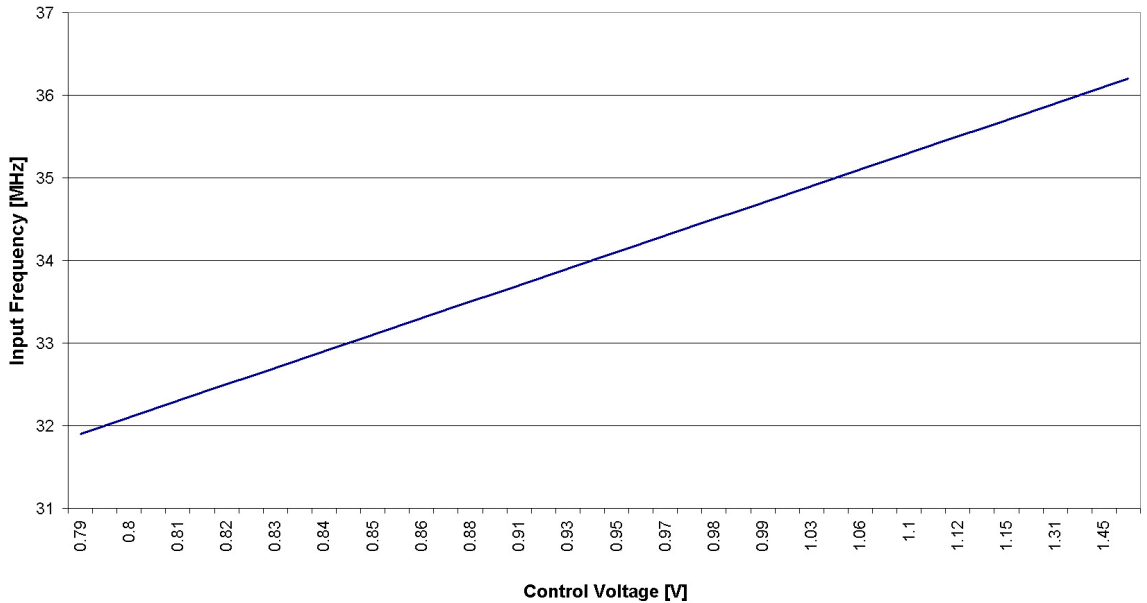


Figure 5.16: Measured detailed curve for digital code 11111 (all capacitors switched in) for Chip 2 with $V_{LOWER} = 1.1$ V and $V_{UPPER} = 1.35$ V at room temperature.

issue. Also, transmission gates were used to switch in the capacitors to the signal path of the delay cell. These switches look like resistors in the on state, which means that there exists an RC delay as opposed to only a current/capacitance delay. This was considered during the design of the delay cell, and it was found that by doubling the width of the switch as the capacitance area doubled would alleviate any non-linear code switching or pulse shrinking/growing concerns, which was not the case. A better approach would have been to turn on the gate on a MOSCAP whose source/drain node would be connected to the signal path as demonstrated by Baronti [39].

The next problem discovered was that there are frequency ranges that the DLL was unable to lock, which differed for each chip, that did not show up in any of the simulations of the VCDL run at those frequencies of concern. This too could be attributed to the RC filters of the switches or the pulse shrinking/growing problem. Some potential improvements to the VCDL will be presented in the next chapter that could help mitigate these problems.

Other reasons for the reduced capture range is the excessive noise on the supply and control node. From simulations the control voltage capture range would be reasonably valid from 600 mV to 1.4 V. However, when you couple the additional noise elements, into this, it would be expected for the range to be reduced. For example, assume the worst case for the peak-to-peak noise described above. Say the control voltage is locked at 1.1 V, and the supply is a nominal 1.8 V. If the worst case supply noise occurred simultaneously, the supply voltage could drop to around 1.65 V for some short duration. From simulations discussed in Chapters 3 and 4, this would reduce the tuning range of the VCDL, thereby causing the control voltage to swing to the rail, which would unlock the DLL. The same could be said if there was a large deterministic jitter component that could potentially cause the DLL output to consistently misalign with the reference clock at the phase detector.

The deterministic jitter is mentioned due to the design error eluded to earlier. Dynamic flip flops were used in place of their static flip flops throughout all of the logic, with

exception (thankfully) of the counter used for the coarse calibration switch registers. These dynamic flip flops were found to toggle high (after being reset low) within 400 s. There was an additional digital block that was added for trial purposes that would constantly run, even when externally disabled due to these dynamic flip flops. This block added additional noise base on the system clock (system was synchronous) into the DLL that would not have been there otherwise.

These invalid capture ranges are mostly likely due to a combination of these discussed occurrences, as opposed to just one alone.

Another finding is that the self-calibrating DLL responded better with $V_{LOWER} = 1.1$ V. Referring to the plots, the capture ranges were more uniform and larger in these plots. This tunable reference voltage would allow for DAC controlled tunability, which would be a useful feature in extreme environments.

One interesting finding is that more calibrations codes existed in the lower frequencies of the DLL capture range of the DLL. This is not surprising since the delay steps were set to be linear steps for time and not frequency.

Despite some of the above problems, the DLL does exhibit the capability to digitally calibrate to a capture range of 35 MHz to 120 MHz at room temperature for a nominal supply. In this case, too many digital steps were added to the delay cell, as there are large areas of frequency overlap between the transfer lines. However, since this for extreme environment application, this was intentionally done as to correct for any drain current variations (delay associated with the drain current to capacitance ratio) that would occur for temperature or voltage deviations.

The next test performed was the extreme environment test, which was described earlier in this chapter. The DLL board was placed in the temperature chamber and the temperature was set at 25°C, and the supply voltage was stepped from 1.6 V to 2.1 V in 0.1 V increments. The maximum capture range was not tested as done above, but rather the maximum and minimum frequencies were found. However, the DLL was spot checked at some the troublesome frequencies, and it was discovered that these dead ranges were smaller at colder temperatures. The temperature was then decrease in 25°C increments to −125°C, and then it was taken up to 175°C. The results are shown in Figures 5.17 - 5.28. The reference voltages were set to $V_{LOWER} = 1.1$ V and $V_{UPPER} = 1.35$ V for these tests. Figures 5.29 and 5.30 show oscilloscope captures of the input and output of chip 1 with a 2.1 V supply at 175°C for the upper and lower lock frequencies. The yellow is the input clock and the green is the output waveform.

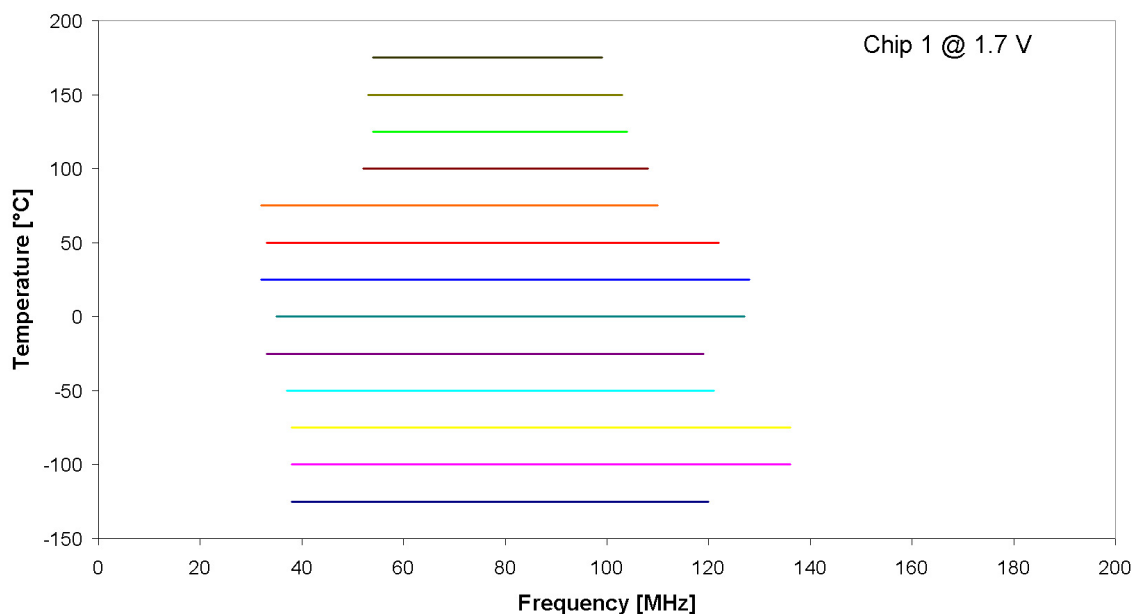


Figure 5.17: Measured temperature data for chip 1 at 1.7 V.

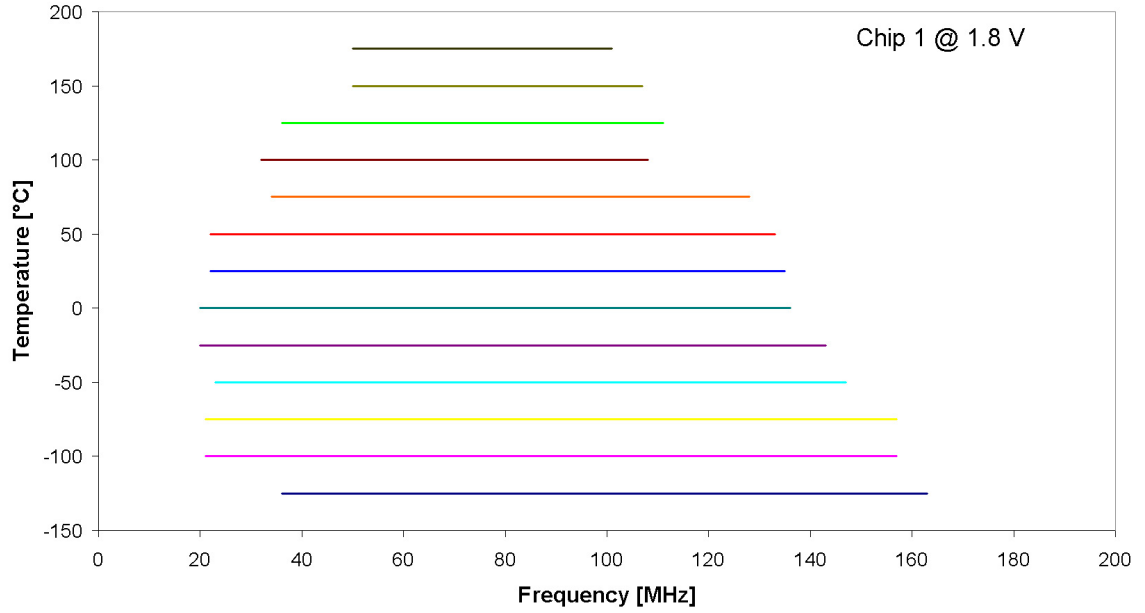


Figure 5.18: Measured temperature data for chip 1 at 1.8 V.

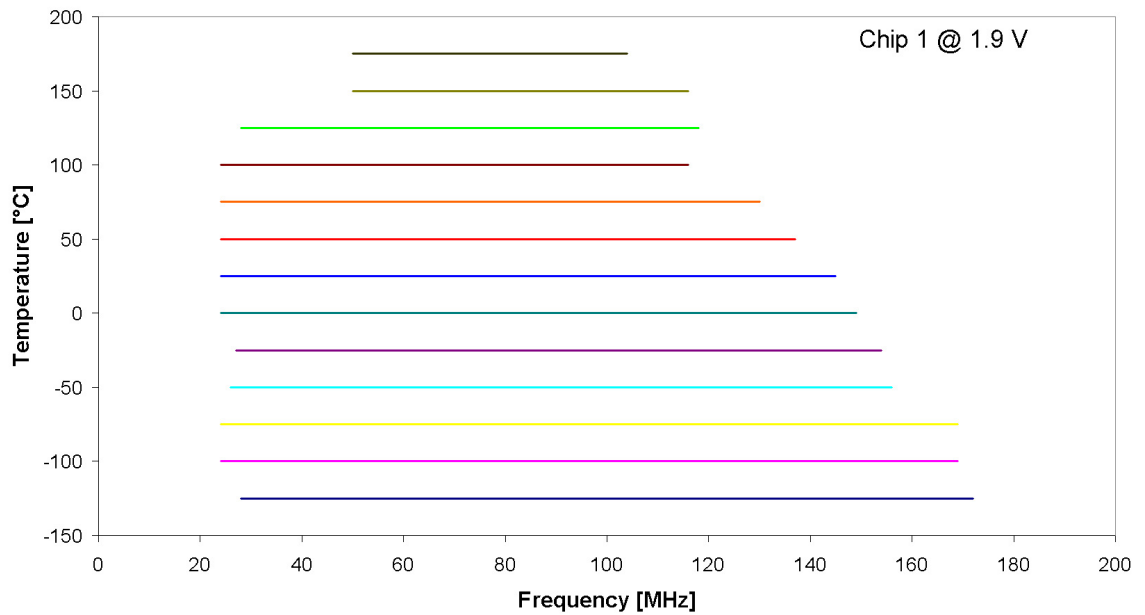


Figure 5.19: Measured temperature data for chip 1 at 1.9 V.

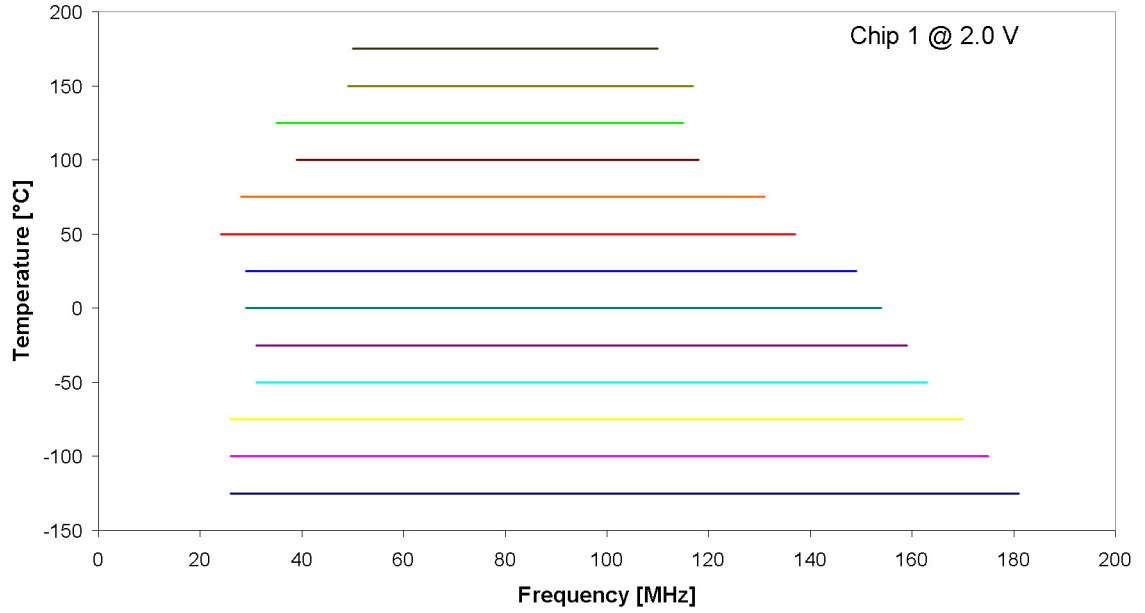


Figure 5.20: Measured temperature data for chip 1 at 2.0 V.

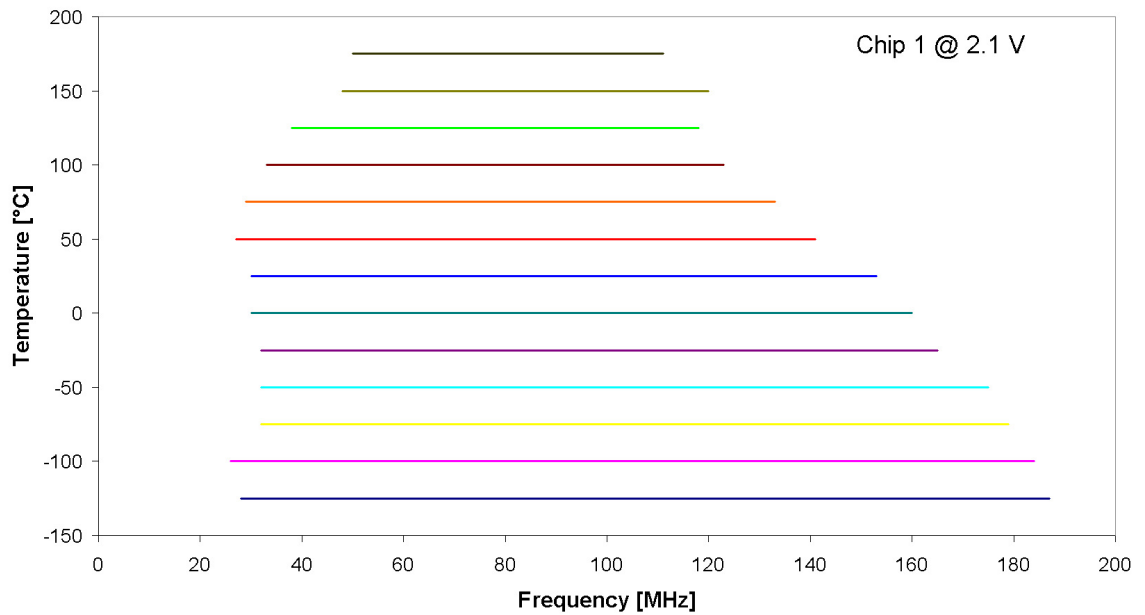


Figure 5.21: Measured temperature data for chip 1 at 2.1 V.

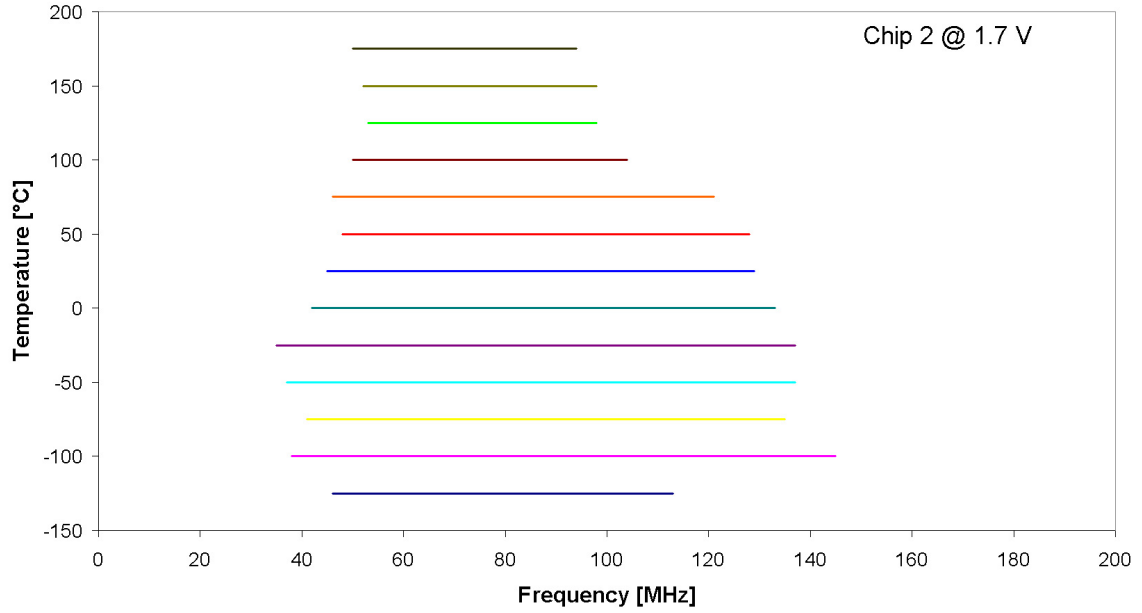


Figure 5.22: Measured temperature data for chip 2 at 1.7 V.

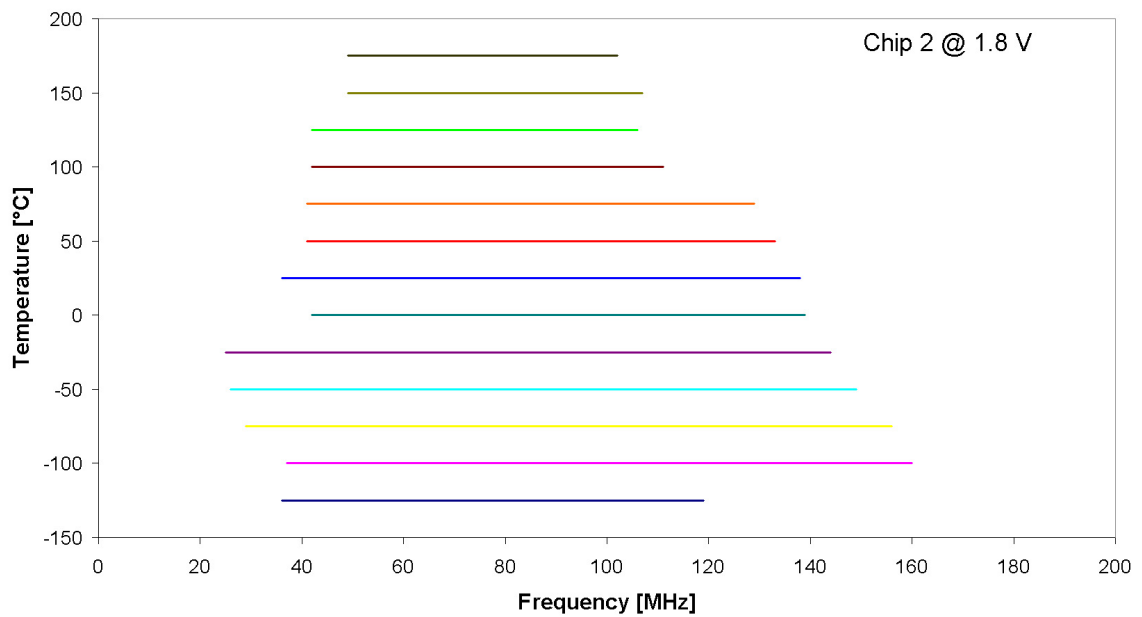


Figure 5.23: Measured temperature data for chip 2 at 1.8 V.

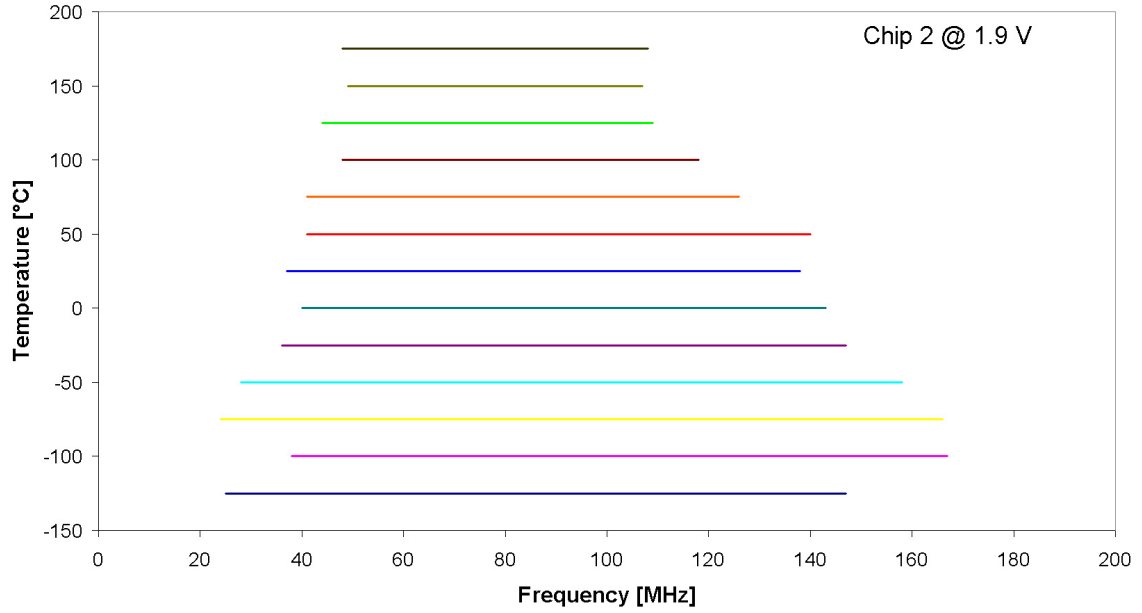


Figure 5.24: Measured temperature data for chip 2 at 1.9 V.

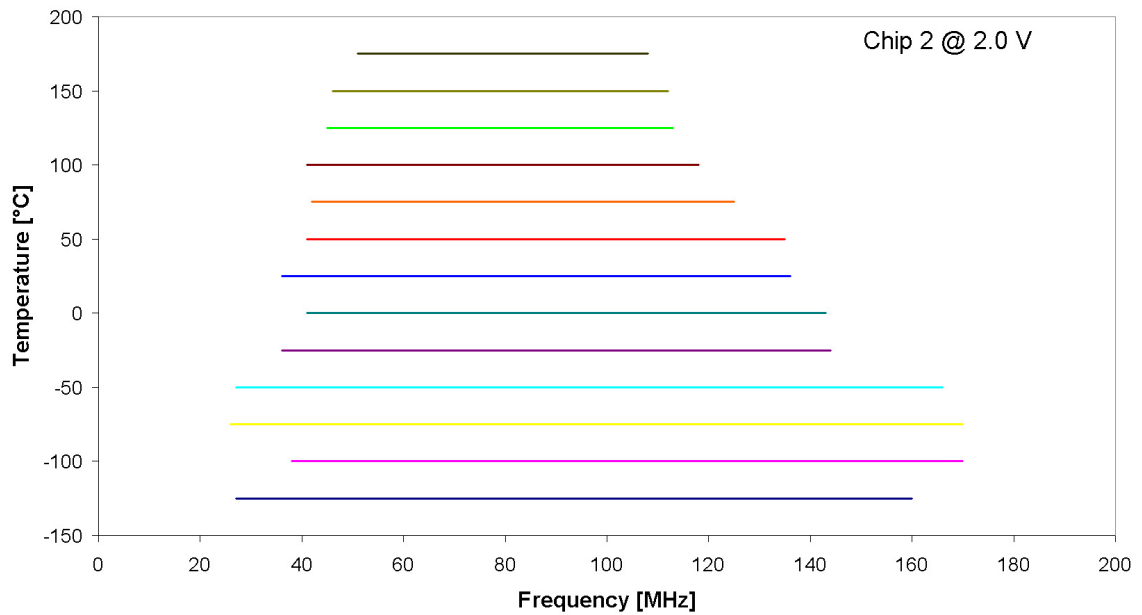


Figure 5.25: Measured temperature data for chip 2 at 2.0 V.

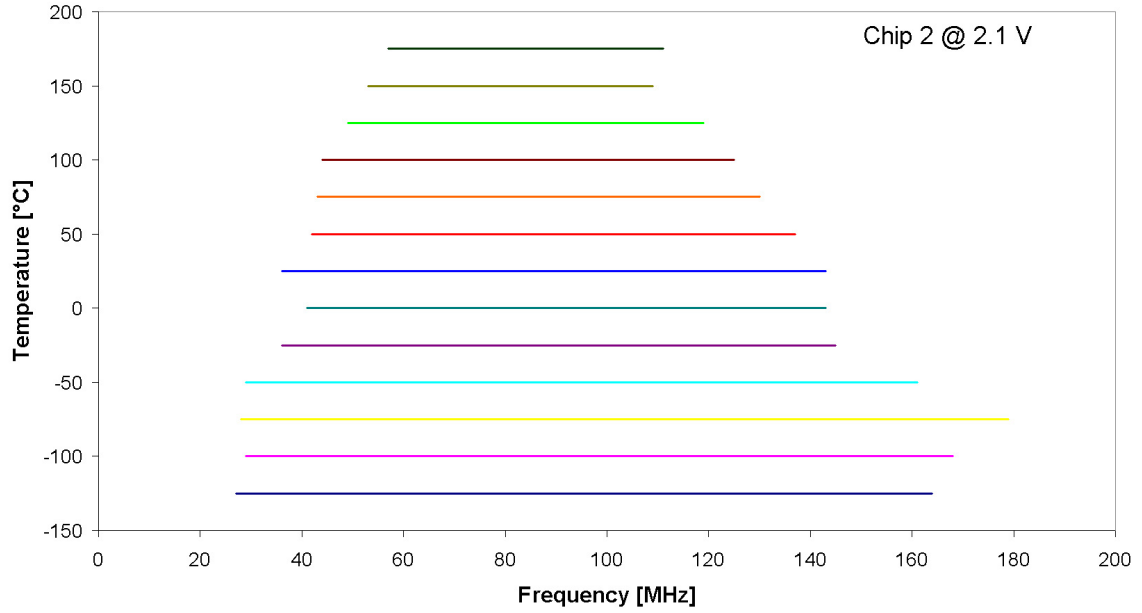


Figure 5.26: Measured temperature data for chip 2 at 2.1 V.

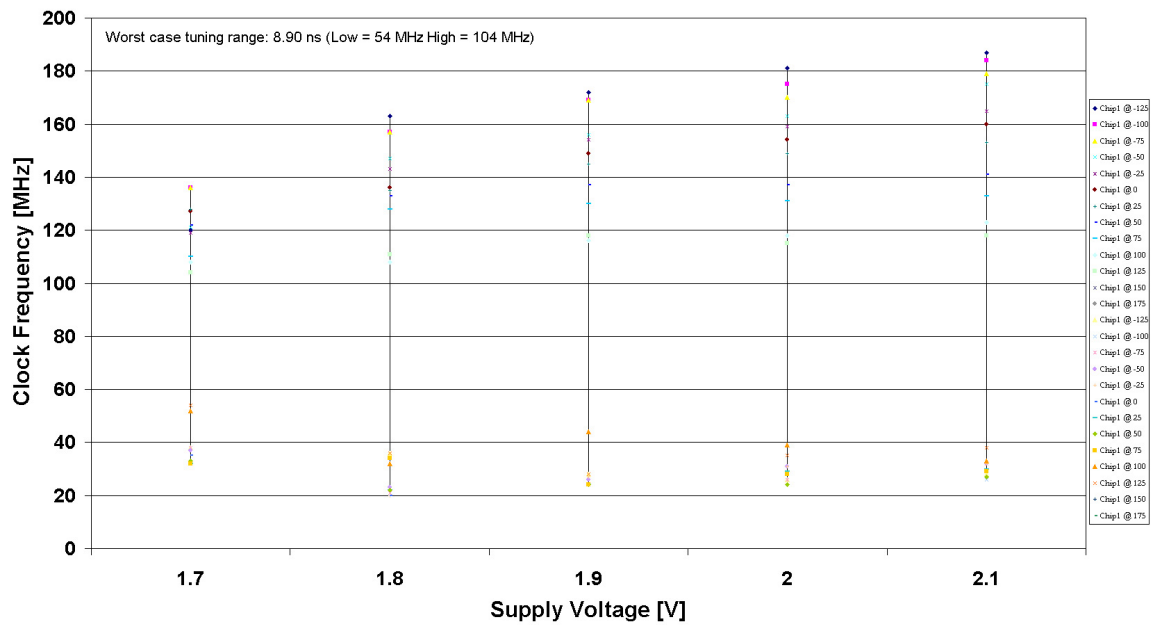


Figure 5.27: Cumulative temperature and voltage measured data for chip 1.

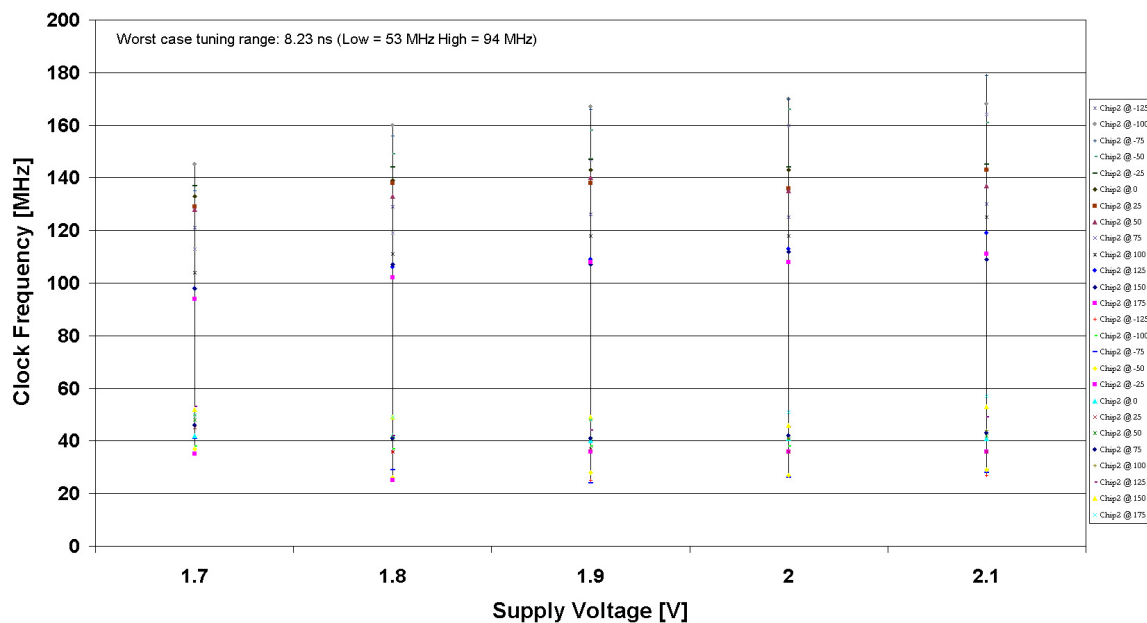


Figure 5.28: Cumulative temperature and voltage measured data for chip 2.

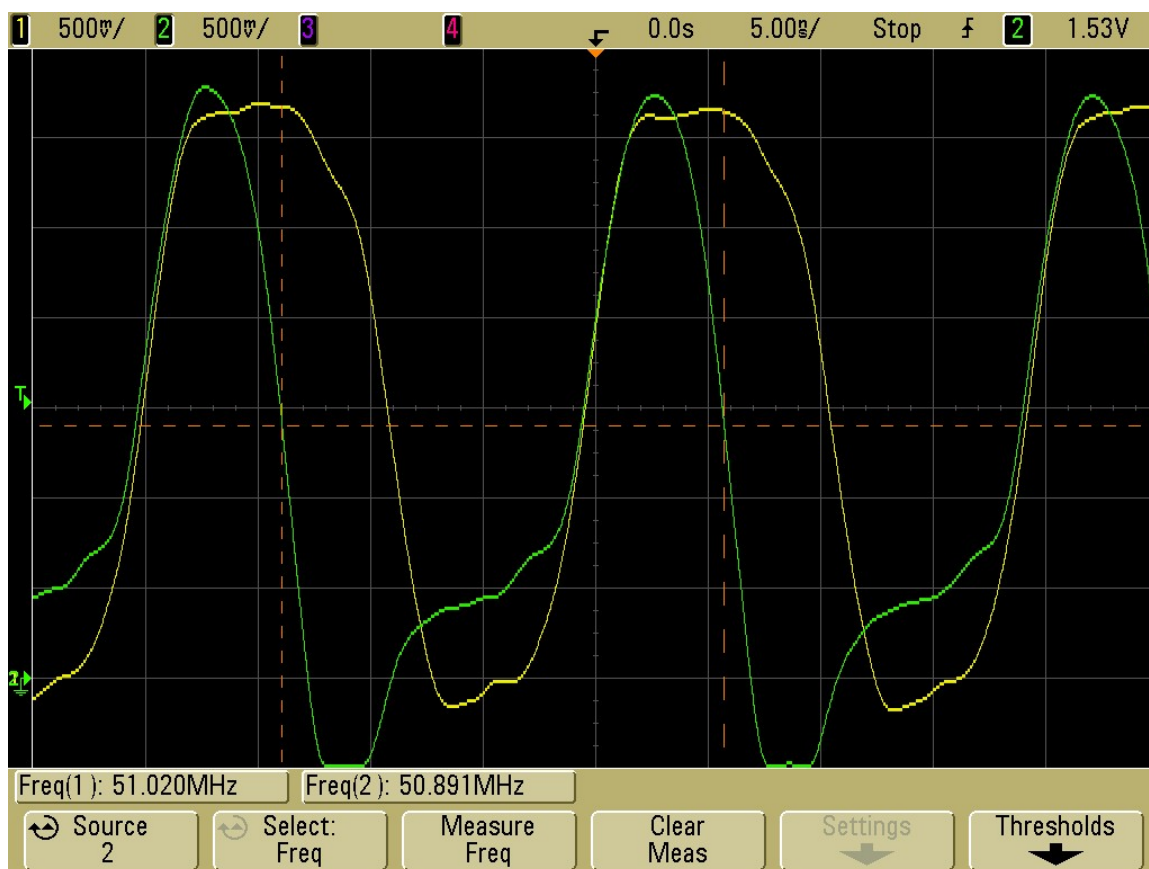


Figure 5.29: Locked DLL waveform at 51 MHz for chip 1 with a 2.1 V supply at 175°C.

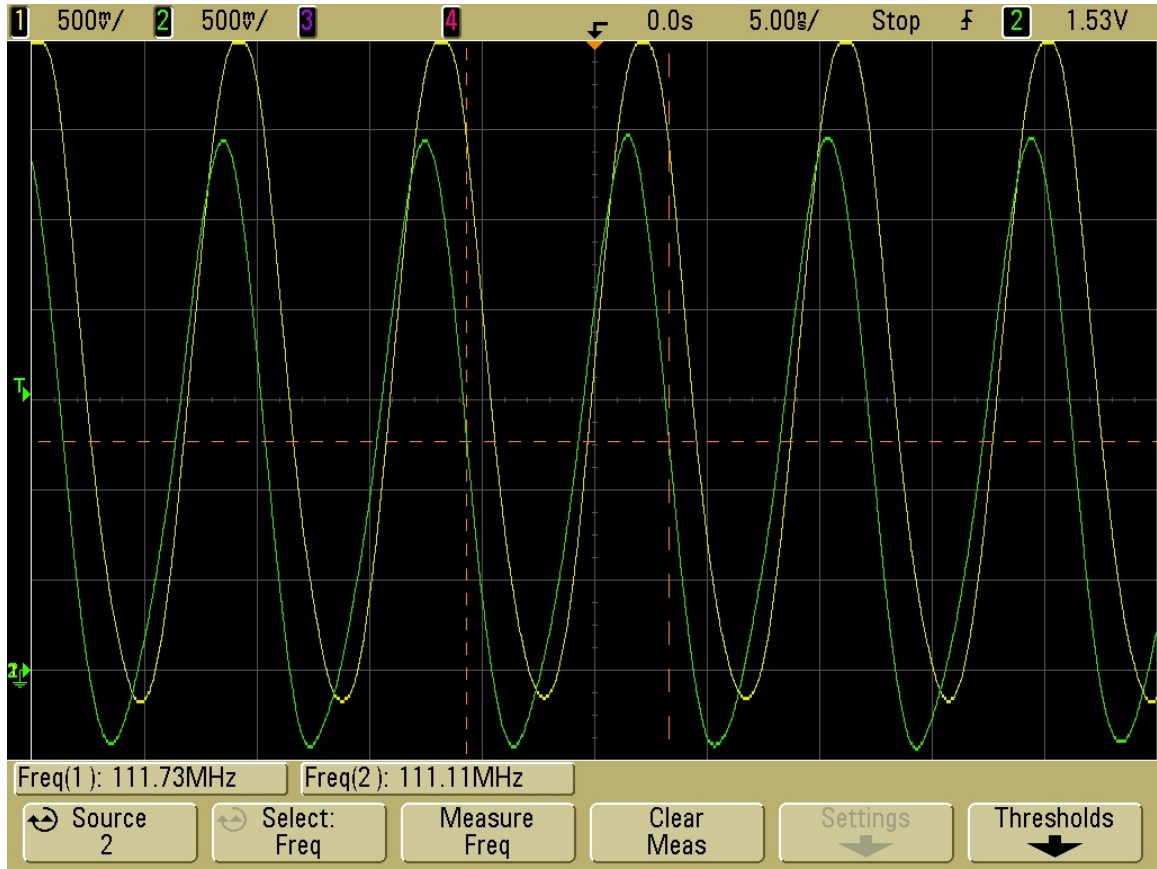


Figure 5.30: Locked DLL waveform at 111 MHz for chip 1 with a 2.1 V supply at 175°C.

Figures 5.17 - 5.26 present the DLL's tuning range across temperature for the individual supply voltages. Figures 5.27 and 5.28 show all of the data from the previous figures in a single plot to demonstrate the cumulative effect of temperature and supply voltage. As expected, the higher temperatures exhibit smaller capture ranges than the colder temperatures.

The first thing noticed is that there was no data for 1.6 V. This is due to the supply noise issues discussed earlier. There were some frequencies at lower temperatures that locked at 1.6 V supplies, but nothing that was reproducible. However, after subsequent tests with a more stable supply, it was found that the DLL could operate down to 1.4 V at 175°C.

The smallest capture range for both chips was found to be 54 MHz to 94 MHz (7.88 ns delay range), which was taken at 1.7 V supplies at 175°C. The largest capture range was found to be 26 MHz to 187 MHz, which was taken with 2.1 V supplies at -125°C .

These results follow the same trend as the demonstrated in the simulations, but were at slower frequencies due to parasitics (even though it was simulated with parasitics). The desired tuning range was 90 MHz to 140 MHz (3.97 ns delay range), which was smaller than the tested range, however this range was aiming for a 1.6 V supply minimum.

These results are very promising for extreme environment applications. This demonstrates that the digital calibration scheme, with some modifications to minimize the noise and other basic DLL architecture modifications, lends itself to be robust against static voltage and supply variations.

5.3.2 System Transient Monitor Test Results

The transient monitor was tested in the manner described earlier. In doing this, special care had to be taken for the dynamic D flip-flop issue. The case where the VCDL became both too fast and too slow were tested. The results are shown in Figures 5.31 and 5.32.

The results demonstrate that in the presence of large transients, that the transient monitor will produce an error. This error could be used to recalibrate the DLL when the time the DLL takes to relock would be longer than that of the coarse calibration.

When the control voltage is pulse high, it takes longer for the lock error to respond. This is because the DLL was already locked at a higher control voltage, and the delay does not change as drastically as when the control voltage drops lower, therefore it takes longer for the transient monitor to catch the error. Possible enhancements to the transient monitor will be presented in the next chapter.

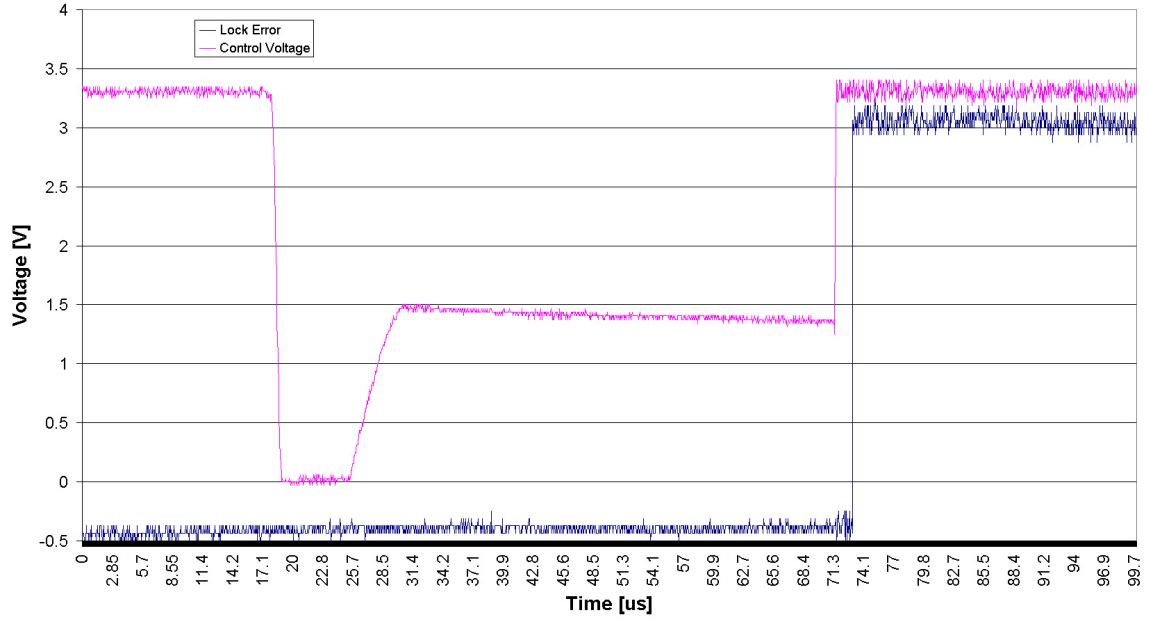


Figure 5.31: Lock error test where the VCDL became too fast

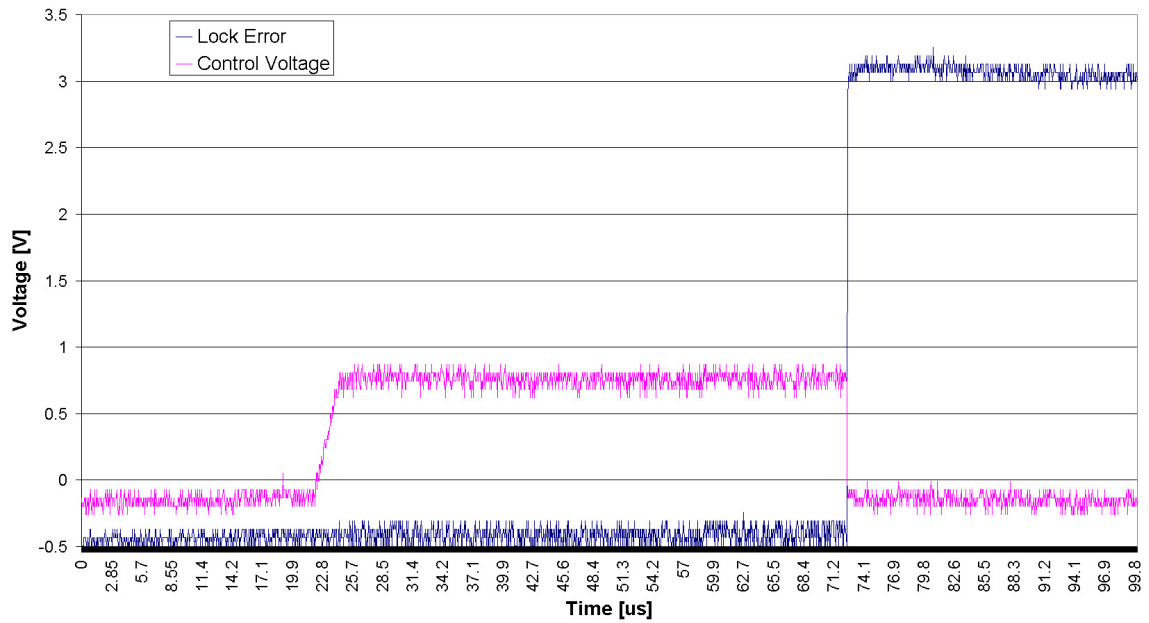


Figure 5.32: Lock error test where the VCDL became too slow

5.3.3 Comparison to Prior Art

Table (5.1) provides a comparison of the previously reported self-calibrating delay locked loops to the one presented here. From the table it can be seen that this work is the only self-calibrating DLL that has presented any temperature and supply voltage variation measurements, nor has there been any presented analog multiphase delay locked loops published in the literature. For direct comparison, the room temperature data was included.

5.4 Summary

This chapter provides the results of the self-calibrating DLL. Whereas there were some issues that need to be addressed in future work, the core concept works.

The digital coarse calibration scheme for analog multiphase DLL proved to work with a supply voltage range of 1.7 V to 2.1 V in a temperature range of -125°C to 175°C . However there are some caveats with the DLL as it stands. Noise from various sources proved to cause problems for the single-ended architecture with a simple charge pump. Also, the missing frequencies need to be addressed in future revisions. In the end, the self-calibrating DLL was able lock to 54 MHz – 94 MHz across the extreme environment range tested.

The transient monitor worked as expected. The slow response on the delay to fast side could be a problem for certain applications, but in general the transient monitor would be extremely useful in extreme environments where radiation, temperature changes, or dc supply variations exist.

Table 5.1: Comparison of prior art self-calibrating delay locked loops to presented work.

	This Work	[35]	[37]	[11]
DLL Type	Analog	Digital	Analog	Digital
Process	0.15μ	0.28μ	0.35μ	0.35μ
Multiphase Outputs	Yes (10)	No	No	Yes (7)
Frequency [MHz]	54 - 94 (22 - 135)*	150 - 200	6 - 130	20 - 85
Delay Range [ns]	7.88 (38.05)*	1.67	159.74	38.24
Supply Variation	1.7 V - 2.1 V	No data	No data	No data
Temperature Range	-125°C to 175°C	Not Reported	Not Reported	Not Reported

* Room temperature data given for direct comparison purposes.

Chapter 6

Future Work

6.1 Introduction

Even though the coarse calibration algorithm and transient monitor work, there is much that can be improved upon. This chapter will provide some future modifications that would improve the overall operation of the self-calibrating DLL. This will include variations of the delay cell (Section 6.2), modifications to the transient monitor (Section 6.3), future testing (Section 6.4) and an additional calibration block to help minimize localize process errors (Section 6.5).

6.2 Alternate DLL Components

The first issue that needs to be addressed with self-calibrating DLL is how to minimize the noise, or make the DLL more resistant to it.

The easiest fix to the fabricated DLL would be use static flip-flop in place of dynamic in places that need to be used to hold values. This however was just a design error, not necessarily an architecture improvement.

One of the most common methods to minimize common mode noise, as well as supply rejection, is to use a differential architecture. One of the first analog DLLs presented [6] used a differential pair as the delay cell. This is also a common technique for ring based VCOs. Figure 6.1 demonstrates the basic architecture of the differential VCDL. However the simple delay cell presented by Maneatis did not include any digital control. Adding capacitors to the output of the delay cells would effectively change the delay just as was done with the delay cell presented here, however these capacitors must be added to both output nodes. Note that this would be done on the delay cells, not the output buffers. This would look something similar to Figure 6.2. If switches are to be used, then special care must be take to ensure that the RC time constant does not cause any potential problems as noted in Chapter 5. The general architecture of the coarse calibration block would not need to be directly modified, assuming that the Tap_{N+} signal is fed back to the phase detector. The biasing of V_{BN} and V_{BP} would be generated in the same process independent manner that was presented by Maneatis [6].

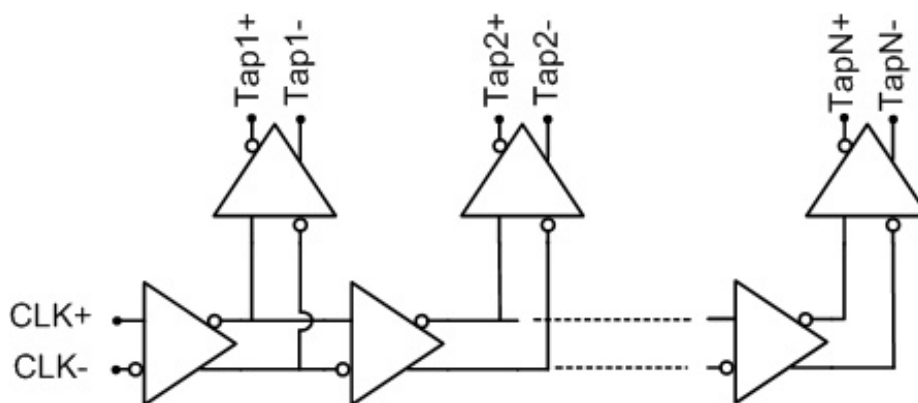


Figure 6.1: Common differential VCDL architecture.

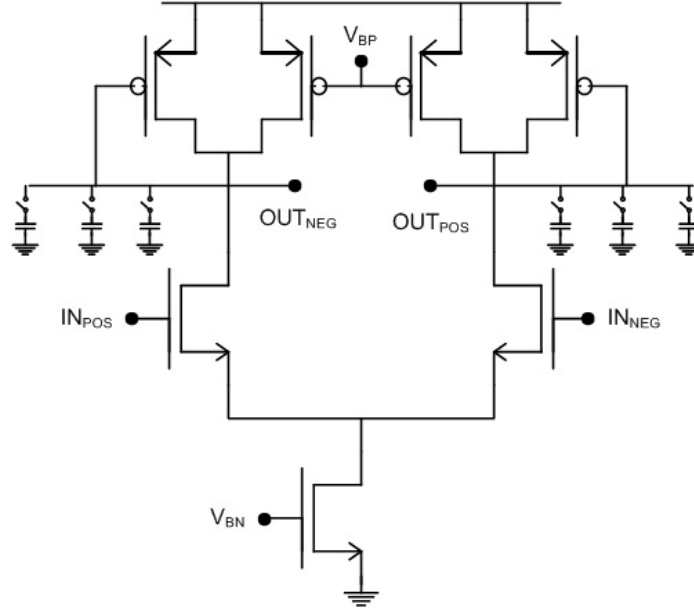


Figure 6.2: Differential pair delay cell with digital calibration.

The above architecture is not a wide swing output, which generally produces lower noise compared to the rail-to-rail output counterparts. However this may not be acceptable in certain applications. Another option would be use the differential inverter architecture shown in Figure 6.3. Figure 6.3b shows the basic delay cell schematic. To make it so it can be calibrated digitally, the bias generator is a simple current DAC. As the PMOS mirror devices are switched in or out, the available current in the inverters will change with it. This concept could also be applied to the delay cell of Figure 6.2. One could also add capacitors to the outputs of the delay cell just as done in this work.

As the temperature increased it was noticed that the implemented delay cell architecture's delay range reduced, but the center delay stayed relatively constant. By combining the capacitor and current DAC approach, it may be possible to keep the capture range constant across temperature.

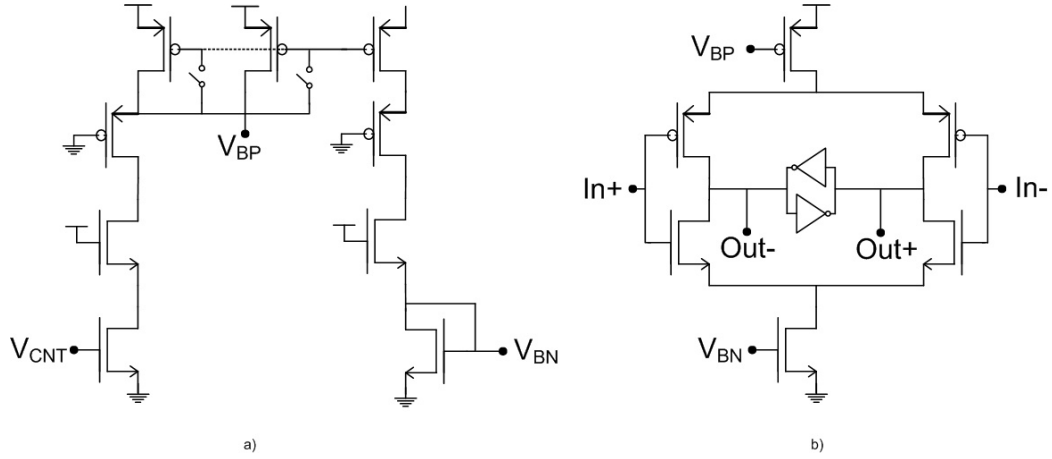


Figure 6.3: a) Bias generator with calibration, and b) Inverter based differential pair delay cell with digital calibration.

Other enhancements the the DLL may include isolation buffers between the control voltage and the bias generator as to minimize any kick back. A fully differential phase detector/control voltage path would also help enhance PSRR.

6.3 Alternate System Transient Monitor

There are some improvements to the transient monitor that would enhance its capabilities.

Another consideration is that the transient monitor is sensitive to the reference clock duty cycle for the the case where the VCDL has become to fast. Using tap7 as the decision for the fast condition. Ideally the tap7 occurs at 7 ns (for 10 ns input clock period), and the input clock falling edge would fall at 5 ns for a 50% duty cycle. This means that the VCDL delay would need to change by more than 2 ns for an error to occur. If the duty cycle decreased by 10%, the falling edge would now occur at 4 ns. For an error to occur in this case, the VCDL would now have to change by more than 3 ns. If the duty cycle increases, the probability that an error is flagged will increase.

Something else that was assumed was that the control voltage was close to the reference V_{LOWER} . Assume that the control voltage is close to the upper limit of the tuning range, it is possible that the DLL would be unable to lock, and the transient monitor is unable to report an out of range error, based on the proposed operation.

A possible solution to these dilemmas is shown in Figure 6.4. By monitoring the control voltage, it can be determined to decide if the DLL will be unable to lock on the upper range (i.e. will not be able to reduce the delay enough). If the control voltage goes above the upper threshold (which could be the same comparator and reference used by the coarse calibration) defining the tuning range, the comparator would trip, indicating an $OutOfRange_{Slow}$ error. To eliminate the problem with the duty cycle, an additional replica delay cell has been added to the output of the DLL. If the delay of the VCDL plus one additional delay occurs before the next reference clock edge, then a lock error would be triggered. Also, the addition of counters to the outputs of the original transient monitor can be used as thresholds that will only trip once a sufficient number of fast or slow errors has occurred. These counters would be continually reset after a certain period of time, or possibly decrement for every cycle that an error does not occur.

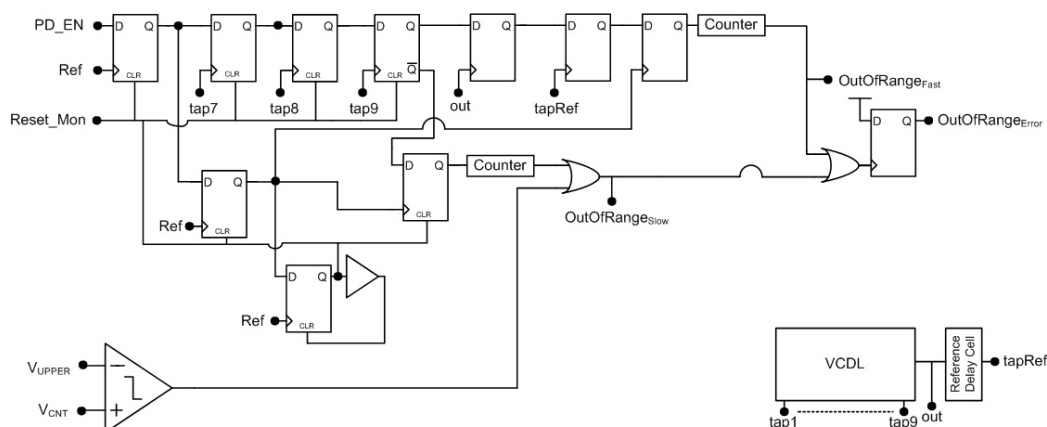


Figure 6.4: Alternative solution to the system transient monitor.

6.4 Further Testing

Beyond the tests performed in this study, there would be some other tests that would help qualify its operation in extreme environments.

If the DLL was to be used in space, or other extreme cold environments, and if the capture range trend shown in Chapter 5 holds, the DLL capture range will continue to increase. However, for higher temperature environments, above 200°C, the DLL must be tested to ensure its operation. To test the DLL at higher temperatures, high temperature PCB boards material, temperature solder, and passive components would be required.

Also, for space based applications, radiation testing would be imperative. This would test the transient monitor’s ability to detect any catastrophic radiation strike, as well as the self-calibrating DLL’s ability to calibrate out any TID effects.

Finally, future revisions of the DLL should incorporate low voltage differential signaling (LVDS) to bring the clocks on and off the chip as a means to minimize EMI.

6.5 Fine Calibration

The main consideration of the coarse calibration block was to minimize PVT effects on the DLL. One thing that it does not deal with is localized (device-to-device) mismatch. The main issue with process mismatch is that a multiphase DLL would exhibit timing mismatch between the different phases. That is to say that the ideal tap-to-tap time difference would be T_{REF}/N , where N is the number of delay cells in the VCDL. For DLL based applications such as frequency synthesis and TDCs, these timing errors would correspond to frequency errors, or spurs, and DNL errors. This has been the focus of multiple DLL papers [39, 42, 43].

The majority of the prior art deals with either complex algorithms or phase blending approaches. A different approach to the previously published work would be to use a reference delay cell that is identical to the reference that each delay cell would be calibrated with. This system is shown in Figure 6.5. After each delay cell has been calibrated to the reference cell, the result would be that each delay cell will have minimized the timing errors due to process mismatch. For further details of the proposed fine calibration architecture, refer to Appendix C.

6.6 Summary

With some modifications to the current design, the self-calibrating DLL can become a much more robust design. By first making the DLL fully differential, many of the common mode noise issues can be minimized. Implementing the proposed variation to the transient monitor will ensure that false errors are kept to a minimum. Further testing, such as irradiation, will help qualify the DLL for extreme environments such as space. Finally, the proposed coarse calibration will help reduce the effects of localized process mismatch with the VCDL, thereby making the tap-to-tap timing errors less.

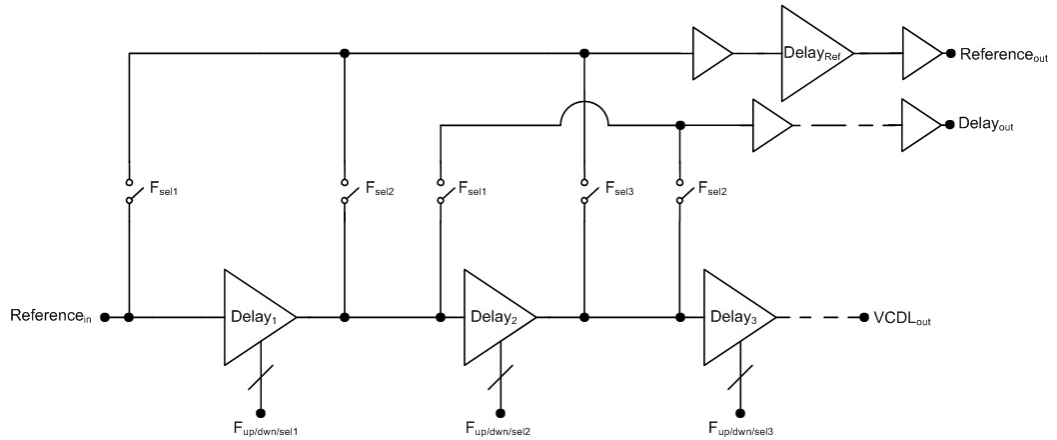


Figure 6.5: Proposed fine calibration system.

Chapter 7

Conclusions

This dissertation presented a self-calibrating analog delay locked loop tailored for extreme environment applications. Test results have been provided demonstrating the abilities of the self-calibrating architecture implemented in a 150-nm SOI CMOS technology.

The following list provides the original contributions of the presented work:

- A self-calibrating technique for a analog multiphase DLLs
- A DLL suitable for extreme environment operation, including noisy supplies
- Measurement results of a DLL over a 300 degree temperature range (-125°C - 175°C)
- Technique for enabling the phase detector to ensure proper locking
- A system transient monitor that can be used to detect large transients in the DLL outputs

Bibliography

Bibliography

- [1] J. Craninckx and M. Steyart, *Wireless CMOS Frequency Synthesizer Design*. Kluwer Academic Publishers, 1998.
- [2] B. D. Muer and M. Steyart, *CMOS Fractional-N Synthesizers, Design for High Spectral Purity and Monolithic Integration*. Kluwer Academic Publishers, 2003.
- [3] G. S. L. R. R. Zhang, “Fast acquisition clock and data recovery circuit with low jitter,” *IEEE Journal of Solid-State Circuits*, vol. 41, pp. 1016–1024, May 2006.
- [4] S. Y. Sun, “An analog PLL-based clock and data recovery circuit with high input jitter tolerance,” *IEEE Journal of Solid-State Circuits*, vol. 24, pp. 325–330, April 1989.
- [5] A. Efendovich and *et al.*, “Fast acquisition clock and data recovery circuit with low jitter,” *IEEE Journal of Solid-State Circuits*, vol. 29, pp. 67–70, January 1994.
- [6] J. Maneatis, “Low-Jitter Process-Independent DLL and PLL Based on Self-Biased Techniques,” *IEEE Journal of Solid-State Circuits*, vol. 31, pp. 1723–1732, November 1996.
- [7] G. Chien and P. Gray, “A 900 MHz Local Oscillator using a DLL-Based Frequency Multiplier Technique for PCS Applications,” *IEEE Journal of Solid-State Circuits*, vol. 35, pp. 1996–1999, December 2000.

- [8] D. J. Foley and M. P. Flynn, "CMOS DLL-Based 2-V 3.2-ps Jitter 1-GHz Clock Synthesizer and Temperature-Compensated Tunable Oscillator," *IEEE Journal of Solid-State Circuits*, vol. 36, pp. 417–423, March 2001.
- [9] A. L. Coban, M. H. Koroglu, and K. A. Ahmed, "A 2.5-3.125-Gb/s quad transceiver with second-order analog DLL-based CDRs," *IEEE Journal of Solid-State Circuits*, vol. 40, pp. 1940–1947, September 2005.
- [10] T. H. Lee, K. S. Donnelly, J. T. C. Ho, J. Zerbe, M. G. Johnson, and T. Ishikawa, "A 2.5-V CMOS delay-locked loop for 18 Mbit, 500 Megabytes DRAM," *IEEE Journal of Solid-State Circuits*, vol. 29, pp. 1940–1947, December 1994.
- [11] C. C. Chung and C. Y. Lee, "A new DLL-based approach for all-digital multiphase clock generation," *IEEE Journal of Solid-State Circuits*, vol. 39, pp. 469–475, March 2004.
- [12] J. Christiansen, "An integrated CMOS 0.15 ns digital timing generator for TDC's and clock distribution systems," in *Nuclear Science Symposium and Medical Imaging Conference*, pp. 690–701, November 1994.
- [13] K. Karadamoglou, N. Paschalidis, E. Sarris, N. Stamatopoulos, G. Kottaras, and V. Paschalidis, "An 11-bit high-resolution and adjustable-range CMOS time-to-digital converter for space science instruments," *IEEE Journal of Solid-State Circuits*, vol. 39, pp. 214–222, January 2004.
- [14] N. Haddad, R. Brown, T. Cronauer, and H. Phan, "Radiation hardened COTS-based 32-bit microprocessor," in *Radiation and Its Effects on Components and Systems*, pp. 593–597, Fifth European Conference on RADECS, 1999.
- [15] D. L. Critchlow, "MOSFET Scaling-The Driver of VLSI Technology," *Proceedings of the IEEE*, vol. 87, pp. 659–667, April 1999.
- [16] J. Craninckx and M. Steyart, *CMOS Circuit Design, Layout, and Simulation*. IEEE Press Series on Microelectronic Systems, 1998.

- [17] R. Farjad-Rad, W. Dally, N. Hiok-Tiaq, R. Senthinathan, M. Lee, R. Rathi, and J. Poulton, "A low-power multiplying DLL for low-jitter multigigahertz clock generation in highly integrated digital chips," *IEEE Journal of Solid-State Circuits*, vol. 37, pp. 1804–1812, December 2002.
- [18] F. Gardner, "Charge-Pump Phase-Lock Loops," *IEEE Transactions on Communications*, vol. 28, pp. 1849–1858, November 1980.
- [19] A. Chandrakasan, W. J. Bowhill, and F. Fox, *Design of High Performance Microprocessor Circuits*. New York, NY: IEEE Press, 2001.
- [20] C. Jia, *A Delay-Locked Loop for Multiple Clock Phases/Delays Generation*. PhD thesis, Georgia Institute of Technology, Atlanta, GA, 2005.
- [21] M. Mansuri and K. Chih-Kong, "Jitter optimization based on phase-locked loop design parameter," *IEEE Journal of Solid-State Circuits*, vol. 37, pp. 1375–1382, November 2002.
- [22] M. Lee, W. Dally, T. Greer, N. Hiok-Tiaq, R. Farjad-Rad, J. Poulton, and R. Senthinathan, "Jitter transfer characteristics of delay-locked loops - theories and design techniques," *IEEE Journal of Solid-State Circuits*, vol. 38, pp. 614–624, April 2003.
- [23] B. Razavi, *Design of Analog CMOS Integrated Circuits*. New York, NY: McGraw-Hill, 2001.
- [24] B. Garlepp, K. Donnelly, J. Kim, P. Chau, J. Zerbe, C. Huang, C. Tran, C. Portmann, D. Stark, Y. Chan, T. Lee, and M. Horowitz, "A portable digital DLL for high-speed CMOS interface circuits," *IEEE Journal of Solid-State Circuits*, vol. 34, pp. 632–644, May 1999.
- [25] J. Brown, "A digital phase and frequency-sensitive detector," *Proceedings of the IEEE*, vol. 59, pp. 717–718, April 1971.

- [26] H. Notani, H. Kondoh, , and Y. Matsuda, “A 622-MHz CMOS Phase-Locked Loop with Precharge-Type Phase Frequency Detector,” *VLSI Symp. Dig. Tech. Papers*, pp. 129–130, June 1994.
- [27] W. Rhee, “Design of High Performance CMOS Charge Pumps in Phase-Locked Loops,” *Proceedings of the IEEE Intl. Symp. on Circuits and Systems*, vol. 1, pp. 545–548, 1999.
- [28] G. E. Moore, “Cramming more components onto integrated circuits,” *Electronics Magazine*, vol. 38, pp. 114–117, April 1965.
- [29] G. E. Moore, “Progress in digital integrated electronics,” *Technical Digest of the International Electron Devices Meeting*, pp. 11–13, 1975.
- [30] R. R. Schaller, “Moore’s Law: past, present, and future,” *IEEE Spectrum*, pp. 53–59, June 1997.
- [31] Z. Liu and V. Kursun, “Leakage Power Characteristics of Dynamic Circuits in Nanometer CMOS Technologies,” *IEEE Transactions on Circuits and Systems II*, vol. 58, pp. 692–696, August 2006.
- [32] J. Kalisz, T. Orzanowski, and R. Szplet, “Delay-locked loop technique for temperature stabilisation of internal delays of CMOS FPGA devices,” *Electronics Letters*, vol. 36, pp. 1184–1185, July 2000.
- [33] E. O’Sullivan, M. Kawaguchi, and A. Shimoda, “A New Auto Lock Circuit Concept Guaranteeing Low Jitter in PLL Frequency Synthesizers Irrespective of Process Variations,” in *1996 Proc. of the Bipolar/BiCMOS Circ. and Tech. Meeting*, pp. 220–223, 1996.
- [34] T. Kobayashi, H. Iwamoto, and T. Hara, “A Digitally Temperature Compensated Compact PLL Module,” *Intl. Frequency Control Symp.*, pp. 969–974, 1997.
- [35] A. Hatakeyama and *et al.*, “A 256-Mb SDRAM Using a Register-Controlled Digital DLL,” *IEEE Journal of Solid-State Circuits*, vol. 32, pp. 1728–1724, November 1997.

- [36] W. B. Wilson, U. K. Moon, K. R. Lakshmikumar, and L. Dai, "A CMOS Self-Calibrating Frequency Synthesizer," *IEEE Journal of Solid-State Circuits*, vol. 35, pp. 1437–1444, October 2000.
- [37] H. H. Chang, J. W. Lin, C. Y. Yang, and S. I. Liu, "A Wide-Range Delay-Locked Loop with a Fixed Latency of One Clock Cycle," *IEEE Journal of Solid-State Circuits*, vol. 37, pp. 1021–1027, August 2002.
- [38] A. Aktas and M. Ismail, "CMOS PLL Calibration Techniques," *IEEE Circ. and Devices Mag.*, pp. 6–11, October 2004.
- [39] F. Baronti, L. Fanucci, D. Lunardini, R. Roncella, and R. Saletti, "On-line Calibration of Non-linearity Reduction of Delay-Locked Delay-Lines," *IEEE International Conference on Electronics Circuits and Systems*, pp. 1001–1005, September 2001.
- [40] B. Minch, "A low-voltage MOS cascode bias circuit for all current levels," *IEEE International Symposium on Electronics Circuits and Systems*, vol. 3, pp. 619–622, January 2002.
- [41] B. G. Kim and L. S. Kim, "A 250MHz - 2 Ghz Wide Range Delay Locked Loop," *IEEE Journal of Solid-State Circuits*, vol. 40, pp. 1310–1321, June 2005.
- [42] H. H. Chang, C. H. Sun, and S. I. Liu, "A Low Jitter and Precise Multi-phase delay-locked loop using Shifted Averaging," *IEEE International Conference on Electronics Circuits and Systems*, pp. 434–435, February 2003.
- [43] H. H. Chang, J. Y. Chang, C. Y. Kuo, and S. I. Liu, "A 0.7-2-GHz Self-Calibrated Delay-Locked Loop," *IEEE Journal of Solid-State Circuits*, vol. 41, pp. 1051–1061, May 2006.

Appendix

Appendix A

Appendix A is supplementary information regarding the system analysis and dynamics of the DLL. This information is provided for more details than was discussed in Chapter 2.

The following s-domain analysis will provides the system equations for a charge pump based DLL. The charge pump, not to be confused with voltage doubler charge pump, was introduced for PLL's by Gardner [18]. In Figures 2.1 and 2.2, there are references to charge pumps that are the element just after the phase detector. The following analysis would be similar for the PLL.

The DLL's loop analysis has been demonstrated thoroughly in [6] and [19], but this section will be used as a review of their work. The phase detector is a sampling process, but to analyze the loop in the s-domain, this must be ignored. This assumption is valid as long as the DLL's lower bandwidth, which will be seen to vary with the input clock frequency, is approximately a decade or more less than the input clock frequency [6]. Figure 2.4 shows the block diagram of the linearized DLL s-domain model. Going around the loop in Figure 2.4, the transfer function for each block are given as second per radian for phase detector, amperes per second for the charge pump, a low pass filter for the loop filter, and radians per volt for the VCDL. These blocks transfer functions will be discussed in more detail later in this chapter. The input and output clock phases are given as Θ_i and Θ_o , with difference

being denoted as Θ_e . T_{REF} is the input clock period, and I_{CP} is the charge pump charging current.

In feedback control theory terms, the DLL takes on the closed-loop form

$$\frac{\Theta_o(s)}{\Theta_i(s)} = \frac{G(s)}{1 + G(s)}, \quad (1)$$

where $G(s)$ is

$$G(s) = \frac{\Theta_o(s)}{\Theta_i(s)} = \frac{K_{PD}K_{VCDL}I_{CP}F_{LF}(s)}{T_{REF}}. \quad (2)$$

For many DLLs, the loop filter $F_{LF}(s)$ is a single capacitor connected from the output of the charge pump to ground. This holds the form

$$F_{LF}(s) = \frac{1}{sC_{LF}} \quad (3)$$

in the s-domain. Plugging equation 3 into 2, and rewriting T_{REF} in terms of angular frequency yields

$$G(s) = \frac{K_{PD}K_{VCDL}I_{CP}}{s2\pi C_{LF}}\omega_{REF}. \quad (4)$$

Substituting 4 into 1, and then rearranging variables gives

$$\frac{\Theta_o(s)}{\Theta_i(s)} = \frac{1}{1 + \frac{s}{\omega_n}}, \quad (5)$$

where ω_N is the closed loop 3-dB frequency given as

$$\omega_N = \frac{K_{PD}K_{VCDL}I_{CP}}{s2\pi C_{LF}}\omega_{REF}. \quad (6)$$

AppendixB

After the first successive edge once the phase detector is enabled (PD_EN goes high), the chain of three D flip-flops actively watches the seventh through ninth taps of the VCDL. In actuality, there will be other D flip-flops before the tap7 D flip-flop that are used to monitor the other earlier phases of the VCDL, just as presented for the coarse calibration circuit, but is drawn this way for simplicity. There are two cases that are of interest when the DLL goes out of lock: 1) the VCDL has become too fast, and 2) the VCDL has become too slow. First consider when the VCDL delay has become so slow that the main DLL will be unable to correct it. Assume the DLL is locked to a reference clock of 100 MHz (10 ns), and the tuning voltage is approximately in the center of the tuning range. The proposed VCDL would ideally have a tuning range of 1 ns around the 10 ns in clock period, or 10 ns to 12 ns. Each of the delay cells will exhibit approximately a 1 ns delay. This means that if tap9 has still not occurred after one clock period, that the VCDL will have approached the edge of its capture range. Referring to Figure 4.20, the middle D flip-flop connected to the output of the PD_EN D flip-flop will go high on the second reference clock edge. If tap9 has still not gone high, its Qbar output will still be high, which will cause OutOfRangeSlow to go high. This causes OutOfRangeError to go high, which would be used to reset the DLL for calibration. After the third reference edge, the lower D flip-flop causes the monitor flip-flops to be reset, with the exception of the OutOfRangeError flip-flop, and then the process repeats. This process is demonstrated in Figure 4.21a. When the delay of the

VCDL becomes too fast, it becomes more difficult to determine how fast it has become. The only other information given by the reference clock is the falling edge, which ideally would occur half of the clock period after the rising edge. Under locked (or close to lock) conditions, tap5 of the VCDL should be going high at this time. Again, assume the VCDL delay is set at approximately 10 ns. If the input reference dropped to 11.11 ns (90 MHz), the VCDL may now be unable to relock since the delta delay step is greater than 1 ns. With each delay cell exhibiting 1 ns of delay, and the ideal reference clock's falling edge occurring at 5.555 ns, tap5 would be high, but tap6 would still be low until 6 ns. Since the VCDL control voltage is close to that of the reference VLOWER as used in the coarse calibration, depending on the value of VLOWER, there is usually a lot of extra tuning capability below, but at the cost of a higher VCDL gain. The delay could easily be adjusted 3 ns slower, if the VLOWER is set at the corner of the curves in Figure 4.9 (around 0.9 V). If this is allowed it would be easy to just monitor the output of the tap6 D flip-flop, and determine if it occurred before the first half of the reference clock cycle. Unfortunately most clocks will not exhibit a perfect 50% duty cycle (e.g. crystal oscillators can vary up to 10%). Only the propagated rising edge of the reference clock inside the VCDL is used to for calculation purposes, but the falling edge of the reference clock is needed to make a decision for the VCDL out of range fast case. For a 100 MHz clock (10n), the variation could easily cause the falling edge of the clock to occur anywhere between 4 ns and 6 ns. As demonstrated in Figure 1, if the DLL is close to lock, the deviation of the falling edge could potentially overlap that of the tap6 rising edge (ideally at 6 ns for locked case), which would leave to false out of range errors. However, tap7 would not be inside this deviation range, leaving it as a good choice for the decision, which is illustrated in Figures 4.20 and 4.21.

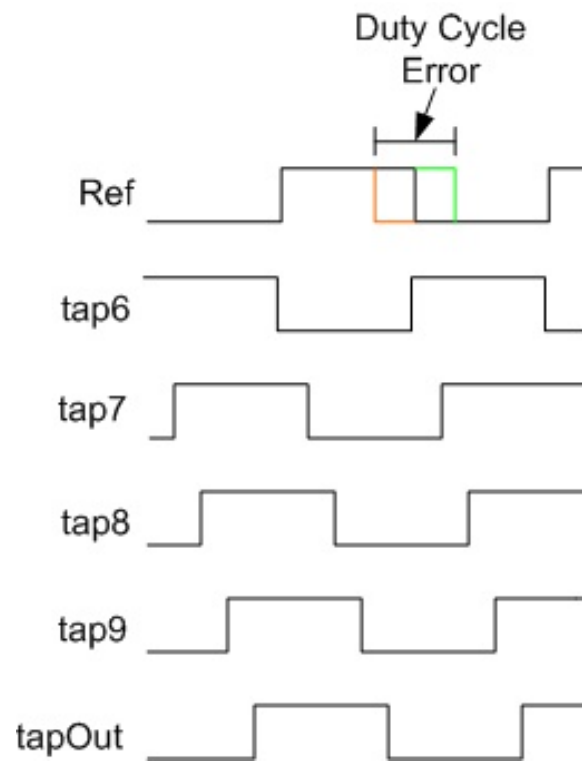


Figure 1: Duty cycle error for VCDL too fast case.

AppendixC

In order to make each of the delay cells equal to one another, some form of calibration must be performed of the individual delay cells. The method proposed here uses a reference delay cell as a reference timing element. This reference delay cell is identical to the delay cells in the VCDL, and has undergone the same coarse calibration. This means that its delay should be close to that of the VCDL delay cells, but it too will differ by some amount based on the process deviation. By comparing the delay of each of the VCDL delay cells to that of the reference delay cell, and individually adjusting them to equal the reference delay, the final result would be that all of the delay cells' delays would be equal.

to incorporate this the the implemented self-calibrating DLL, an extra system must be added as show in Figure 2. This can be done by selecting the individual delay cells' inputs and outputs, and using those signals as the test signals. This basic function is shown by the switches in Figure 6.5. For example, assume Delay2 is being calibrated. The input of Delay2 (output of Delay1) is connected to the input of Delay_{Ref}, and the output is connected to the DelayOut path. The time it takes for the signal from the input of Delay2 through the Reference_{Out} path will consist of two buffer delays and the reference delay cell delay ($2 \cdot T_{BUF} + T_{DELAYREF}$). The Delay_{Out} path consists of two buffer delays and the Delay2 delay ($2 \cdot T_{BUF} + T_{DELAY2}$). The difference between these two paths is the difference between the two delay cells ($T_{DELAYREF} - T_{DELAY2}$). If the difference should be adjusted

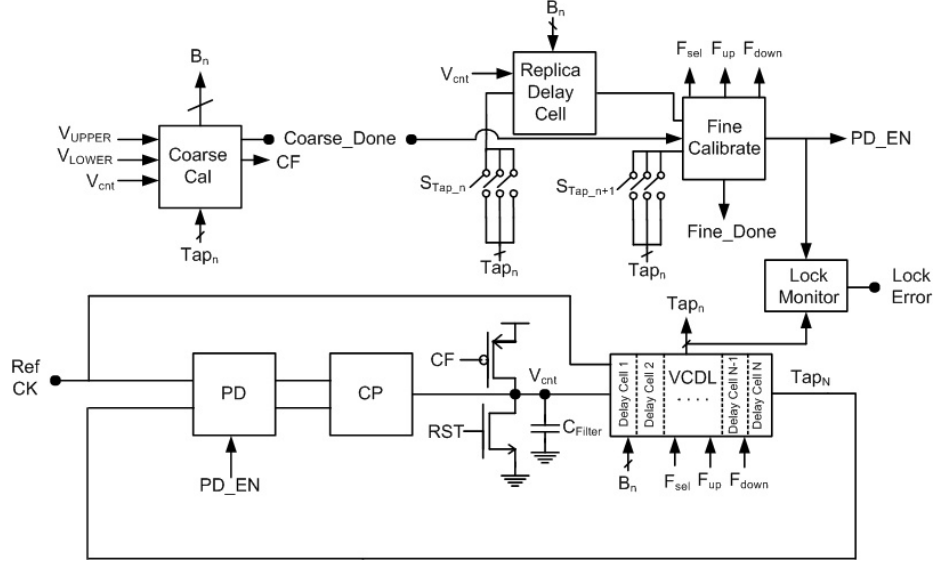


Figure 2: Self-calibrating DLL with coarse and fine calibration capabilities.

to equal zero on every delay cell, the result will be that each of the delay cells delays will be equal to each other. The buffers are used to ensure that similar capacitive loads are seen by both of the signals. Any timing error difference between these buffers would be result in a global error on all of the delay cells, which would not cause any tap-to-tap timing errors. The fine calibration algorithm is shown in Figure 3.

Upon the global reset, which is the same used in the coarse calibration algorithm, all of the fine calibration logic and registers will be reset in a known state. It will be important to be able to test the effectiveness of the proposed fine calibration technique; therefore the circuit must have option to be bypassed by some enable pin. This enable bit is controlled by both the user and the coarse calibration done bit. If the fine calibration circuit is enabled the algorithm continues, otherwise the fine calibration algorithm is completed. At the reset state, the first VCDL delay cell is selected for comparison via the delay cell pointer. The fine calibration circuit tests to see whether the selected delay cell has either become faster or slower than the reference delay cell compared to the previous test. This would be false for

the first comparison. Specifically, the in the delay cell under calibration had been lagging the reference delay cell, and is now leading, that delay is finished calibrating. On the other hand, if the delay cell had been leading and is now lagging, one capacitor is removed, and then the algorithm continues. On the subsequent tests, if the direction of the lead/lag position has changed (e.g. the selected delay was leading but now is lagging), the jumps past the timing comparison section, otherwise it continues. The reference cell delay is compared to the selected delay cell delay, and if it is faster the lead counter is incremented, otherwise the lag counter is incremented. Even though it is not directly shown, it is possible for the both registers to be incremented due to the implementation of the delay differencing stage. This would occur if the two delays were within 5 ps from one another. This operation is repeated until the comparison counter completes. Once the comparison counter is done, the two counter outputs are compared using a digital comparator. If the lag counter is less than the lead counter, meaning the selected delay cell is slower than the reference delay cell, the selected delay cell delay is decreased. If the lead counter is greater than the selected delay cell delay is increased. If the counters are equal, the algorithm jumps to test if the last delay cell had just been tested. The fine calibration circuit then checks to see if the selected delay cell's fine delay word can be adjusted any more. If it can, the previous steps repeat until the direction of the lead and lag position has change, otherwise the algorithm continues. If the last delay cell has now been calibrated, the fine calibration has been completed. However, if there are more delay cells that need to be calibrated, the delay pointer is incremented, the various counters are reset, and the whole process repeats on the next delay cell.

Ideally the fine calibration circuit should be able to determine whether the selected VCDL delay cell is leading or lagging the reference delay cell after one test point. However, in the presence of noise and other factors such as the potential for small thresholds shifts during switching, a single test point could result in a false result. Therefore the comparison counter was added to allow for multiple test points, effectively creating a statistical average of leading and lagging. Digital comparison of the lead and lag counters determines whether

the selected delay cell is leading, lagging, or equal to the reference delay cell. Ideally, for the equals case, there would be an approximation buffer zone that would be used to allow close counter values to be considered equal. For example, counters with the values 100000 (32) and 011111 (31) would show that the register holding 101000 is larger even though there is only a difference of one. If not buffer zone was used, the digital comparator would show that the selected delay cell would be lagging, however with values this close together, it would be difficult to accurately assume this in the presence of non-idealities. However, the sum of the lead and lag registers will not necessarily equal 64 (26), since it is possible for the phase detector to show that the delays are equal for very small delay differences (within 5 ps) between the test and reference delay cells. For example, if the reference and test delay cells were within the 5 ps range, it would be possible for the two registers to hold values of 110001 (49) and 111100 (60), which would report an equals. In this case, the comparison would be valid, despite the large difference. If the delay cell's delay difference was not small, the chance for both registers to be incremented at the same time would be small. Despite the potential for error by not using the approximation buffer zone, the required additional logic needed to test for all of the cases would have been excessive. Instead, due to the complexity of the approximation approach, the proposed DLL uses two 6-bit counters for the lead and lag registers, with the comparison counter being 6-bits too, and compares the two MSBs while discarding the remaining bits. Even though it is possible for the case discussed above (registers with 32 and 31) to occur due to noise and other environmental aspects causing cycle-to-cycle deviations in the reference and test delay cell's delay, these deviations must exceed the 5 ps window set by the phase detector.

A block diagram of the fine calibration circuit is shown in Figure 4, including the VCDL for completeness. The blocks correspond to the flow chart of Figure 3, with the exception of the delay cell pointer, which consists of the signal buses S_{tap} and F_{sel} inside the fine calibration logic.

For demonstration purposes, the digital fine calibration control has been incorporated to the delay cell used in this work as shown in Figure 5. Only the NMOS coarse calibration capacitors are shown in Figure 5, but there are complementary PMOS capacitors as discussed in Chapter 4. The only components that differ from the previously described delay cell are the delay cell fine calibration control logic and the MOSCAPs, with their control switches. There are $2(N+1)$ MOSCAPs (half PMOS and the other half NMOS) that make up the digital fine calibration delay steps. From hence forth, only the NMOS capacitors will be discussed, but the exact same logic applies to PMOS capacitors.

The sizes and number of the MOSCAPs in the delay cell of Figure 4.19 are based on multiple factors. Since these factors set the digital fine calibration time step and range of the delay cell, there will be a tradeoff between the accuracy of the technique and the area that the capacitors/switches occupy. The delay of that branch (T_{d2}) of the delay cell is given by Equation 4.4 above. CL2 is being adjusted as the fine calibration MOSCAPs are being added or removed, just as the case of the coarse calibration, and the current $i_{dn,p}$ is the source/sink current of the inverter. After running Monte Carlo simulations on the individual delay cell, the standard deviation delay change of the delay cell was found to be 80 ps. This would include variations in the various transistors and capacitors. Based on these results, the fine calibration delay range must be able to change at least 80 ps when the maximum or minimum fine calibration codes are applied. For this proposed DLL, the fine calibration delay range was chosen to change 96 ps. For demonstration purposes, this range is broken up into 8 steps of 12 ps, which means that there are a total of 16 fine calibration capacitors ($N=16$). In other words, for every F_{up} , or F_{down} , the delay cell will increase by either -12 ps, or decrease by +12 ps. This step size can be reduced by using smaller capacitors, but at the cost of more stages, hence larger silicon area.

The switch array of Figure 6.5 is controlled by the fine calibration logic, and will switch the input of selected delay cell to the reference delay cell path, and its output to the VCDL output (or $Delay_{Out}$) path.

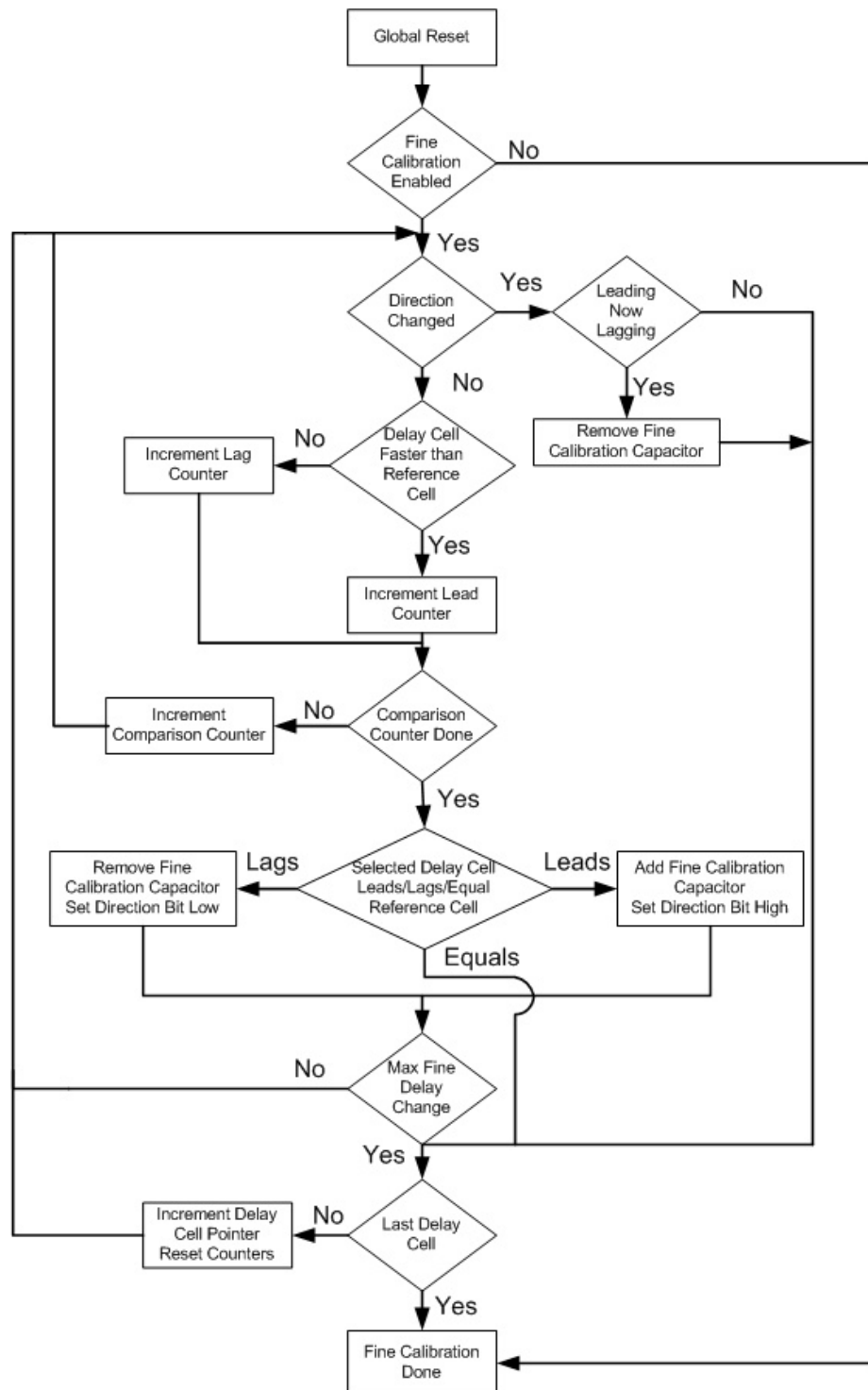


Figure 3: Fine calibration algorithm.

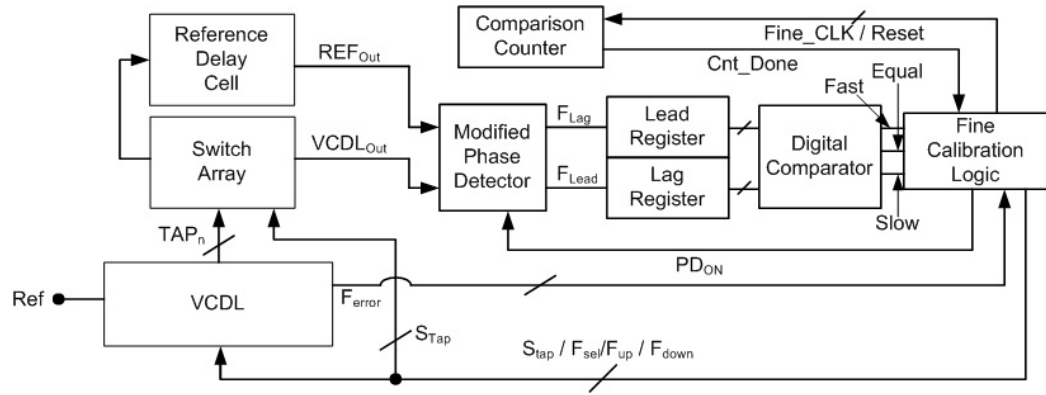


Figure 4: Fine calibration block diagram.

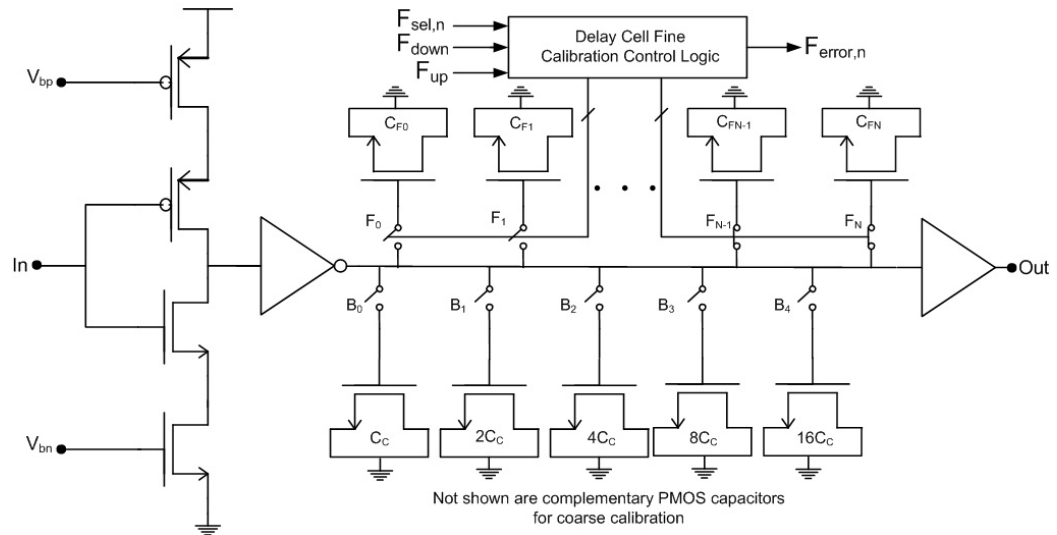


Figure 5: Delay cell with fine calibration digital control.

Phase detectors usually operate with no dead zone to ensure that the transfer characteristic of the phase detector is linear (i.e. no zero gain section of the transfer characteristic). This means that for the main phase detector used in the DLL, both the Up and Down signals go high for every clock comparison, but for the proposed fine calibration circuit, the only information needed would be that the reference signal led the slave signal (i.e. Up='1' and Down='0'). The reverse outputs would be required for the reference signal lagging the slave signal.

To implement this function, the pseudo-digital phase detector in Figure 6 was developed. This modified phase detector is based on the basic PFD architecture, but a small dead zone was added. If the redrawn latches were D flip-flops, this simplified form of the modified phase detector resembles that of the standard phase detector, but with the exception of the extra control signals. The dashed box consists of three stages: the first stage consisting of the four leftmost transistors, the second stage consisting of the next five transistors, and finally a D flip-flop. The first and second stage is actually a modified dynamic D-flip, which has been altered to allow for more functionality. Assume that the modified phase detector is enabled ($en = '1'$). In this condition the second stage of the block is active due to the PMOS being on. If en was low, then that stage is has no path to V_{DD} , thereby disabling the phase detector. Due to different pull-down NMOS transistors, after a reset condition, signals $rstb$ and rst , the outputs of the three stages, upa , $dwna$, up , and $down$, are a logic '0'. This reset will occur due to multiple events. First, if either the up or down signals are high, $rstb$ will go high, thereby resetting both the first and second stages. In the event that reference clock, the slave signal, $rstb$, and the enable bar bit are all '0', the third stage D flip-flops will be reset. Since signals $dwna$ and upa are low after a reset, the PMOS device of the first stages is active, waiting for the reference and slave signals to go high. Assume that ref is leading slave by some time (small or large). The output of the upper (up path) latch first stage will ideally start charging up before that of the lower (down path) latch. As the first stage output approaches the threshold of the second stage input, the second stage output will begin to charge towards V_{DD} . For the ref leading case, upa will charge up

earlier than dw_{na} . Since up_{a} is connected to the PMOS of lower latch's first stage, the first stage will begin to turn off as up_{a} rises towards V_{DD} . Signal up_{a} is also connected to NMOS devices in the lower latch's first and second stages. As up_{a} approached V_{TN} , the first and second stage outputs are pulled back towards ground, thereby resetting that latch. The dw_{na} signal will start the same process on the upper latch, but since up_{a} started earlier, the end result will be that up_{a} will continue to charge up will dw_{na} discharges. Signal up_{a} will continue to rise until the D flip-flop switching point has been reached, at which time the latch output up will go high. Signal up causes $rstb$ to go high thereby resetting the first and second stages of the latches, and once the input signals $slave$ and ref go low again, signal rst goes high thereby resetting the D flip-flops. This process has been simulated in Spectre, and the results are shown in Figure 4.22 for the case where $slave$ leads ref by 8.7 ps. As can be seen in the middle graph, the dw_{na} and up_{a} signals initially raise at a faster rate, but as they approach the thresholds of the NMOS reset devices, the charging slows down. Both signals are close together and still charging towards V_{DD} (around 260.25 ns), but the dw_{na} is charging slightly faster since it started charging earlier. As dw_{na} continues to get larger than up_{a} , the opposite latch will begin to pull towards ground quicker, eventually to the point where up_{a} pulls away, and latches the output D flip flop.

There might be some events where up and $down$ are both high. This could occur if the signals are perfectly overlapping, or so close that the difference between up_{a} and dw_{na} is no longer large enough to cause the feedback reset action to work. In simulation, signals as close as 5 ps apart were able to be discerned from one another. Regardless, if the input signals are this close, it is safe to assume that the reference delay cell and the selected VCDL delay cell have approximately the same delay.

Another concern is that the with process parameter mismatch, that one latch in the pseudo-digital phase detector will be faster than the other. If this is the case, the error will be seen by all of the tested delay cells, which would be a static delay error that is added (or subtracted) from each delay cell. This means that all of the VCDL delay cells delay would

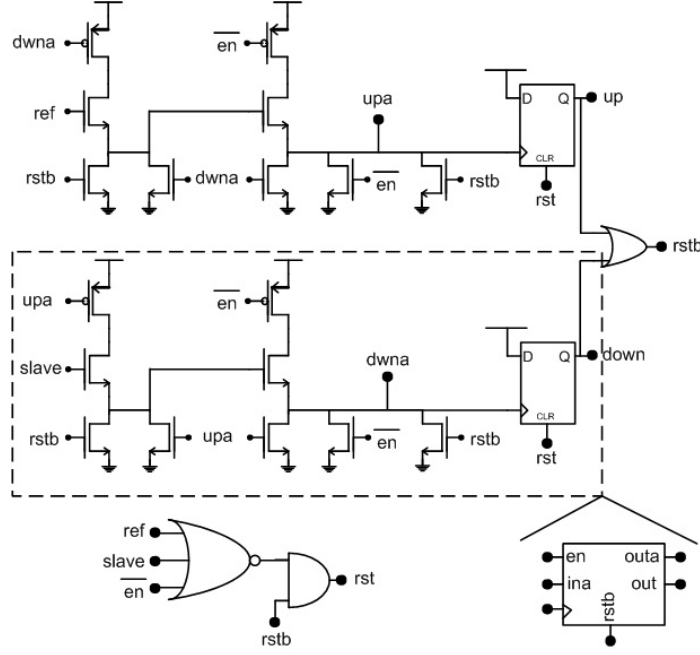


Figure 6: Proposed pseudo-digital phase detector used for the fine calibration system.

still be approximately equal to each other, but not necessarily equal to the reference delay cell.

The other major concern would be noise and stored charge seen at the different nodes throughout the test path. This would result in potential differences between the comparison nodes, in particular upa and dwna. For this reason, the comparison counter is used such that a large enough sampling of the modified phase detectors outputs can be averaged, thereby giving a more statistically accurate result.

Full DLL simulations have been performed with a slightly modified version of the proposed fine calibration scheme with positive results. This concept could also be applied to a differential topology with some minor changes.

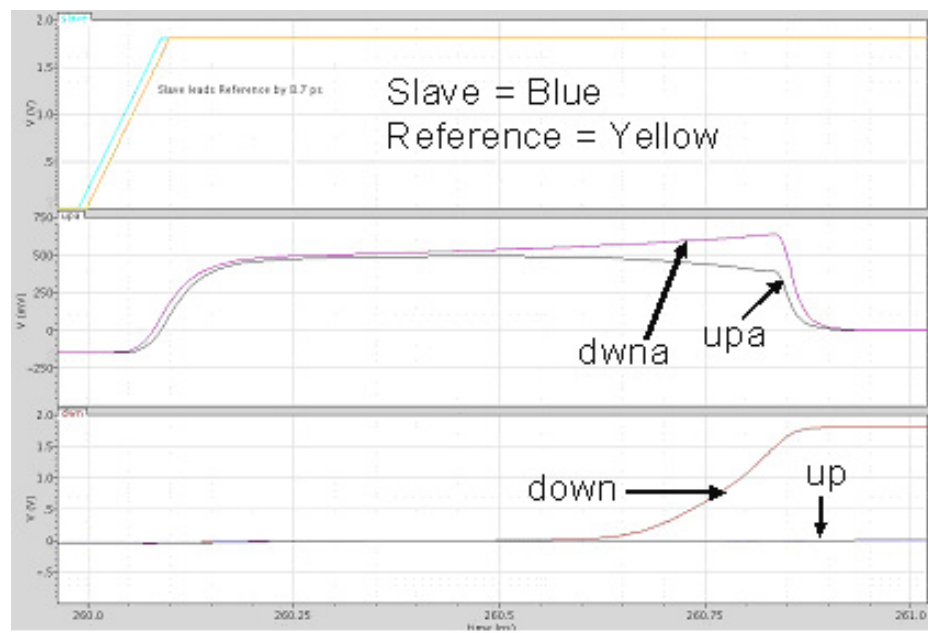


Figure 7: Simulation of the modified phase detector where slave leads the reference clock.

Vita

James Vandersand Jr. was born on July 13, 1978 in Burleson, TX. He lived there until he completed high school in 1996, and then went to Mississippi State University. He received his Bachelors and Masters degree in Electrical Engineering from MSU in 2000 and 2002, respectively.

James has been slowly working on his Ph.D. degree in Electrical Engineering since August 2002 under the supervision of Dr. Benjamin Blalock. The major focus of his research included CMOS analog and mixed-signal integrated circuit design, which included delay locked loops, operational amplifiers, comparators, and other analog building blocks. In particular, the circuits were intended for extreme environment applications such as space. He also worked on other projects involving novel devices including SOI MESFETs and G4-FETs.

During the final semester of his Ph.D., he joined Cadence Design Systems, Inc. Cary, NC where he works on memories. His other major life quest is to stay physically active by means of kung fu, tai chi, and general fitness activities.