Digital Phase Tightening for Improved Spatial Resolution in millimeter-Wave Imaging Systems

by

Ke Lu

B.S., University of California (2006)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Masters of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2009

Author ....
Department of Electrical Engineering and Computer Science
December 22, 2008

Certified by. ....
Charles G. Sodini
LeBel Professor, Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by ....
Terry P. Orlando
Chairman, Department Committee on Graduate Theses

Digital Phase Tightening for Improved Spatial Resolution in millimeter-Wave Imaging Systems

by

Ke Lu

Submitted to the Department of Electrical Engineering and Computer Science
on December 22, 2008, in partial fulfillment of the
requirements for the degree of
Masters of Science in Electrical Engineering and Computer Science

Abstract

Imaging systems using millimeter-wave frequencies allow for the possibilities of vehicular radar and concealed weapons detection. By using silicon technology, the integration of millimeter-wave circuits can reach new levels that were previously impossible. This thesis discusses the challenge and design of a mm-wave imaging system using a technique called digital phase tightening for improved spatial resolution. Digital phase tightening uses feedback and oversampling to accurately measure the amplitude and phase of an incoming signal. Furthermore, it can be implemented using only a delay-lock loop, an analog-to-digital converter, and a counter. A proof of conecpt system utilizing a 2.4GHz delay-lock loop with supporting circuitry is designed in 90nm CMOS. Test results demonstrate a proof of concept system with a measured DLL resuolution of 41.7ps that consumes 36mW of power. The goal of the system is to reduce the jitter of phase measurements to the order of femto-seconds. In the proto system, the quantization error is larger than the Gaussian noise; therefore, significant improvements in the accuracy of the phase measurements were not obsereved.

Thesis Supervisor: Charles G. Sodini
Title: LeBel Professor, Electrical Engineering and Computer Science

Acknowledgments

Working on the millimeter-wave active imaging project has taught me countless things. The most important realization I made is recognizing that I could not have completed this project without the help of numerous people who are many times more talented than I am.

First, I would like to thank my advisor, Prof. Charles Sodini, for giving me the opportunity to work on this project and guiding me through it. Charlie taught me how to identify problems, solve them, and most importantly, communicate to another person on how the problems were solved. Additionally, I greatly appreciate Charlie for putting in so much time and effort during the winter holiday season to ensure that I graduate on time.

I would like to thank Helen Kim of Lincoln Laboratory, Prof. Harry Lee, and Prof. Greg Wornell for helping me to understand and design the circuits for the millimeter-wave imaging system.

I would like to especially thank Khoa Nguyen for helping me with everything imaginable that comes with designing a CMOS chip, Johnna Powel for providing guidance on constructing my circuits and writing my thesis, and Anthony Accardi for explaining digital phase tightening and making sure that I understood it completely.

I would like to thank the past and present member of the Sodini/Lee group for providing me with their insights: Jack Chu, Mark Spaeth, John Fiorenza, Jeff Feng, Albert Chow, Kevin Ryu, Mariana Markova, David He, Ivan Nausieda, Matt Guyton, Eric Winokur, and Sunghyuk Lee. Furthermore, I would like to thank Suzanne Piccarini for helping me receive my reimbursements in time to pay for monthly bills.

Also, I would like to thank some of my fellow graduate students for their immense help; without their expertise and wisdom I would have not been able to finish my project: Belal Helal, Matt Park, Jose Bohorquez, Payam Lajevardi, Chun-Ming Hsu, Pat Mercier, Phil Godoy, Byungsub Kim, and Sanquan Song.

Contents

1 Introduction **11**

1.1 Active Imaging via Beamforming 11

1.2 Previous Work 15

2 MMW Active Imaging System **17**

2.1 Digital Phase Tightening 17

2.1.1 Algorithm 17

2.1.2 Implementation 21

2.2 System Overview 24

2.3 Averaging for Noise Reduction 27

2.4 Using Noise to Improve Phase Estimation 28

3 Proof of Concept Integrated Circuit for Phase Tightening **34**

3.1 DLL Operation And Architecture 34

3.1.1 Delay Element And Delay Line 37

3.1.2 Phase Detector, Charge Pump And Low Pass Filter 41

3.1.3 Frequency Divider for Duty Cycle Correction 46

3.2 Comparator 48

3.3 Digital Processing 51

4 Testing of Integrated Circuits **53**

4.1 Decreased Output Swing 55

4.2 Test Results for Functionality 61

5 Conclusion **65**

5.1 Summary 65

5.2 Future Work 66

List of Figures

1-1 Image formation for conventional digital camera using a lens to focus [1] 12

1-2 Incident wave received by a two element array for beamforming [1] . . 13

1-3 The beam pattern is set to receive waves from angle θ_T but the approaching wave is at angle θ 14

1-4 Tradional digital beamforming receiver with different phase delays at each element 16

2-1 Lock state of digital phase tigtening algorithm with positive edge triggered sampling (a) Input signal with a phase offset of ϕ from the reference clock (b) Sampling clock with a phase offset θ (c) Reference clock 19

2-2 Shifting the sampling clock to lock onto positive-sloped zero crossings when (a) θ the phase offset of the sampling clock is initially greater than ϕ the phase offset of the signal (b) θ is initially smaller than ϕ . 20

2-3 Due to the resolution of the phase shifts, the sampling clock will toggle around the positive-sloped zero crossings 21

2-4 Locking time can be reduced in half when alternating samples are used to lock onto both positive-sloped and negative-sloped zero crossings . 22

2-5 Chopping inverts the input for every other sample, which effectively turns all zero crossings to postived-sloped 22

2-6 System level block diagram for digital phase tightening. The DLL generates a shifted version of the reference clock for the sampling clock of the ADC. The digital processing block determines the direction of the shifts using the ADC's outputs. 23

2-7 The phase offset of the input signal ϕ is in between $M \cdot 2\pi/N$ and $(M-1)\cdot 2\pi/N$; thus, the sampling clock's phase offset θ toggles between the two values. (a) The input signal after chopping with 4 sample points (b) The reference clock 25

2-8 MMW active imaging system. One transmitter and 1000 receiver elements all share a 100MHz reference clock. 26

2-9 Noise effecting phase and magnitude measurements (a) Ideal input sampled with no noise indicates that the sampling clock's phase offest θ should not be changed (b) Noise changes the sample values from the ideal case and indicates θ should be increased. Furthermore, the amplitude measurement also changed. 29

2-10 True value of ϕ lies between quantization levels 0 and 1 30

2-11 When there is no significant amount of noise present, the measured value of ϕ alternate between 0 and 1 every cycle 30

2-12 Value of ϕ with a small noise $n(t)$ lie between quantization levels 0 and 1 30

2-13 Value of ϕ with a larger noise $n(t)$ lie between quantization levels -1, 0, and 1 31

2-14 With an increased noise, the measured value of ϕ switches between -1, 0, and 1 randomly 31

2-15 Calculated quantization error of the phase estimations for values of ϕ between 0 and 1 with a Gaussian noise source with rms jitter of 0.5 steps 32

2-16 Quantization error of the phase estimations for values of ϕ between 0 and 1 when the Gaussian noise source has a rms jitter of (a) 0.35 steps (b) 0.45 steps 33

3-1 Block diagram of the implemented system on chip. External 2.4GHz signal is used for the reference clcok and external 0.6GHz signal is used as the comparater's input. The system outputs 5 bits at 1.2GHz . . . 35
3-2 Traditional DLL architecture with phase detector, low pass filter, and delay line 36
3-3 Modified DLL architecture using a multiplexer to select different phase offsets from the reference signal and a frequency detector to generate 50% duty cycle at the output 37
3-4 Schematic view of delay element with cross coupled inverters at the output for positive feedback 38
3-5 Transistor view of delay element 38
3-6 The amount of skew reduction by the delay element at its output when the differential input is skewed 39
3-7 The effect of the edge transition rate of the input on the rise and fall times of the output after buffering by the delay element 40
3-8 The effect of the amplitude of the input on the ouput swing after buffering by the delay element 41
3-9 Delay line: a chain of identical delay elements 41
3-10 (a) Propagation delay of single delay element with changing control voltage
(b) Resulting frequency of delay line due to changes in the control voltage of the delay elements 42
3-11 SCL latch 43
3-12 Charge pump 45
3-13 After the RST signal is released at 10ns, the control voltage of the DLL settles within 10% of its final value after 40ns 46
3-14 Low Pass Filter 47
3-15 Frequency divider 48

3-16 Effectiveness of using the frequency divider as a duty cycle corrector to ensure 50% dutcy cycle at the output when the input has a varying duty cycle 49
3-17 Regenerative latch as comparator 50
3-18 Digital circuitry includes an adder to keep the state of the DLL and combinatorial logic to wrap around from 0 to 23 and 23 to 0 52

4-1 Schematic level description of the test setup with key input and output signals. An input signal at 1/4 the frequency of the reference clock is delayed and sampled by the comparator. The comparator's outputs adjust the comparator clcok and measure the delay by changing the 5bits of the counter. 54
4-2 PCB used for testing 54
4-3 The counter can only represent delays up to a length of $1/f$; thus, an addtional state, the number of times that the counter has wrapped around, needs to be recorded to represent delays up to $4/f$. 55
4-4 Circuit toplogy of SCL buffer 55
4-5 The core circuits of the chip and the buffers for driving the output signals off chip use separate supplies. The buffers drive 50Ω loads with parasitics from bond wires, the packaging and the PCB traces. 56
4-6 Simulated and measured voltage swings at the output 57
4-7 Layout of the chip showing the thin wires that contribute high parasitic resistances 57
4-8 Parastic resistance $R1$ is placed between the supply voltage and the buffers 58
4-9 Calculated values for parasitic resistances $R1$ and $R2$ 58
4-10 Parastic resistance $R2$ is placed between OUT and V_X 59

4-11 The plots show results for simulations using the automatic parasitic extraction in Cadence, using two manually added resistors, and using three manually added resistors. The plots are for (a) output voltage low (b) output voltage high (c) swing. 60

4-12 The digital output when the reference clock is at 1.8GHz and the input is delayed by 700ps. The output cycles between a low value and a high value. 61

4-13 System block diagram with registers 62

4-14 The 5bits of the digital output with varying delay of the input signal 63

4-15 Change in the 5bits of the digital output when the delay is changed by 500ps, where the expected value is 12 64

Chapter 1

Introduction

Due to advances in silicon and digital processing technology, millimeter-wave (MMW) imaging solutions with high antenna array density are now viable at a low cost. The MMW frequency spectrum, spanning from 30 to 300GHz, features properties that are highly suitable for imaging. The MMW, capable of penetrating through opaque objects such as clothing and inclement weather such as fog, are reflected by metal, making them advantageous over waves in the infrared and visible spectrums, which are highly attenuated in such conditions. Furthermore, wavelengths in the MMW frequencies are small enough to provide sufficient spatial resolution for imaging, which makes them advantageous over microwave frequencies. Most importantly, MMW frequencies are low enough that they can be produced from antennas driven by today's electronic circuits. Because of these advantages, specific bands of the MMW spectrum, such as the 77GHz, 94GHz, and 120GHz bands, have been allocated for imaging applications such as vehicular radar and concealed weapons detection.

1.1 Active Imaging via Beamforming

The MMW active imaging system utilizes the method of beamforming instead of the traditional method of capturing an image through a lens. In a conventional digital camera, light from the scene passes through a lens onto the focal plane as shown in Figure 1-1 [1]. A sensor array placed at the focal plane captures the light energy from the scene. Each element of the sensor array is in one-to-one correspondence with a part of the scene; thus, each element corresponds to a pixel in the resulting digital image. The value of each pixel is obtained by integrating the electromagnetic radiation received by the corresponding array element over the exposure time and quantizing this value. However the conventional optical detection mechanisms can only measure the magnitude of the incoming light; thus losing the phase information.

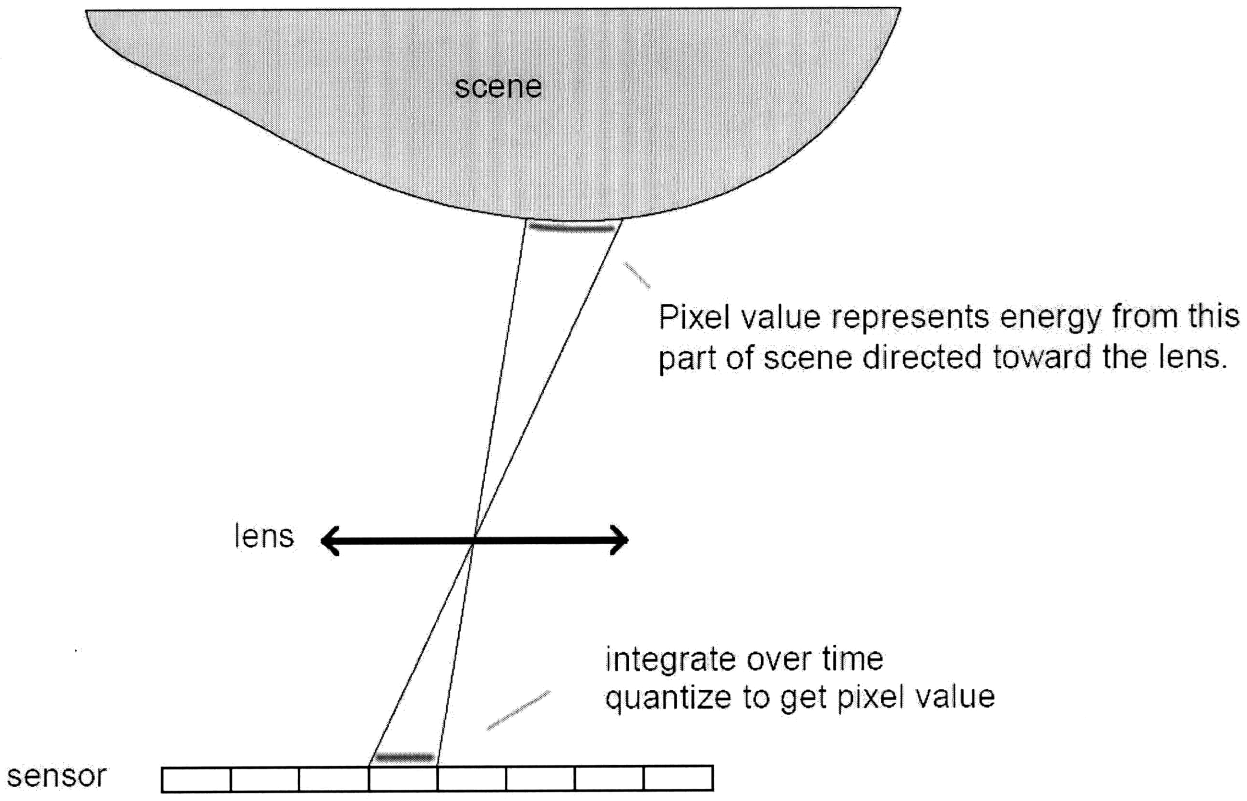

scene
Pixel value represents energy from this part of scene directed toward the lens.
lens
integrate over time
quantize to get pixel value
sensor

Figure 1-1: Image formation for conventional digital camera using a lens to focus [1]

Alternatively, the MMW active imaging system utilizes beamforming where only an antenna array is used to capture both the phase and the magnitude in order to form an image. Using constructive and destructive interferences, a beamformer adds different delays to different receiver nodes to select a specific pattern of radiation.

This concept can be demonstrated with just two array elements. Separate the two elements by a distance of d and place the physical origin of the x-axis midway in between them as shown in Figure 1-2. Assuming that the elements are isotropic then the signal recorded at one element is just a time delayed by $\frac{d}{c} \cdot cos(\theta)$ version of the other. Consider a harmonic plane wave with amplitude A and angular frequency w propagating along at an angle θ from the end of the array at the speed of light c. Choose the time origin for the t-coordinate such that the signal $A \cdot sin(wt)$ is observed at x = 0.

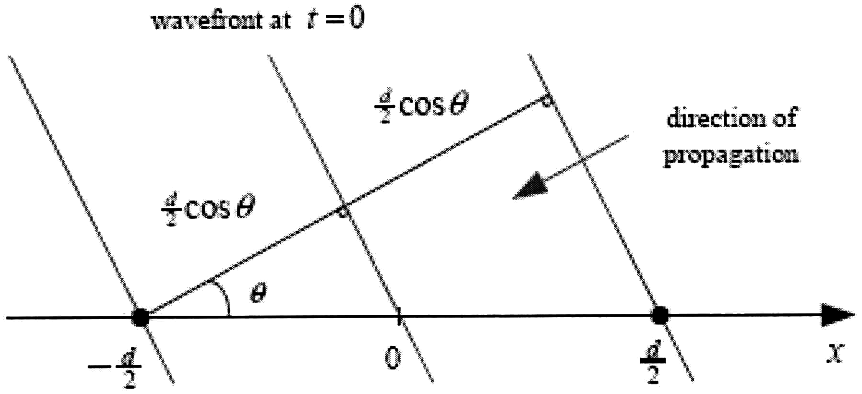


Figure 1-2: Incident wave received by a two element array for beamforming [1]

In real applications, the array will receive waves from different directions and different frequencies. The underlying circuitry will filter out waves that are not at the frequency of interest w. Beamforming can estimate the wave arriving from any angle θ_T by using the signals recorded at each element. A specific delay of $\pm\frac{d}{2c}cos(\theta_T)$ is applied to each element such that the recorded wavefronts for a plane wave arriving at the target angle θ_T are aligned in time. Note that if radiation is only present in the direction represented by $\theta = \theta_T$ then beamforming produces the correct result $A \cdot sin(wt)$. Furthermore, if a wave from another angle is present then it will be attenuated by harmonic interference. Equation 1.1 represents the result of interference [1].

$$\hat{A} = A \cdot cos\left\{\frac{\pi d}{\lambda}\left[cos(\theta_T) - cos(\theta)\right]\right\} \tag{1.1}$$

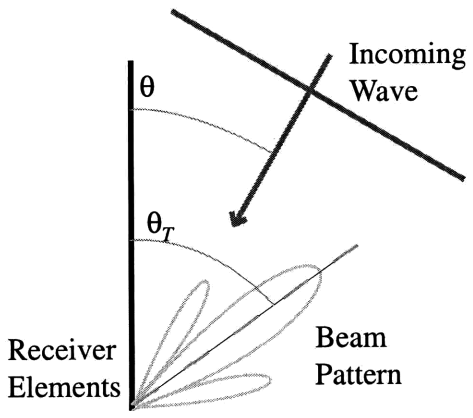


Figure 1-3: The beam pattern is set to receive waves from angle θ_T but the approaching wave is at angle θ

$\hat{A}$ is the measured amplitude of the signal $A \cdot sin(wt)$ coming from the angle θ when the beamformer is tuned to receive from the angle θ_T as illustrated in Figure 1-3. When normalized to $A = 1$, Equation 1.1 is defined as the beam pattern for the array. From the gemoetery in Figure 1-2, θ and θ_T take on values between 0 and π. As expected, $\hat{A}$ is maximum when $\theta_T = \theta$. As the angle of the approaching wave moves away from θ_T, $\hat{A}$ begins to decrease, which defines a main lobe of sensitivity pointing in the target direction. However, $\hat{A}$ increases again as θ moves even farther from θ_T, which traces out a sidelobe. The sidelobes represent ambiguities in directions that are unable to be resolved by the array.

When extrapolating to an array with multiple elements, Equation 1.2 can be used to calculate the pixel resolution ρ [1]. λ is the wavelength, L is the length of the antenna array, and r is the distance to the target. As an example, when beamforming at a MMW frequency of 77GHz using a 2m array and observing a target at 30m, we have a resolution of $\rho = 5.2$cm.

$$\rho \approx 0.8859\frac{r\lambda}{L} \tag{1.2}$$

The system drives its transmitter antenna with a transistor based amplifier to actively illuminate scenes and generate images based on the reflected electromagnetic waves. As stated previously, the system can steer the beam electronically to acquire different regions of the target image by applying a suitable delay to each receiver and adding the result. The recorded magnitude and phase information at each receiver element can be combined at a central processing unit to estimate the radiation incident from the particular direction of the steered beam. Electronic steering is advantageous because no moving mechanical parts are involved. Unconstrained by the physics of a lens, digital image formation provides a greater level of computational flexibility. However, a great deal of control of the phase is required for accurate electronic steering as beamforming works by constructive interference at the intermediate frequency (IF). For example, 1ps of jitter at a 1GHz IF results in a blurring of $\sim$0.2m from a distance of 30m. This places a very stringent specification on the amount of phase noise and timing jitter (the relationship between the two is discussed later in the thesis) in the system. The motivation for digital phase tightening, discussed in chapter 2, is designing a system that can accurately measure the magnitude and the phase and also calculate the phase entirely in the digital domain. The objective of this thesis is a proof of concept circuit for digital phase tightening.

1.2 Previous Work

In traditional digital beamforming receivers described in Haynes, the receivers manipulate the phase information before the digitizing of the electromagnetic energy [2]. The delay between signals received at different antennas is generated in the analog domain by down mixing the signals with phase shifted versions of the same sine wave as shown in 1-4. Only after this step are the signals digitized by the analog-to-digital converter (ADC). Using this setup, Mead et al. presented a digital beamforming receiver architecture with 128 elements that was capable of producing 2500 pixels by generating 50 beams in a 25 degrees field of view to image objects between 100 and 1600 meters away from the system [3].

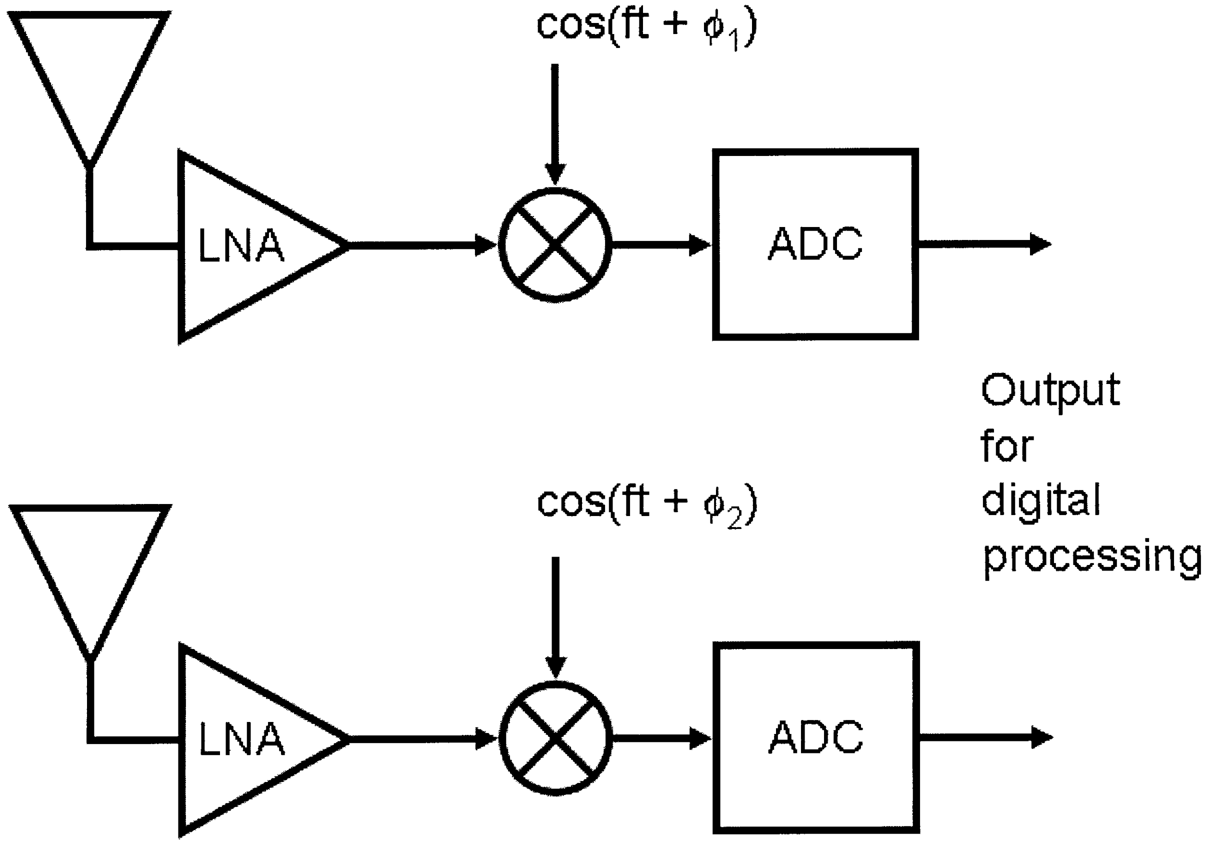


Figure 1-4: Tradional digital beamforming receiver with different phase delays at each element

Our system, as described in the next chapter, processes the phase signal purely in the digital domain. Although the overall system is novel, the key circuit elements, such as the core delay-locked loop (DLL), are motivated by previous designs. Helal et al. demonstrated a 1.6GHz DLL with 1.41ps of jitter and 6mW of power in 130nm CMOS [4].

Chapter 2

MMW Active Imaging System

This chapter is structured to first present the algorithm for measuring phase in the digital domain. Then it describes the schematic level implementation of the algorithm and how it fits into the overall architecture of the MMW active imaging system. Lastly, reduction in noise by averaging due to the specifications of the system is discussed.

2.1 Digital Phase Tightening

This secton will elaborate on the digital phase tightening methodology first mentioned in Accardi [1].

2.1.1 Algorithm

The Nyquist-Shannon sampling theorem states that if a bandlimited signal has maximum frequency f then all of the information about the signal can be obtained by sampling it at a rate of $2f$ samples per second or greater. However, it is difficult to analyze the samples. Often complicated digital signal processing (DSP) blocks are required. For example, even a set of samples generated from the signal in Equation 2.1 requires DSP to determine its phase ϕ and magnitude A.

$$A \cdot sin(2\pi f \cdot t + \phi) \tag{2.1}$$

Given the assumptions that the signal is of the form Equation 2.1 and f is known then the digital phase tightening algorithm can easily extract A and ϕ by sampling at a rate of $4f$ samples per second. The two assumptions are justified in the following sections.

The digital phase tightening algorithm can be best understood by first looking at its end result as shown in Figure 2-1. If the algorithm samples at f samples per second then in the finished state the sampling clock is shifted by θ from the reference clock to sample the signal at exactly its positive-sloped zero crossings. This shifted phase θ is exactly the same value as ϕ.

The algorithm reaches its end goal of sampling at exactly the positive-sloped zero crossings by shifting the phase of the sampling clock based on the sign of each sample. As shown in Figure 2-2, θ is decreased if the sample is positive and is increased if the sample is negative. Eventually the samples become zero and the algorithm is in lock and stops changing θ.

However, only a finite number of steps of a uniform phase shift size can be implemented; thus the samples may never become zero. Instead, the algorithm results in θ alternating between two values that correspond to the smallest possible positive value and the smallest possible negative value for a sample. As shown in Figure 2-3, this becomes the new definition for locking as the average of the two values for θ is the best estimation for ϕ. The trade off is that smaller step size results in greater resolution for the estimation and larger step size results in locking in fewer cycles. Since the algorithm stays locked, the initial locking time is not as important as the resolution is. This gives the algorithm its name because the sampling points move toward and tighten around the zero crossings.

By doubling the sampling rate to $2f$ samples per second, the even samples can lock onto the negative-sloped zero crossings while the odd samples lock onto the positive-sloped zero crossings. Now θ is shifted twice for each period of Equation

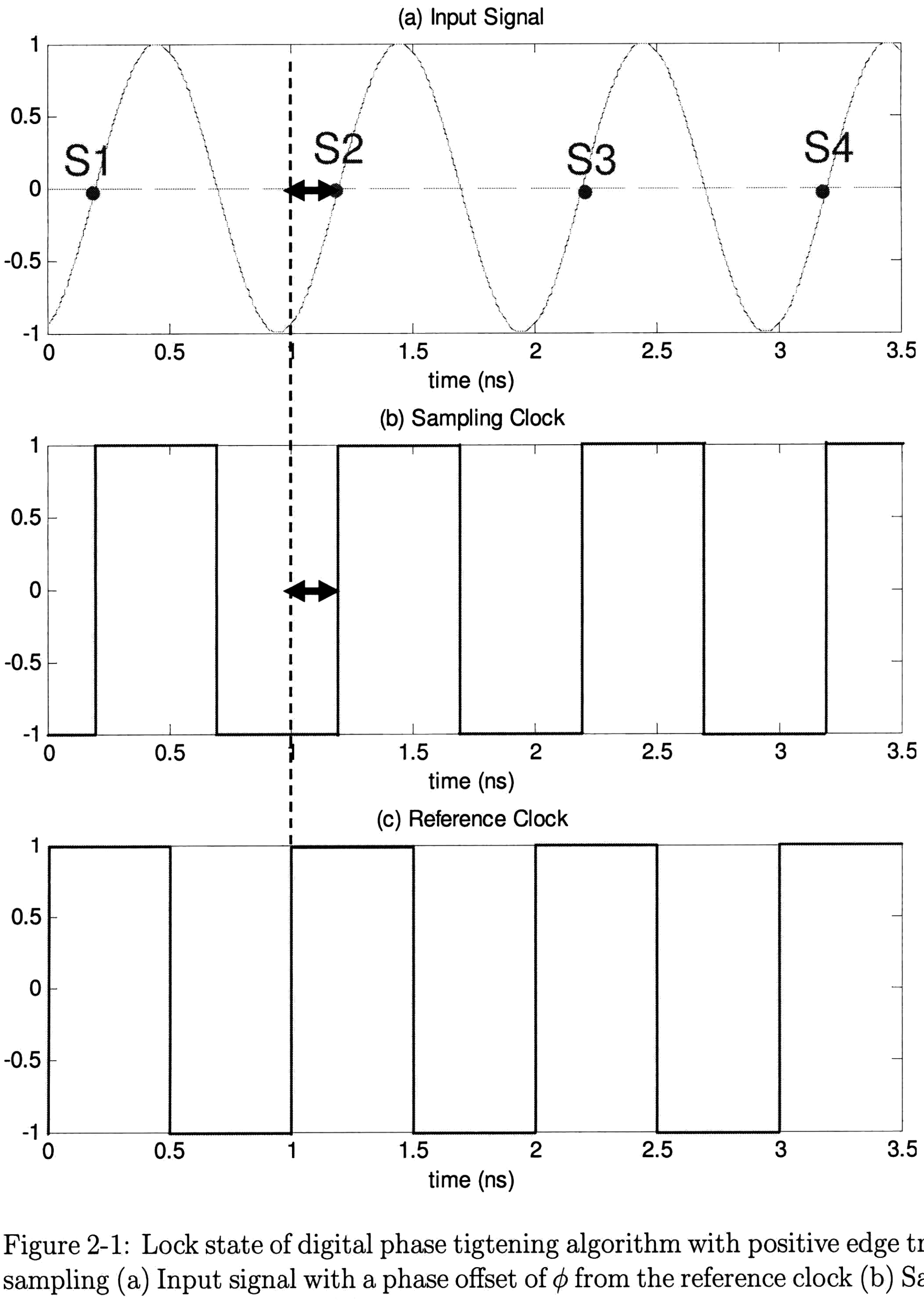


Figure 2-1: Lock state of digital phase tigtening algorithm with positive edge triggered sampling (a) Input signal with a phase offset of ϕ from the reference clock (b) Sampling clock with a phase offset θ (c) Reference clock

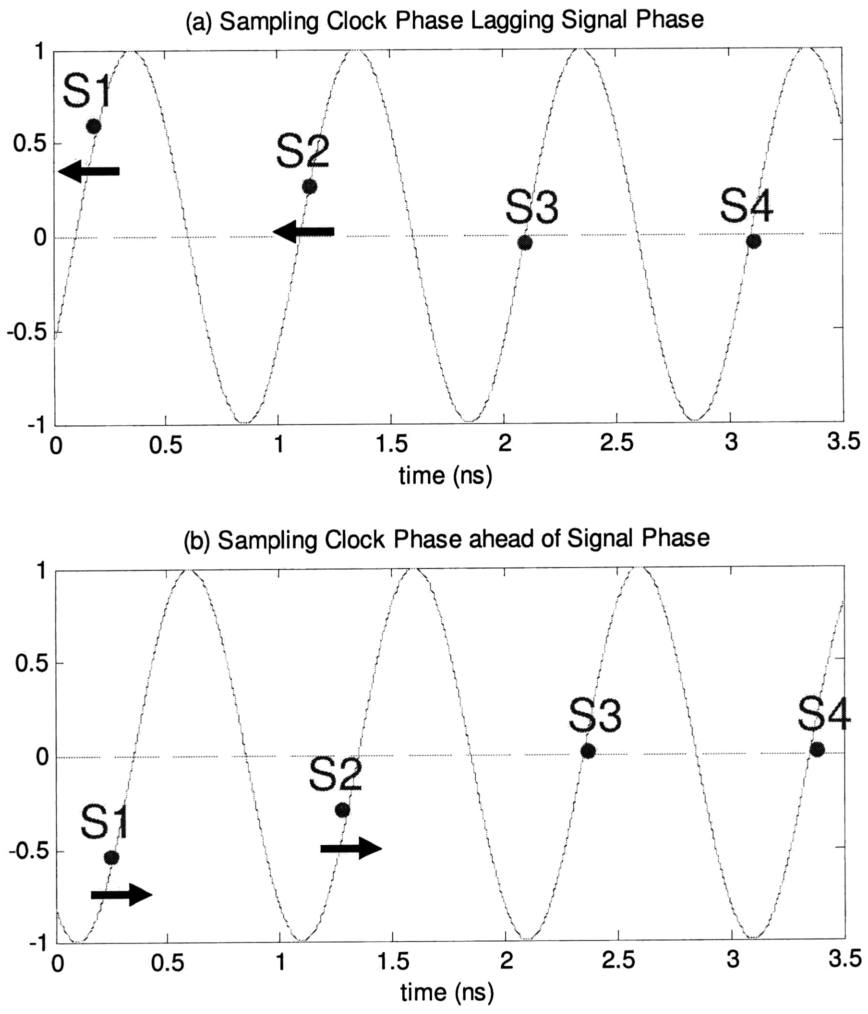


Figure 2-2: Shifting the sampling clock to lock onto positive-sloped zero crossings when (a) θ the phase offset of the sampling clock is initially greater than ϕ the phase offset of the signal (b) θ is initially smaller than ϕ

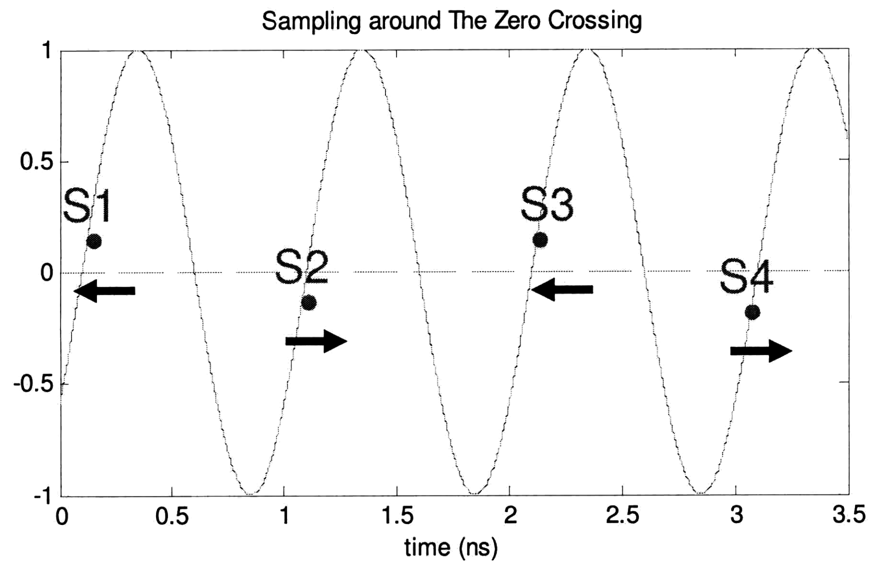


Figure 2-3: Due to the resolution of the phase shifts, the sampling clock will toggle around the positive-sloped zero crossings

2.1 instead of only once, thus the locking speed has doubled as shown in Figure 2-4. However, locking onto both types of zero crossings requires additions to the algorithm, which translates to additional circuitry for implementation. This can be avoided by inverting Equation 2.1 for every even sample. As shown in Figure 2-5, only positive-sloped zero crossings exist. This process, called chopping, is easily implemented and is described in a later section as it also has circuit related benefits.

Lastly, when the sampling rate is doubled again to $4f$, the new sets of samples, between the sets of samples used for locking, automatically fall on the maximum and the minimum of Equation 2.1. Thus A is determined.

2.1.2 Implementation

One of the main advantages of digital phase tightening is that it is easily implemented with established components as shown in Figure 2-6. The DLL shifts its output, the sampling clock, from its input, the system clock, to lock onto the zero crossings of the ADC's input. Assume the input is of the form Equation 2.1 and f is 1GHz, thus the

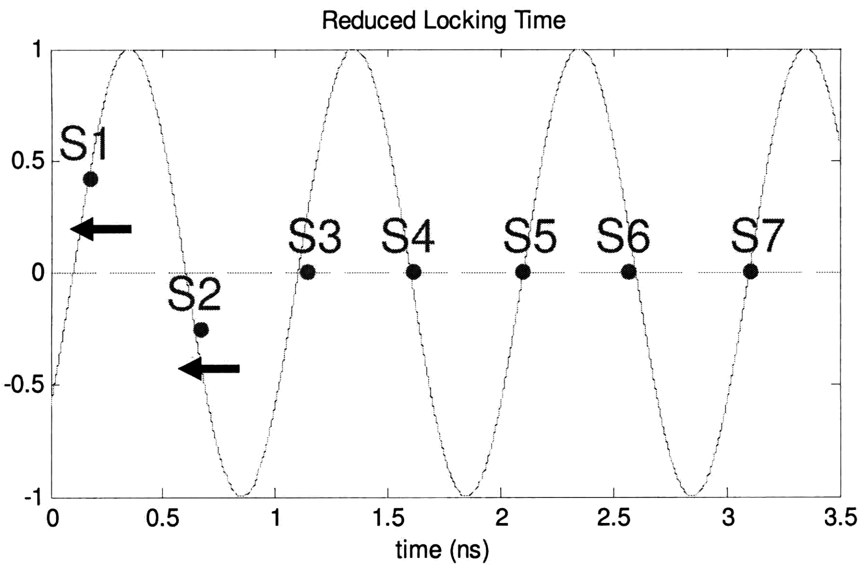


Figure 2-4: Locking time can be reduced in half when alternating samples are used to lock onto both positive-sloped and negative-sloped zero crossings

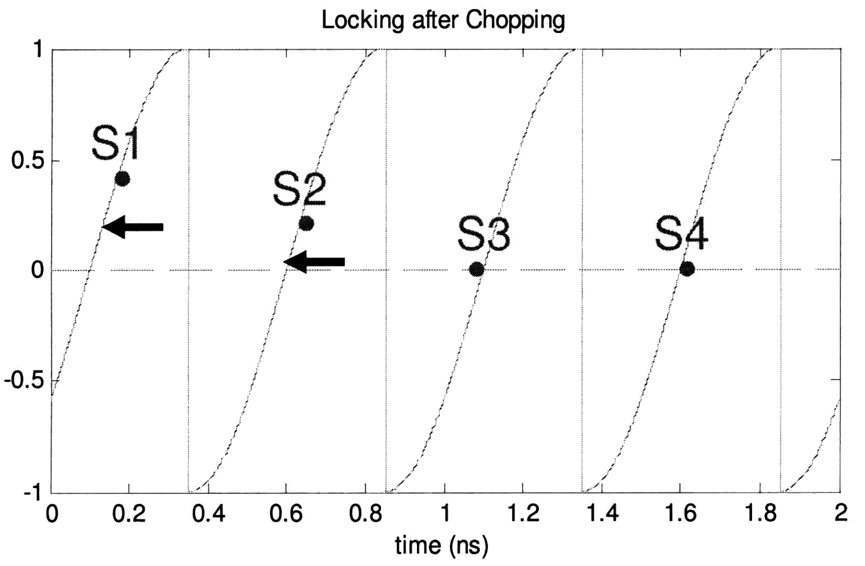


Figure 2-5: Chopping inverts the input for every other sample, which effectively turns all zero crossings to postived-sloped

ADC samples at 4 G samples/s. The digital processing block simply determines the sign of every odd sample. This information guides the DLL to shift the ADC sampling such that the odd samples become locked near the zero crossings. After which, the even samples, falling in between the zero crossing, become the measurements for the amplitude A. Additionally, the front-end of the ADC applies multiplexers to invert the input signal at every other odd sample to simplify the circuitry for digital processing as discussed previously. This technique, called chopping, also reduces the offset noise in ADCs [5].

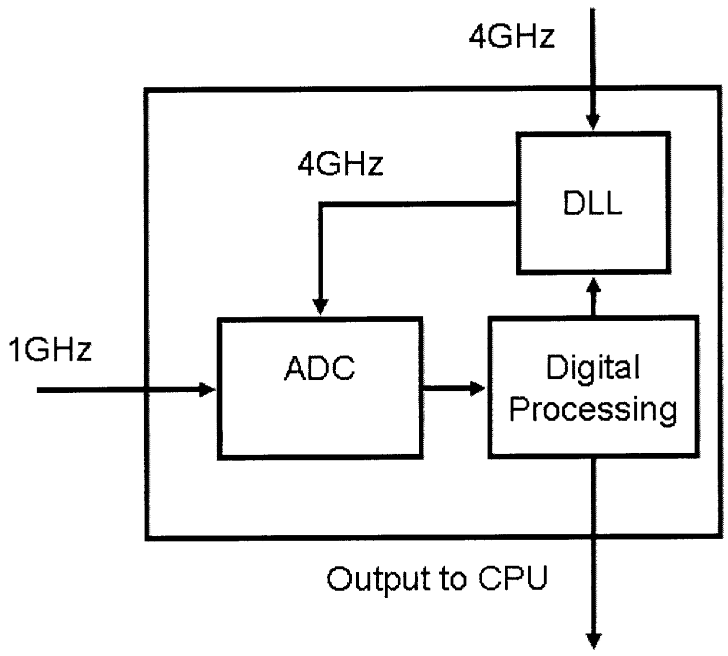


Figure 2-6: System level block diagram for digital phase tightening. The DLL generates a shifted version of the reference clock for the sampling clock of the ADC. The digital processing block determines the direction of the shifts using the ADC's outputs.

The number of cycles required for the feedback loop to become locked depends on the resolution of the DLL. The DLL can shift its output clock by steps of size $2\pi/N$, where N is the number of delay elements in the delay line. Note that N such shifts completely cover an entire period; thus locking must occur in less than N shifts. The obtainable resolution for the phase ϕ depends only on N if the ADC has a threshold at zero volts. There exists an integer M that satisfies Equation 2.2.

$$0 \leq M \cdot 2\pi/N - \phi \leq 2\pi/N \tag{2.2}$$

When the DLL shifts θ to $M \cdot 2\pi/N$ then the ADC, regardless of its step size, will sample a positive value and the DLL will shift θ to $(M-1) \cdot 2\pi/N$. For the next phase correction, the ADC will sample a negative value and the DLL will shift θ back to $M \cdot 2\pi/N$. This process is repeated as illustrated in Figure 2-7. It follows that ϕ is in between $M \cdot 2\pi/N$ and $(M-1) \cdot 2\pi/N$.

On the other hand, the estimation of amplitude A depends on both the ADC resolution and the DLL resolution because it is the quantized value of a sample taken in a range of $[-2\pi/N, 2\pi/N]$ around $\phi + \pi/2$.

2.2 System Overview

The MMW active imaging system, shown in Figure 2-8, is a planar array of 1,000 elements. Each array element consists of a Per-Antenna Processing (PAP) unit, along with the corresponding antenna for receiving radiation. Each element has a heterodyne architecture with a 76GHz phase lock loop (PLL) and a 1GHz IF. On the transmitter side, a single antenna radiates a 77GHz sinewave. Both the receivers and the transmitter share the same reference clock. Ideally, the signal transmitted and received is a spectrally pure 77GHz tone; however the mismatches in center frequency between the PLLs of the 1000 elements is equivalent to an expansion of the bandwidth of the received signal to 200MHz [1].

The PLL is used to shift the amplified received signal down to IF. A low pass filter (LPF) rejects the high frequency image and other sources of interferences. The result is converted to the digital domain. Using a clock derived by dividing down from the PLL, the analog-to-digital converter (ADC) samples at twice the Nyquist rate, 4Gs/s, to avoid aliasing. Since the antennas use the same reference source as the ADC and the DLL it becomes feasible to assume that their frequencies are in

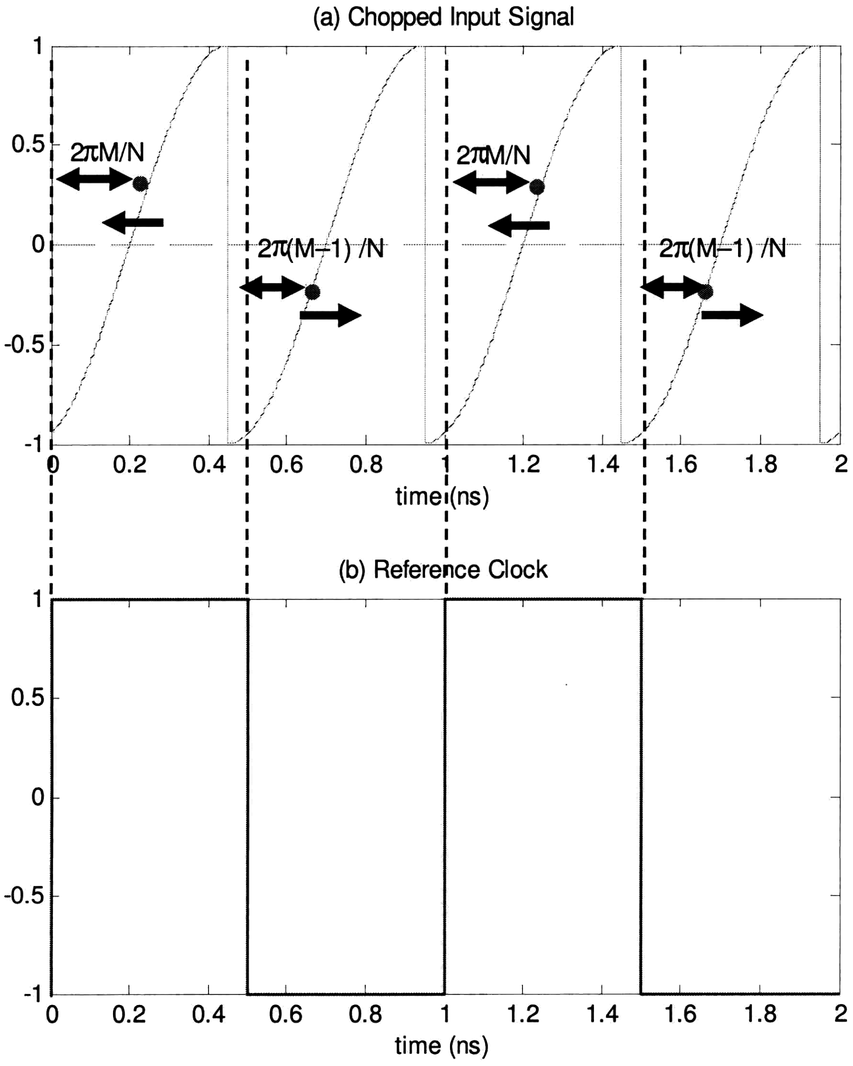


Figure 2-7: The phase offset of the input signal ϕ is in between $M \cdot 2\pi/N$ and $(M-1) \cdot 2\pi/N$; thus, the sampling clock's phase offset θ toggles between the two values. (a) The input signal after chopping with 4 sample points (b) The reference clock

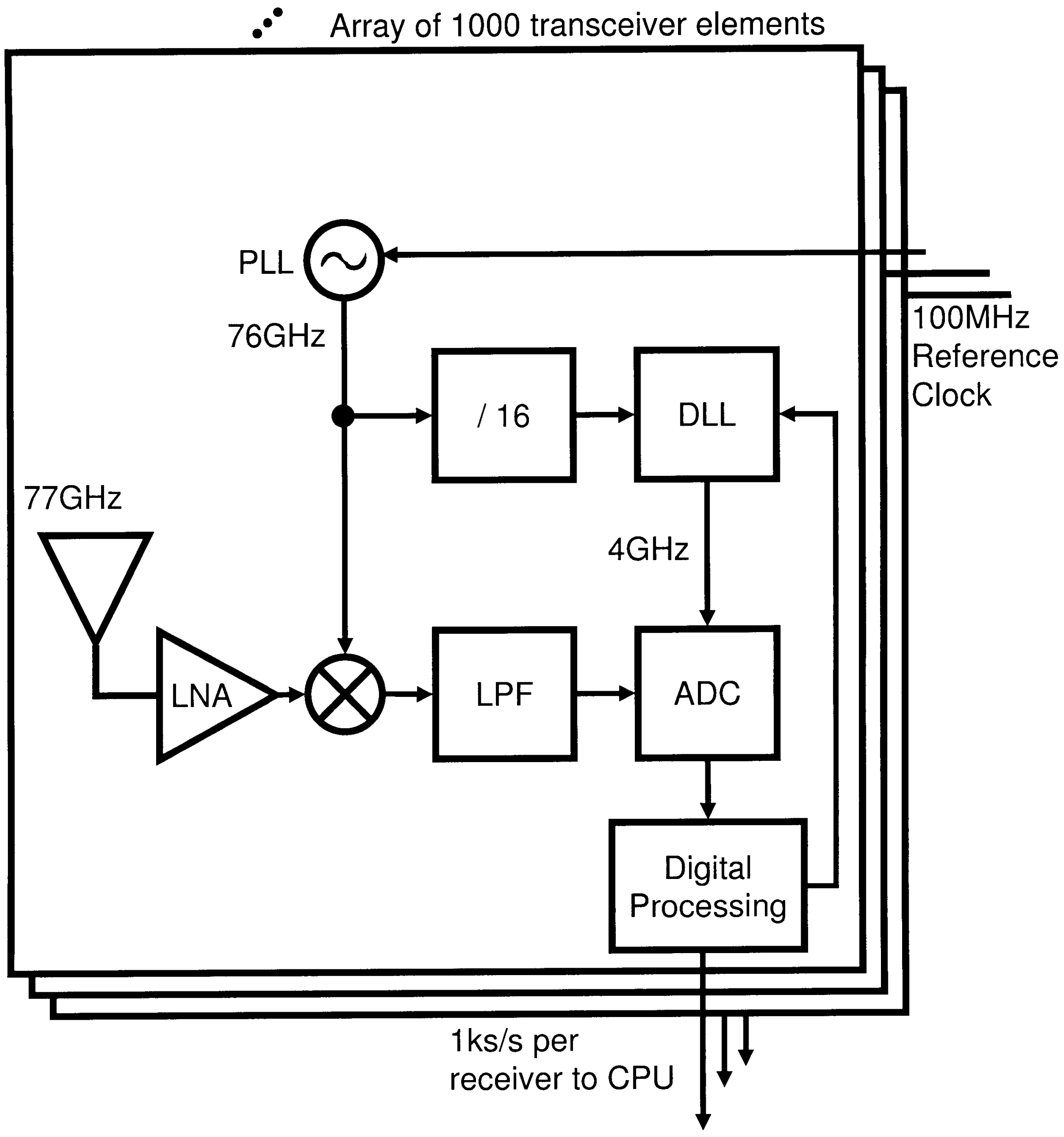


Figure 2-8: MMW active imaging system. One transmitter and 1000 receiver elements all share a 100MHz reference clock.

lock. By itself, a single received signal represents the superposition of waves reflected towards its antenna from every point in the scene. The central processing unit (CPU) combines the information from each PAP to resolve distinct pixels. A system clock of 100MHz is used to synchronize the CPU and all 1,000 PAPs. Having a low frequency system clock circumvents the high cost of implementing an overly complicated and highly lossy clock distribution system. Furthermore, the data rate from each PAPs to the CPU of 1ks/s avoids the use of power-hungry circuitry for digital transmission at high data rates across long distances. By reducing the data rate from 4Gs/s to about 1ks/s in each of the 1000 elements, the digital blocks attempt to reduce the noise using redundancy before sending the data to the CPU. This method of oversampling for noise reduction is described in the next section.

2.3 Averaging for Noise Reduction

The input signal has phase noise $\Phi(t)$ and amplitude noise $a(t)$ is shown in Equation 2.3. All of the noise sources in this chaper are assumed to be white with zero mean.

$$(1 + a(t)) \cdot A \cdot sin(2\pi f \cdot t + \phi + \Phi(t)) \tag{2.3}$$

One of the biggest advantages, following from the simplicity of extracting ϕ is the ability to average the phase information to reduce the error in measurement. Similarly, averaging can reduce the amplitude noise of the measurement of A. However, phase noise shifts the sampling off of the peak amplitude, which always results in a measured value smaller than the actual A. Another method is to ignore averaging and use only the maximum and the minimum values to calculate A; this method is most accurate if amplitude noise is assumed to be insignificant. The best approach to measuring A will be determined in future work.

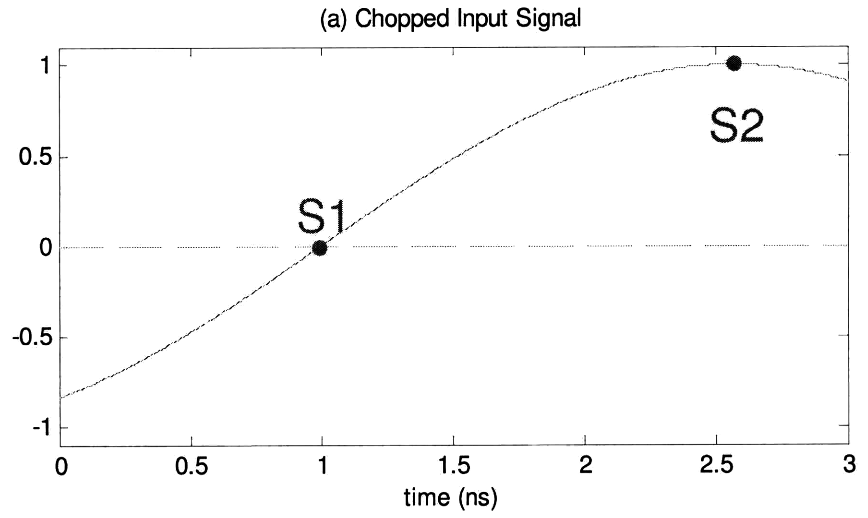
Noise in the system causes erroneous measurements for ϕ as shown in Figure 2-9. One noise source of concern is phase noise from the PLL. The current PLL has the following profile for phase noise: -85dBc/Hz at 1MHz offset from 76GHz, -90dBc/Hz at 10MHz offset, and -120dBc/Hz at 100MHz offset. This corresponds to a jitter of $\sim$0.4ps. Assuming the noise source is white, then averaging L samples reduces the standard deviation of the noise by $\sqrt{L}$. Given the 1ks/s data rate and the 4Gs/s sampling rate, where half of the samples are used for phase estimation, L is 2,000,000. This averaging reduces the 0.4ps jitter from the PLL into an effective jitter of 280as. However, there are physical limits that prevent the DLL resolution to be lower than 10ps, which introduces a limitation on the accuracy of the phase estimation. The following section will describe this limitation and how it can be resolved.

2.4 Using Noise to Improve Phase Estimation

The finite step size of the DLL introduces deterministic quantization errors. In a signal without noise the quantization error is the distance from ϕ to the nearest DLL step. This is illustrated in Figure 2-10 with the step size normalized to 1. When the DLL is shifting its output by 0, call this state 0, it will detect that its output needs to be delayed to move towards ϕ; thus, in the next cycle the DLL will shift its output by 1, call this state 1. In state 1, the DLL needs to advance its output to move towards ϕ. In other words, state 0 transitions to state 1 and state 1 transition to state 0 each time with probability 1 as shown in Figure 2-11. Since the DLL is forced to increment or decrement every cycle, the quantization error alternates between $-\phi$ and $1 - \phi$. For any value of ϕ between 0 and 1, the estimated value for ϕ will be the average of the DLL steps which is 0.5. If the step sizes are 17.5ps then the worst quantization error is $\pm$8.75ps which occurs when ϕ is very close to 0 or 1.

If the signal has a noise $n(t)$ with a small peak-to-peak value then $n(t)$ has no effect on the phase estimation due to the quantization error. As a result, averaging does not improve the phase estimation when the noise is much smaller than the quantization level. As shown in Figure 2-12, the case with small noise results in the same scenario as Figure 2-11.

If $n(t)$ is uniformly distributed between -0.5 and 0.5 then more than two states will be reached for any value of ϕ. When enough samples are taken, the expectation of the phase estimate equals ϕ. Given the example in Figure 2-13 where $\phi = 0.25$,



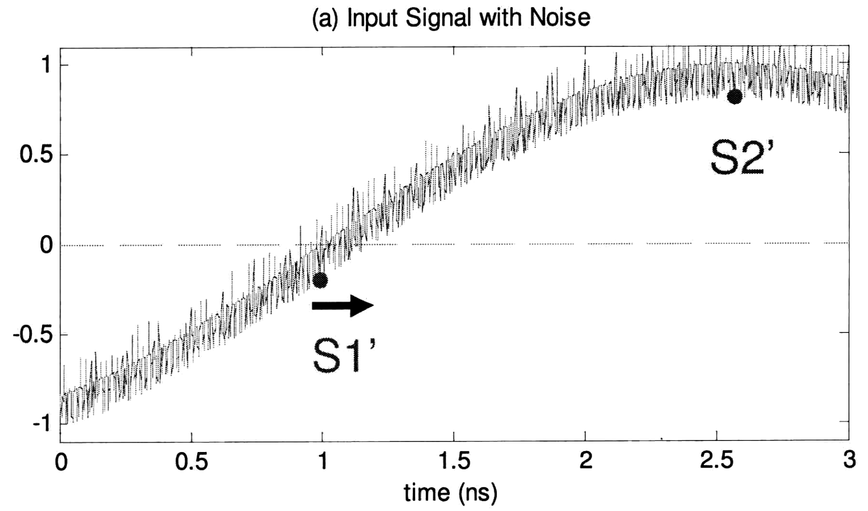


Figure 2-9: Noise effecting phase and magnitude measurements (a) Ideal input sampled with no noise indicates that the sampling clock's phase offest θ should not be changed (b) Noise changes the sample values from the ideal case and indicates θ should be increased. Furthermore, the amplitude measurement also changed.

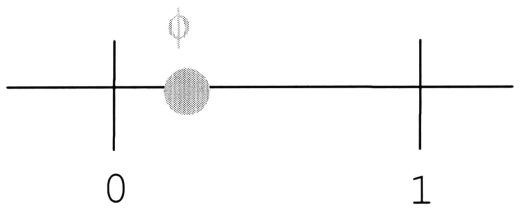


Figure 2-10: True value of ϕ lies between quantization levels 0 and 1

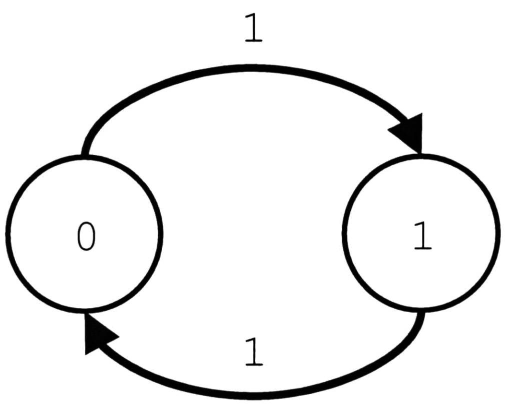


Figure 2-11: When there is no significant amount of noise present, the measured value of ϕ alternate between 0 and 1 every cycle

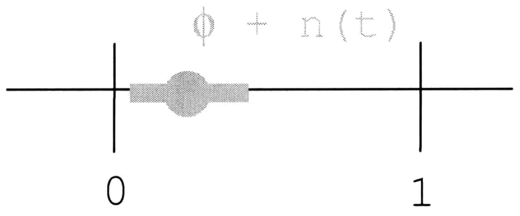


Figure 2-12: Value of ϕ with a small noise $n(t)$ lie between quantization levels 0 and 1

the DLL reaches state -1 in addition to 0 and 1. Figure 2-14 illustrates that at state 0 the probabilities of the next state being -1 and 1 are 0.25 and 0.75 respectively. Since states -1 and 1 always transition to state 0, half of the DLL steps are at 0. Furthermore, state 1 is expected to be reached three times more than state -1. Equation 2.4 shows that the expectation of the phase estimate equals ϕ as quantization error has been reduced to zero. When the peak-to-peak value of the uniformly distributed noise is not 1 or another positive integer, the quantization error will not be completely eleminiated [6].

$$.125 \cdot -1 + .50 \cdot 0 + .375 \cdot 1 = .25 \tag{2.4}$$

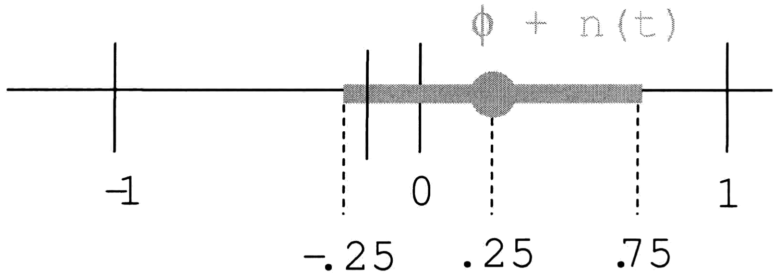


Figure 2-13: Value of ϕ with a larger noise $n(t)$ lie between quantization levels -1, 0, and 1

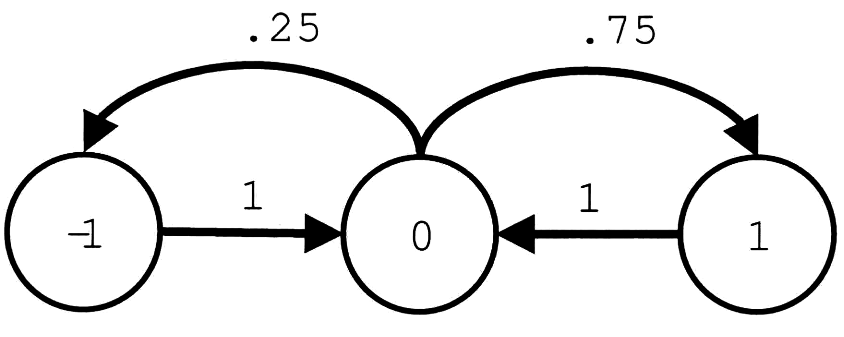


Figure 2-14: With an increased noise, the measured value of ϕ switches between -1, 0, and 1 randomly

The total noise contribution from the input signal, the ADC, and the DLL is usually enough to correct the quantization error [7]. A process called dithering can be applied to intentionally add more noise to ensure that quantization errors are reduced. The noise in a real system has a Gaussian distribution. When the root mean square (rms) jitter of a Gaussian noise source is 0.5 steps or greater, quantization error is essentially eliminated and the system noise can be reduced by averaging [7]. Figure 2-15 shows the calculated quantization error for values of ϕ between 0 and 1 when the system has a Gaussian noise source with a rms jitter of 0.5 steps or 8.75ps. Figure 2-16 shows the quantization errors when the rms jitter is 0.35 and 0.45 steps for comparison. The quantization error decreases exponentially when the rms jitter changes from 0.35 to 0.5 steps. By averaging 2,000,000 samples of a signal with 8.75ps rms jitter, the phase estimation will have a jitter of 6.2fs and a quantization error determined by Figure 2-15. Note that the uncertainty of the phase estimation, represented by the jitter, is significantly larger than the deterministic offset, which is equivalent to the quantization error.

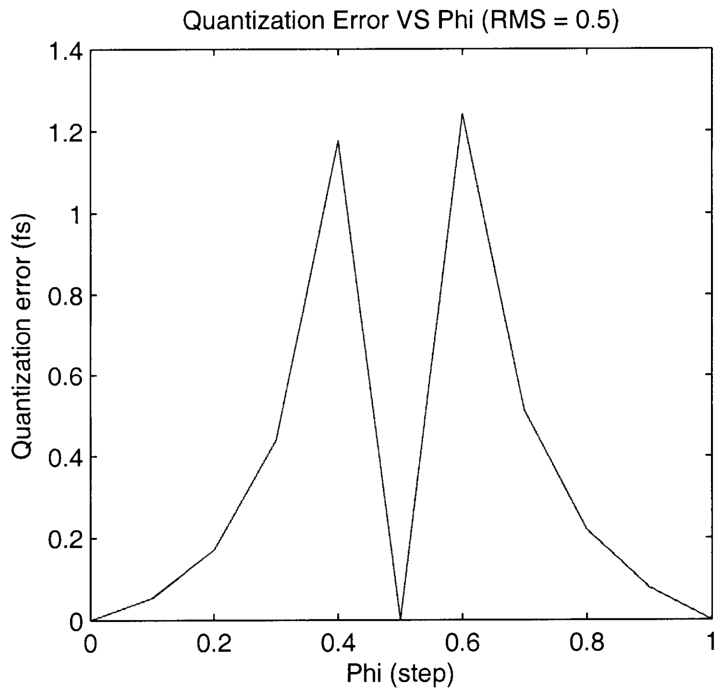


Figure 2-15: Calculated quantization error of the phase estimations for values of ϕ between 0 and 1 with a Gaussian noise source with rms jitter of 0.5 steps

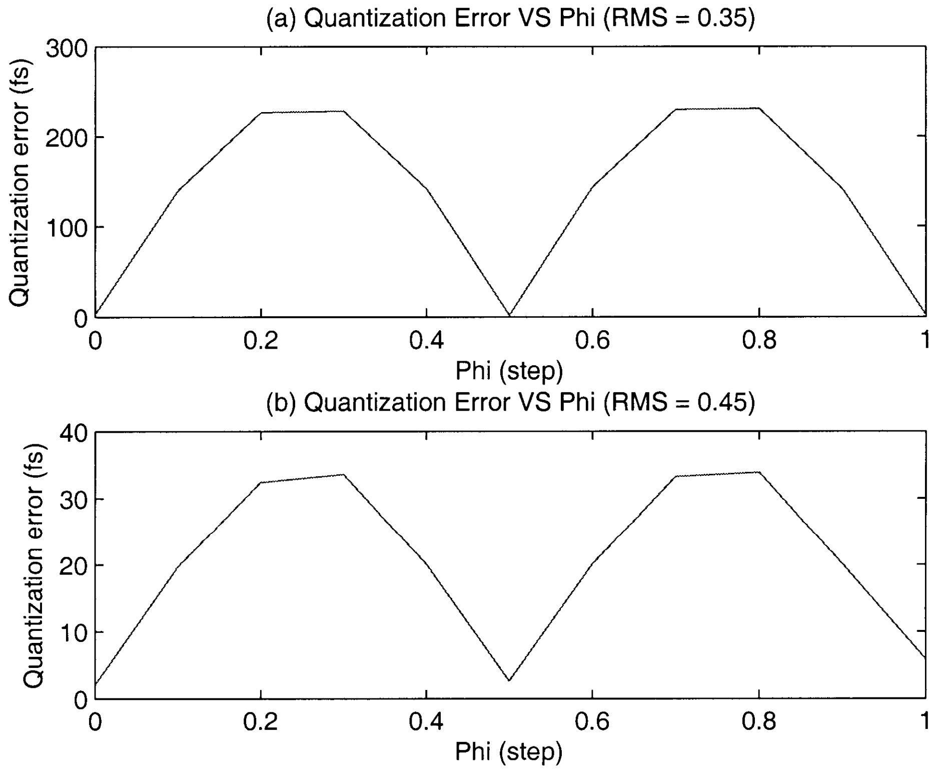


Figure 2-16: Quantization error of the phase estimations for values of ϕ between 0 and 1 when the Gaussian noise source has a rms jitter of (a) 0.35 steps (b) 0.45 steps

Chapter 3

Proof of Concept Integrated Circuit for Phase Tightening

The system incorporates enough abstractions to allow the key blocks to be designed and analyzed separately. The DLL, the ADC, and the digital processing block are designed and implemented for a proof-of-concept system, as shown in Figure 3-1. Due to constraints placed by the technology, the input signal to the DLL is reduced to 2.4GHz and the input signal to the comparator is reduced to 0.6GHz. The clock is derived by dividing the output from the 76GHz PLL; however, a signal generator will be used to produce the 2.4GHz signal. Furthermore, the ADC is replaced by a 1-bit comparator that samples at 1.2Gs/s. Sampling at the Nyquist frequency, the system, using the method described in the previous section, is able to find the zero crossings to determine the phase information.

3.1 DLL Operation And Architecture

A close cousin of the PLL, the DLL consists of three main blocks: the delay line, the phase detector (PD) and the low pass filter (LPF) as shown in Figure 3-2. Normally, the DLL is used to buffer the incoming clock signal; thus, giving only one output. The DLL operates by passing the input signal through a delay line and comparing its output to the input signal. The PD and LPF extracts this error signal to control the

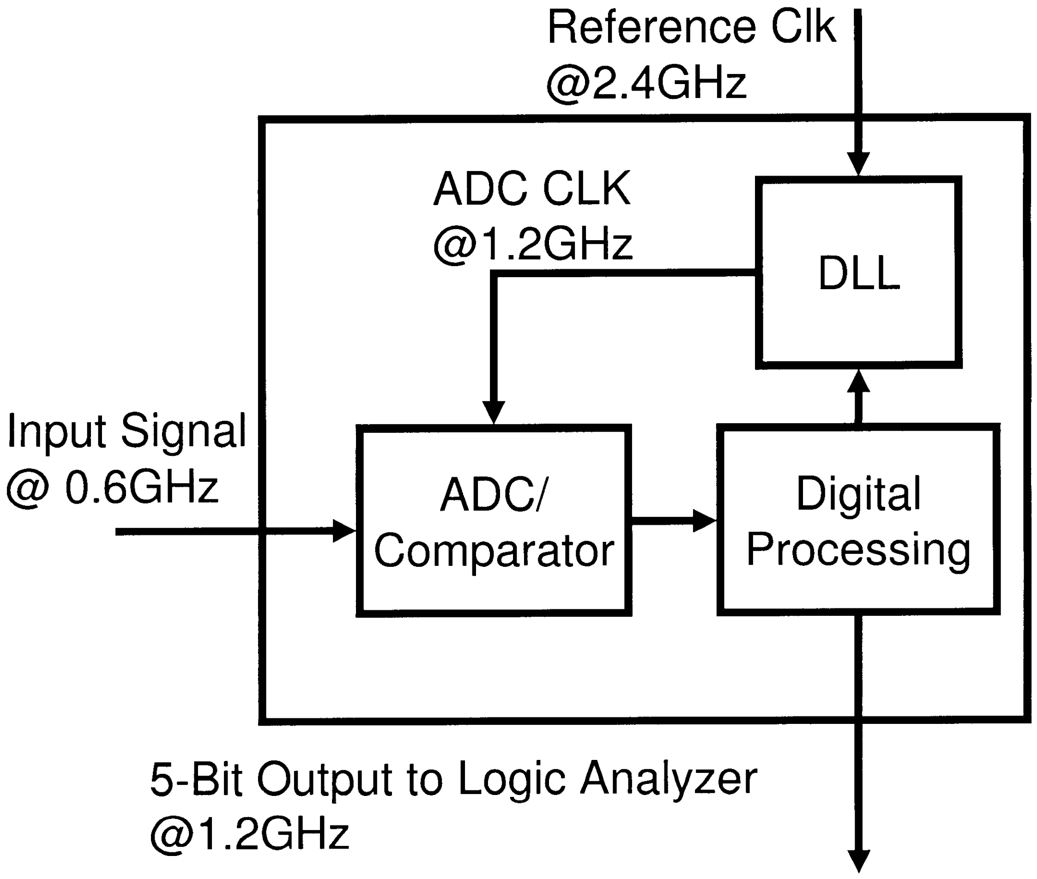


Figure 3-1: Block diagram of the implemented system on chip. External 2.4GHz signal is used for the reference clcok and external 0.6GHz signal is used as the comparater's input. The system outputs 5 bits at 1.2GHz

delay through the delay line. The standard architecture can be modified for other applications.

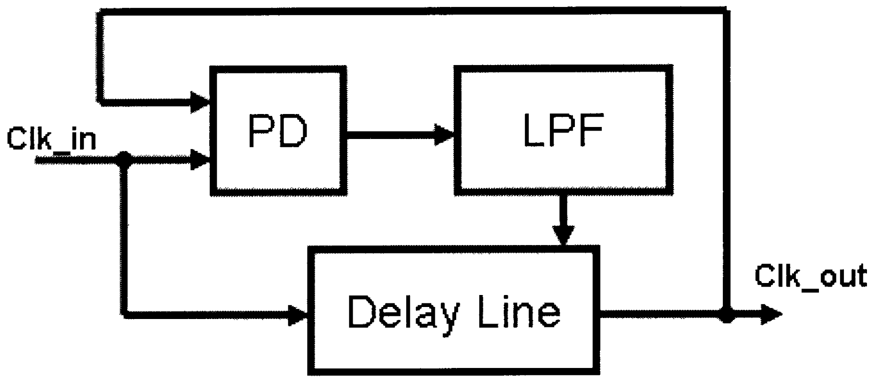


Figure 3-2: Traditional DLL architecture with phase detector, low pass filter, and delay line

Our application requires selecting from multiple phases; thus an architecture allowing multiple outputs is required. The modified architecture shown in Figure 3-3 is implemented [8]. The core DLL is almost identical to the canonical DLL shown in Figure 3-2; however, the delay line outputs shifted versions of the input at 24 different phases. Each phase is $\pi/12$ radians or 17.5ps from the next one. The multiplexer then passes a selected phase to the frequency divider, which buffers the output signals and sets its duty cycle to fifty percent [9].

The delay line divides the 420ps long period of a 2.4GHz input signal into 24 equal sized steps of 17.5ps, where each step is selected by a unique digital code of the DLL. However, the DLL output has a 840ps long period and a 1.2GHz frequency; 24 steps of 17.5ps each cannot cover the entire 840ps period. Therefore each digital code corresponds to two delay steps that are π radians apart at 1.2GHz. However, the delay steps progress incrementally and wraps around; thus, combining the history of the digital codes with the current digital code uniquely determining the delay step.

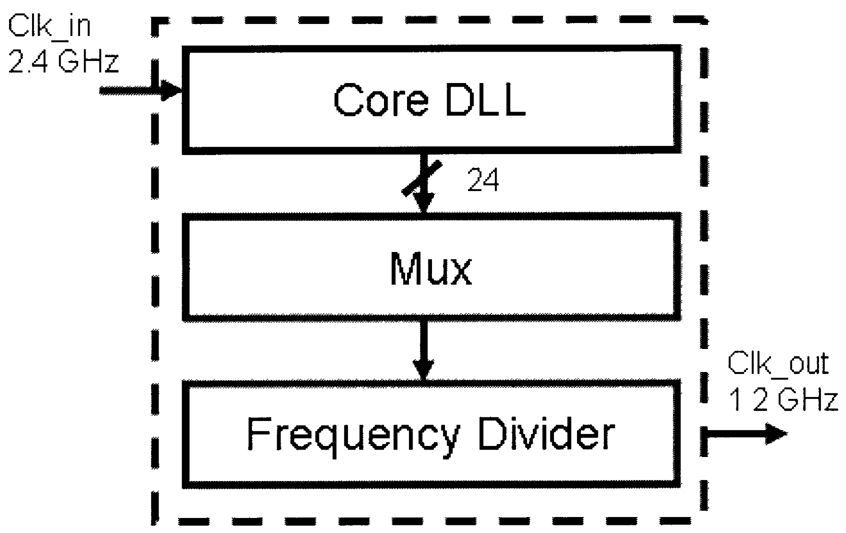


Figure 3-3: Modified DLL architecture using a multiplexer to select different phase offsets from the reference signal and a frequency detector to generate 50% duty cycle at the output

3.1.1 Delay Element And Delay Line

The delay elements are four current starved inverters; the pseudo-differential configuration is as shown in Figure 3-4 and the detailed transistor level schematic is show in Figure 3-5 [4]. The control voltage sets the maximum amount of current allowed in the NMOS transistor. Current I_D is related to propagation delay t_{delay} by Equation 3.1 [10]. Equation 3.1 models the propagation delay as the time it takes I_D to charge or discharge a load capacitor C to a fixed voltage $V_{DD}/2$.

$$t_{delay} = \frac{C \cdot V_{DD}/2}{I_D} \tag{3.1}$$

Since only the pull down current is affected, there is asymmetry between the rise time and the fall time. Asymmetry causes skew between the positive and negative terminals, which results in unpredictability for the propagation time. Tying both outputs to each other through a pair of secondary inverters reduces this problem. Figure 3-6 shows that passing a signal with an initial skew through two delay elements significantly reduces the skew at the output [11]. A smaller ratio between the size of the primary inverter pair and the size of the secondary pair can correct more skew;

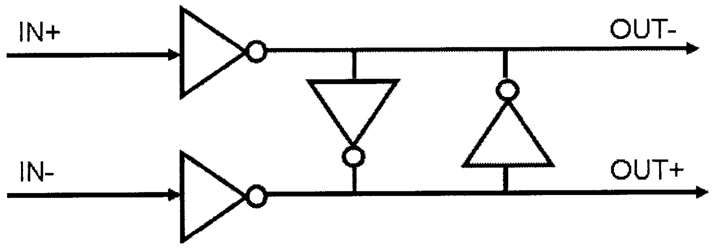


Figure 3-4: Schematic view of delay element with cross coupled inverters at the output for positive feedback

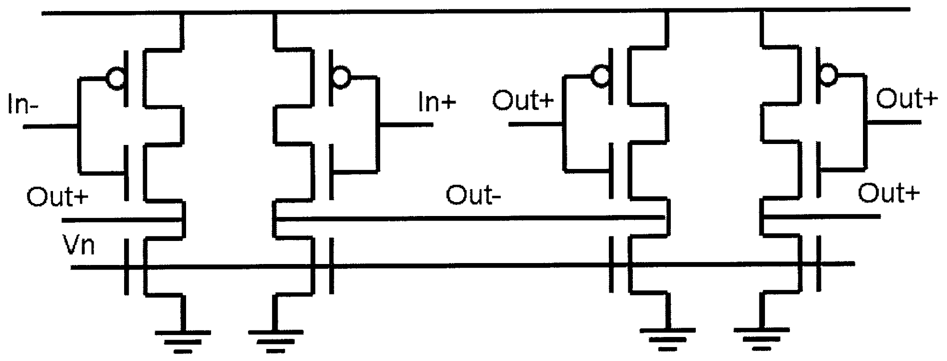


Figure 3-5: Transistor view of delay element

however, a larger secondary pair presents more loading and slows the propagation time. A ratio of four to one is chosen to allow for a fastest propagation time, when the control signal is at V_{DD}, of 16.5ps. This is only slightly slower than 12.5ps, which is the propagation delay of the unit size inverter. The input signal has a swing of V_{DD}, 60ps rise and fall times, and variable skew.

A chain of two delay elements can also be used to buffer weak clock signals. Figure 3-7 shows the rise and fall times at the output when the input signal has a varying rise and fall time. Figure 3-8 shows the amplitude at the output when the input signal has a varying amplitude.

The output of the buffers feed into the delay line. Buffering is essential because

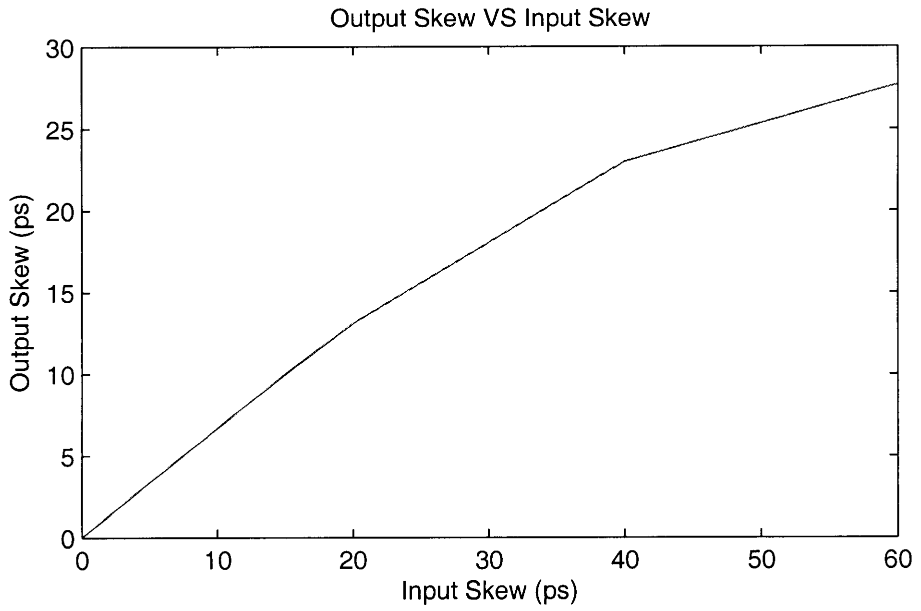


Figure 3-6: The amount of skew reduction by the delay element at its output when the differential input is skewed

the signal should be identical through each delay element of the delay line. The delay line, as in Figure 3-9, is a chain of N delay elements. When the DLL is in lock, the output of the last delay element should have the same phase as the input to the DLL and the resolution is of the form Equation 3.2. Also, the phase offset after the xth delay element is $x \cdot 2\pi/N$ radians or $x \cdot 420/N$ ps for 2.4GHz.

$$t_{delay} = \frac{period}{N} \tag{3.2}$$

The delay line is the main source of noise in the DLL. The total noise contribution from the delay line scales linearly with the number of delay stages [12]. Equation 3.3 is used to estimate the noise generated by the delay line [13]. On the other hand, the resolution of the delay line is inversely proportional to the number of delay elements. The incremental increases in noise eventually outweigh the improvements in resolution. Using 24 stages, the delay line has a resolution of 17.5 ps and contributes 3.7 ps of jitter.

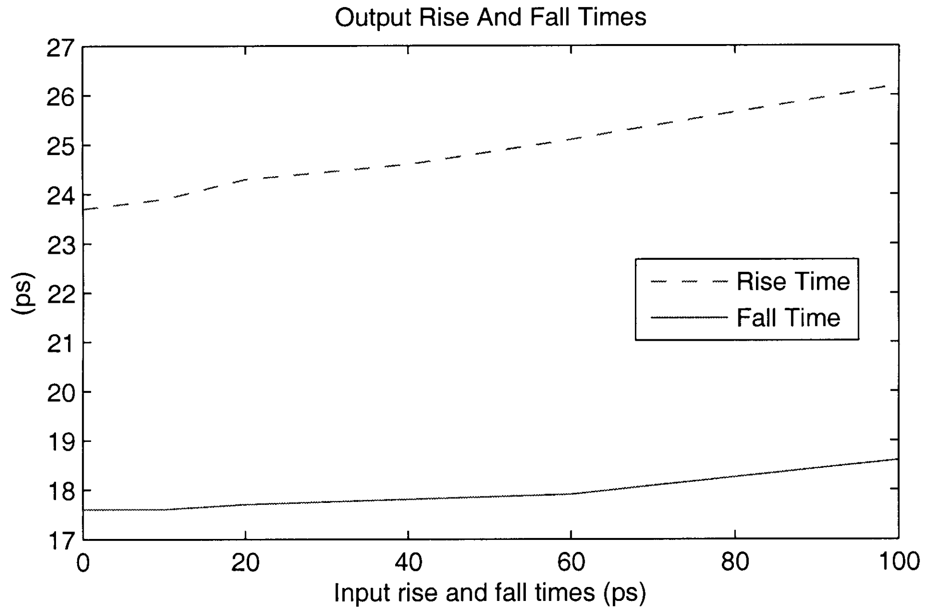


Figure 3-7: The effect of the edge transition rate of the input on the rise and fall times of the output after buffering by the delay element

$$t_{jitter,total} = N \cdot t_{jitter,stage} = N \cdot \sqrt{\frac{2kT\gamma r_o C}{\frac{W}{L}\mu C_{ox}(V_{DD} - V_T)^2}} \tag{3.3}$$

There are additional limits on the resolution of the DLL. Equation 3.2 shows the resolution cannot be smaller than the smallest possible t_{delay}. However, there is a technique, phase interpolation, that allows for the averaging of the phases from the delay lines [14]. Unfortunately, the phase is determined by combining the currents from different delay elements, which produces high noise and non-linearity.

Furthermore, 24 can be factored as in $24 = 2 \cdot 3 \cdot 4$. This allows for the implementation of a cascade of a 2:1, 3:1, and 4:1 multiplexers with buffers in between. A chain of multiplexers is much more robust than a N:1 multiplexer.

Lastly, Figure 3-10 shows the delay of each delay element and the locking frequency for the DLL as a function of the control voltage. The DLL is flexible enough to work in the range of frequencies between 400MHz and 2.5GHz. However, note that with decreasing frequency the charge pump (CP) pumps current over a longer time during each cycle, which results in greater control voltage changes and greater frequency

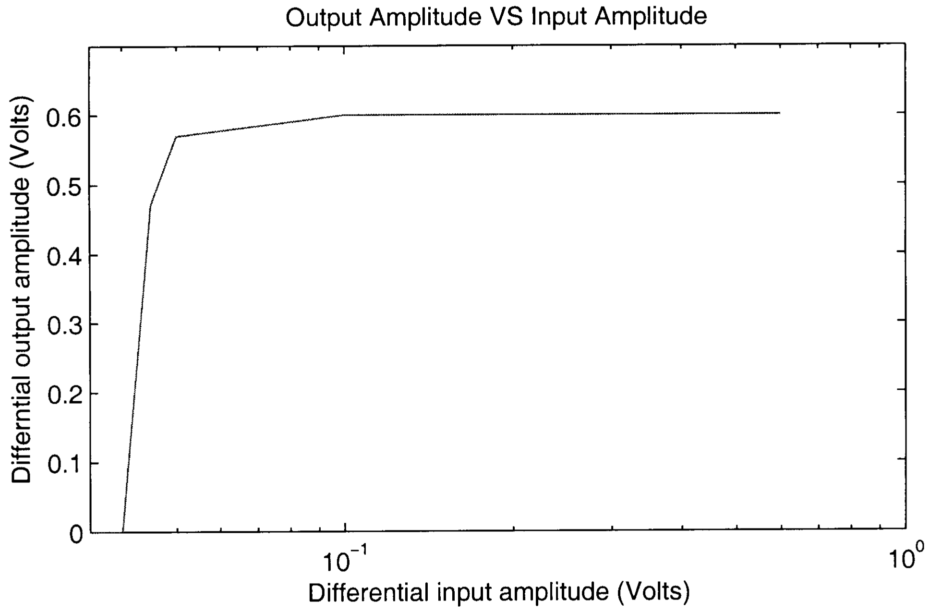


Figure 3-8: The effect of the amplitude of the input on the ouput swing after buffering by the delay element

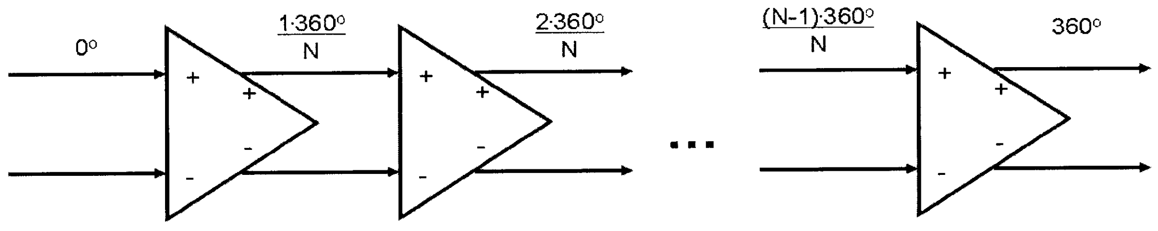

0°
$\frac{1\cdot 360^\circ}{N}$
$\frac{2\cdot 360^\circ}{N}$
$\frac{(N-1)\cdot 360^\circ}{N}$
360°
+
-
...

Figure 3-9: Delay line: a chain of identical delay elements

changes. Eventually, the frequency changes so greatly that the DLL overshoots the phase correction and falls out of lock; thus, the DLL cannot be slowed down to operate at any arbitrary low frequency.

3.1.2 Phase Detector, Charge Pump And Low Pass Filter

The PD, CP, and LPF comprise the feedback control portion of the DLL. In a simple DLL, an exclusive-or (XOR) gate is used to compare the delay line output to the input signal. When the XOR gate is used to compare two signals, the smallest delay difference it can resolve is larger than the gate's propagation delay [15]. The fastest

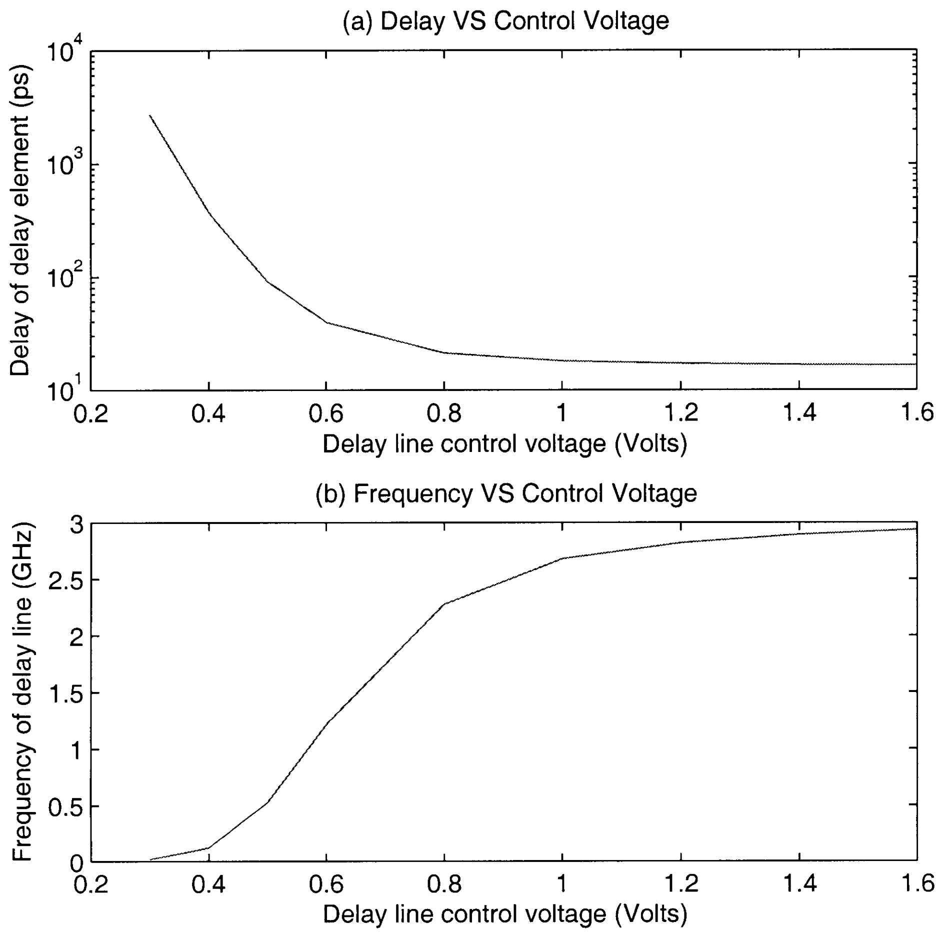


Figure 3-10: (a) Propagation delay of single delay element with changing control voltage
(b) Resulting frequency of delay line due to changes in the control voltage of the delay elements

combinational gate, the inverter, has a propagation delay or 12.5ps; thus the XOR gate, significantly slower than the inverter, cannot resolve delay differences that are close to the step size of the DLL.

A faster phase detector can be constructed using source-coupled logic (SCL). A latch constructed using SCL, shown in Figure 3-11, can detect a small phase difference because it is differential and uses current switching instead of relying on a voltage signal [16]. When the CLK signal is high, the circuit functions as an ordinary differential amplifier with the output value changing with the input. When CLK_bar is high, the circuit is in the hold stage. The two cross coupled NMOS transistor are in positive feedback and hold the output value constant. These phase detectors can detect phase differences as small as 4ps, which is more than three times smaller than the resolution of the XOR gate.

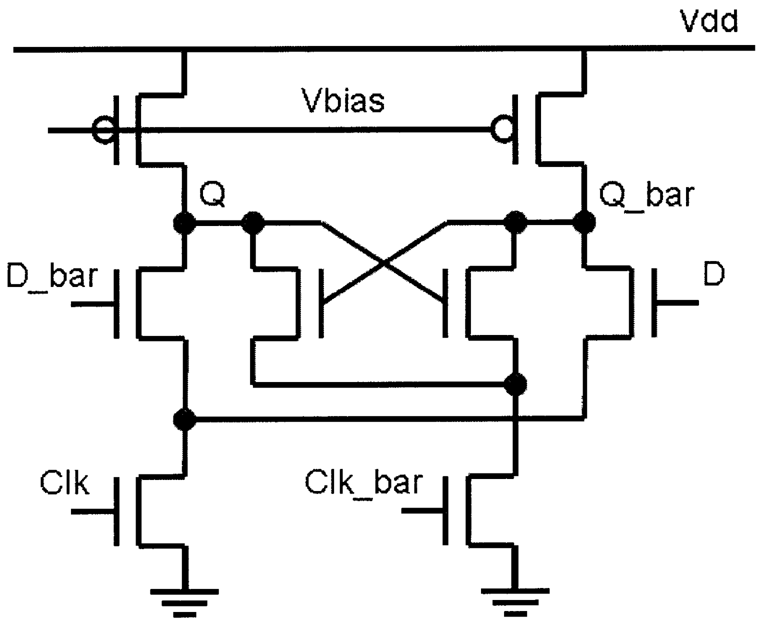


Figure 3-11: SCL latch

The ideal function of the PD is described as follows. Two latches are constructed, one for the UP signal and one for the DOWN signal. When the DLL input is connected to CLK and CLK_bar of the latch and the last delay element's output V_{last} is connected to the differential input of the latch, D and D_bar, a faster V_{last} generates a voltage high for DOWN. DOWN remains high until the next positive edge of the DLL input. And when DOWN is high, the CP decreases the voltage feeding the LPF, which increases the propagation delay of each delay element and the total propagation delay of the delay line. Eventually, the delay line is slowed so that V_{last} matches the input to the DLL. However, when the input signal is leading then the latch remains unresponsive. The latch generating the UP signal is analogous with the exception that the complimentary output of the latch is used as UP. When the phase difference becomes too small to be detected by the phase detectors, the DLL enters the tristate stage where the delay between V_{last} and the input remains constant. This is usually in the neighborhood of 3 to 4ps.

The outputs of the SCL latches drive the input of a single-ended charge pump. Although differential charge pumps have the advantages of low switch mismatch, higher output voltage range, and better immunity to supply and substrate noise, the single-ended topology is chosen because, in addition to having a lower power consumption during tri-state operation, it does not require an additional loop filter and common mode feedback circuitry [17]. Furthermore, mismatch between the pull-up PMOS and the pull-down NMOS has minimized effect to this situation because the rate of frequency change over one output cycle is low such that any given cycle will not have a significant impact on the output frequency of the DLL.

Figure 3-12 shows the topology of the chosen charge pump design. Transistors M1 and M6 are for active control of the charge pump. The switches are placed at the source to force both devices to always be in saturation, which allows better matching. Additionally, this topology switches faster because each input drives only a single transistor so that there is lower parasitic capacitance.

When both of these devices are off, the CP enters tri-state operation where power is saved because there is no dynamic current flowing from supply. However, there is a maximum leakage current of 400nA present through transistors M1 and M6. The voltage value changes by only one to two mili-volts during a cycle when the CP is active and driving 8μA; therefore, the leakage current has a trivial effect on the output voltage.

Transistors M2 through M5 are a part of the current mirror network which is cascaded to increase the output impedance so that the current variation is less sensitive to the output voltage. Lastly, C1 and C2 are added to reduce the charge coupling to

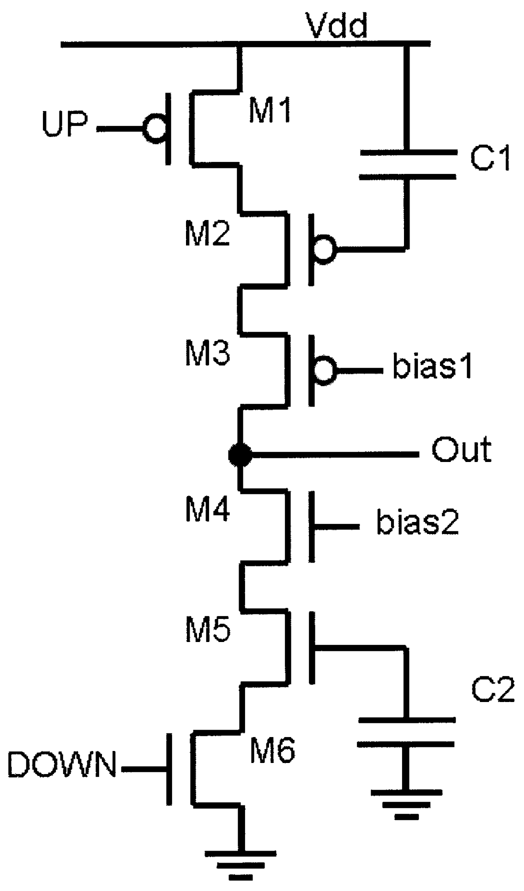


Figure 3-12: Charge pump

the gate and to help enhance the switching speed.

In addition to the CP, an off-chip signal RST can also affect the control voltage for the delay line. RST is used to initialize the control voltage at V_{DD} such that the bottom transistor in each delay element is in strong inversion at the start of operations. RST achieves this by driving a large pull-up PMOS transistor into strong inversion. Once RST is released, the CP and the rest of the system will adjust to lock onto the input signal. Simulations show that locking occurs in less than 200 clock cycles or 100ns as shown in Figure 3-13. Furthermore, due to the sensitivity of the PD and the level of the leakage current in tri state, the DLL remains in lock indefinitely. However, when it falls out of lock, it will return to the locked state in fewer clock cycles than the initial locking time.

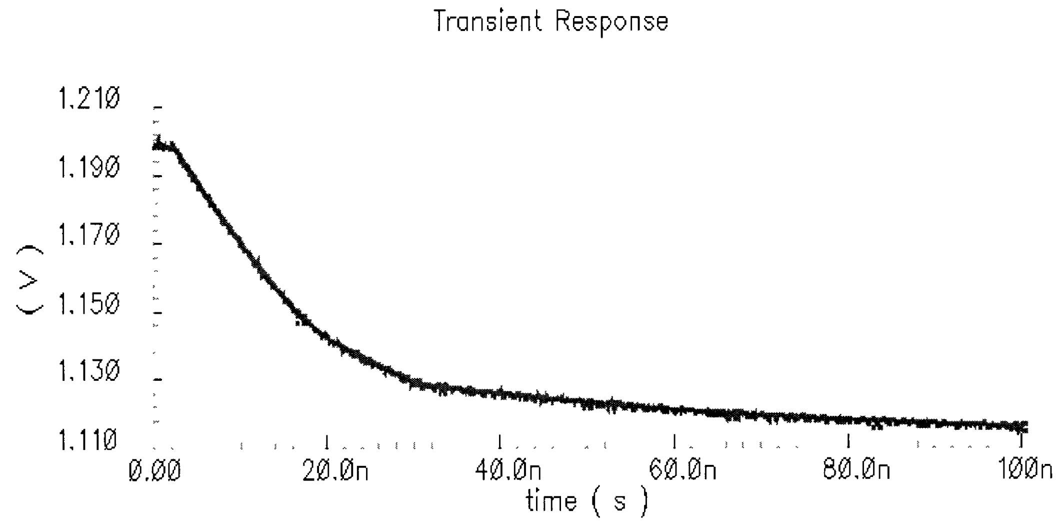


Figure 3-13: After the RST signal is released at 10ns, the control voltage of the DLL settles within 10% of its final value after 40ns

A second order LPF, shown in 3-14, integrates the output of the CP and converts it to the control voltage for the delay line. The loop filter is a complex impedance in parallel with the input capacitance of the delay line; however, C1 and C2 are sized 500fF and 100fF respectively such that they dominate the capacitance seen at this node. The total gate capacitance from the transistors on the delay line is 150fF. Furthermore, C1 and C2's sizing also mitigate the affect of leakage current; thus, reducing the risk of the DLL drifting out of lock. The disadvantage of higher capacitance is that it increases the time required for locking. As stated earlier, this is not essential as the DLL only needs to lock once. The shunt capacitor C1 is placed to remove discrete voltage steps at the control voltage of the delay line due to instantaneous changes in the CP current output.

3.1.3 Frequency Divider for Duty Cycle Correction

As stated earlier, the primary use of the DLL is to generate a buffered clock signal with a 50 percent duty cycle. Due to the changing rise and fall times of the delay

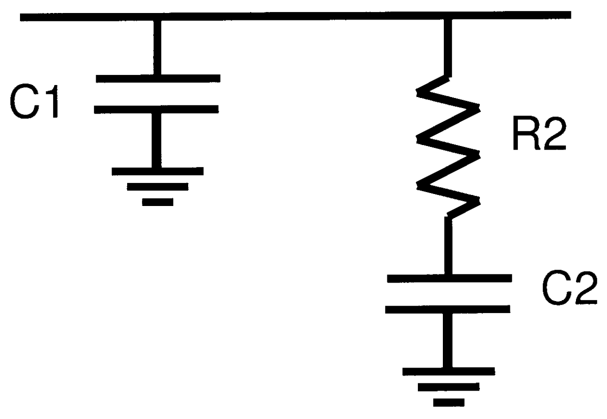


Figure 3-14: Low Pass Filter

elements, the output of the delay line can stray far from a 50 percent duty cycle. The conventional DLL uses a duty cycle correction (DCC) circuit to address this problem. The DCC circuit works by comparing points on the rising edge of the output against a threshold value via an amplifier. However, a signal with a rise time smaller than 20ps is too fast to be adjusted by a typical DCC circuit.

Taking advantage of a higher frequency reference source divided down from the 76GHz PLL, a frequency divider replaces the conventional DCC circuit. The delay line drives the frequency divider directly and every positive edge from the delay line cause the frequency divider's output to toggle. Thus, the frequency divider's output will remain high for one period of the delay line's output and remain low for one period of the delay line's output also. Since the delay line's output is periodic, this guarantees that the frequency divider outputs a signal with a 50 percent duty cycle. This system varies from the conventional DLL in that the frequency of the final output is half the frequency of the input instead of being equal to it.

The schematic of the frequency divider is shown in Figure 3-15. It is two latches tied together in the standard configuration. SCL latches, identical to the ones used for phase detection, are used because a static CMOS divider cannot operate at 2.4GHz [18]. One disadvantage of the SCL divider is that the circuit has static current. While

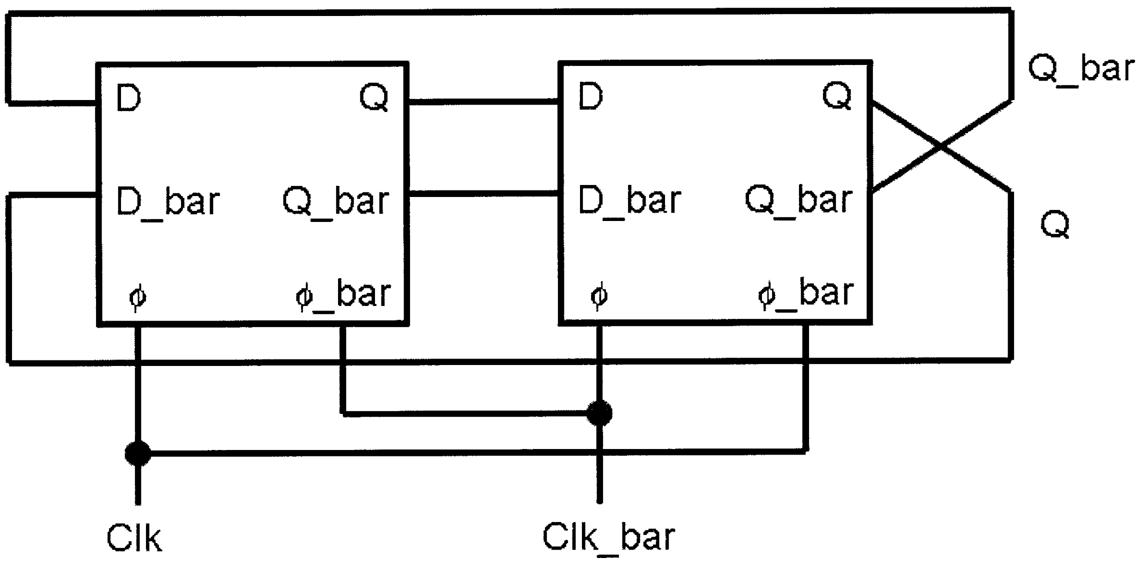


Figure 3-15: Frequency divider

operating at 2.4GHz, the SCL divider consumes 7.48mW. Figure 3-16 demonstrates the effectiveness of the divider at fixing the duty cycle. Lastly, the divider adds flexibility to the system as it can operate with an input in the frequency range of 10MHz to 20GHz.

3.2 Comparator

A regenerative latch, as shown in Figure 3-17, is used as the comparator to measure the input signal. Often, a preamplifier with low gain, 2 to 4, accompanies the comparator. However, at higher frequencies, the preamplifier rarely has gain above unity, but it is necessary to remove the "kickback." "Kickback"' is when the turning on and off of transistors in the latch affect the input voltage to the latch [19]. We have opted to remove the preamplifier stage because it slows down the comparator and simulations show that kickback is insignificant. When EN is high, the latch functions as a differential amplifier with positive feedback at the output. Transistors M3 and M5 amplify the small input differential voltage. Transistors M1 through M4 amplifies the differential signal at nodes X and Y by pulling the small analog value to a digital

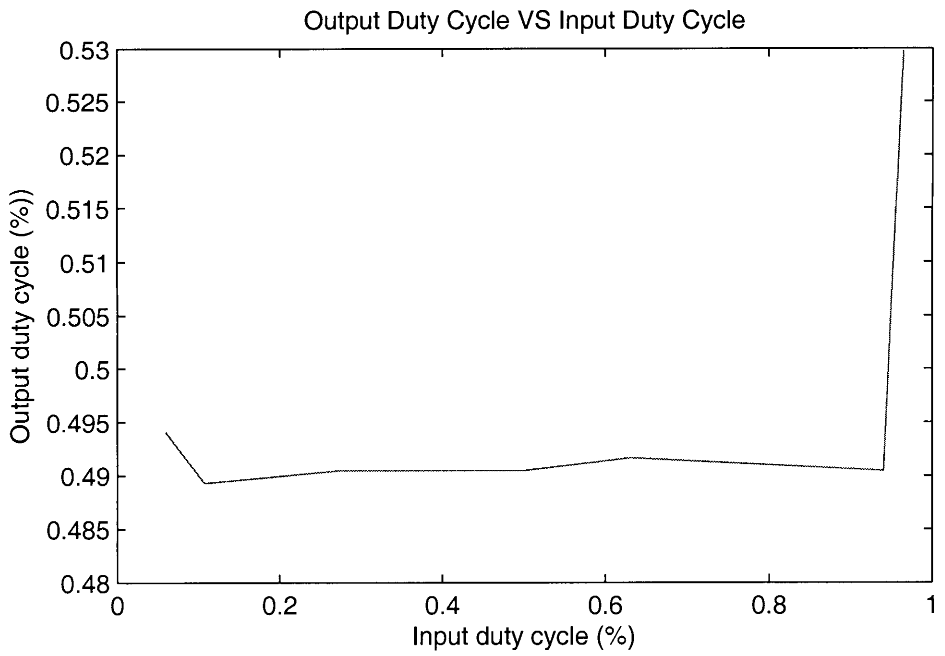


Figure 3-16: Effectiveness of using the frequency divider as a duty cycle corrector to ensure 50% dutcy cycle at the output when the input has a varying duty cycle

value. The comparator enters the reset phase when EN is low. M13, the tail current source, is turned off to preserve power. During this stage, it is important to erase the state of the comparator; otherwise, the comparator will fight against changing its outputs, causing hysteresis. To prevent hysteresis nodes X and Y are shorted together by M11 and VO and VOB are shorted together by M12. Furthermore, transistors M7 through M10 precharge the nodes X, Y, VO and VOB to V_{DD} so that they can make a fast switch at the next decision cycle. Removing memory from the comparator reduces the rate of error and also decreases the time it needs to make a decision. Lastly, low Vt transistors are used to increase the speed of the comparator.

The differential input signal is repeated in Equation 3.4 for convenience. The rate of change around the zero crossing is Equation 3.5. If Δv is set to the comparator resolution then the comparator cannot accurately measure a value sampled within $\Delta t/2$ of the zero crossing. Equation 3.6 reflects this explicitly. The comparator is unable to make a correct decision when the inputs differ by less than 20mV. For a typical value of A equal 400mV and f equal 600MHz, the 20mV corresponds to a

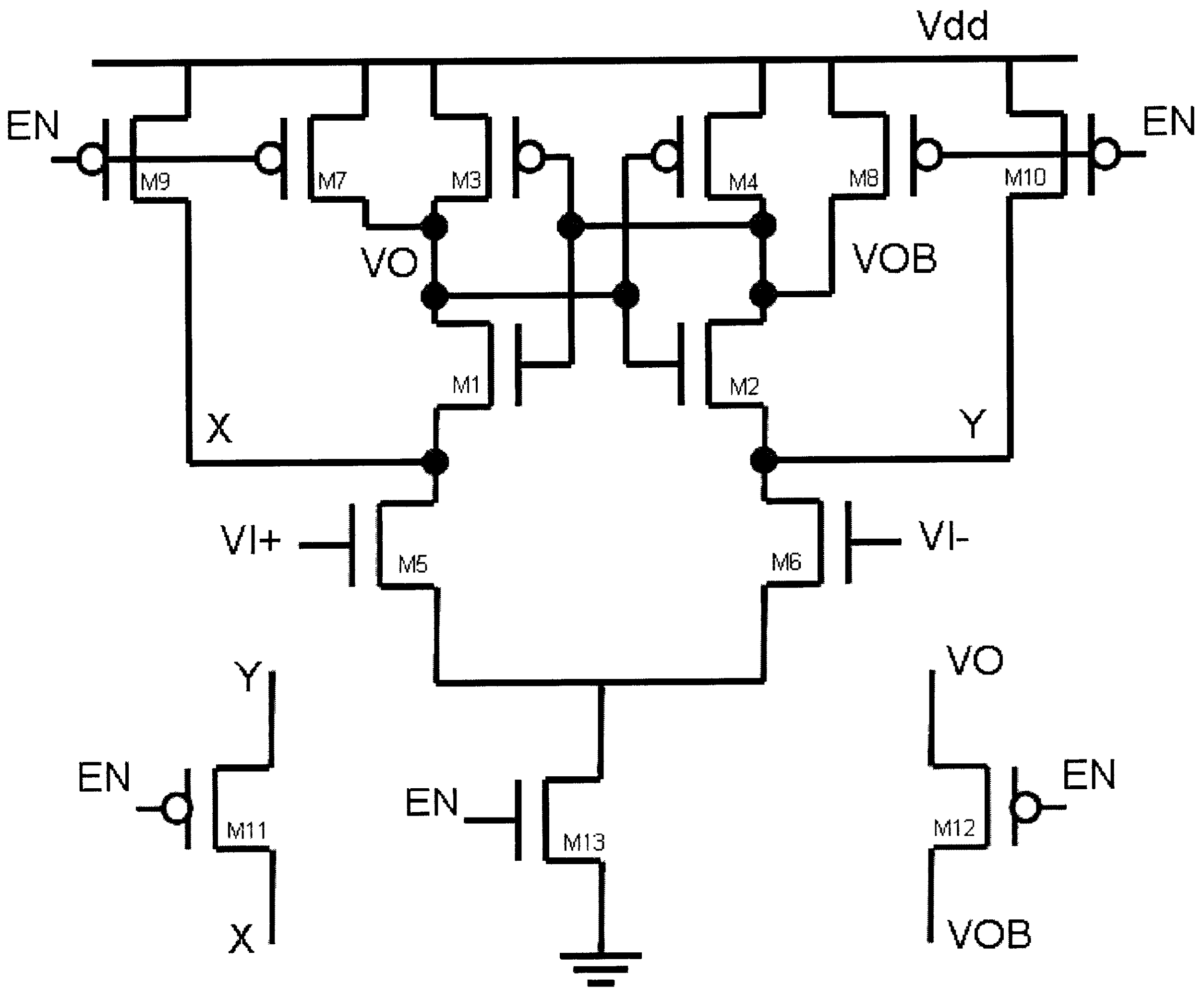


Figure 3-17: Regenerative latch as comparator

range of 13.3ps around the zero crossings that cannot be correctly determined.

$$A \cdot sin(2\pi f \cdot t + \phi) \tag{3.4}$$

$$\frac{\Delta v}{\Delta t} = 2\pi A \cdot f \tag{3.5}$$

$$\Delta t = \frac{\Delta v}{2\pi A \cdot f} \tag{3.6}$$

3.3 Digital Processing

As stated previously, one of the main advantages of digital phase tightening is the reduction of high speed power hungry digital circuitry. As shown in Figure 3-18, only adders, NAND gates, NOR gates, and multiplexers are needed to implement digital phase tightening. Implemented in static CMOS logical, the blocks operate robustly at 1.2GHz and can function at lower frequencies. A five-bit mirror cell full adder, utilizing two's complement arithmetic, is used to implement the counter that keeps the state of the DLL [20]. Every clock cycle, output from the comparator tells the counter to decrement or increment, which corresponds to decreasing or increasing the phase shift θ. Note that a five-bit adder can represent 32 different values, but the DLL has only 24 states. The logic block forces the adder to wrap around from 23 to 0 and from 0 to 23.

Additional combinational logic is used to translate the adder's output into select signals for the DLL's multiplexers. The 0 to V_{DD} digital output of the adder is also converted into differential SCL and brought off-chip. Although, the SCL buffers resolve the difficulty of bringing a 1.2GHz signal off-chip, they consume a disproportionate amount of power. All of the buffers consume a total of 160mW, while the core circuitry uses only 60mW. In the full system, off-chip data from the core circuitry can be power gated.

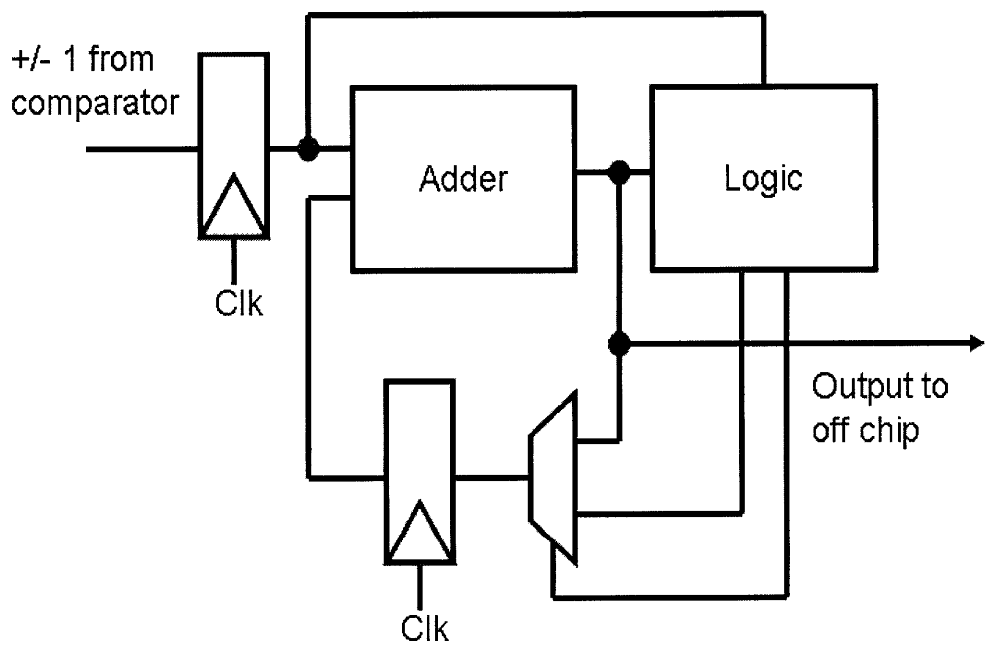


Figure 3-18: Digital circuitry includes an adder to keep the state of the DLL and combinatorial logic to wrap around from 0 to 23 and 23 to 0

Chapter 4

Testing of Integrated Circuits

The test setup for the chip fabricated in 90nm CMOS is illustrated in Figure 4-1. An Agilent dual channel 81133A vector generator generates a differential square wave at frequency f for the reference clock and another differential signal at $0.25f$ with variable delay for the input to the comparator. The output of the counter are 5bits of differential digital signals. The outputs are viewed with an Agilent DSA80000B oscilloscope. Furthermore, there is a reset signal to the DLL that pulls the control voltage for the delay line up to the supply voltage. The differential clock to the comparator is also brought off of the chip and can be viewed for debugging purposes. The chip is encased in a Quad Flat No leads (QFN) package. Lastly, the printed circuit board (PCB) for testing is shown in Figure 4-2.

The goal of the setup is to show that the digital bits track the change of the delay for the input to the comparator. The counter, using 24 states to track a period of the reference clock, wraps around at 23 back to 0. However, the 24 states correspond to one period of $1/f$, while the delay for the $0.25f$ input signal can range from 0 to $4/f$. As illustrated in Figure 4-3, the counter needs to wrap around four times to represent all possible delay values. Therefore, the number of times the counter wraps around has to be observed in order to determine the delay.

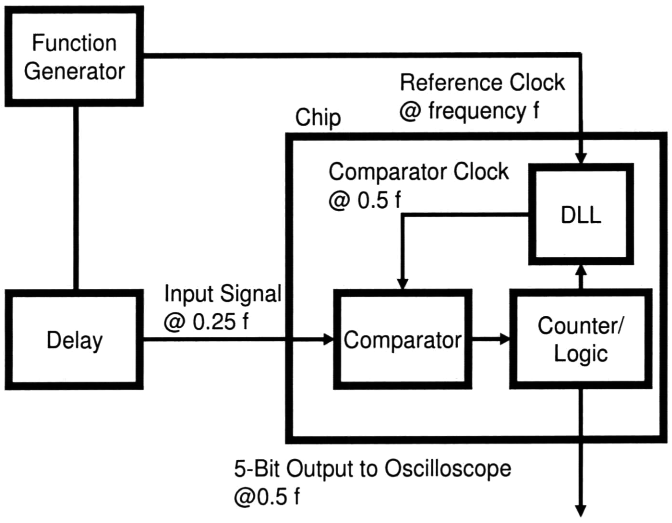


Figure 4-1: Schematic level description of the test setup with key input and output signals. An input signal at 1/4 the frequency of the reference clock is delayed and sampled by the comparator. The comparator's outputs adjust the comparator clcok and measure the delay by changing the 5bits of the counter.

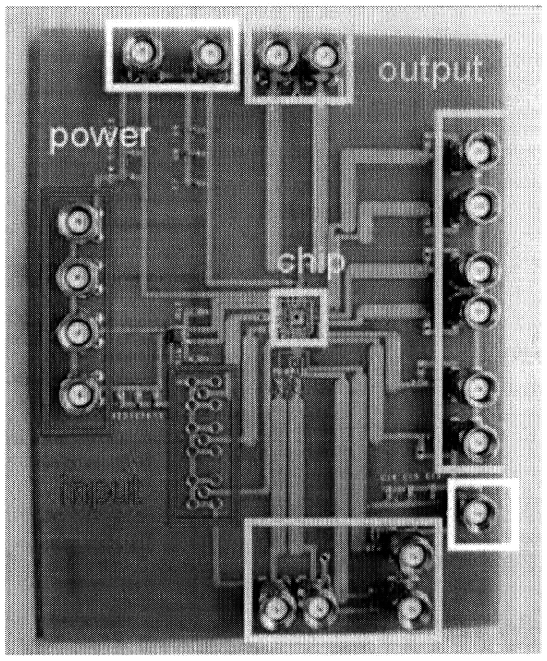


Figure 4-2: PCB used for testing

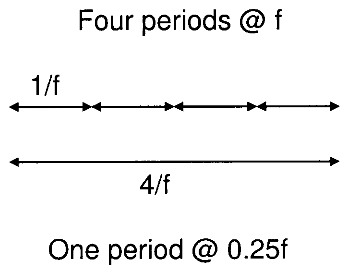


Figure 4-3: The counter can only represent delays up to a length of $1/f$; thus, an addtional state, the number of times that the counter has wrapped around, needs to be recorded to represent delays up to $4/f$.

4.1 Decreased Output Swing

As mentioned previously, SCL buffers, shown in Figure 4-4, are used to drive the outputs. The buffers are designed to drive a 50Ω load and all of the parasitic capacitances and inductances introduced by the bond wires, the packaging pads and the PCB traces. A separate voltage source is used so that the large current spikes from the buffers will not disturb the operations of the core circuitry. Figure 4-5 illustrates this.

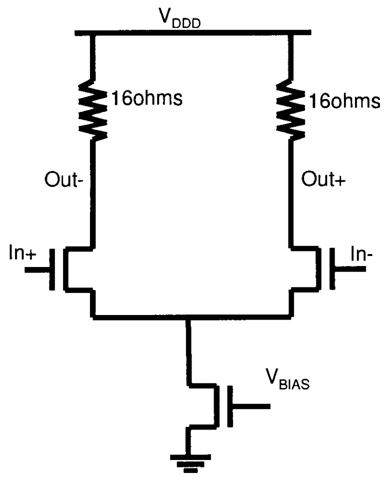


Figure 4-4: Circuit toplogy of SCL buffer

The measured swing on a single end of the SCL buffer is significantly less than

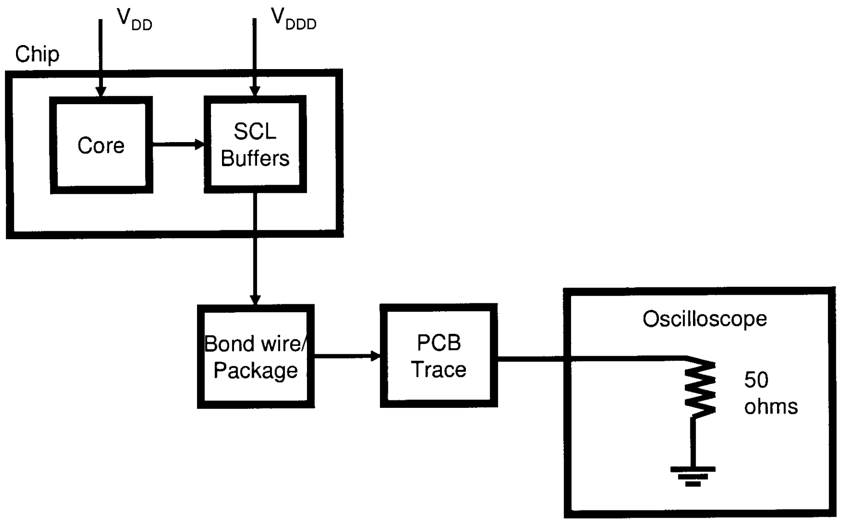


Figure 4-5: The core circuits of the chip and the buffers for driving the output signals off chip use separate supplies. The buffers drive 50Ω loads with parasitics from bond wires, the packaging and the PCB traces.

the simulated swing as shown in the plots of Figure 4-6. The swing remains low even when the frequency of the reference clock is reduced to 1kHz. This suggests that parasitic inductances and capacitances are not reducing the swing. The cause of this is speculated to be parasitic resistances due to poor layout. As show in Figure 4-7, the wire widths for the voltage supply and the outputs are 5μm and 2.4μm respectively. These wires are very thin and will produce extremely high parasitic resistances. The wire carrying the supply voltage V can be modeled as a resistor between the voltage source and all of the buffers as shown in Figure 4-8. This reduces the effective supply voltage V_{EFF} to $V_{DDD} - I \cdot R1$, where I is the measured current drawn from the supply source. As the buffers are designed to draw more than 100mA, even a small value for $R1$ can severely reduce V_{EFF}. When the buffer swings high, the measured voltage by a high impedance source at the output is V_{EFF} because the transistor is off and no current can flow through it. Thus, Equation 4.1 can be applied to calculate $R1$. Figure 4-9 shows the calculated values of $R1$ for different values of V_{DDD}. The average value of $R1$ is 3.2Ω.

$$R1 = \frac{V_{DDD} - V_{EFF}}{I} \tag{4.1}$$

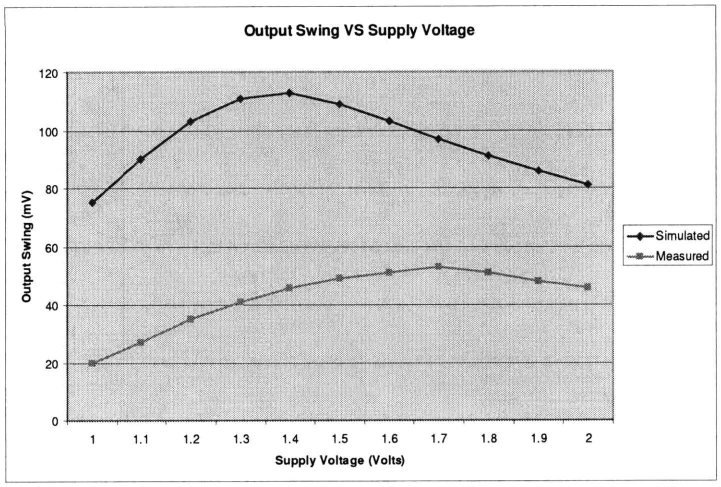


Figure 4-6: Simulated and measured voltage swings at the output

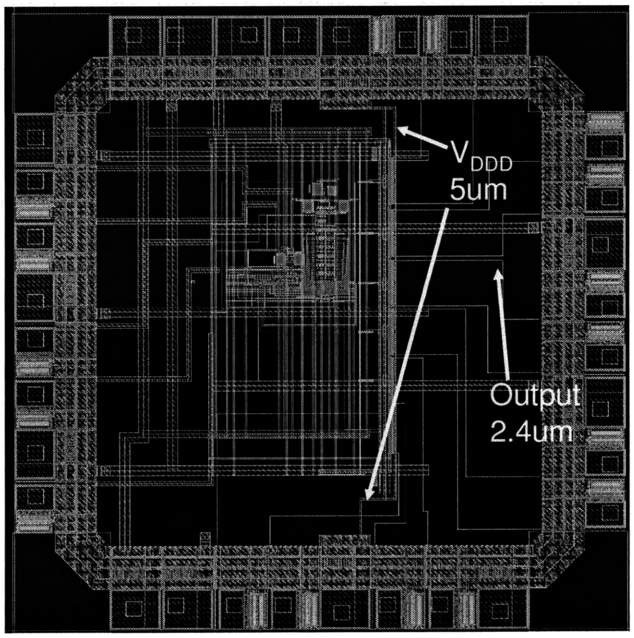


Figure 4-7: Layout of the chip showing the thin wires that contribute high parasitic resistances

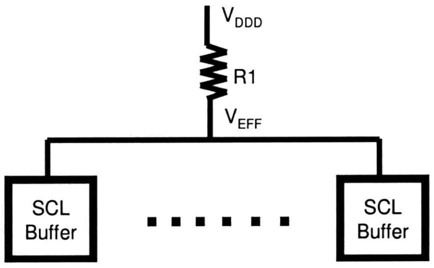


Figure 4-8: Parastic resistance $R1$ is placed between the supply voltage and the buffers

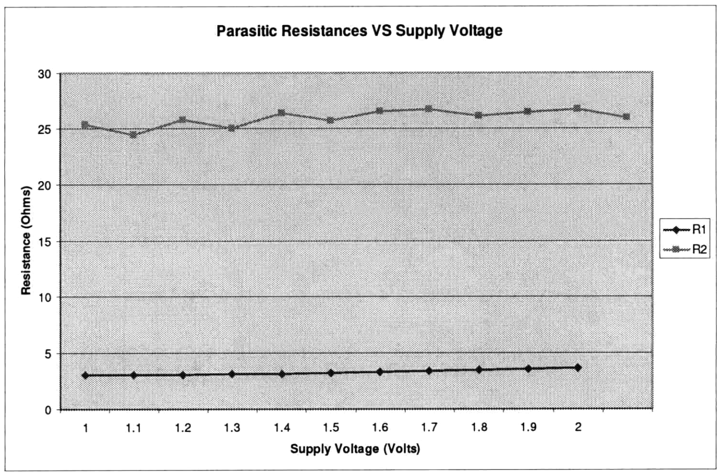


Figure 4-9: Calculated values for parasitic resistances $R1$ and $R2$

The parasitic resistance at the output can be modeled as a resistor $R2$ between OUT and V_X as show in Figure 4-10. When the output is high, attaching the oscilloscope forms a series of three resistors between V_{EFF} and ground. $R2$ is calculated using Equations 4.2 and the results are plotted in Figure 4-9. The number 66 is the sum of the resistances from the 50Ω load of the oscilloscope and the 16Ω pull-up resistor in the buffer. The average value for $R2$ is 25.9Ω.

$$R2 = \frac{V_{EFF}}{V_X} \cdot 50 - 66 \tag{4.2}$$

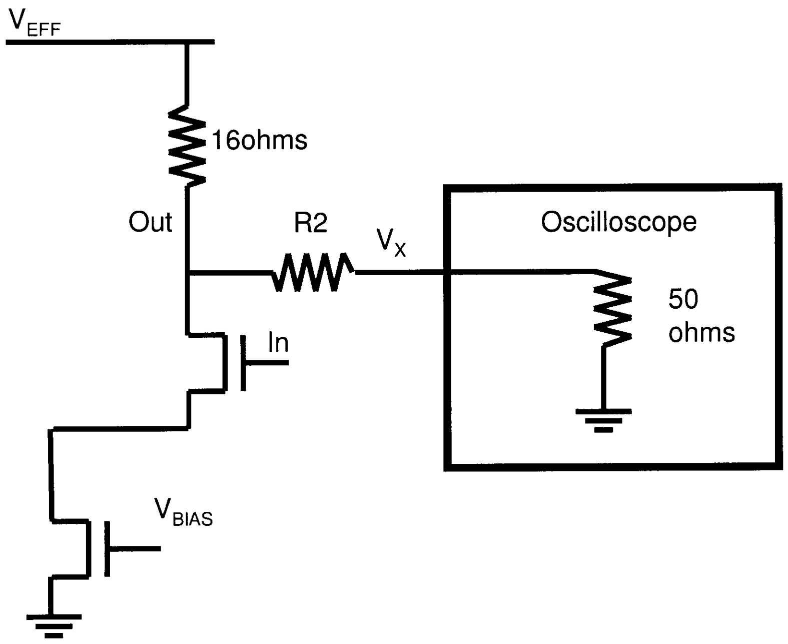
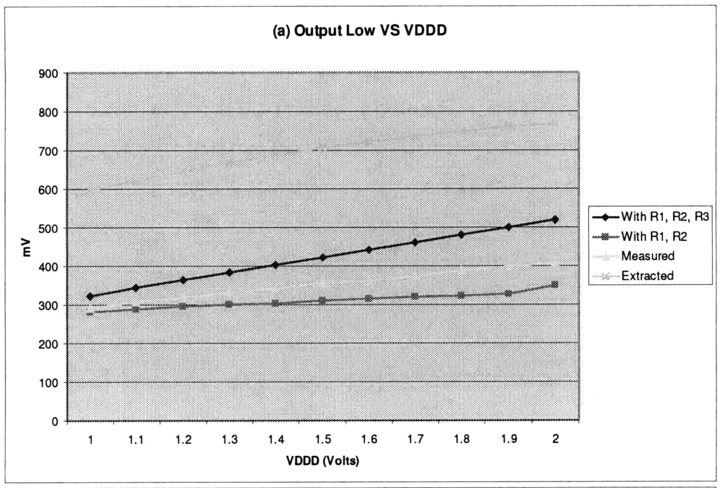
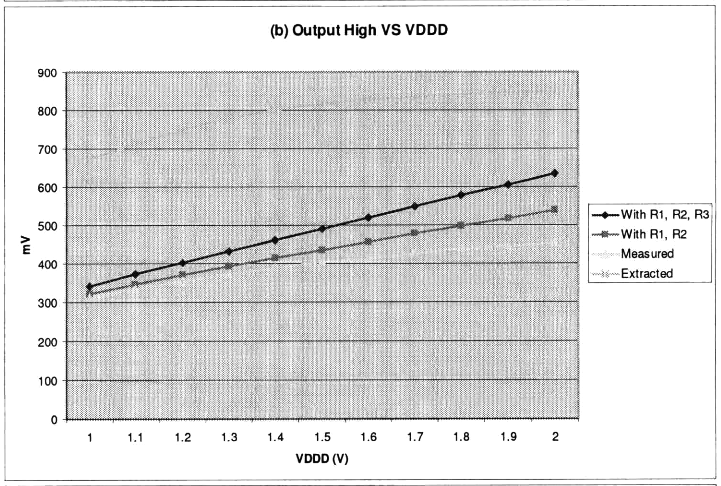


Figure 4-10: Parastic resistance $R2$ is placed between OUT and V_X

Figure 4-11 plots the buffers' output voltage low, output voltage high, and the difference of the two as the swing for three simulation setups and the measurement results for comparison. One simulation is generated using automated parasitic extraction of the layout in Cadence, while the other two use resistors that are manually inserted. One such simulation includes $R1$ and $R2$, the parasitic resistors discussed previously, and the other simulation includes an additional parasitic resistor $R3$. Analogous to $R1$, which models the resistances between the buffers and V_{DDD}, $R3$ models the resistances between the buffers and ground. As there is no accurate formula to determine $R3$, $R3$ is approximated as being equal to $R1$. This is a reasonable approximation as both resistors are connected to the voltage source. When all three parasitic resistors are added in, the simulations results are the closest to the measured values; they are especially close for low values of V_{DDD} between 1V and 1.3V. Even though the output swing is heavily degraded by parasitic resistances, the rest of the chip can still demonstrate functionality.





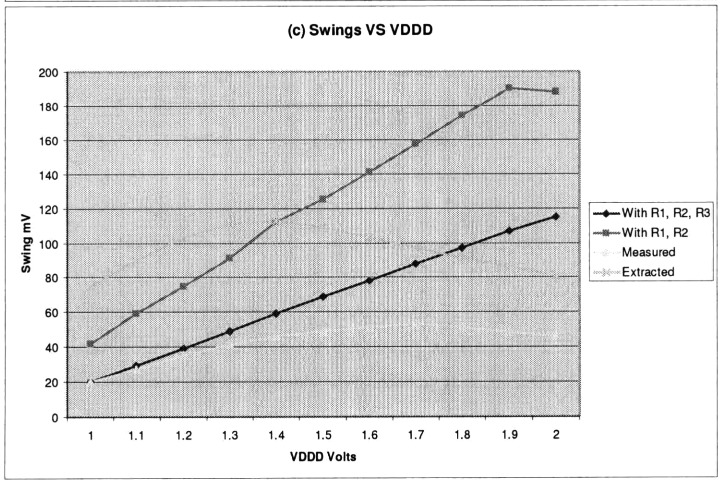


Figure 4-11: The plots show results for simulations using the automatic parasitic extraction in Cadence, using two manually added resistors, and using three manually added resistors. The plots are for (a) output voltage low (b) output voltage high (c) swing.

4.2 Test Results for Functionality

Figure 4-12 shows the typical outputs of the counter on the oscilloscope. Ideally, the counter outputs should alternate between two adjacent values when the chip has locked onto the input signal; however, the output in Figure 4-12 show that the output starts at 6, rises to 11, returns to a low value at 4, rises to 11 and falls back to 6. The drop to 4 instead of a 6 can be accounted by noise in the system that caused the comparator to make a wrong decision. Otherwise, the behavior of the output can be described as repeatedly incrementing to a high value and decrementing to a low value as opposed to repeatedly incrementing by one and then decrementing by one. This behavior is called over-counting and is due to the placement of two registers in the feedback path that controls the generated comparator clock as shown in Figure 4-13.

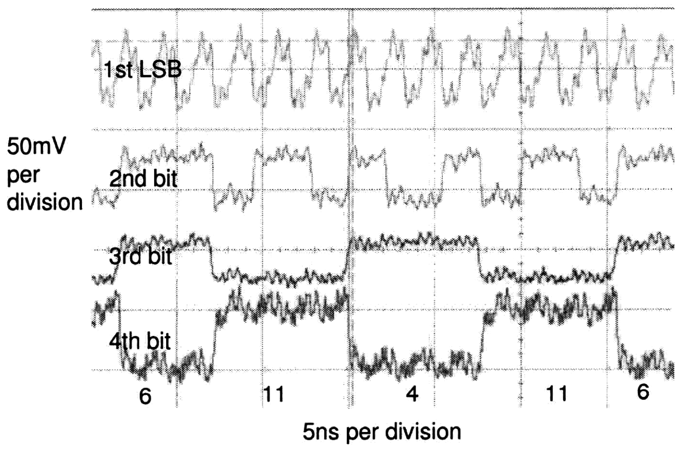


Figure 4-12: The digital output when the reference clock is at 1.8GHz and the input is delayed by 700ps. The output cycles between a low value and a high value.

Without the registers, the comparator clock would be adjusted based on the output from the previous cycle; however, due to the registers, the comparator clock is adjusted based on the output from three cycles ago. This behavior can be explained in the following example. The system advances its clock to lock onto the delay of the input when the phase of the comparator clock lags behind the phase of the input.

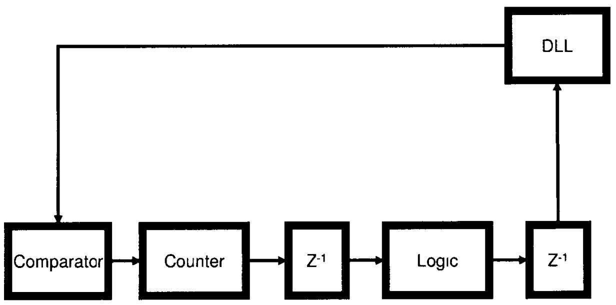


Figure 4-13: System block diagram with registers

When the comparator clock is advanced ahead of the input, the comparator will signal the system to delay the comparator clock; however, this signal does not affect the system until three cycles later. This results in the comparator clock being advanced by two extra steps, which increments the counter two extra times. Similarly, when the system is delaying the comparator clock to lock onto the delay of the input, it will decrement the counter two extra times.

Even though the registers cause over-counting, they are essential for the functioning of the system. First of all, the registers resolve timing issues for the digital counter and the logic circuits. Furthermore, the registers control the select signals to the multiplexers in the DLL. As mentioned in Chapter 3, the multiplexers determine the phase shift of the comparator clock. If the signals to the multiplexers did not change synchronously then the comparator clock can be shifted by the wrong value. Although two sets of registers were used, using only one set of registers can prevent the errors due to timing issues and enforce synchronization. In this case, the feedback signal is only delayed by two cycles instead of three. In the complete system described in Chapter 2, only every other cycle is used to measure the delay. The cycles in between are used to measure the magnitude and they do not affect the comparator clock. Therefore, in the complete system, having a delay of two cycles causes the comparator clock to shift during the next cycle used for locking onto the delay (which is two cycles later). This becomes identical to a system that uses every cycle to shifts the comparator clock with adjustments taking effect immediately in the following cycle; thus, over-counting is eliminated.

Figure 4-14 shows the digital outputs' measured high and low values due to over-counting and the average of the two values versus a variable delay when the reference clock is at 1GHz. The delay of the 0.25GHz input signal is incremented by steps of 100ps. Since 24 steps correspond to one period at 1GHz (1ns), 12 steps correspond to 500ps and each step corresponds to 41.7ps. Figure 4-15 plots the difference between the average digital outputs between delay values that are 500ps apart. Figures 4-14 and 4-15 demonstrate that the chip can lock onto a reference signal and determine the delay of the input signal. The plotted values are determined by observing the outputs for hundreds of cycles for each delay. Matlab simulations show that the outputs would have reached a greater range for each delay value if enough Gaussian noise to remove the quantization error had been present.

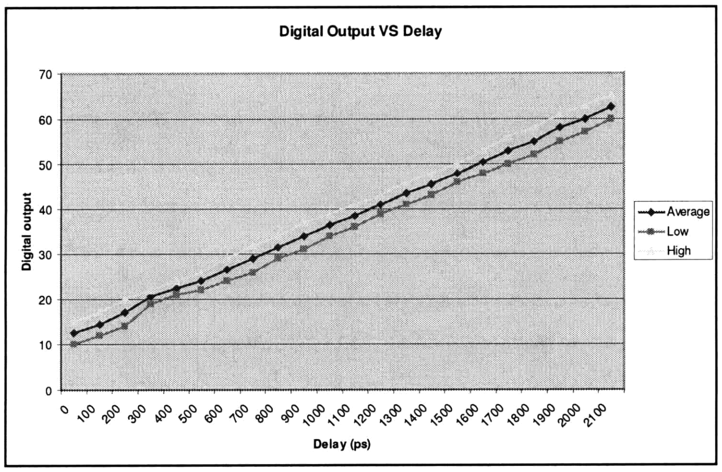


Figure 4-14: The 5bits of the digital output with varying delay of the input signal

The system is not able to function with a 2.4GHz reference clock; the cause is most likely parasitic resistances as described in the previous section.

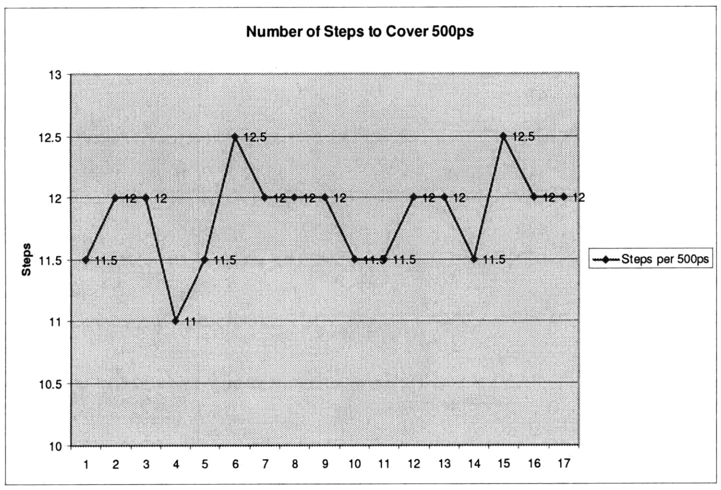


Figure 4-15: Change in the 5bits of the digital output when the delay is changed by 500ps, where the expected value is 12

Chapter 5

Conclusion

This chapter describes future work and summarizes this thesis.

5.1 Summary

As discussed in this thesis, digital phase tightening can be constructed from well-researched circuit blocks and greatly improves the phase and magnitude resolution of beamforming by oversampling. Furthermore, the digital phase tightening does not need to use high speed power hungry digital circuits.

The chip fabricated in 90nm CMOS implements a proof of concept system that can lock onto a 1GHz reference clock and can detect a phase delay with a resolution of 41.7ps. The performance of this chip was hindered by parasitic resistance due to poor layout. Using thicker wires will reduce the parasitic resistance; thus, increasing the output swing and allowing the circuit to run at a faster frequency. Additionally, another bottleneck is the speed of the delay elements which limits the resolution of the phase estimation. However, this is dependent on minimum features and will scale with technologies. Furthermore, as discussed in Chapter 2, noise in the system can be used to help improve the resolution of the phase estimation.

A full digital phase tightening system can be implemented by replacing the comparator with a high resolution ADC. The ADC can be utilized to measure the amplitude of the input signal in every other cycle. Inserting a clock cycle in between each cycle used for phase estimation also eliminates the over-counting as described in Chapter 4.

In conclusion, digital phase tightening, using high resolution, high speed ADCs and DLLs, is a novel system that can be applied to many applications such as MMW imaging.

5.2 Future Work

The end goal of the test setup is to demonstrate that the system can produce phase estimations that are accurate on the order of femto-seconds. As described in Chapter 2, the accuracy can be limited by both the quantization error and the noise in the system. Furthermore, when the noise is Gaussian and surpasses a threshold of RMS jitter greater than 1/2 of the quantization step size then the quantization error is removed. Once the quantization error is removed, digital phase tightening obtains high accuracy by reducing the noise through oversampling and averaging.

The quantization step size is equivalent to the resolution of the DLL, which is $1/(24 \cdot f)$, where f is the frequency of the reference clock. When the reference clock is slowed, the quantization step size increases and thus more noise is needed to eliminate the quantization error. For example, when f is 2.4GHz, the step size is 17.4ps and the required RMS jitter for the Gaussian noise is 8.7ps. And when the system is running at the reduced speed in the test setup, f is 1GHz, the step size is 41.7ps and the RMS jitter required for the Gaussian noise is 20.8ps. The noise source in the system does not have an RMS jitter greater than 20.8ps so quantization error is not eliminated. Therefore, the current chip could not produce phase estimations that are accurate on the order of femto-seconds.

However, if the system could have performed at a higher speed then the process for demonstrating its accuracy can be described as follows. The average of 2,000,000 digital outputs is taken as the phase estimation. This average scaled by the DLL resolution equals the measured delay in units of time. For example, if the resolution is 17.4ps and the averaged value is 3.5 then the measured delay is 60.8ps. Multiple phase estimations can be taken for a fixed delay at the input. Each phase estimation can be treated as one sample, and the standard deviation and the mean of the samples can be calculated. The standard deviation scaled by the DLL resolution is the RMS jitter of the reduced noise. This number, which should be on the order of femto-seconds, determines the accuracy of the system.

Bibliography

[1] A. Accardi, "Digital compensation in imaging systems," Massachuesetts Institute of Technology, 2007.

[2] T. Haynes, "A primer on digital beamforming," [Online], Spectrum Signal Processing, 1998, http://spectrumsignal.com/.

[3] J. B. Mead, "Digital beamforming receiver architectures for microwave remote-sensing applications," in *Geoscience and Remote Sensing Symposium Proceedings, 1998. '98. 1998 IEEE International*, 6-10 Jul. 1998, pp. 132–134.

[4] B. M. Helal, M. Z. Straayer, G.-Y. Wei, and M. H. Perrott, "A low jitter 1.6 GHz multiplying DLL utilizing a scrambling time-to-digital converter and digital correlation," *IEEE Symposium on VLSI Circuits*, pp. 166–167, June 2007.

[5] J.-E. Eklund and F. Gustafsson, "Digital offset compensation of time-interleaved ADC using randomchopper sampling," in *Circuits and Systems, 2000. Proceedings. ISCAS 2000 Geneva. The 2000 IEEE International Symposium on*, vol. 3, 28-31 May 2000, pp. 447–450.

[6] K. C. Pohlmann, *Principles of Digital Audio.* McGraw-Hill Professional, 2005.

[7] W. Kester, "ADC input noise: The good, the bad, and the ugly. is no noise good noise?" Analog Devices, 2006.

[8] S. Sidiropoulos and M. A. Horowitz, "A semidigital dual delay-locked loop," *Solid-State Circuits, IEEE Journal of*, vol. 32, pp. 1683–1692, 1997.

[9] Y.-J. Jeon, J.-H. Lee, H.-C. Lee, K.-W. Jin, K.-S. Min, J.-Y. Chung, and H.-J. Park, "A 66-333-Mhz 12-mW register-controlled DLL with a single delay line and adaptive-duty-cycle clock dividers for production DDR SDRAMs," *IEEE Journal of Solid-State Circuits*, vol. 30, pp. 2087–2092, Nov. 2004.

[10] R. T. Howe and C. G. Sodini, *Microelectronics: an integrated approach.* Prentice Hall, 1997.

[11] A. Rantala, D. Martins, and M. Aberg, "A DLL clock generator for a high speed A/D-converter with 1 ps jitter and skew calibrator with 1 ps precision in 0.35 um CMOS," *Analog Integrated Circuits and Signal Processing*, vol. 50, pp. 69–79, January 2007.

[12] T. C. Weigandt, B. Kim, and P. R. Gray, "Analysis of timing jitter in CMOS ring-oscillators," in *Proce. ISCAS*, vol. 4, 1994, pp. 27–30.

[13] M. E. Waltari and K. A. I. Halonen, *Circuit Techniques for Low-Voltage and High-Speed A/D Converters.* Springer, 2002.

[14] M. Combes, K. Dioury, and A. Greiner, "A portable clock multiplier generator using digital CMOS standard cells," *Solid-State Circuits, IEEE Journal of*, vol. 31, pp. 958–965, 1996.

[15] R. Farjad-Rad, W. Dally, H.-T. Ng, R. Senthinathan, M.-J. E. Lee, R. Rathi, and J. Poulton, "A low-power multiplying DLL for low-jitter multigigahertz clock generation in highly," *Solid-State Circuits, IEEE Journal of*, vol. 37, pp. 1804–1812, 2002.

[16] A. Tajalli and M. Atarodi, "Linear phase detection using two-phase latch," *Electronics Letters*, vol. 39, pp. 1695–1696, 2003.

[17] W. Rhee, "Design of high-performance CMOS charge pumps in phase-locked loops," in *Circuits and Systems, 1999. ISCAS '99. Proceedings of the 1999 IEEE International Symposium on*, vol. 2, June 1999, pp. 545–548.

[18] B. A. Floyd, "A CMOS wireless interconnect system for multigigahertz clock distribution," Ph.D. dissertation, University of Florida, 2001.

[19] D. A. Johns and K. Martins, *Analog Integrated Circuit Design.* John Wiley & Sons, Inc., 1997.

[20] J. M. Rabaey, A. Chandrakasan, and B. Nikolic, *Digital Integrated Circuits: A Design Perspective.* Prentice Hall, 2003.