CMOS Exactly Solvable Chaotic Oscillator

by

Casey Fendley

A thesis submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Auburn, Alabama
December 14, 2019

Keywords: CMOS, Chaotic Oscillator

Approved by

Robert N. Dean, Chair, McWane Endowed Professor of Electrical and Computer Engineering
Thaddeus Roppel, Associate Professor of Electrical and Computer Engineering
Edmon Perkins, Assistant Professor of Mechanical Engineering
Aubrey Beal, Assistant Professor of Electrical Engineering, University of Alabama-Huntsville

Abstract

The design and simulation of a complementary metal-oxide-semiconductor (CMOS) exactly solvable chaotic oscillator is presented in this work. The goal of this work is to design and simulate a high-frequency implementation of a piece-wise linear exactly solvable chaotic oscillator. The oscillator is designed using 45nm CMOS technology, and models the system accurately to around 50MHz. The CMOS model's performance is also evaluated at higher frequencies. SPICE simulation is used for circuit verification and the results are compared with numerical solutions of the model.

In this work, an approach for design and fabrication of an integrated circuit is also discussed to lay the groundwork for future integrated circuit development of exactly solvable systems. This integrated circuit was designed and simulated within the Cadence Virtuoso Analog Design Environment. Fabrication was done through MOSIS.

Acknowledgments

I would like to thank my friends and family for their unwavering support and encouragement for furthering my education. Additionally, I would like to thank my advisor, Dr. Robert Dean, and my committee members Dr. Thaddeus Roppel, Dr. Edmon Perkins, and Dr. Aubrey Beal.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

High frequency chaotic waveforms have applications in many different areas in electrical engineering. Two popular applications for this type of nonlinear waveform are radar [5] and communications [27]. Chaotic signals have a wide bandwidth and in the long-term they are non repeating waveforms. However, conventionally in the presence of noise, these signals require high-quality synchronization and complicated nonlinear filter techniques to be useful [24]. One way to overcome these limitations is to use exactly solvable chaotic waveform generators. Exactly solvable chaos is different in the sense that an exact analytical solution exists for these systems. From the solution, a matched filter can be constructed for the system in order to maximize the signal-to-noise ratio in the presence of noise [2].

The exact solvable model of interest in this work has been fully described in [1], and a low frequency circuit implementation has been realized in [24]. Additionally, a circuit implementation up into the MHz frequency range has also been described in [35]. Both of these circuit designs mentioned previously are implemented using printed circuit board designs accompanied with commercial off the shelf parts. The main goal of this work is to present a higher frequency circuit model of the exactly solvable system previously mentioned using integrated circuit process technology. The circuit designs presented in this work are simulated using a BSIM4 SPICE model based on 45nm complimentary metal-oxide-semiconductor (CMOS) technology.

A description for general chaos can be found in Chapter 2. This chapter also contains the description of the Lorenz system to illustrate some of the characteristics of chaos. A circuit model for the Lorenz system is presented in Chapter 2 to serve as an example of how nonlinear

differential equations can be solved using analog computation. Additionally, a mathematical description for the exactly solvable system of interest in this work is presented as well.

Simulation of all the circuit designs presented in chapter 2, 3, and 4 are carried out in SPICE. The SPICE simulator used for design verification is LTspiceXVII which is maintained by Analog Devices. This simulator was chosen due to its availability and speed. SPICE models for all the circuits presented can be found throughout this work and also in the Appendix. Chapter 3 contains a block level design description of each piece of the CMOS exactly solvable oscillator along with some simulations describing the operation of individual circuits. Chapter 4 contains the simulation results of the system model as a whole. Chapter 5 provides an overview of the design process that was used to fabricate a custom integrated circuit using the Cadence Analog Design Environment.

Chapter 2

Background

2.1 Chaos

Chaos in the normal context of the word implies ”randomness” and ”disorder”, however in the mathematical sense it is quite the opposite. Chaos Theory is a branch of mathematics that studies nonlinear dynamical systems that are sensitive to initial conditions. Chaotic systems may look like they are random, but actually they a deterministic. In other words, for a given input into a chaotic system, the output will be the same every time. Another characteristic of these nonlinear dynamical systems are their long term aperiodicity. Some of the popular examples of chaos include systems like the Lorenz system[3] or the Rössler system[4]. Chaos has been found to exist in many different disciplines of science, which include Economics[17], Electronics[6, 7, 8], Chemistry[16], and Meteorology[3]. In this chapter, an example chaotic system will be examined, as well as the main focus of this work which is exactly solvable chaos.

2.1.1 The Lorenz System

Perhaps the most popular example of chaos is demonstrated with the Lorenz System. This system was discovered by Edward Lorenz in 1963 when he made a simplification to the Navier-Stokes equations for atmospheric modeling. This simplification became known as the Lorenz system, and can be described by the following three equations:

$$\dot{x} = \sigma(y - x) \tag{2.1}$$

$$\dot{y} = -x(\rho - z) - y \tag{2.2}$$

$$\dot{z} = xy - \beta z \tag{2.3}$$

where x, y, and z are the state variables and σ, ρ, and β are control parameters [3]. The system has two quadratic nonlinearities (xy and xz). Although this system looks relatively simple on the surface, for the control parameters σ=10, ρ=28, and β=8/3, it is capable of producing complex dynamics.

In order to show the complex dynamical behavior of the system, an analog circuit was built and simulated in LTspice. The circuit schematic is shown in Fig. 2.1 and is based on the design of Paul Horowitz [12]. Inverting operational amplifiers are used with a capacitor in feedback to perform integration on the state variables. Integrated analog multipliers are used to create the quadratic nonlinearities. All of the operational amplifiers and analog multipliers are powered with a differential supply voltage of $\pm$15V.

The operational amplifiers used in the simulation are LT1057s which are high speed precision JFET amplifiers made by Analog Devices [10]. The AD633 analog multipliers used in the circuit are also manufactured by Analog Devices [11]. The AD633 is capable of performing multiplication of two differential signals, which works well for performing the nonlinearity required for the circuit. The resistors in the circuit set the values of the control parameters σ, ρ, β. Resistance values are normalized with respect to a nominal resistance value ($R_{nom} = 1M\Omega$). For this simulation, the control parameter $\sigma = R_{nom}/R_1 = R_{nom}/R_2 = 10$. Similarly, $\rho = R_{nom}/R_3 = 28$ and $\beta = R_{nom}/R7 = 8/3$. The capacitor values set the bandwidth of the system. Also, the resistor ratios have been adjusted to compensate for the output attenuation of the analog multiplier [13].

The circuit was simulated in LTspiceXVII and the trajectories were plotted in MATLAB in order to better understand the system. Figure 2.2 shows two separate simulations of the state variable x. The initial conditions for y and z were kept constant, but the initial condition for x was set to $0.5mV$ and $0.501mV$. Notice how both trajectories are nearly identical for the first second of the simulation and how they diverge into different trajectories while the only difference is the slight variation of 1mV in the initial condition of x. This demonstrates one of

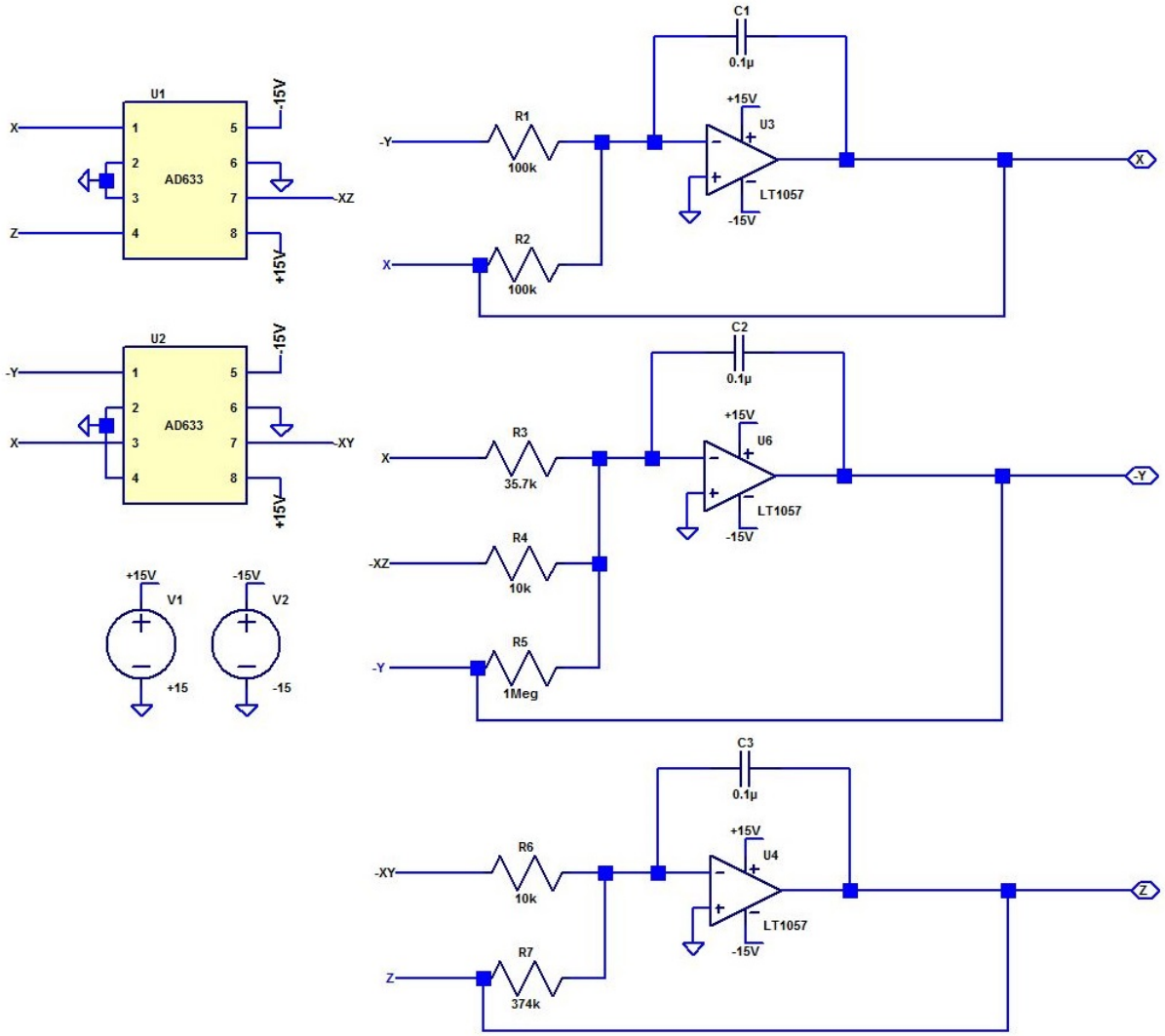

Figure 2.1: Schematic for Lorenz Circuit

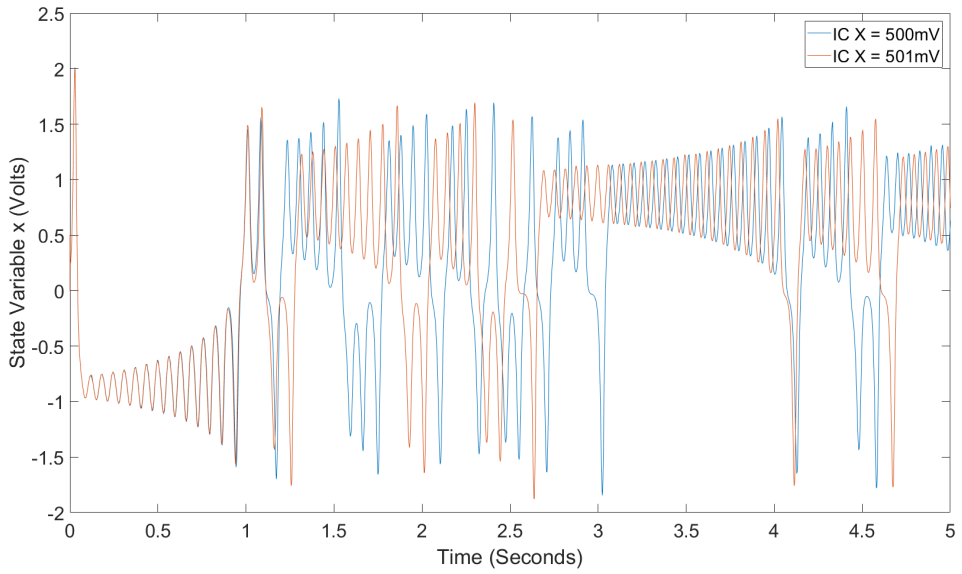

Figure 2.2: Time Series Data of the State Variable x from the Lorenz Circuit Simulation

the fundamental characteristics of chaos, which is a system's sensitivity to initial conditions. The aperiodic nature of the oscillations can also be observed from the time series data. This long term, irregular oscillation is also a characteristic for chaos[9].

The plot shown in Figure 2.3 is the phase space plot of the system. This is perhaps the most famous plot of the system, known as the "Lorenz Butterfly". The phase space plot allows a good view of the strange attractors. The dense bands around the attractor are also another common characteristic of chaos.

2.2 Exactly Solvable Chaos

Though some of the nonlinear systems mentioned in the previous section are capable of being solved with numerical integration or analog computation, none of them have an exact analytical solution. Numerical solutions can certainly provide some insight into the behavior of these complex dynamical systems, but a closed form solution provides for a different type of analysis. Exactly solvable systems have been discovered that exhibit complex chaotic behavior while also having an analytical solution [18, 22, 1, 23, 24].

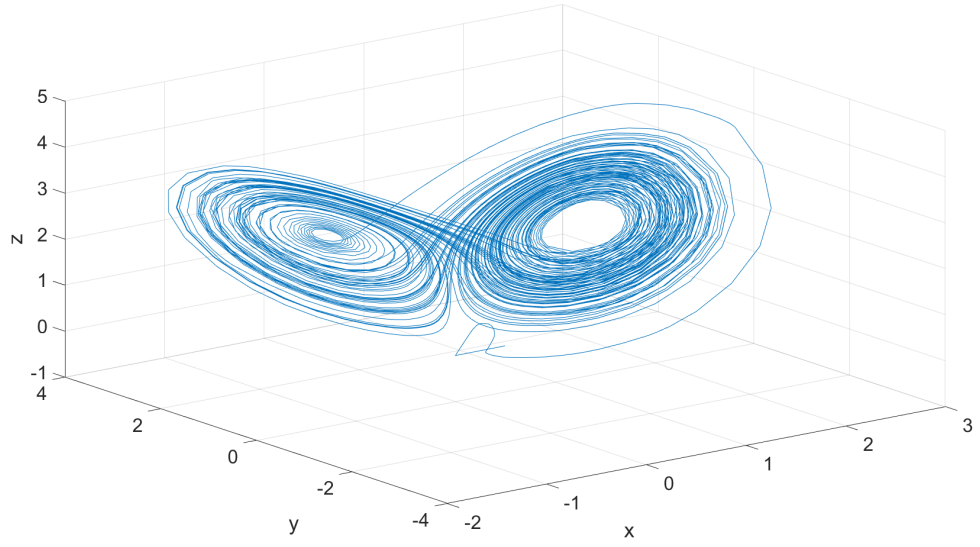

Figure 2.3: Phase Space Plot of State Variables x, y, z from Lorenz Circuit Simulation

2.2.1 Exact Shift-Band Oscillator

This work mainly focuses on a manifold piecewise-linear system described by Saito and Fujita [20]. The hybrid dynamical system of interest consists of a continuous state $u(t)$ and a discrete state $s(t)$. The continuous-time dynamics is described by the differential equation

$$\ddot{u} - 2\beta\dot{u} + (\omega^2 + \beta^2)(u - s) = 0 \tag{2.4}$$

where ω and β are fixed parameters satisfying $\omega = 2\pi$ and $0 < \beta \leq ln2$. Transistions in the discrete dynamics are defined by the guard condition

$$\dot{u}(t) = 0 \Rightarrow s(t) = sgn(u(t)), \tag{2.5}$$

meaning the discrete state $s(t)$ is set to the sign of $u(t)$ whenever the derivative of the continuous state diminishes and $s(t)$ maintains this value until the guard condition is next met. In this system, the signum function is defined as

$$sgn(u) = \begin{cases} +1, & u \geq 0 \\ -1, & u < 0, \end{cases} \tag{2.6}$$

where the choice that $sgn(0) = +1$ is arbitrary and chosen for completeness [24].

A MATLAB simulink model was constructed to demonstrate the dynamical behavior of the system. Fixed step ode4 (Runge-Kutta) numerical integration was used to solve the system. Figure 2.4 shows a typical time series simulation where $\omega = 2\pi$ and $\beta = ln2$. The growing oscillation can be seen around the fixed point in the continuous state until the guard condition in (2.5) is met and then the growing oscillation is flipped to the new fixed point. The phase space plot of the of the system is shown in Figure 2.5. The phase space for the system has a Lorenz like structure with the two strange attractors. The dense bands around the attractors show the complex dynamical behavior of the system. By Sampling the continuous state $u(t)$ at integer times and plotting the future values versus the previous values (u_{n+1} vs u_n), the map in Figure 2.6 is generated. This map is topologically conjugate to the known chaotic one dimensional map known as the dyadic transformation, Bernoulli map, or bit shift map. The slope of the line is determined by e^{β}, where β controls the negative damping and is also the lyapunov exponent for the system.

The analytical solution for this system can be found for initial conditions at time $t = t_n$ such that $u(t_n) = u_n$, $\frac{du}{dt}(t_n) = 0$, and $s(t_n = s_n$, where $s_n = sgn(u_n)$. The solution can be given as the linear convolution

$$\begin{aligned} u(t) &= \textstyle\sum_{m=0}^{\infty} s_m \cdot P(t - t_n - m) \\ s(t) &= \textstyle\sum_{m=0}^{\infty} s_m \cdot \phi(t - t_n - m) \end{aligned} \tag{2.7}$$

where $t \geq 0$ and each symbol $s_m = \pm 1$. The fixed basis functions are

$$P(t) = \begin{cases} (1 - e^{-\beta})e^{\beta t}(cos\ \omega t - \frac{\beta}{\omega} sin\ \omega t), & t < 0, \\ 1 - e^{\beta(t-1)}(cos\ \omega t - \frac{\beta}{\omega} sin\ \omega t), & 0 \leq t < 1, \\ 0, & 1 \leq t, \end{cases} \tag{2.8}$$

and

$$\phi(t) = \begin{cases} 0, & t < 0, \\ 1, & 0 \leq t < 1, \\ 0, & 1 \leq t. \end{cases} \tag{2.9}$$

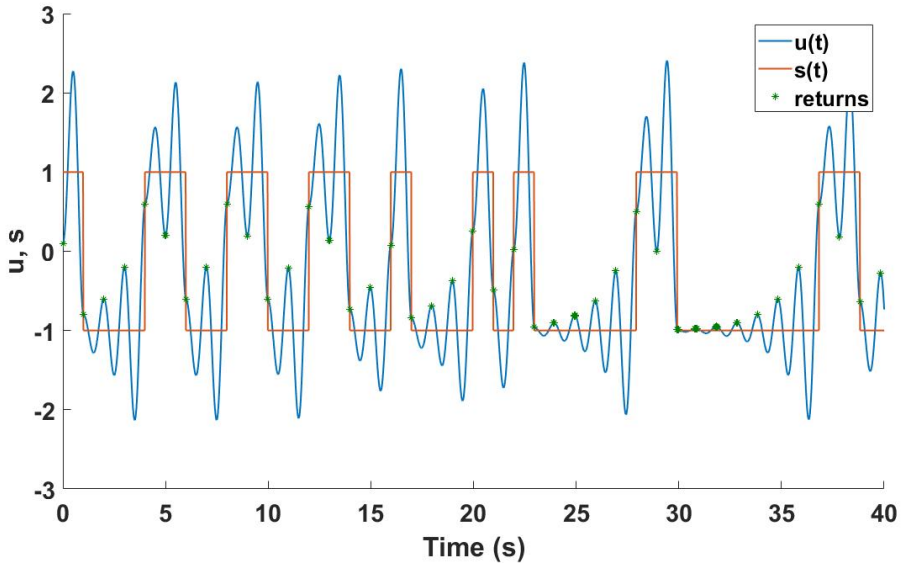

Figure 2.4: Time Series Simulation of MATLAB Simulink Model

The solution shows that the oscillator's state only depends on the current and future symbols output from the oscillator [29]

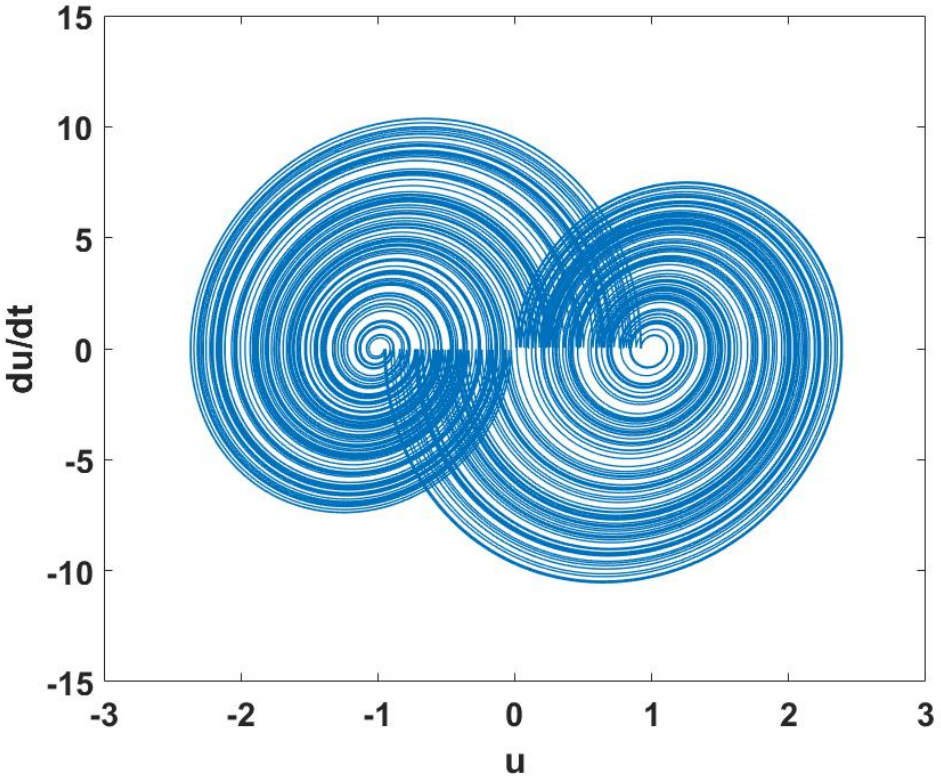

Figure 2.5: Phase Space Projection from Numerical Simulation

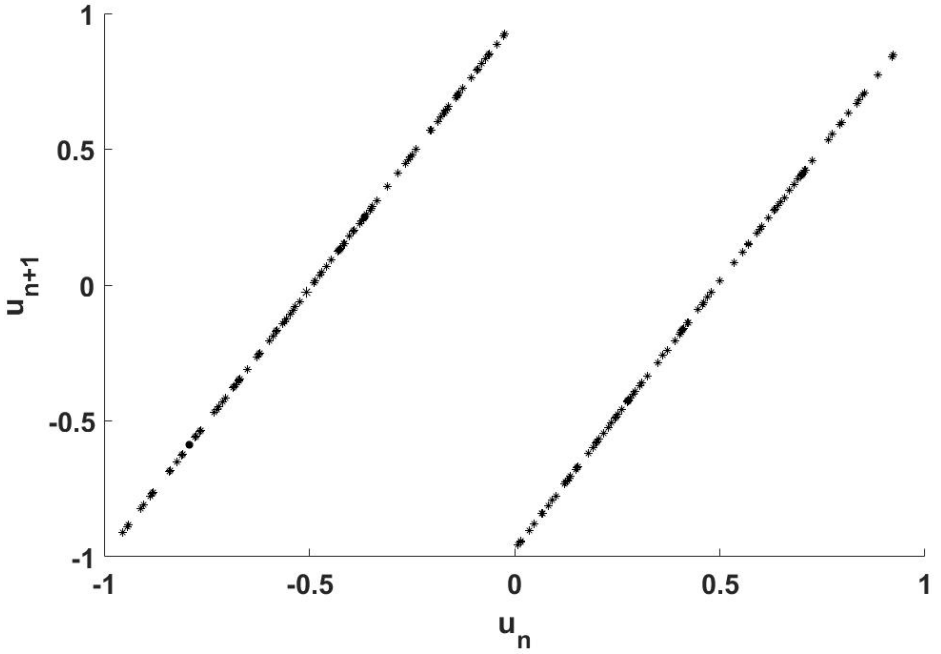

Figure 2.6: Chaotic Shift Map for $\beta = ln2$

Chapter 3

CMOS Shift-Band Oscillator

The CMOS implementation of the hybrid dynamical system consists of a continuous time section and a discrete time section. The design of each section is explained in detail within this chapter. All circuit designs were done using a BSIM4 SPICE Model, which is listed in Appendix A. The BSIM4 model used in this work models a 45nm CMOS Technology [21]. A full schematic for the design can also be found in the Appendix.

3.1 Continuous-time Section

To implement the continuous dynamics of the exact solvable shift band oscillator, a -RLC circuit is used. In particular, a series -RLC circuit is used to model the second order continuous dynamics. The inductor value required for the circuit can be quite large, so an active implementation is used. Additionally, in order to realize a negative resistance, active components have to be used. Active filter design techniques are applied to implement the series -RLC network. Active filters utilize integration in order to realize the differential equations they represent. This work uses a G_m-C integrator, which consists of a transconductor and a capacitor [30]. These integrators are commonly used in active filter design for integrated circuits. The transconductor used in this case is an operational transconductance amplifier. Figure 3.1 shows the G_m-c implementation of the continuous-time section of the oscillator. The following section describes the operation and design of the operational transconductance amplifier used in the synthesis of the unstable ladder network.

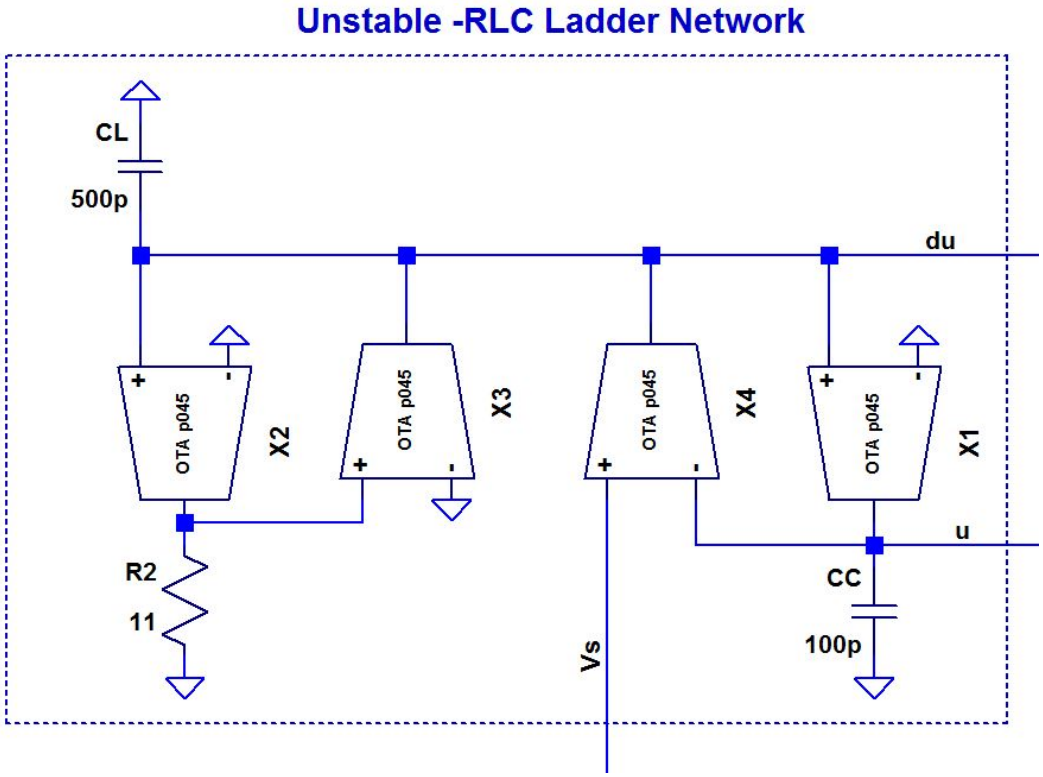

Figure 3.1: Operational Transconductance Amplifier Implementation of Unstable -RLC Ladder Network

3.1.1 Operational Transconductance Amplifier

The operational transconductance amplifier (OTA) can be defined as an amplifier where all nodes are low impedance except the input and output nodes. The OTA is typically modeled as a voltage controlled current source since its main function is to take a differential voltage input and output a single or differential current. A symmetrical single stage OTA topology can be seen in Figure 3.1. This OTA topology consists of a differential amplifier with a current source load. The voltage gain is given by

$$A_v = \frac{V_{out}}{V_{inp} - V_{inm}} = g_{m1,2} \cdot (r_{o6} \parallel r_{o8}) \tag{3.1}$$

where $g_{m1,2}$ is the transconductance of the input transistors that make the differential pair. r_{o6} is the output resistance of the pmos current mirror and r_{o8} is the output resistance of the nmos current mirror. The transconductance and output resistance of a MOSFET transistor in

saturation is given by

$$g_m = \sqrt{2\mu_n c_{ox} \frac{W}{L} I_D} \tag{3.2}$$

and

$$r_o \approx \frac{1}{\lambda I_D} \tag{3.3}$$

where W and L describe the physical geometry of the device and I_D is the drain current. The transconductance parameter, $\mu_n c_{ox}$, and the channel-length modulation, λ, are determined by the process technology used and are not able to be adjusted by the designer [19]. While the voltage gain of an OTA can be useful to know, a more useful figure of merit is the total transconductance of the circuit. The total transconductance of the circuit is given by the following relationship

$$G_m = \frac{I_{out}}{V_{inp} - V_{inm}} \tag{3.4}$$

where $I_{out} = I_{D6} - I_{D8}$. For the case where the nmos and pmos current mirrors are sized 1:1, the total transconductance of the OTA is equal to the transconductance of the input transistors of the differential pair $g_{m1,2}$.

It is also useful to look at the frequency characteristics of the amplifier. For higher frequencies, the output voltage can be written as

$$V_{out} = I_{out} \cdot \frac{1}{j\omega C_L} = g_{m1,2} V_{in} \cdot \frac{1}{j\omega C_L}. \tag{3.5}$$

In this case C_L is the load capacitance and $\omega = 2\pi f$. Dividing both sides of equation 3.5 by the input voltage, V_{in}, and computing the absolute value of both sides to get rid of the imaginary component gives

$$\left| \frac{V_{out}}{V_{in}} \right| = \frac{g_{m1,2}}{2\pi f C_L}. \tag{3.6}$$

The unity-gain frequency is defined as the point where the open-loop gain is unity [25]. By setting the voltage gain equal to 1 in equation 3.6 and solving it for f gives equation 3.7. This equation is useful for estimating the unity gain frequency of the OTA topology shown in Figure

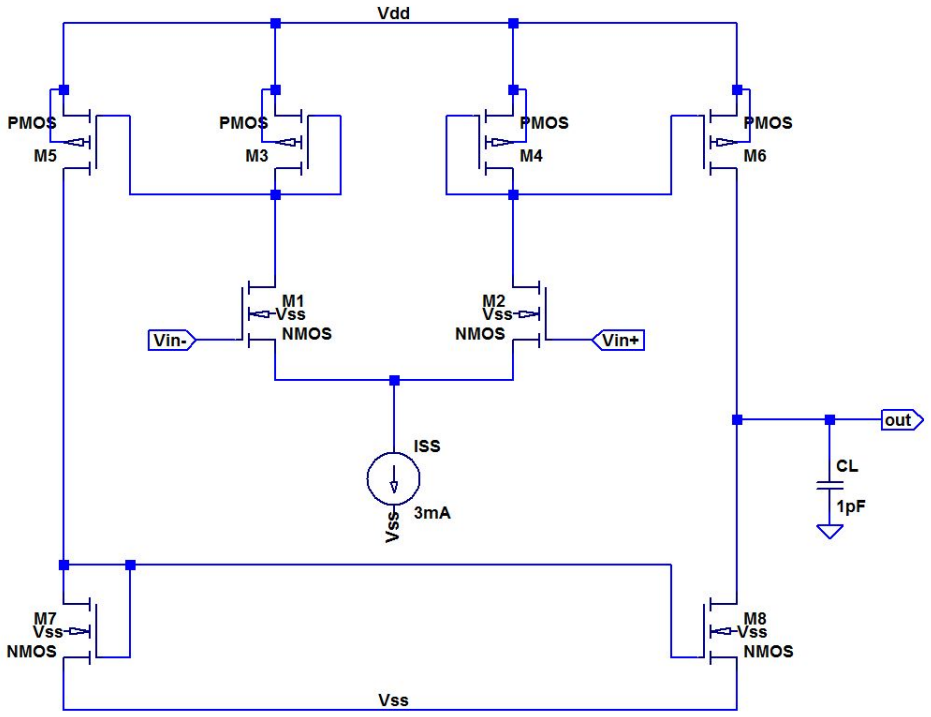

Figure 3.2: Single Stage Symmetrical OTA Topology

3.1.

$$f_{un} = \frac{g_{m1,2}}{2\pi C_L} \tag{3.7}$$

3.1.2 OTA Design

The design process for the OTA in Figure 3.1 uses what are considered short-channel devices. Devices with channel lengths well below 1 μm are considered short-channel devices. The reason for this distinction between long and short channel devices is that the square-law current-voltage characteristics of the MOSFET starts to become a less accurate model as channel lengths get smaller [25]. While there are models that can be used to calculate bias points for short-channel effects, a simpler design approach is used in this work. The design process used in this work is based on simulations of the SPICE model to determine the bias points. This graphical approach is used by Razavi[26]. The bias points can easily be determined for

Parameter	Value
Vdd	1V
Vss	-1V
V_{thn}	0.45V
V_{thp}	-0.45V
Length	0.18μm
C_L	1pF
f_{un}	1.5 GHz

Table 3.1: Symmetrical OTA Design Parameters

the short-channel devices by looking at simulations of the current-voltage characteristics for a single transistor.

Before beginning the design process, it is important to set some design parameters. The design parameters are shown in table 3.1. The first step in the design of the symmetrical OTA shown in Figure 3.1 is to determine the transconductance of the input differential pair. To find this value, equation 3.7 can be rearranged to solve for the transconductance. For the given values in the table 3.1, calculating the transconductance from equation 3.7 yields a value of $9.4mA/V$ or approximately $10mA/V$. Once the value of g_m required for the input n-type transistors is known, the other bias points can be determined from simulation.

The next step is to plot the current-voltage characteristic in SPICE for a single n-type transistor and obtain the bias points. Figure 3.2 shows the current-voltage characteristic (left) for the n-type transistor and the transconductance (right). These plots are generated by a DC sweep of the Gate to Source voltage, V_{GS}, while keeping the Drain to Source voltage, V_{DS}, constant. V_{DS} is chosen such that the transistor operates in saturation. The left plot in Figure 3.2 shows the drain current. The plot on the right in Figure 3.2 is obtained by taking the derivative of the drain current with respect to V_{GS}. This simulation was performed at various transistor widths to demonstrate the effects of transistor sizing on the operating points. By adjusting the size of the transistor, it is possible to obtain specific values of g_m and I_D at a given V_{GS}. For the design in this work, V_{GS} is set to approximately $700mV$ which is denoted by the dashed black line in the plots of Figure 3.2. It is important to select a value for V_{GS} that still provides an adequate overdrive voltage, $V_{ov} = V_{GS} - V_{thn}$. The speed of the amplifier is directly related to the overdrive voltage [25]. Once an approximate V_{GS} is selected, the idea

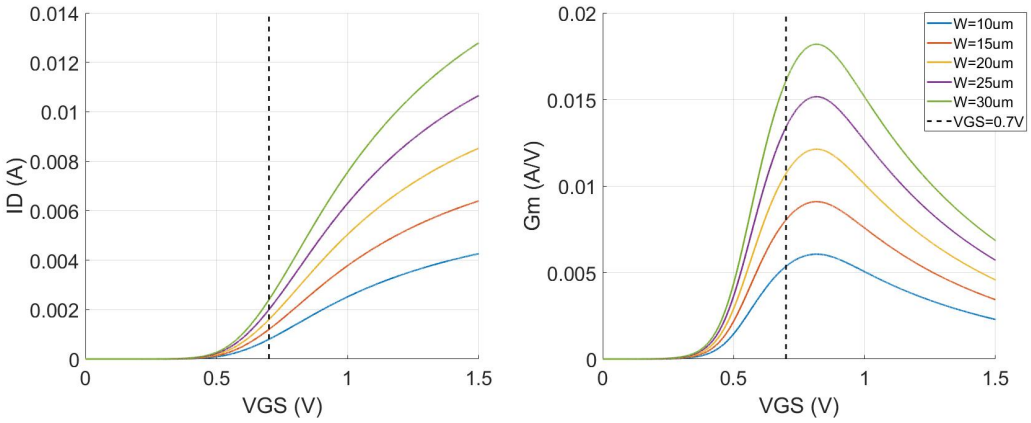

Figure 3.3: (Left) IV Characteristic for n-type MOSFET at various widths. (Right) Transconductance Characteristic for n-type MOSFET at various widths.

is to adjust the width of the transistor until the desired g_m is achieved at the selected gate to source voltage. With the geometry of the transistor known, the drain current can be found using the same method on the IV characteristic. Once the drain current is known, the source current, or the current through ISS shown in Figure 3.1 is found by doubling the drain current. This same process can be repeated to find the bias points for the p-type transistors located within the circuit.

Using this method, the width for the NMOS transistor was found to be $20\mu m$. While the width for the PMOS transistor is $155\mu m$. The drain current at the given geometry is around $1.5mA$. Doubling the drain current gives a source current, I_{SS}, of $3mA$. Now that the geometry for the n-type and p-type transistors and the source current are known, the design for the OTA can be simulated to test the performance.

3.1.3 OTA Simulation

Voltage Transfer Characteristic Simulation

The voltage transfer characteristic shows the total output voltage versus the total input differential voltage. The transfer characteristic for the amplifier designed in the previous section can be seen in Figure 3.3. This plot is created by grounding the inverting terminal of the amplifier and performing a DC sweep at the non-inverting terminal. The output voltage is then plotted versus the input voltage. The straight line in the center of the plot from $V_{in} \approx \pm 350mV$

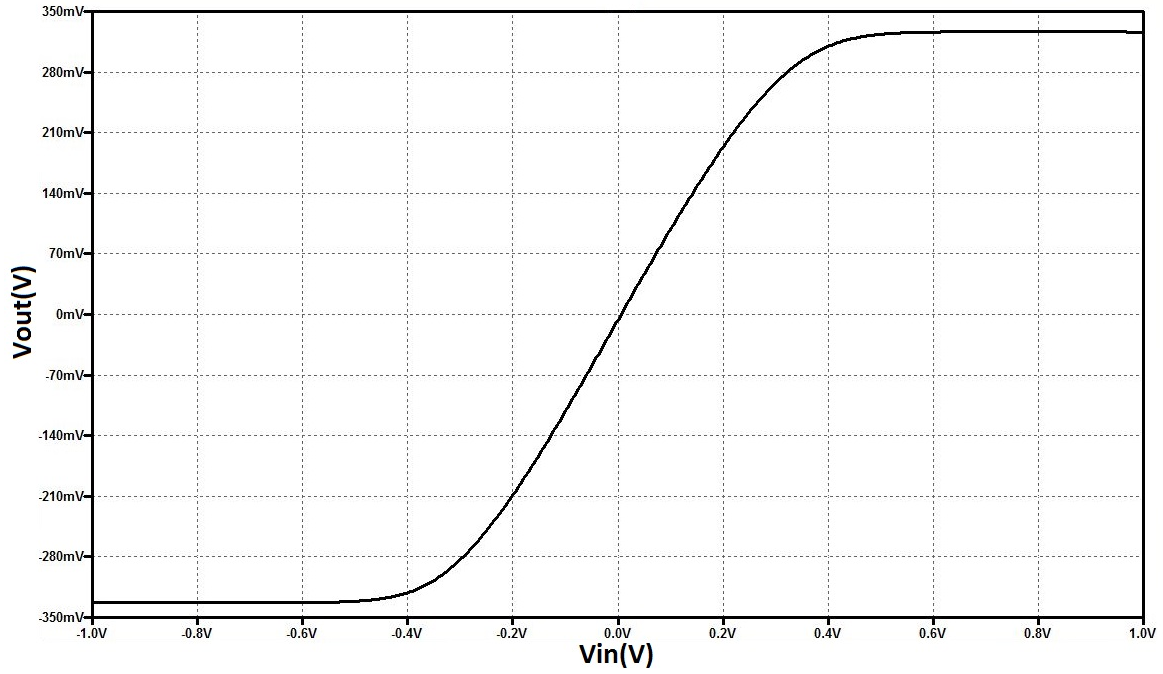

Figure 3.4: Simulated Voltage Transfer Characteristic of the Operational Transconductance Amplifier

is known as the linear region, where the slope is approximately equal to the open loop gain [19]. Input differential voltages greater than this will cause the amplifier to enter its nonlinear region of operation, also known as the saturation region.

AC Sweep Simulation

The frequency response of an amplifier can be simulated by performing an AC sweep simulation. To do this, an AC signal must be applied to the non-inverting input of the amplifier while the inverting input is grounded. The AC signal's frequency is then swept across a specified range. The open loop gain of the amplifier may be looked at by plotting $\left|\frac{V_{out}}{V_{in}}\right|$ versus frequency. The open loop gain for the transconductance amplifier designed in the previous section is shown in Figure 3.4. From Figure 3.4, the unity gain bandwidth of the amplifier can be seen at the point where the curve intersects 0dB. The unity gain bandwidth for the amplifier designed in the previous section is around 1.56GHz, which is very close to the frequency that it was designed for.

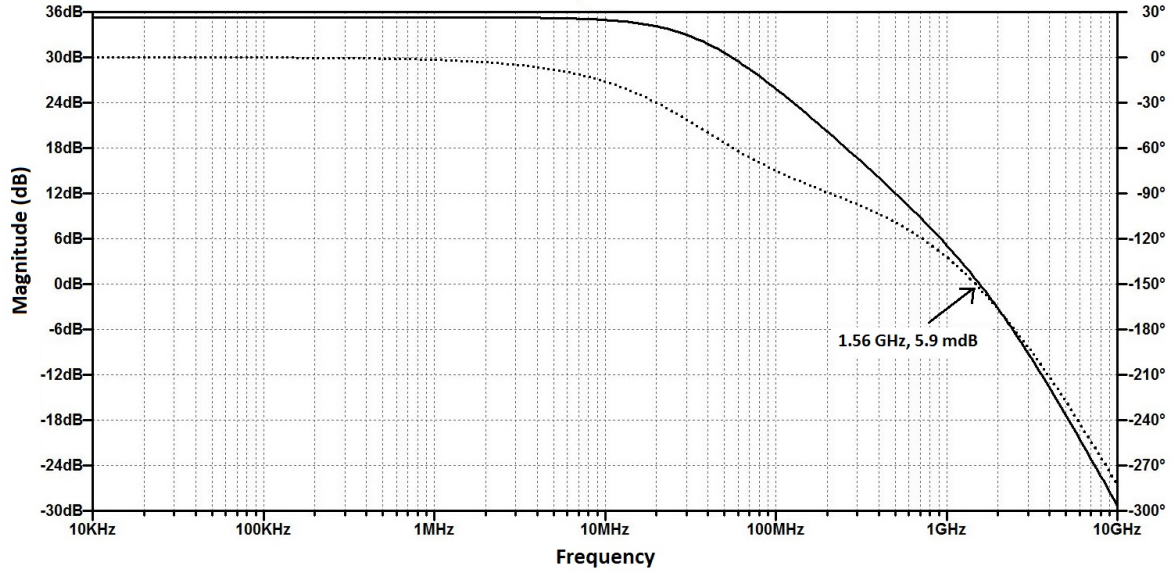

Figure 3.5: Simulated Open-Loop Gain of the Operational Transconductance Amplifier

3.2 Discrete-Time Section

The discrete-time section of the oscillator is responsible for determining the discrete switching condition to the -RLC network. The image shown in Figure 3.3 shows the schematic of the discrete-time circuit. There are multiple ways of implementing the discrete-time switching condition[28]. The circuit in this work consists of a D-type latch, a zero crossing detector, and differential high speed buffer that can be used to apply the signum function to a signal. A zero crossing detector determines when the derivative $\frac{du}{dt}$ goes to zero. The D-type latch is used to sample the sign of u whenever a zero crossing is detected. A resistor divider then scales the output of the latch to the appropriate magnitude for the forcing function. The components used for the discrete switching condition are broken down and explained within the next sections.

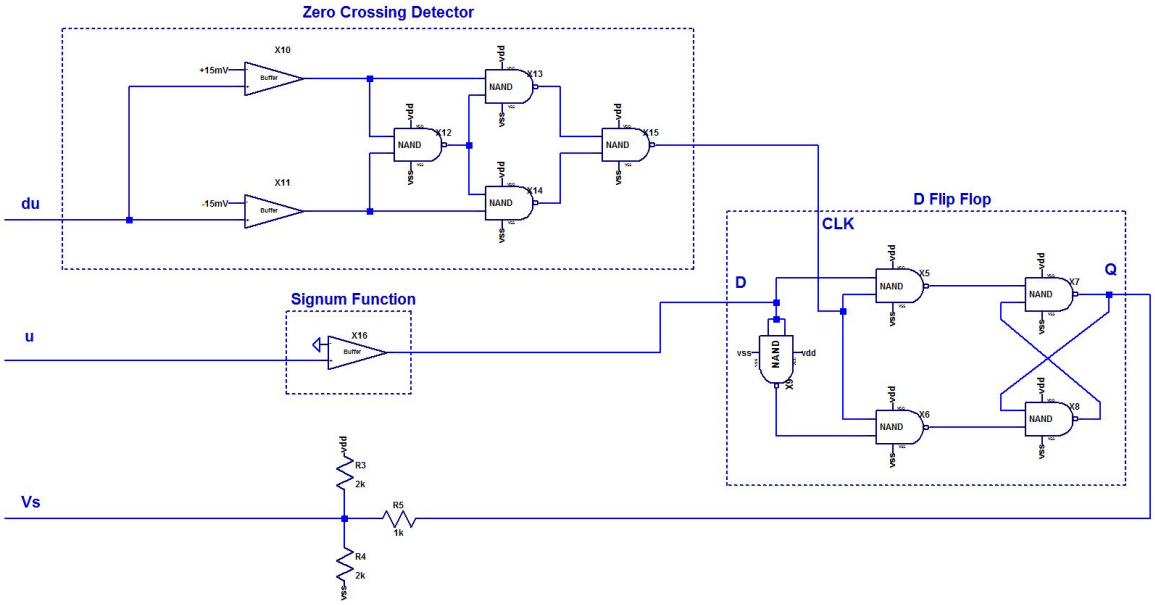

Figure 3.6: Schematic of CMOS Buffer Used as a Signum Function

3.2.1 Signum Function

The main purpose of the Signum or sign function is to convert the continuous state u into a discrete signal. Mathematically, the signum function can be defined for the real variable x as:

$$sgn(x) = \begin{cases} -1 & , x < 0 \\ 0 & , x = 0 \\ 1 & , x > 0. \end{cases} \tag{3.8}$$

The circuit used to accomplish this mathematical function can be seen in Figure 3.7. This circuit consists of an NMOS and PMOS self biased differential amplifier connected in parallel with three inverters connected to the output in series. The inverters are used to help shape the output into a smoother discrete signal. The circuit behaves like a buffer with a differential input, where the output changes state when

$$Vinp > Vinm \rightarrow out = vdd \tag{3.9}$$

and

$$Vinp < Vinm \rightarrow out = vss \tag{3.10}$$

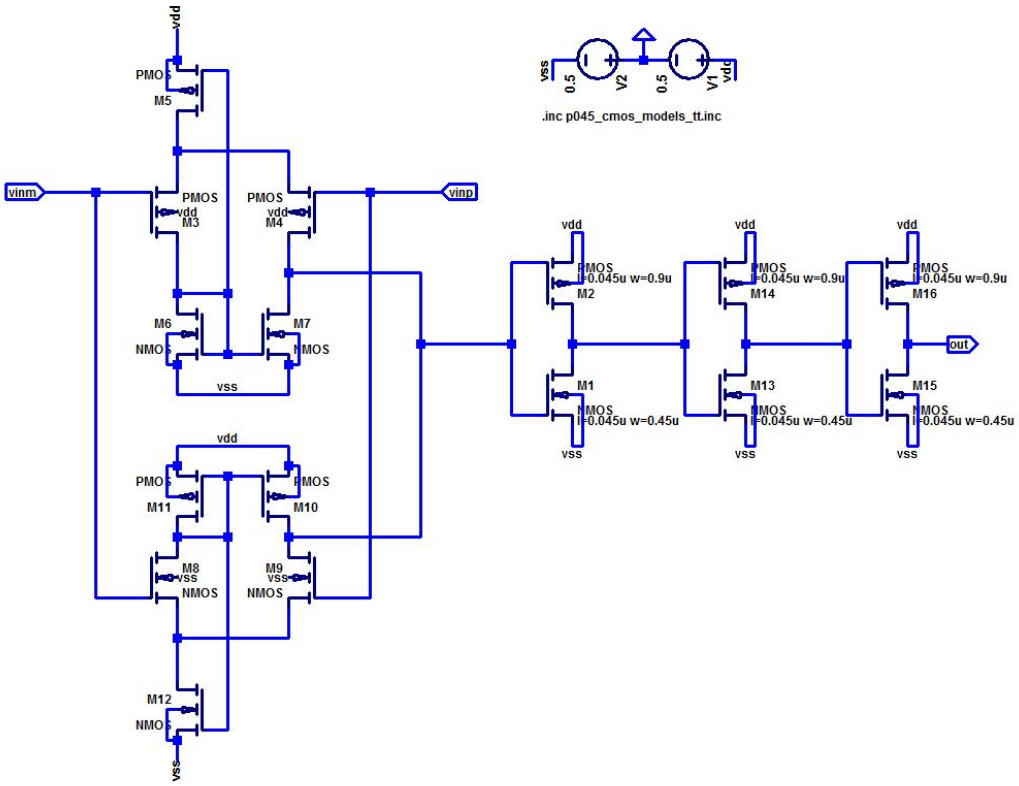

Figure 3.7: Schematic of the Discrete-time Section in the Oscillator Circuit

In this case, vdd is considered a logic level ”High” and vss is considered a logic level ”Low.” Using the NMOS and PMOS differential amplifiers in parallel allows the circuit to operate better with input signals approaching vss or vdd. By using the buffers in parallel, the complementary nature results in a buffer that is robust and works over a wide range of operating voltages [25].

3.2.2 D Type Latch

The D latch is used to provide the logic for the discrete switching condition. A NAND gate architecture is used to implement the D Latch. When the clock signal of the latch is high, the output will reflect the logic level which is present at the D input. When the input of the D latch changes on a low clock signal, the output remains the same. The output of the latch corresponds to the discrete state $s(t)$ for the shift-band oscillator. The clock input to the latch is connected to the output of the zero-crossing detector while the D input is connected to the output of the signum function. Therefore, on zero crossings of the derivative, the output of the

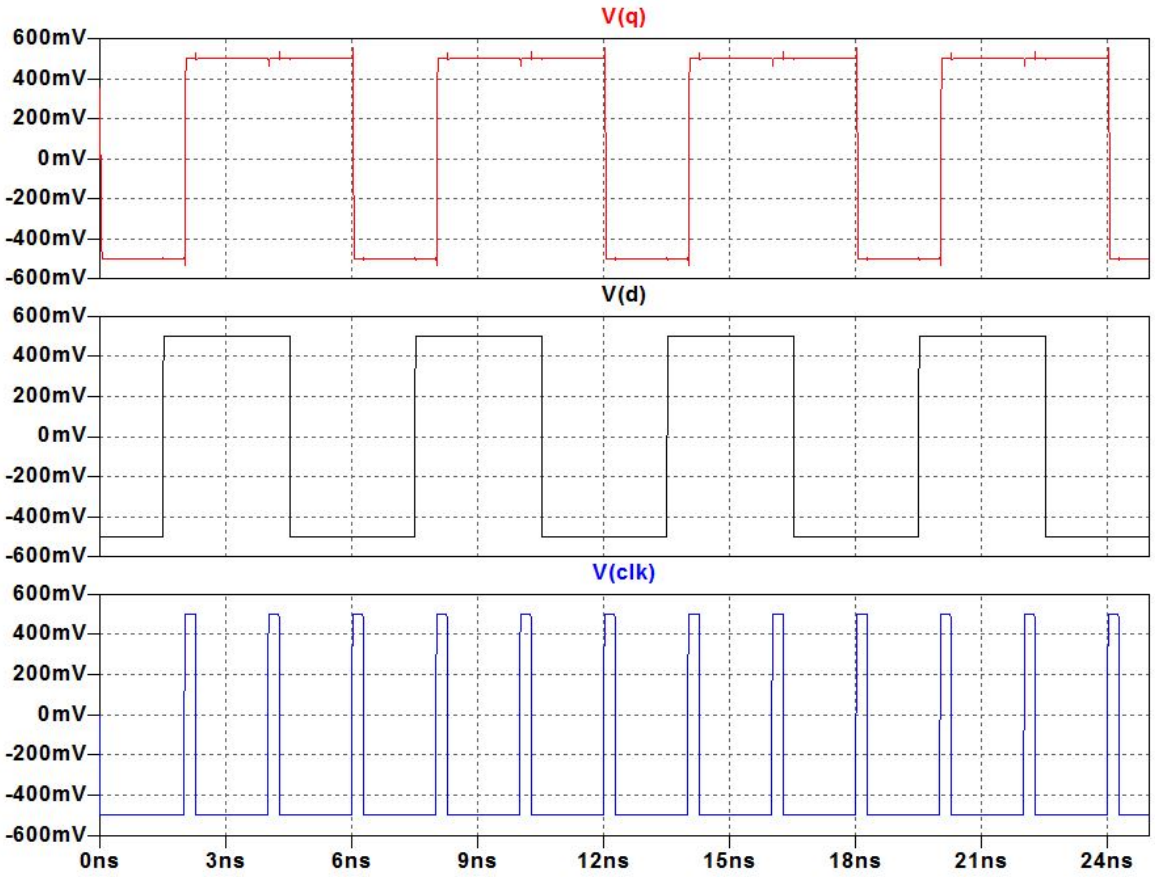

Figure 3.8: SPICE Simulation of D-Type Latch

latch will be set to the $SGN(u(t))$, thus satisfying the switching condition for the system. The schematic for the D latch can be seen in Figure 3.6. The netlist and schematic can also be seen in the Appendix. A simulation of the latch can be seen in Figure 3.8. The propagation delay through the latch is approximately $20ps$.

3.2.3 Zero Crossing Detector

The schematic for the zero-crossing detector (ZCD) can be seen in Figure 3.6. The circuit consists of two differential input buffers previously discussed in section 3.2.1 and 4 NAND gates from the previous section. The NAND gates are configured to create an XOR gate. The XOR gate output only goes high when only one of the two inputs is high and its output is low when the inputs are both high or low. The buffer's inverting input is biased with +15mV and the other input is biased with -15mV. When the input signal goes in between the two bias voltages as it crosses zero, only one of the buffer's output will be high, while the other is low.

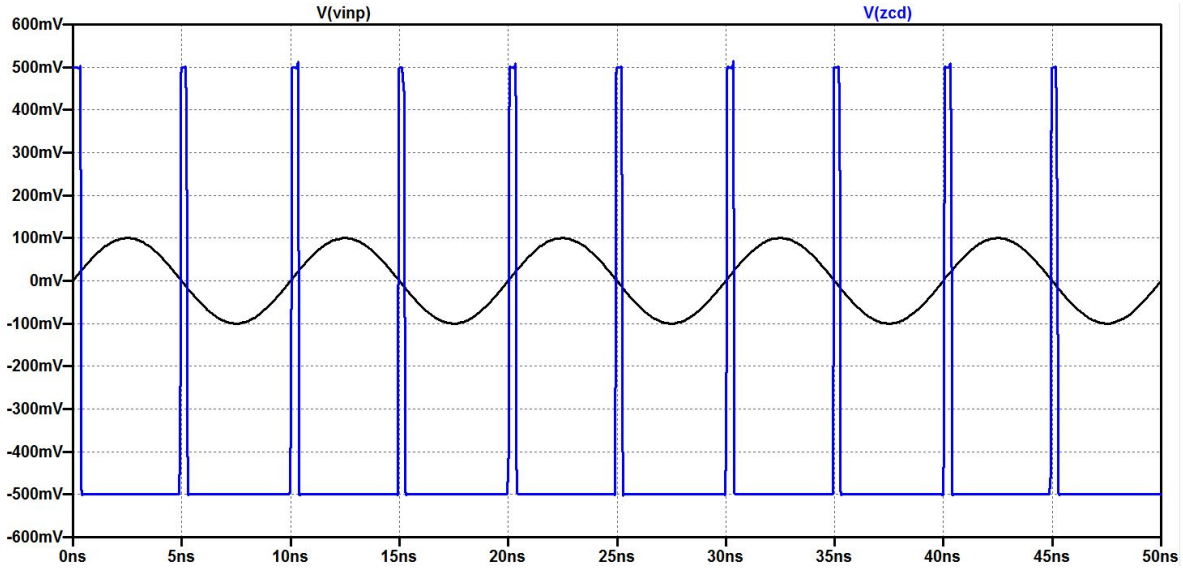

Figure 3.9: SPICE Simulation of Zero-Crossing Detector

This causes the output of the XOR gate to go high while the input signal is in between the two bias voltages. The hysteresis of the circuit can be controlled by tuning the bias voltages on the inverting terminal of the buffers. Figure 3.9 shows a SPICE simulation for the ZCD with a 100MHz sine wave input. For this particular input, the zero crossing detector has a worst case delay of approximately 200ps.

Chapter 4

Simulation

The main focus of this work is to evaluate how a CMOS implementation of the exactly solvable chaotic system could model the hybrid dynamical system from Chapter 2. In this chapter, The process for simulating the CMOS oscillator in LTspiceXVII is discussed as well as the results from simulation. The continuous-time section and discrete-time section from the previous chapter were combined into a single schematic. This schematic can be seen in Figure C.1 located within Appendix C. A netlist for the simulation is also contained within the Appendix.

Transient analysis is used to compute the results for all the circuit simulations within this section. The transient analysis command **.tran 0 4e-6 0 0.1e-9** is used to compute the simulation. In the command, the 4e-6 corresponds to length of time the simulation runs, and the 0.1e-9 is the maximum allowable time step. For the simulation to work correctly it is also important to include the command **.ic V(u)=0.029mV V(du)=0V**. This command sets the initial conditions for the continuous states. The discrete-time section requires a differential voltage of $\pm 15mV$ for the hysteresis of the zero-crossing detector. A $\pm 500mV$ voltage is also required to power the logic devices. Additionally, the OTAs are powered with a differential voltage supply of $\pm 1V$.

The first simulation is a $2.5\mu s$ transient simulation of the oscillator. Figure 4.1 shows the time series for the continuous state $u(t)$ and $s(t)$, as well as the successive returns. From looking at the time series simulation, it would seem that the CMOS implementation produces a similar result to the ideal mathematical simulation previously shown in Chapter 2. The discrete state has some minor distortion on the high side. To get a better idea of the dynamical behavior

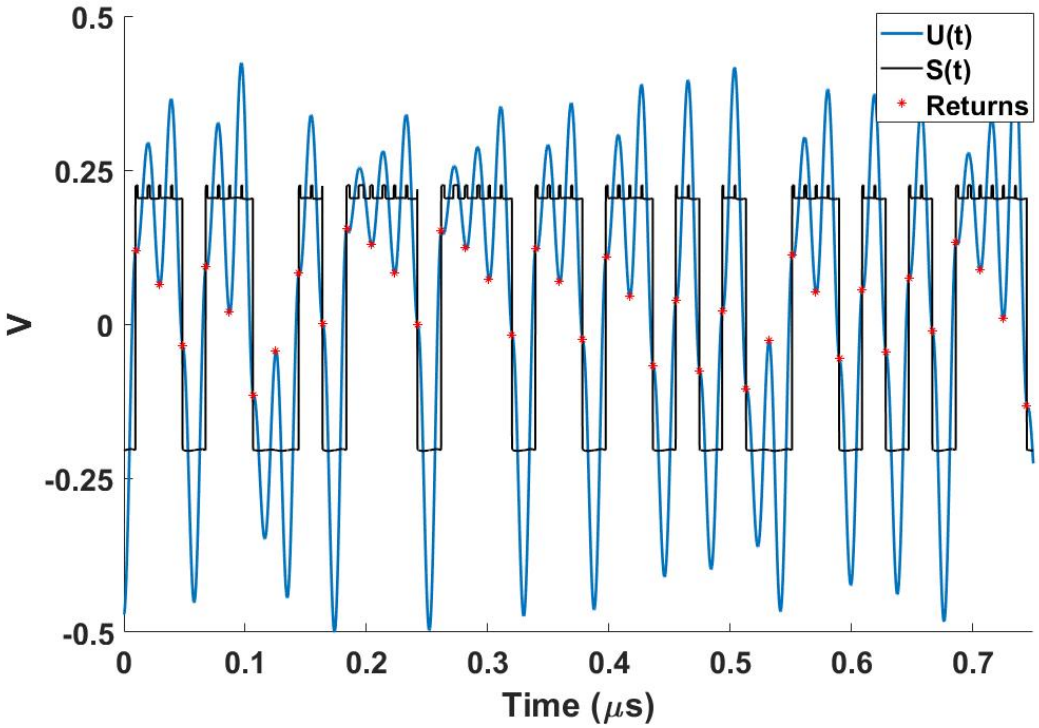

Figure 4.1: SPICE Time Series Simulation of CMOS Exactly Solvable Chaotic Oscillator

of the system, the phase space can is plotted in Figure 4.2. To generate the phase portrait of the system, the continuous state, $\frac{du}{dt}$, is plotted versus u. From the phase portrait, the two strange attractors are present, as well as the thick bands around them.

While on the surface it appears that the system models the dynamics of the hybrid system presented in Chapter 2, the successive returns need to be plotted to see if the exactly solvable nature is still maintained. Figure 4.3 plots the successive returns from the simulation data. This plot is created by plotting future values of u versus current values of u at return times T, u_{n+1} vs. u_n. The return times can be measured from the simulation data by using changes in the discrete state, $s(t)$, and zero-crossings of the derivative, $\frac{du}{dt} = 0$. A MATLAB script was created to do just that, and can be viewed in the Appendix. The average return time, T, for the system can be computed by averaging the time difference between returns. The average return time for this simulation gives $T = 19.23ns$. Taking the inverse of T gives the operating frequency of the CMOS oscillator, which is approximately 51.91MHz. The return map in Figure 4.3 shares a close resemblance to the map from chapter 2.

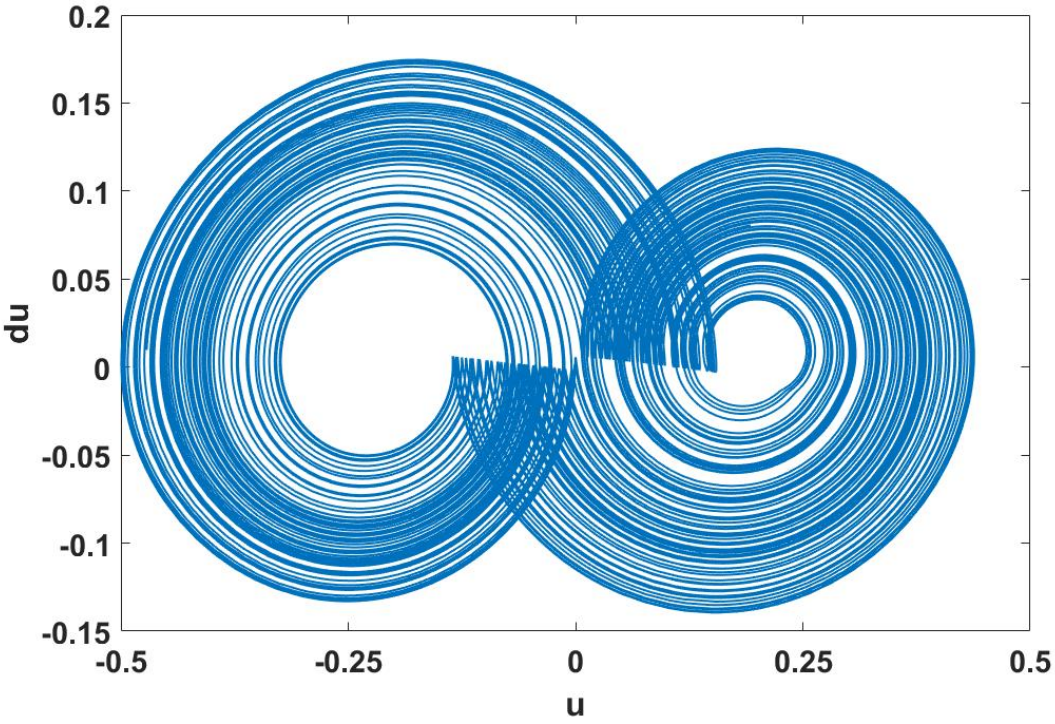

Figure 4.2: Phase Space Plot from Simulation Data

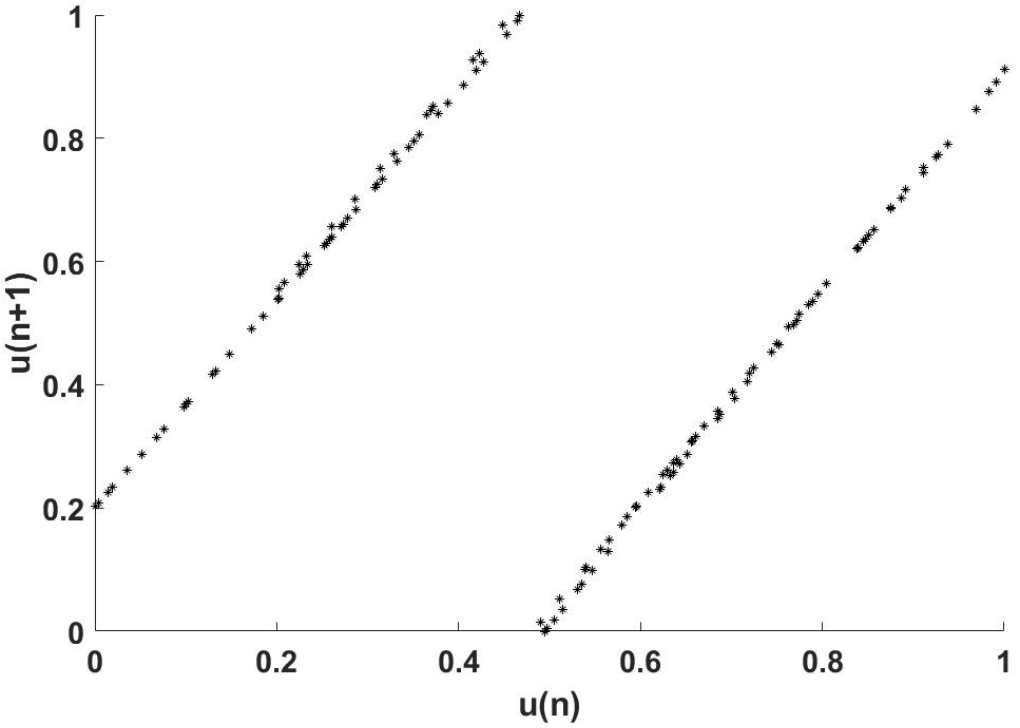

Figure 4.3: Successive Return Map from Simulation Data

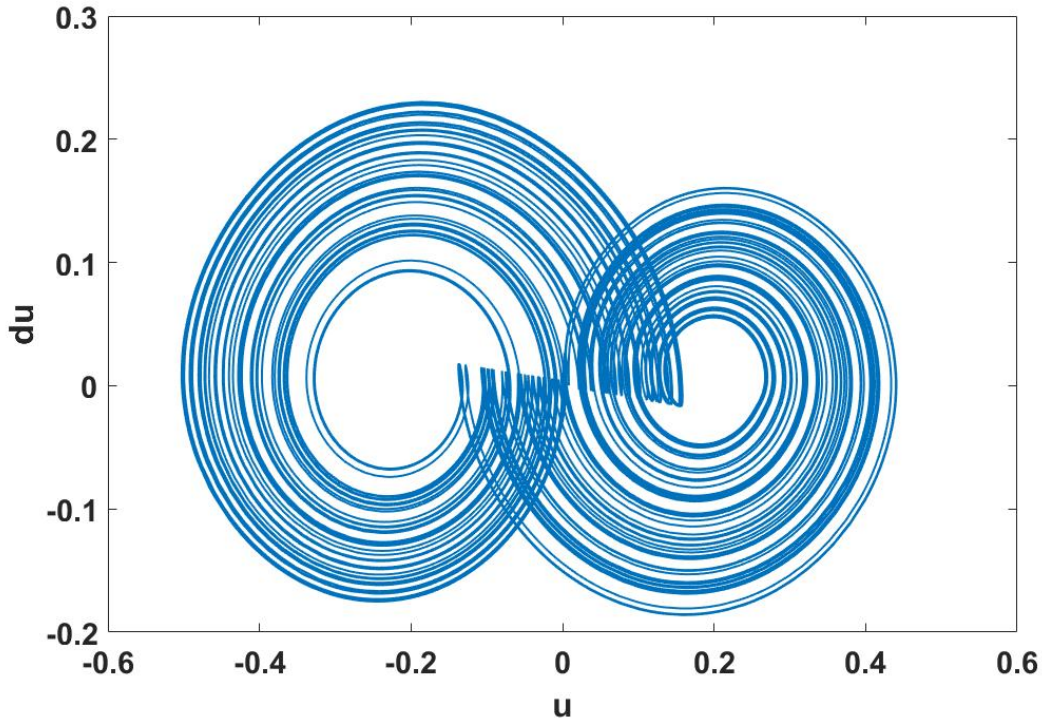

Figure 4.4: Phase Portrait from Higher Frequency Simulation

The -RLC network in the continuous time section is responsible for controlling the frequency of the oscillator. By tuning the passive values within the continuous-time circuit, the frequency can be increased or decreased. The values of CL, R2, and CC in the CMOS oscillator schematic where changed to evaluate the performance of the circuit at higher frequencies. The new values are given as, $CL = 100pF$, $R2 = 2\Omega$, and $CC = 50pF$. Simulating the circuit with the new values increases the frequency of oscillation to around 70MHz. The phase space of the simulation at this frequency can be seen in Figure 4.4. The phase space for this simulation starts to show distortion in the center. This distortion is caused by the propagation delay in the discrete time circuit. For the system to operate correctly, the propagation delay in the discrete time circuit needs to be minimized. The delay causes the return map to deviate from ideal exactly solvable return map.

Chapter 5

Integrated Circuit Design and Fabrication

This chapter presents an overview for designing a custom analog CMOS integrated circuit. While the design and fabrication of an exactly solvable system into an integrated circuit is beyond the scope of this work, the process for design used in this chapter will hopefully lay the groundwork for future work. Integrated circuit design was carried out using the Cadence Virtuoso Analog Design Environment. The CMOS technology used for the design is the C5 $0.5\mu m$ by ON Semiconductor [31]. This process technology allows for a minimum transistor gate length of $0.6\mu m$. A process design kit (PDK) for this process was obtained through MOSIS.

The Cadence integrated circuit design tools can be complicated to set up correctly. A guide for setting up the environment is listed within the Appendix. Information regarding additional learning resources can also be found within the Appendix.

5.1 Design

In order to begin a design, a process design kit (PDK) needs to be obtained. The PDK contains the device models and information concerning the technology used in the design. Data sheets, device characteristics, and instructions are contained within the PDK and may need to be referenced for design purposes. The foundry with which the design is to be fabricated typically has PDKs for all the different technologies they can manufacture.

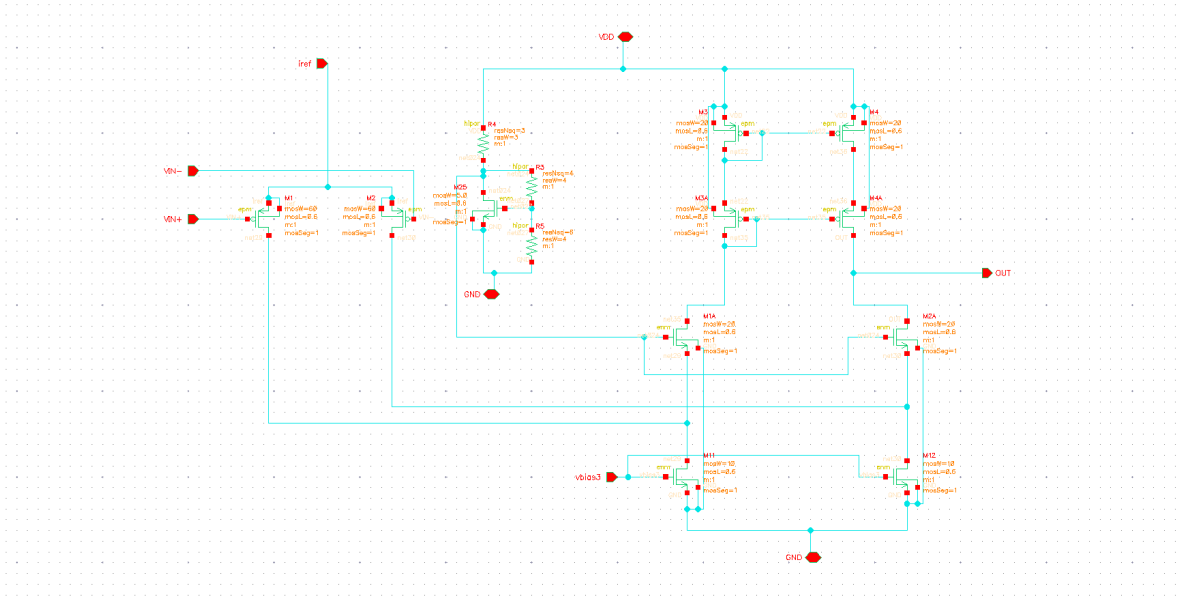

Figure 5.1: CMOS Folded-Cascode Operational Amplifier

5.1.1 Schematic

Once the technology has been selected and the PDK has been configured with Cadence, the next step is to use the Virtuoso schematic editor to create the circuit. Figure 5.1 shows the schematic for a folded-cascode operational amplifier. When creating the schematic, it is important to only use circuit components located within the PDK library. Using components outside of the PDK library can yield different simulation results. Additionally, components located within the PDK library usually already come with a pre-defined defined standard cell layout. It can be beneficial to use a hierarchical design approach for chip design. Create pins for all inputs and outputs of the circuit. For example, in the circuit shown in Figure 5.1, pins were created for inputs such as power, ground, bias voltages, etc. When the schematic is complete, a circuit symbol should be created so the circuit can be used in simulations and other circuits.

5.1.2 Simulation

To simulate a design using Cadence, another schematic needs to be created. This schematic is where the circuit symbol creation from the last section is used. The symbol can be placed along with all the different input and output stimuli required for the circuit to operate properly. Figure 5.2 shows an example of a schematic made for simulation. The circuit symbol has pin

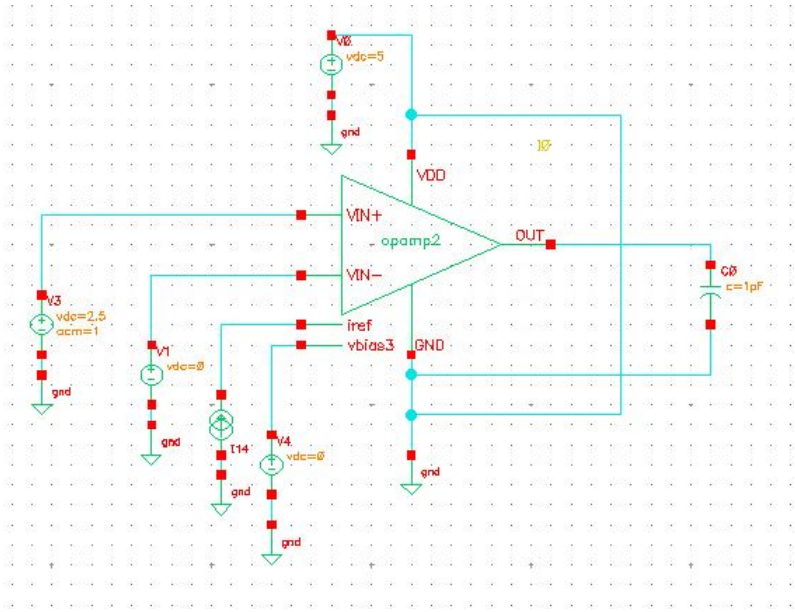

Figure 5.2: Schematic for simulation

connections for all the required connections. Using ideal circuit elements from other libraries in this schematic is acceptable since it is just used for simulation and characterization. Once the schematic for simulation has been created, the circuit can be simulated by launching the Analog Design Environment (ADE). From the ADE window, a variety of simulations can be performed from transient analysis, AC analysis, Monte Carlo, etc. Monte Carlo analysis is recommended to ensure that physical variations in devices do not have disastrous affects on circuits. From the ADE window it is possible to use third-party simulators, such as HSPICE, if desired [32].

5.2 Layout

Once a circuit has been designed and verified with simulation, the next step is the physical layout of the circuit. Physical layout can be a complicated process, but there are many physical layout techniques that can be utilized to improve the robustness and reliability of integrated circuits. Common centroid layout and using transistors in parallel to create large devices can help reduce physical device mismatch [21]. Figure 5.3 shows a fully custom integrated circuit physical layout. This cell not only includes the circuits that are to be fabricated, but it also

includes things like bond pads, guard rings, electrostatic discharge (ESD) protection, and orientation marks. All of these item requirements will differ depending on the process used for fabrication. A physical design rules document should be contained within the PDK received from the foundry that will contain specific instructions on how to create these structures.

Once a physical layout is completed, a design rule check (DRC) simulation can be done to ensure the design meets the foundry specifications. Any rule violations can be checked within the physical design rules document for clarification. It is also a good idea to perform a layout versus schematic (LVS) simulation. The LVS simulation is capable of extracting a netlist from the physical layout, along with parasitic elements such as resistor-capacitor (RC) structures within the silicon. This netlist is then compared with the schematic netlist to ensure everything matches. The extracted netlist can also be simulated with parasitic elements to ensure it does not affect the operation of the circuit.

5.3 Test

The next step in the design process is hardware verification. Once the fully custom physical layout has been fabricated by the foundry, it needs to be tested. In order to test the die, it needs to be wire bonded and packaged like the photo in Figure 5.4 ,or it can be tested under a probe station. If the die is wire bonded and packaged, great care needs to be taken in order to avoid damaging the circuits.

5.4 Summary

It is the hope that this chapter can provide some basic knowledge for the design and fabrication of custom integrated circuits. Documentation for the Cadence Analog design environment can be found within the the Cadence install directory. The documents include user guides and tutorials, and are quite comprehensive. Additionally, the documentation that comes along with the process design kit is crucial for designing successful chips. The website accompanying the textbook in [25] is also an excellent resource for all things CMOS integrated circuit design.

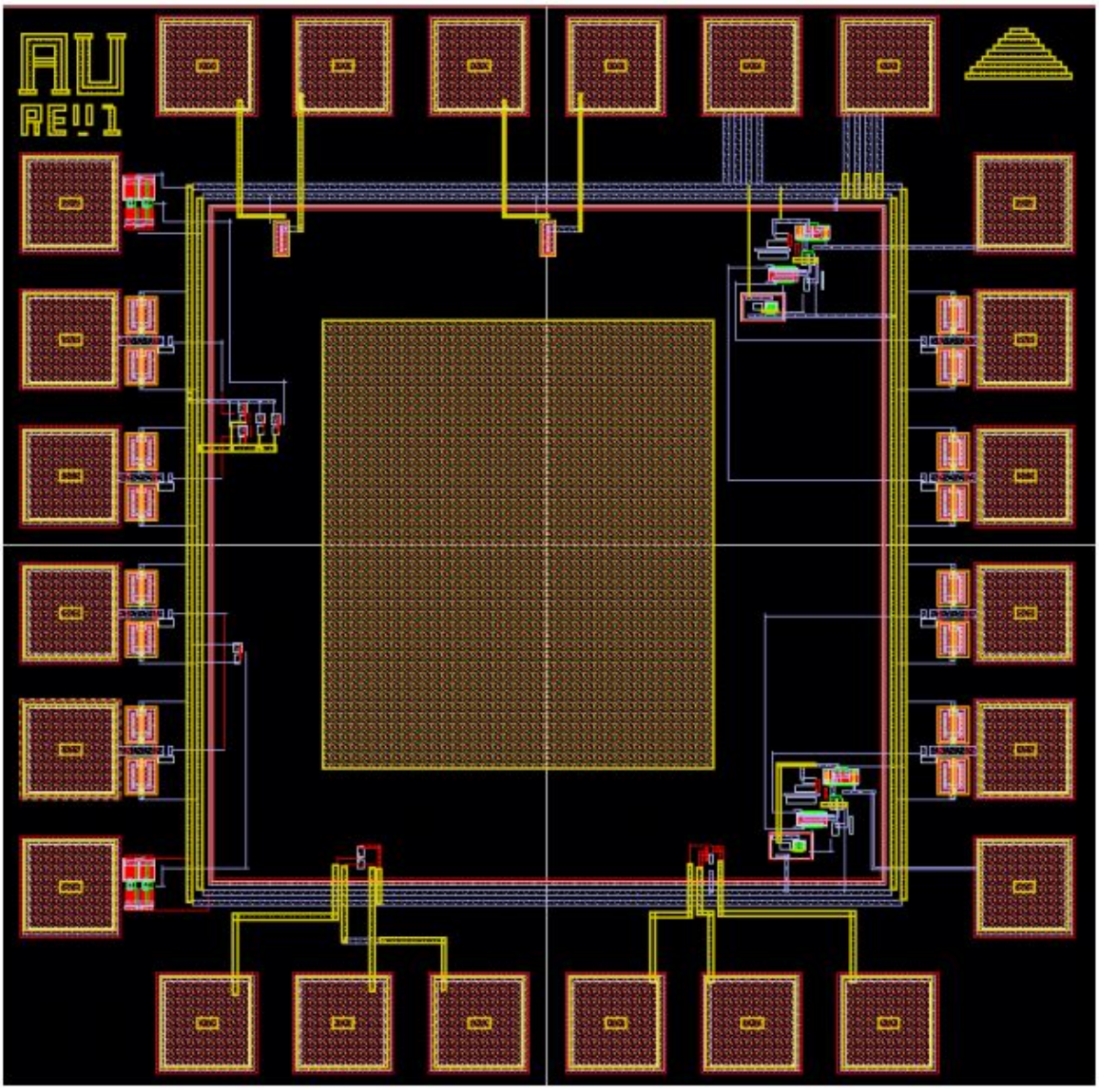

Figure 5.3: Fully Custom Physical Layout

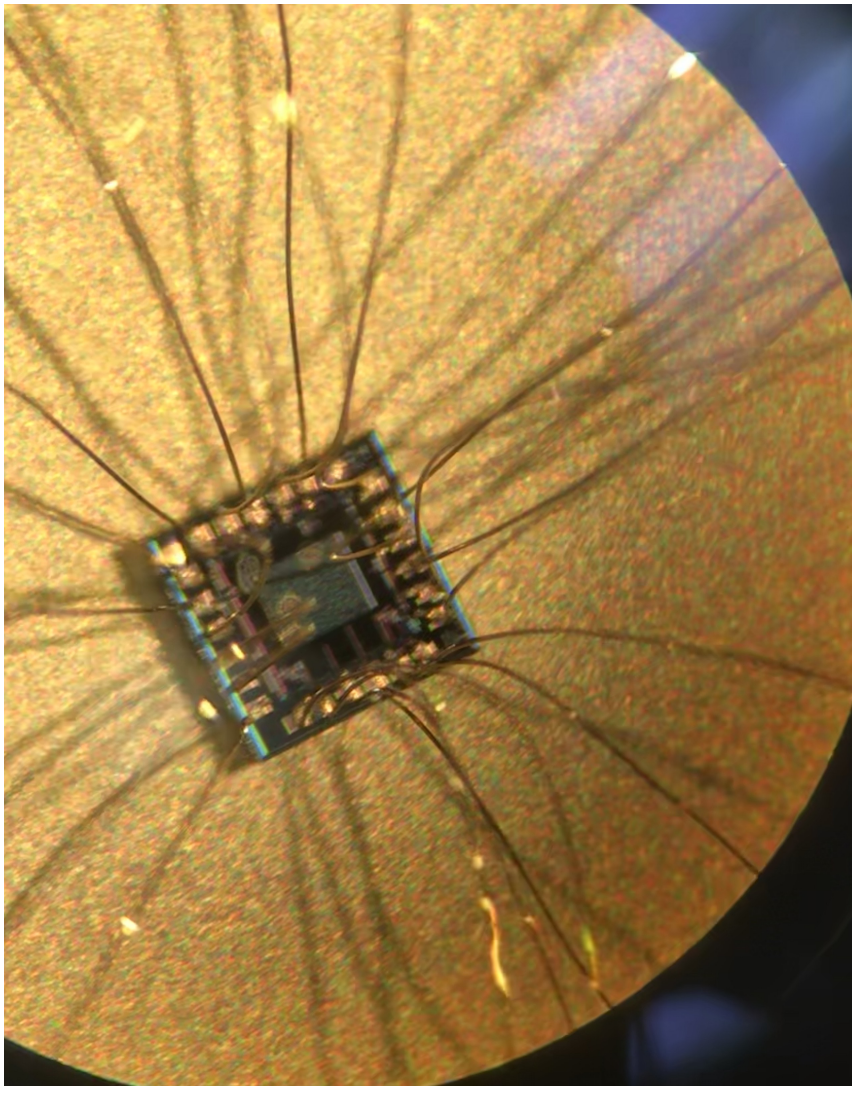

Figure 5.4: Photo of Die under microscope after wire bonding

Chapter 6

Conclusion and Future Work

It has been shown that high frequency chaotic waveform generation can be simulated using CMOS devices. The CMOS implementation was able to model the exactly solvable system up to approximately 50 MHz in SPICE simulations. The design was done using a 45nm CMOS process with a BSIM4 SPICE model. The system utilizes an active -RLC network with a discrete time switching circuit. The speed of the system is easily tuned by changing values of the passive components within the active -RLC circuit. These passives were tuned so that the operating frequency of the system was increased to 70 MHz. However, due to physical limitations in the design, the performance of the system is greatly diminished at higher speeds.

One physical limitation is the finite bandwidth of the operational transconductance amplifiers used in the active -RLC network. For future designs, one way this may be mitigated is by using modern Silicon Germanium (SiGe) process technology. For example, high performance 90nm SiGe BiCMOS has heterojunctuion bipolar transistors (HBT) with $f_T = 300GHz$ and $f_{MAX} = 360GHz$ [33]. BiCMOS technology allows the use of FETs to be fabricated on the same wafer as bipolar transistors. Using these bipolar devices for the input differential pair of the operational transconductance amplifiers could potentially allow for wideband operation. Active filters operating in the gigahertz range have been synthesized from amplifiers made with SiGe HBTs [14].

Another limitation for high frquency implementation of the system is the propagation delay in the discrete time section of the circuit. The delay through the digital circuitry causes significant distortion in the dynamics of the system. Shrinking the device size can certainly minimize the propagation delay within the digital circuitry, but it does not solve the problem.

A technique for compensation of this imperfect switching has been studied for hybrid systems in [34]. Although this seems counter intuitive, short term behavior can be reliably predicted for this hybrid system. Future work on higher frequency implementation could benefit greatly by designing compensation technique, such as this, in hardware.

References

[1] N.J. Corron, "An Exactly Solvable Chaotic Differential Equation," *Dynamics of Continuous, Discrete and Impulsive Systems A*, vol. 16, pp. 777788, 2009.

[2] Seth D. Cohen and Daniel J. Gauthier, "A pseudo-matched filter for chaos," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 22, no. 3, pp. 033148, 2012.

[3] E.N. Lorenz, "Deterministic Nonperiodic Flow" *Journal of Atmospheric Sciences*, vol. 20, pp. 130-141, 1963.

[4] O.E. Rössler, "An Equation for Continuous Chaos," *Physics Letters A*, vol. 57, no. 5, 1976.

[5] A.N. Beal, J.N. Blakely, N.J. Corron and R.N. Dean, "High frequency oscillators for chaotic radar," *Proc. of the SPIE Conf. on Radar Sensor Technology XX*, 98290H (12 May 2016).

[6] R. Brown, "Generalizations of the Chua equations," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 40, no. 11, pp. 878884, 1993.

[7] Benjamin K. Rhea, Edmon Perkins, and Robert N. Dean, "High frequency realization of non-autonomous nonlinear transistor circuit," AIP Advances, vol. 9, 2019, 8pp.

[8] R.C. Harrison, B.K. Rhea, F. T. Werner and R.N. Dean, A 4 MHz chaotic oscillator based on a jerk system, *Proc. of the 4th International Conference on Applications in Nonlinear Dynamics*, Denver, CO, Aug 28 Sep. 1, 2016, pp 41-51.

[9] S.H. Strogatz, *Nonlinear Dynamics And Chaos: With Applications To Physics, Biology, Chemistry, And Engineering*, CRC Press, First Edition, 2000.

[10] Analog Devices, *LT1057/LT1058 Dual and Quad, JFET Input Precision High Speed Op Amps*, 10578fd.pdf

[11] Analog Devices, *AD633 Low Cost Analog Multiplier*, AD633.pdf, 2015

[12] P. Horowitz, "Build a Lorenz Attractor", Harvard Univ., Cambridge, MA, USA. [Online]. Available: http://seti.harvard.edu/unusual_stuff/misc/lorenz.htm

[13] C.S. Pappu, B.C. Flores, P.S. Debroux, and J.E. Boehm, "An Electronic Implementation of Lorenz Chaotic Oscillator Synchronization for Bistatic Radar Applications," *IEEE Transactions on Aerospace and Electronic Systems* PP. 1-1. 10.1109/TAES.2017.2680661, 2017.

[14] S.P. Voinigescu et al., "Design methodology and applications of SiGe BiCMOS cascode opamps with up to 37-GHz unity gain bandwidth," *IEEE Compound Semiconductor Integrated Circuit Symposium, 2005. CSIC '05.*, Palm Springs, CA, USA, 2005, pp. 3 pp

[15] M.P. Kennedy, "Chaos in the Colpitts oscillator", *IEEE Trans. Circuits Syst. I*, vol. 41 pp. 771-774, 1994.

[16] J.L. Hudson and J.C. Mankin, "Chaos in the BelousovZhabotinskii reaction," *The Journal of Chemical Physics* vol. 74, 6171, 1981.

[17] B. LeBaron, Chaos and Nonlinear Forecastability in Economics and Finance, *Phil. Trans. R. Soc. Land. A*, vol. 348, no. 1688, pp. 397404, 1994.

[18] R.W. Rollins and E.R. Hunt, "Exactly Solvable Model of a Physical System Exhibiting Universal Chaotic Behavior" *Physical Review Letters*, vol. 49, no. 18, 1982.

[19] Richard C. Jaeger and Travis N. Blalock, *Microelectronic Circuit Design*. McGraw-Hill Education, Fourth edition (March 1, 2010)

[20] T. Saito and H. Fujita, "Chaos in a manifold piecewise linear system," *Electronics and Communications in Japan (Part I: Communications)*, vol. 64, no. 10, pp. 917, 1981.

[21] Tony Chan Carusone, David Johns, Kenneth Martin, *Analog Integrated Circuit Design.* John Wiley & Sons Inc., Second Edition, (December 13, 2011)

[22] K. Umeno, "Method of constructing exactly solvable chaos," *Phys. Rev. E,*, vol. 55, no. 5, 1997.

[23] N. J. Corron, R. M. Cooper, and J. N. Blakely, Analytically solvable chaotic oscillator based on a first-order filter, *Chaos: An Interdisciplinary Journal of Nonlinear Science,* vol. 26, no. 2, p. 023104, 2016.

[24] N. Corron, J. Blakely, and M. Stahl, "A matched filter for chaos," *Chaos: An Interdisciplinary Journal of Nonlinear Science,* vol. 20, no. 2, pp. 023123, 2010.

[25] R. Jacob Baker, *CMOS Circuit Design, Layout, and Simulation.* Wiley-IEEE Press, New Jersey, Third Edition, 2010.

[26] Behzad Razavi, *Design of Analog CMOS Integrated Circuits.* McGraw-Hill Education, Second Edition, 2016.

[27] B. K. Rhea, F. T. Werner, R. C. Harrison, A. N. Beal, and R. N. Dean, Electronic chaotic oscillator realization with potential uses in communication systems, *IMAPS 14th International Conference and Exhibition on Device Packaging,* 2017.

[28] A. N. Beal, "Analog Computation of a High Frequency Exactly Solvable Chaotic Communication System Using State Variable Networks," Dissertation, Auburn University, 2012.

[29] Ned J. Corron, Jonathan N. Blakely, and Shawn D. Pethel, "Communicating with Exactly Solvable Chaos," in; Henry Leung, ed., Chaotic Signal Processing, pp. 49-84 (SIAM, 2014).

[30] B. Wilamowski and J. Irwin, *The Industrial Electronics Handbook.* CRC Press, Boca Raton, FL, Second Edition, 2011.

[31] *C5:* $0.5\mu m$ *Process Technology,* C5-D.PDF, ON Semiconductor, 2016. [Online]. Available: `https://www.onsemi.com/pub/Collateral/C5-D.PDF`

[32] *Virtuoso Analog Design Environment L Datasheet*, virtuoso-analog-design-environment-l-ds.pdf, Cadence, 2011. [Online]. Available: `https://www.cadence.com/content/dam/cadence-www/global/en_US/documents/tools/custom-ic-analog-rf-design/virtuoso-analog-design-environment-l-ds.pdf`

[33] J. J. Pekarik et al., A 90nm SiGe BiCMOS technology for mm-wave and high-performance analog applications, *Proc. BCTM,* pp. 92-95, 2014

[34] N. J. Corron, Compensation for imperfect switching in a chaotic hybrid oscillator circuit, in *Circuits and Systems (MWSCAS), 2015 IEEE 58th International Midwest Symposium on,* pp. 1-4, IEEE, 2015.

[35] A. Beal, J. Bailey, S. Hale, R. Dean, M. Hamilton, J. Tugnait, D. Hahs, and N. Corron, Design and simulation of a high frequency exact solvable chaotic oscillator, in *MILCOM 2012-2012 IEEE Military Communications Conference,* pp. 16, IEEE, 2012.

Appendices

Appendix A

SPICE MODELS

```
* Customized PTM 45 NMOS

.model  NMOS  nmos  level = 54

+version = 4.0    binunit = 1    paramchk= 1    mobmod  = 0
+capmod  = 2      igcmod  = 1    igbmod  = 1    geomod  = 1
+diomod  = 1      rdsmod  = 0    rbodymod= 1    rgatemod= 1
+permod  = 1      acnqsmod= 0    trnqsmod= 0

* parameters related to the technology node
+tnom = 27    epsrox = 3.9
+eta0 = 0.0049    nfactor = 2.1    wint = 5e-09
+cgso = 1.1e-10    cgdo = 1.1e-10    xl = -2e-08

* parameters customized by the user
+toxe = 1.75e-09    toxp = 1.1e-09    toxm = 1.75e-09    toxref = 1.75e-09
+dtox = 6.5e-10    lint = 3.75e-09
+vth0 = 0.471    k1 = 0.53    u0 = 0.04359    vsat = 147390
+rdsw = 155    ndep = 3.3e+18    xj = 1.4e-08

* secondary parameters
+ll      = 0            wl      = 0           lln     = 1           wln     = 1
+lw      = 0            ww      = 0           lwn     = 1           wwn     = 1
+lwl     = 0            wwl     = 0           xpart   = 0
+k2      = 0.01         k3      = 0
+k3b     = 0            w0      = 2.5e-006    dvt0    = 1           dvt1    = 2
+dvt2    = -0.032       dvt0w   = 0           dvt1w   = 0           dvt2w   = 0
+dsub    = 0.1          minv    = 0.05        voffl   = 0           dvtp0   = 1.0e-009
+dvtp1   = 0.1          lpe0    = 0           lpeb    = 0
+ngate   = 2e+020       nsd     = 2e+020      phin    = 0
+cdsc    = 0.000        cdscb   = 0           cdscd   = 0           cit     = 0
+voff    = -0.13        etab    = 0
```

```
+vfb     = -0.55        ua      = 6e-010       ub      = 1.2e-018
+uc      = 0            a0      = 1.0          ags     = 1e-020
+a1      = 0            a2      = 1.0          b0      = 0            b1      = 0
+keta    = 0.04         dwg     = 0            dwb     = 0            pclm    = 0.04
+pdiblc1 = 0.001        pdiblc2 = 0.001        pdiblcb = -0.005       drout   = 0.5
+pvag    = 1e-020       delta   = 0.01         pscbe1  = 8.14e+008    pscbe2  = 1e-007
+fprout  = 0.2          pdits   = 0.08         pditsd  = 0.23         pditsl  = 2.3e+006
+rsh     = 5            rsw     = 85           rdw     = 85
+rdswmin = 0            rdwmin  = 0            rswmin  = 0            prwg    = 0
+prwb    = 6.8e-011     wr      = 1            alpha0  = 0.074        alpha1  = 0.005
+beta0   = 30           agidl   = 0.0002       bgidl   = 2.1e+009     cgidl   = 0.0002
+egidl   = 0.8

+aigbacc = 0.012        bigbacc = 0.0028       cigbacc = 0.002
+nigbacc = 1            aigbinv = 0.014        bigbinv = 0.004        cigbinv = 0.004
+eigbinv = 1.1          nigbinv = 3            aigc    = 0.012        bigc    = 0.0028
+cigc    = 0.002        aigsd   = 0.012        bigsd   = 0.0028       cigsd   = 0.002
+nigc    = 1            poxedge = 1            pigcd   = 1            ntox    = 1

+xrcrg1  = 12           xrcrg2  = 5
+cgbo    = 2.56e-011    cgdl    = 2.653e-10
+cgsl    = 2.653e-10    ckappas = 0.03         ckappad = 0.03         acde    = 1
+moin    = 15           noff    = 0.9          voffcv  = 0.02

+kt1     = -0.11        kt1l    = 0            kt2     = 0.022        ute     = -1.5
+ua1     = 4.31e-009    ub1     = 7.61e-018    uc1     = -5.6e-011    prt     = 0
+at      = 33000

+fnoimod = 1            tnoimod = 0

+jss     = 0.0001       jsws    = 1e-011       jswgs   = 1e-010       njs     = 1
+ijthsfwd= 0.01         ijthsrev= 0.001        bvs     = 10           xjbvs   = 1
+jsd     = 0.0001       jswd    = 1e-011       jswgd   = 1e-010       njd     = 1
+ijthdfwd= 0.01         ijthdrev= 0.001        bvd     = 10           xjbvd   = 1
+pbs     = 1            cjs     = 0.0005       mjs     = 0.5          pbsws   = 1
+cjsws   = 5e-010       mjsws   = 0.33         pbswgs  = 1            cjswgs  = 3e-010
+mjswgs  = 0.33         pbd     = 1            cjd     = 0.0005       mjd     = 0.5
+pbswd   = 1            cjswd   = 5e-010       mjswd   = 0.33         pbswgd  = 1
+cjswgd  = 5e-010       mjswgd  = 0.33         tpb     = 0.005        tcj     = 0.001
+tpbsw   = 0.005        tcjsw   = 0.001        tpbswg  = 0.005        tcjswg  = 0.001
+xtis    = 3            xtid    = 3

+dmcg    = 0e-006       dmci    = 0e-006       dmdg    = 0e-006       dmcgt   = 0e-007
+dwj     = 0.0e-008     xgw     = 0e-007       xgl     = 0e-008
```

```
+rshg    = 0.4          gbmin   = 1e-010       rbpb    = 5           rbpd    = 15
+rbps    = 15           rbdb    = 15           rbsb    = 15          ngcon   = 1
```

```
* Customized PTM 45 PMOS

.model  PMOS  pmos  level = 54

+version = 4.0    binunit = 1    paramchk= 1    mobmod  = 0
+capmod  = 2      igcmod  = 1    igbmod  = 1    geomod  = 1
+diomod  = 1      rdsmod  = 0    rbodymod= 1    rgatemod= 1
+permod  = 1      acnqsmod= 0    trnqsmod= 0

* parameters related to the technology node
+tnom = 27    epsrox = 3.9
+eta0 = 0.0049    nfactor = 2.1    wint = 5e-09
+cgso = 1.1e-10    cgdo = 1.1e-10    xl = -2e-08

* parameters customized by the user
+toxe = 1.85e-09    toxp = 1.1e-09    toxm = 1.85e-09    toxref = 1.85e-09
+dtox = 7.5e-10    lint = 3.75e-09
+vth0 = -0.423    k1 = 0.491    u0 = 0.00432    vsat = 70000
+rdsw = 155    ndep = 2.54e+18    xj = 1.4e-08

*secondary parameters
+ll      = 0            wl      = 0            lln     = 1            wln     = 1
+lw      = 0            ww      = 0            lwn     = 1            wwn     = 1
+lwl     = 0            wwl     = 0            xpart   = 0
+k2      = -0.01        k3      = 0
+k3b     = 0            w0      = 2.5e-006     dvt0    = 1            dvt1    = 2
+dvt2    = -0.032       dvt0w   = 0            dvt1w   = 0            dvt2w   = 0
+dsub    = 0.1          minv    = 0.05         voffl   = 0            dvtp0   = 1e-009
+dvtp1   = 0.05         lpe0    = 0            lpeb    = 0
+ngate   = 2e+020       nsd     = 2e+020       phin    = 0
+cdsc    = 0.000        cdscb   = 0            cdscd   = 0            cit     = 0
+voff    = -0.126       etab    = 0
+vfb     = 0.55         ua      = 2.0e-009     ub      = 0.5e-018
+uc      = 0            a0      = 1.0          ags     = 1e-020
+a1      = 0            a2      = 1            b0      = -1e-020      b1      = 0
+keta    = -0.047       dwg     = 0            dwb     = 0            pclm    = 0.12
+pdiblc1 = 0.001        pdiblc2 = 0.001        pdiblcb = 3.4e-008     drout   = 0.56
+pvag    = 1e-020       delta   = 0.01         pscbe1  = 8.14e+008    pscbe2  = 9.58e-007
+fprout  = 0.2          pdits   = 0.08         pditsd  = 0.23         pditsl  = 2.3e+006
+rsh     = 5            rsw     = 85           rdw     = 85
+rdswmin = 0            rdwmin  = 0            rswmin  = 0            prwg    = 3.22e-008
+prwb    = 6.8e-011     wr      = 1            alpha0  = 0.074        alpha1  = 0.005
+beta0   = 30           agidl   = 0.0002       bgidl   = 2.1e+009     cgidl   = 0.0002
+egidl   = 0.8
```

```
+aigbacc = 0.012        bigbacc = 0.0028       cigbacc = 0.002
+nigbacc = 1            aigbinv = 0.014        bigbinv = 0.004        cigbinv = 0.004
+eigbinv = 1.1          nigbinv = 3            aigc    = 0.69         bigc    = 0.0012
+cigc    = 0.0008       aigsd   = 0.0087       bigsd   = 0.0012       cigsd   = 0.0008
+nigc    = 1            poxedge = 1            pigcd   = 1            ntox    = 1

+xrcrg1  = 12           xrcrg2  = 5
+cgbo    = 2.56e-011    cgdl    = 2.653e-10
+cgsl    = 2.653e-10    ckappas = 0.03         ckappad = 0.03         acde    = 1
+moin    = 15           noff    = 0.9          voffcv  = 0.02

+kt1     = -0.11        kt1l    = 0            kt2     = 0.022        ute     = -1.5
+ua1     = 4.31e-009    ub1     = 7.61e-018    uc1     = -5.6e-011    prt     = 0
+at      = 33000

+fnoimod = 1            tnoimod = 0

+jss     = 0.0001       jsws    = 1e-011       jswgs   = 1e-010       njs     = 1
+ijthsfwd= 0.01         ijthsrev= 0.001        bvs     = 10           xjbvs   = 1
+jsd     = 0.0001       jswd    = 1e-011       jswgd   = 1e-010       njd     = 1
+ijthdfwd= 0.01         ijthdrev= 0.001        bvd     = 10           xjbvd   = 1
+pbs     = 1            cjs     = 0.0005       mjs     = 0.5          pbsws   = 1
+cjsws   = 5e-010       mjsws   = 0.33         pbswgs  = 1            cjswgs  = 3e-010
+mjswgs  = 0.33         pbd     = 1            cjd     = 0.0005       mjd     = 0.5
+pbswd   = 1            cjswd   = 5e-010       mjswd   = 0.33         pbswgd  = 1
+cjswgd  = 5e-010       mjswgd  = 0.33         tpb     = 0.005        tcj     = 0.001
+tpbsw   = 0.005        tcjsw   = 0.001        tpbswg  = 0.005        tcjswg  = 0.001
+xtis    = 3            xtid    = 3

+dmcg    = 0e-006       dmci    = 0e-006       dmdg    = 0e-006       dmcgt   = 0e-007
+dwj     = 0.0e-008     xgw     = 0e-007       xgl     = 0e-008

+rshg    = 0.4          gbmin   = 1e-010       rbpb    = 5            rbpd    = 15
+rbps    = 15           rbdb    = 15           rbsb    = 15           ngcon   = 1
```

Appendix B

MATLAB Code

B.1 Code for Plotting Return Map from LTspice Data

```
% This script finds the locations of the returns for the shift-band
% oscillator using data from LTspice Simulations. The raw plot data from
% LTspice can be saved as txt file and the txt file can be imported into
% MATLAB for post processing. This file will locate the successive returns
% and plot the return map.
clearvars; clc; close all;

% LTspice raw plot file
x = load('20MHz_CMOS.txt');
t = x(:,1);
ud = x(:,2);
u = x(:,3);
s  = x(:,4);

% Hysteresis Values if used
hys_high = 0.015;
hys_low  = -0.015;

count = 0;
zx = zeros;

% This loop records all indices where the derivative falls in between
% the hysteresis high and low value.
```

```
for j=1:length(ud)
    if ud(j) <= hys_high && ud(j) >= hys_low
        zx(count+1,1) = j;
        count = count + 1;
    end
end

maxS = max(s);
minS = min(s);
% This loops gets rid of indices where zero crossings > and <
% the magnitude of S
for ii=1:length(zx)
   if u(zx(ii)) > maxS-0.02 || u(zx(ii)) < minS
   zx(ii) = 0;
   end

end
zx(zx==0) = [];

% This loop averages out the duplicate duplicate return indices
count = 1;
for i=1:length(zx)-1
   if zx(i+1) - zx(i) == 1
        z(i,count) = zx(i);
   else
       count = count + 1;
   end
end
z(z==0) = NaN;
zx = round(mean(z,1,'omitnan'));



% Normalizes data between 0 and 1
% RTU = mat2gray(u(zx));
RTU = u(zx);
```

```
% Plots the return map data
figure();
hold on;
for i=1:(length(RTU)-1)
    plot(RTU(i), RTU(i+1), '*k');
end
xlabel('u(n)'); ylabel('u(n+1)');
hold off;

figure();
hold on;
t=t*10^6;
plot(t,u,t,s)
plot(t(zx),u(zx),'*')
xlim([0 2])
hold off;

figure();
plot(u,ud)
xlabel('u');ylabel('du');

time=t(zx);
for iii=1:length(zx)-1
   period(iii) = time(iii+1) - time(iii);
end

averageT = mean(period)
```

B.2 Simulink Model for Second Order System

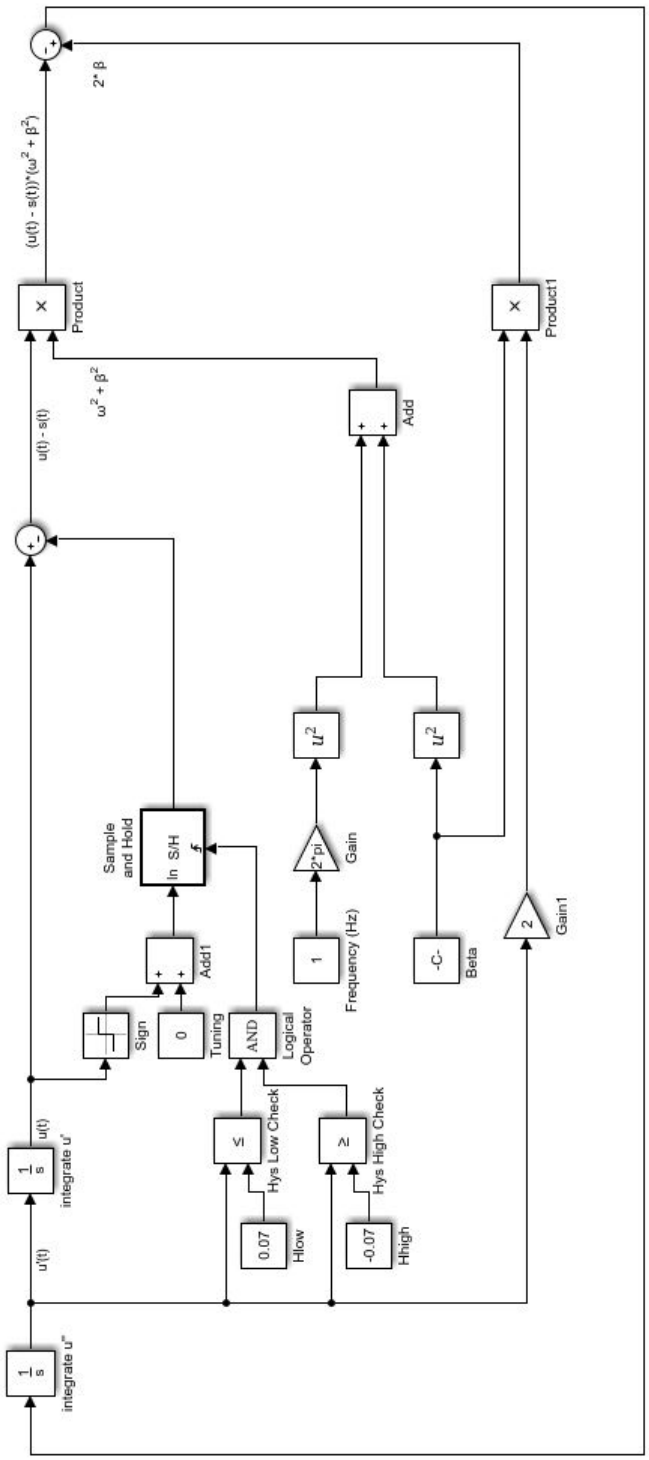

Figure B.1: MATLAB Simulink Model of Exactly Solvable System

Appendix C

Spice Schematics and Netlists

C.1 Lorenz Circuit

```
* C:\Users\cmf0052\Documents\LTspiceXVII\Thesis\Casey_Lorenz.asc
XU1 X 0 0 Z -15V 0 -XZ +15V AD633
XU2 -Y 0 X 0 -15V 0 -XY +15V AD633
XU3 0 N001 +15V -15V X LT1057
R1 N001 -Y 100k
R2 N001 X 100k
C1 X N001 0.1
V1 +15V 0 +15
V2 -15V 0 -15
XU6 0 N002 +15V -15V -Y LT1057
XU4 0 N003 +15V -15V Z LT1057
R3 N002 X 35.7k
R4 N002 -XZ 10k
R5 N002 -Y 1Meg
C2 -Y N002 0.1
C3 Z N003 0.1
R6 N003 -XY 10k
R7 N003 Z 374k
* AD633 Pinout\nPin 1  =  X1
*Pin  =  +Vs\nPin 2  =  X2
*Pin  =  W\nPin 3  =  Y1
*Pin  =  Z\nPin 4  =  Y2
*Pin  =  -Vs
```

```
* X
* -Y
* Z
.tran 0 10 0 0.001 startup
.lib C:\Users\cmf0052\Documents\LTspiceXVII\AD633.cir
.lib LTC.lib
.backanno
.end
```

C.2 CMOS Exactly Solvable Oscillator Schematic and Netlist

```
CL du 0 500p
CC 0 u 100p
R2 0 N011 11
XX1 du 0 u ota_045
XX2 du 0 N011 ota_045
XX3 N011 0 du ota_045
XX4 Vs u du ota_045
XX5 N007 N004 N008 vdd vss nand_p045
XX6 N004 N013 N012 vdd vss nand_p045
XX7 N008 N010 N009 vdd vss nand_p045
XX8 N009 N012 N010 vdd vss nand_p045
XX9 N007 N007 N013 vdd vss nand_p045
V1 +15mV 0 15m
V2 -15mV 0 -15m
XX10 +15mV du N001 highspeedbuffer
XX11 -15mV du N005 highspeedbuffer
XX12 N001 N005 N003 vdd vss nand_p045
XX13 N001 N003 N002 vdd vss nand_p045
XX14 N003 N005 N006 vdd vss nand_p045
XX15 N002 N006 N004 vdd vss nand_p045
V3 vdd 0 0.5
V4 0 vss 0.5
XX16 0 u N007 highspeedbuffer
R3 vdd Vs 2k
```

```
R4 Vs vss 2k
R5 N009 Vs 1k

* block symbol definitions
.subckt ota_045 Vin+ Vin- out
M3 N001 N001 Vdd Vdd PMOS l=0.18u w=155u
M4 N002 N002 Vdd Vdd PMOS l=0.18u w=155u
M1 N001 Vin- N003 Vss NMOS l=0.18u w=20u
M2 N002 Vin+ N003 Vss NMOS l=0.18u w=20u
I1 N003 Vss 3.3mA
V1 Vdd 0 1
V2 Vss 0 -1
M6 out N002 Vdd Vdd PMOS l=0.18u w=155u m=3
M5 N004 N001 Vdd Vdd PMOS l=0.18u w=155u
M7 N004 N004 Vss Vss NMOS l=0.18u w=20u
M8 out N004 Vss Vss NMOS l=0.18u w=20u m=3
.inc p045_cmos_models_tt.inc
.ends ota_045

.subckt nand_p045 a b out vdd vss
M3 out a vdd vdd PMOS l=0.045u w=2.25u
M4 out b vdd vdd PMOS l=0.045u w=2.25u
M5 out a N001 vss NMOS l=0.045u w=1.125u
M6 N001 b vss vss NMOS l=0.045u w=1.125u
.inc p045_cmos_models_tt.inc
.ends nand_p045

.subckt highspeedbuffer vinm vinp out
M1 N004 N003 vss vss NMOS l=0.045u w=0.45u
M2 N004 N003 vdd vdd PMOS l=0.045u w=0.9u
M3 N001 vinm N002 vdd PMOS l=0.045u w=0.9u
M4 N003 vinp N002 vdd PMOS l=0.045u w=0.9u
M5 N002 N001 vdd vdd PMOS l=0.045u w=0.9u
M6 N001 N001 vss vss NMOS l=0.045u w=0.45u
M7 N003 N001 vss vss NMOS l=0.045u w=0.45u
```

```
M8 N006 vinm N007 vss NMOS l=0.045u w=0.45u
M9 N003 vinp N007 vss NMOS l=0.045u w=0.45u
M10 N003 N006 vdd vdd PMOS l=0.045u w=0.9u
M11 N006 N006 vdd vdd PMOS l=0.045u w=0.9u
M12 N007 N006 vss vss NMOS l=0.045u w=0.45u
V1 vdd 0 0.5
V2 0 vss 0.5
M13 N005 N004 vss vss NMOS l=0.045u w=0.45u
M14 N005 N004 vdd vdd PMOS l=0.045u w=0.9u
M15 out N005 vss vss NMOS l=0.045u w=0.45u
M16 out N005 vdd vdd PMOS l=0.045u w=0.9u
.inc p045_cmos_models_tt.inc
.ends highspeedbuffer

.model NMOS NMOS
.model PMOS PMOS
.lib C:\Users\cmf0052\Documents\LTspiceXVII\lib\cmp\standard.mos
.tran 0 4u 0 0.1n
.ic V(u)=.029mV V(du)=0V
* Unstable -RLC Ladder Network
* gm=30m
* D Flip Flop
* Zero Crossing Detector
* Signum Function
* D
* CLK
* Q
* Vs
* u
* du
.backanno
.end
```

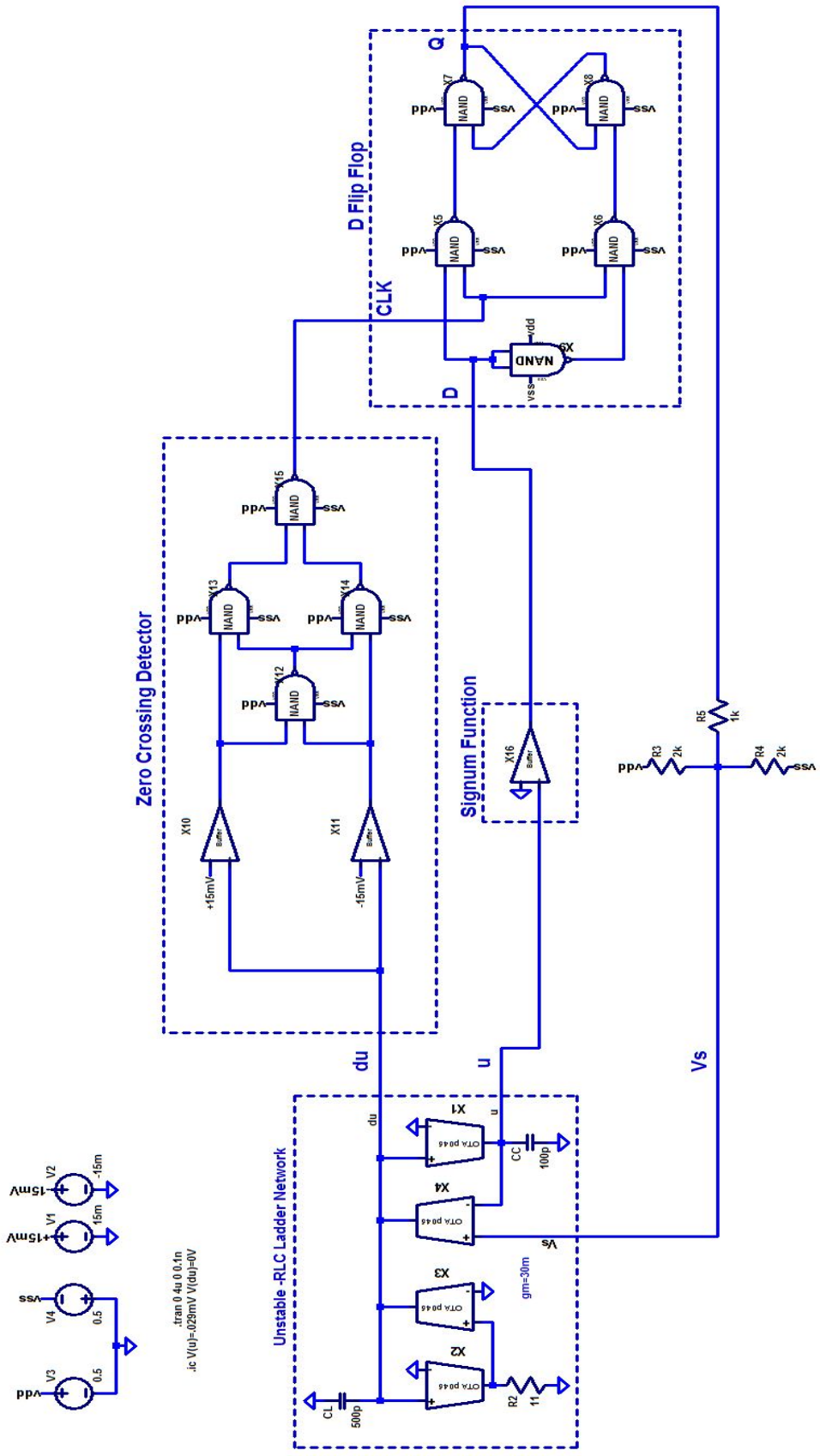

Figure C.1: CMOS Exactly Solvable Shift-Band Oscillator Schematic

Appendix D

Cadence Setup Guide

This tutorial will explain how to setup the Cadence Analog Design Environment (ADE) at Auburn University. Keep in mind that things within this tutorial may change as time goes on. However, at the time of this writing in order to access Cadence, one has to connect remotely to the Linux Lab. Currently, SecureCRT is the recommended SSH client software to use in order to accomplish this. Guides can be found on the university website for proper setup of the SSH client.

Once a connection has been established using the SSH client, a visual interface between Windows and Linux needs setup. Several options are available for doing this. Currently Network Services has a guide for using Xming, but any Virtual Network Computing (VNC) software can be used. Personally, I prefer using TigerVNC, but there are many other alternatives available online. Figure D.1 shows the visual interface betweeen windows and linux using TigerVNC.

With a visual interface setup between Windows and Linux, the next step is creating the environment for Cadence. It is recommended to create a new directory where you would like to save all the files you create with Cadence. This can be done using the graphical interface or through the terminal window. Before accessing Cadence, several files need to be added to this directory to set environment variables. The first file needed is a ".bashrc" (or .cshrc) file. This is a shell file that creates environment variables for simulators, PDKs, and license files. A ".cdsenv" and ".cdsinit" are also needed within the directory. These files configure the environment further by linking model files and providing error messages if things are not configured properly. Another required file within the directory is called "cds.lib." This file

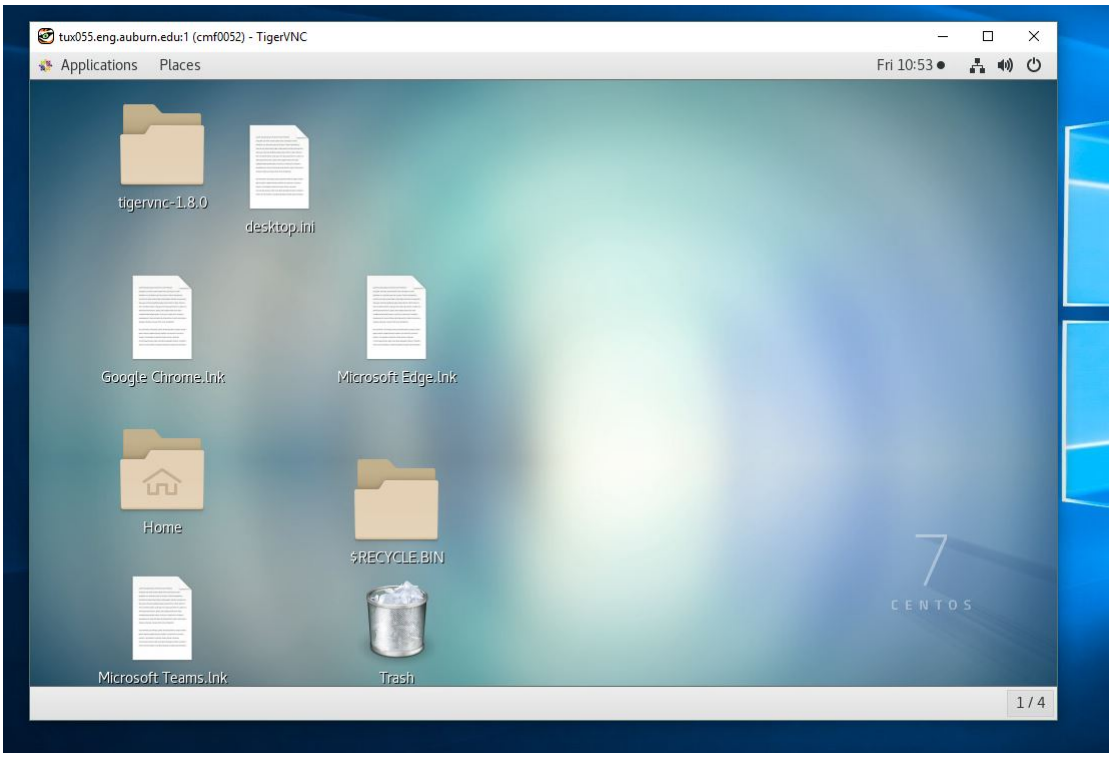

Figure D.1: Screenshot of VNC software

contains paths to part libraries used by Virtuoso. Once all files are configured with the correct paths and environment variables, typing the command ”virtuoso &” in the terminal window within the newly created directory will activate the software.

Typically, the process design kit will include samples of each of the required files listed along with instructions for proper configuration. Sample files can also be found within the Cadence IC install directory. Documentation on all the software tools associated with Cadence and tutorials can be found within the install directory.