UNIVERSITÉ DU QUÉBEC EN OUTAOUAIS

MODÉLISATION ET VÉRIFICATION DES CONCEPTIONS DE SIGNAUX MIXTES

MÉMOIRE PRÉSENTÉ

COMME EXIGENCE PARTIELLE

DU PROGRAMME DE MAÎTRISE EN SCIENCES ET TECHNOLOGIES DE L'INFORMATION

PAR

KOUHOUL ABDERAOUF

DECEMBRE 2018

Jury d’évaluation

Président du jury: Prof. Ilham Benyahia

Examinateur interne: Prof. Michael Korwin-Pawlowski

Directeur: Prof. Larbi Talbi

Co-directeur: Tahar Haddad, PhD

Résumé

Le progrès dans les technologies de Silicon et la mise à l'échelle de la taille des transistors ont permis de placer des millions de transistors dans de petites puces, de réduire la consommation d'énergie, d'augmenter les performances et par conséquent de réduire les coûts de production. D'un autre côté, l'intégration de circuits analogiques et numériques sur une même puce est devenue une exigence fondamentale dans les conceptions les plus récentes. Cependant, de nombreux problèmes ont été soulevés tels que l'interfaçage analogique avec le numérique, l'horloge et la vérification et la validation.

Cette recherche porte sur la modélisation et la vérification des conceptions de signaux mixtes. En fait, comme les dispositifs analogiques et numériques sont intégrés ensemble, la modélisation et la vérification des systèmes de signaux mixtes deviennent très critiques et une tâche difficile pour assurer l'exactitude des conceptions indépendamment des disparitions entre les simulateurs analogiques et numériques. Ainsi, trouver des approches, des méthodologies et des outils pour surmonter l'écart entre les simulateurs analogiques et numériques afin de fournir un cadre pour la vérification des signaux mixtes a été une nouvelle direction de recherche dans l'électronique moderne. La modélisation comportementale des conceptions analogiques est l'approche adoptée dans cette recherche car elle permet la simulation des conceptions analogiques dans des simulateurs numériques.

Dans ce mémoire, un cadre réutilisable et flexible pour développer un modèle comportemental est proposé. Comme preuve de concept, nous avons conçu un modèle physique d'un oscillateur commandé par tension analogique (VCO), puis développé un modèle comportemental suivant le cadre proposé pour montrer la faisabilité du cadre proposé. Après cela, nous fournissons des idées sur la façon d'intégrer les modèles comportementaux VCO dans un verrou de boucle de phase analogique (PLL). Enfin, sur la base de la réalisation physique (Layout) du VCO, nous fournissons une réflexion sur la façon d'augmenter le modèle comportemental pour inclure les variations de température, de puissance et de processus de fabrication.

Abstract

Progress in silicon technologies and scaling of transistors size has allowed putting millions of transistors in a small chip, reduce power consumption, increase performances and consequently reducing production cost. On the other hand, integrating analog and digital circuits on the same chip has become a fundamental requirement in most recent designs. However, many issues have been risen such as interfacing analog with digital, clock and verification and validation.

This research is about the modeling and verification of mixed-signal designs. As analog and digital devices are integrated, modeling and verification of mixed-signal systems become highly critical and a challenging task to ensure the correctness of designs regardless of the disappearances between analog and digital simulators. Thus, finding approaches, methodologies, and tools to overcome the gap between analog and digital simulators to provide a framework for mixed-signal verification has been new research direction in modern electronics. Behavioral modeling of analog designs is the approach adopted in this research as it allows the simulation of analog designs within digital simulators.

This dissertation proposes a reusable and flexible framework for developing behavioral models. As a proof concept, we have designed a physical model of an analog voltage-controlled oscillator (VCO), then, developed a behavioral model following the proposed framework to show the feasibility of the proposed framework. After that, we provide insights on how to integrate the VCO behavioral models in an analog phase loop locker (PLL). Finally, based on the layout of the VCO, we provide a short reflection on how to augment the behavioral model to include temperature, power and process variation.

Acknowledgments

Dedication

Contents

Table of figures

List of Tables

Acronyms

Application Specific Integrated Circuits	ASIC
Computed Aided Design	CAD
Backward Euler	BE
Numerical integration	NI
Programmable Gain Amplifiers	PGA
Kirchhoff Current Law	KCL
Kirchhoff Voltage Law	KVL
System on Chip	SoC
Analog/Digital Converter	ADC
Digital/Analog Converter	DAC
Delta Sigma Modulator	DSM
Phase Locked Loop	PLL
Loop Filter	LF
Error Detector	ED
Frequency Divider	FD
Voltage Controlled Oscillator	VCO
Phase Frequency Detector	PFD
Switched Capacitor	SC
Analog Mixed Signal	AMS
Hardware Description Language	HDL

Universal Validation Methodology	UVM
Open Validation Methodology	OVM
Phase Noise	PN
Finite Element Method	FEM
Finite Difference Method	FDM
Cadence Analog Simulator	Spectre
Synopsys Digital Simulator	VCS
Mentor Graphics Digital Simulator	QuestaSim
Complementary Metal Oxide Semi-Conductor	CMOS
Fin Field Effect Transistor	FinFet

Chapter 1

Introduction, Context and Problem statement

Advancement in silicon manufacturing technologies has resulted in transistors' size downscaling, integrating millions of transistors in small chips, reducing power consumption, improve performances and consequently reducing production cost. On the other hand, the need for integrating analog and digital circuits on the same chip has become a fundamental requirement in most recent designs. An example of devices in which digital components are combined with analog components is smartphones. For instance, smartphones have DSP engine which is a fully digital device, interacting with an RF receiver (oscillator, mixers, filtersetc.) which is analog devices.

Not surprisingly, integrating analog and digital devices has risen many issues such as interfacing analog with digital, clocking, signal integrity, and verification.

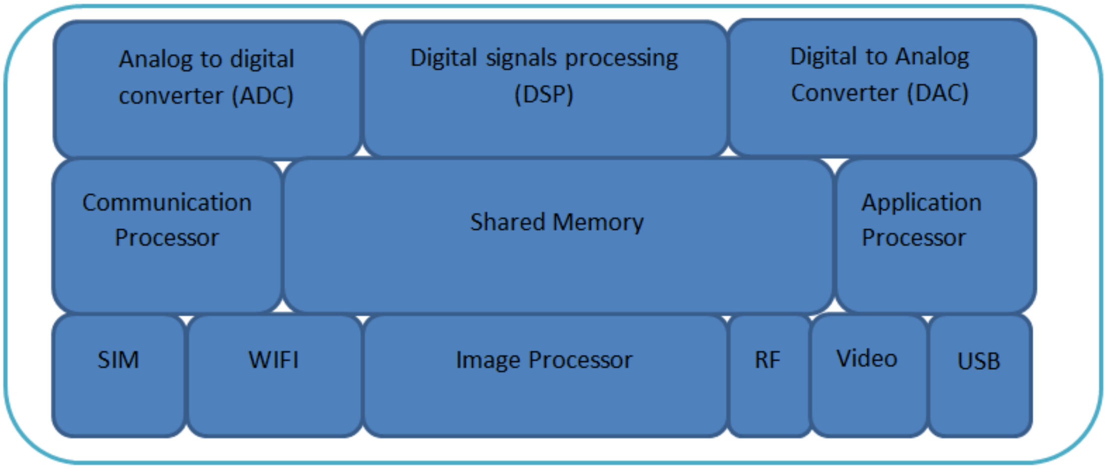

Analog to digital converter (ADC)
Digital signals processing (DSP)
Digital to Analog Converter (DAC)
Communication Processor
Shared Memory
Application Processor
SIM
WIFI
Image Processor
RF
Video
USB

Figure 1. Analog and digitals modules in modern electronics.

1.1 Context

The orientation of this research is the ***modeling and verification of mixed-signal designs***. Considering verification while modeling analog circuits becomes critical tasks as planning verification ahead reduces the complexity of behavioral models used in verification. In this research, throughout a design process, we identified and highlighted the metrics of a behavioral model at functional abstraction level. For clarity purpose, the term verification might have many interpretations, however, along with this dissertation, we define verification as the ***process of demonstrating the functional correctness of an electronic design.*** Also, the terminologies model and design are used interchangeably. Finally, Analog and digital simulators are the primary tools of verification of electronics.

The role of verification in modern designs is ensuring the delivery of devices that meet design specifications and users' requirements. Mixed-signal validation has been driven by the evolution of consumer products such as cell phones, entertainment devices, wearable gadgets, and so on. The time spent on verification and according cost is increasing at a high rate; driven particularly by the continuous increase of modern designs complexity, IP's and embedded processors. Consequently, the number of functionalities and state space have exploded, so current tools and methods of verification become inefficient, time and resource consuming. Also, the growing need of data transmission speed for applications such as streaming services, voice IP, multimedia applications, virtual reality, real-time applications, pushed to the development of high-speed designs that additionally add issues related to power and signal integrity and clock distribution

Mixed-signal verification aims to avoid function logic error while integrating digital and analog designs. From our industry experience, functional logic errors present the most frequent failures in most modern designs and become more frequent in the case of a mixed-signal system as the complexity increases considerably. Therefore, mixed-signal validation is a critical task to ensure production of SoC that meets the requirement of freeness of functional errors.

1.2 Problem Statement

Mixed-Signal Verification challenges are direct consequences of the discrepancies between digital and analog simulators in many aspects such as clocking event, input sampling, and output computing. Analog designers process continuous input, and output signals, and simulators provide metrics such as phase noise and oscillation frequency for a VCO for instance. From analog designers, these metrics are enough to validate the design. On the other hand, digital designers process a set of two states (0 and 1) inputs and outputs. Validation of digital designs is mainly realized by comparing design outputs to the output of a logical function describing the logic of the digital designs.

Three approaches have been proposed to mixed signal validation. The first approach is providing extensions to digital simulators to support analog modules. The second approach is providing extensions to analog simulators to support validation of digital designs. Another approach is behavioral modeling: modeling the behavior of analog designs to reflect the functional logic that is usable within digital simulators.

The importance of mixed-signal validation drives this research whereas analog and digital designs are put side to side in all most recent high-speed designs. Mixed-Signal validation becomes highly critical to ensure the functional correctness of any design and on the other hand, becomes a challenging task due to the discrepancies between analog and digital simulators. The focus, in this research, is how to overcome the gap between analog and digital simulators to provide a framework for mixed-signal validation. The framework should be reusable and adaptable.

1.3 Contribution

Methodology for developing behavioral models: In chapter 3, we provide arguments why behavioral models are of great value and in many cases, are the unique solution to system verification of modern designs. Writing behavioral models is a tedious task as developers who are not designers handle it. Moreover, industry recommendations require that developers should not be involved in the design process to ensure the development of behavioral models at a high level of functional abstraction. To overcome this situation, we proposed a generic and reusable HDL code for the development of behavioral models of analog circuits to achieve system verification of mixed signal with digital simulators.

Physical Design of a VCO: To identify the main links between modeling and verification, an analog VCO has been designed following a design approach using Cadence Spectre analog simulator. Starting from provided requirements and following a design approach, we run simulations to validate the design. During each simulation, we highlight the critical elements of verification and provide an analytical model to be used as bed test. These performances are used to validate the behavioral model.

Design a behavioral model of the VCO: The analog model, and throughout simulations, we have extracted the key elements and construct an analytical model of the behavioral model. Using the generic HDL code, a System Verilog code of a VCO behavioral model with accordance to the methodology is proposed. Validation of the behavioral model towards the analog design is realized by comparing the performances of the analog design obtained by the analog simulator Cadence Virtuoso versus the performances simulated on digital simulators.

Design a behavioral model of a PLL: One of the usefulness of behavioral modeling is allowing simulation of large analog circuits such as an analog PLL. VCO is at the heart of each PLL and using behavioral model requires identifying a chain of interactions between VCO and other components such as a phase detector and a charge pump. To this end, we derived a PLL model that integrates the VCO behavioral model.

Study the effect of Layout on the behavioral model of VCO and PLL: Simulating aging, temperature and process variation effects on analog designs remains a main topic of research. In high-speed designs, these factors become critical and no longer negligible or avoidable during layout. In this direction, we provide a reflection on how to include layout considerations at behavioral abstraction. To this end, the VCO behavioral model is augmented by elements of frequency variations due to a tuning voltage and phase noise.

This dissertation is organized as follows:

After introducing the research context and presenting the problem statement in the first chapter, we provide details on digital and analog simulators with the focus on the discrepancies. Then, mixed-signal verification: definitions, tools, and challenges are presented in chapter 3. The proposed framework for mixed-signal validation is given in chapter 4, followed by a design of an analog VCO in chapter 5. The behavioral model of the analog VCO is presented in chapter 6 with more details on the mathematical model and mapping links to the analog model. Also, chapter 6 introduces the design of a behavioral model of a PLL and short reflection on the effect of Layout and the performances and to reflect the results in the behavioral model.

Chapter 2

Overview of Digital and Analog Simulation

Analog and digital simulators are the primary tools of verification of physical designs. Chapter 2 constitutes a short introduction to the simulation filed of electronic circuits. It provides definitions and a brief presentation of the fundamental of digital and analog simulators.

2.1 Overview of Simulation of electronic circuits

Simulation refers to the modeling of a circuit, its function and estimate its performances. Simulation techniques are used to verify or validate:

- Outputs and metrics
- Assumptions and assertions
- Logic and functions
- Performances

2.1.1 Parts of a Simulator

A simulator, analog or digital, is a software composed of three main components. The ***Kernel*** is the core of the simulator that performs the evaluation and Computes signals at every point of interest. In the case of analog simulators, the kernel is mainly a numerical solver of differential and algebraic equation equations using numerous methods (Newton Raphson iterative methods, LU matrix decomposition). The core of a digital simulator is logic rule evaluator and lookup tables. ***Circuit/Input description*** module responsible for capturing the schematic, spice netlist, HDL and other formats of designs and translate into internal formats to be simulated by the kernel. The ***Stimuli generator*** provides Stimulus to the circuit- modeled in a netlist. Simulation results are presented by an ***Output Interface module*** in charge of plotting simulation outputs in form of graphs, animations, tables, etc.

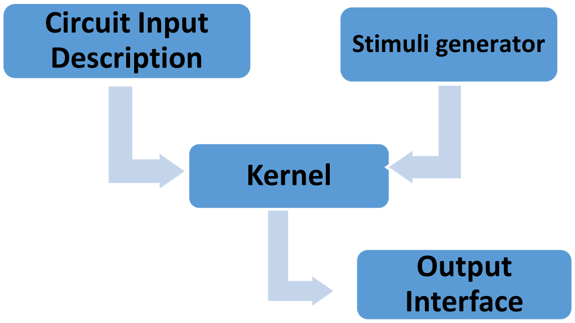


Figure 2. Structure of an analog simulators.

2.1.2 Signal States

Depending on type of simulators and level of a circuit, an electric signal on net or ports represents a state variable with possible values:

- Two-states (0, 1) are used for combinational logic with zero-delay.
- Three-states (0, 1, X) are required for timing hazards and sequential logic initialization. (X uninitialized)
- Four-states (0, 1, X, Z) to represent the states of MOS devices.
-

2.1.3 Simulation Levels

The physical or abstracted type of components and required accuracy and precision, define simulation levels. A designer usually runs one or more simulations at different levels.

2.1.3.1 Device level simulation

- Single semiconductor device is simulated (e.g. Diode, transistor).
- Charge distribution is computed and in time and space.
- The Finite Element Method (FEM) and Finite Difference Method (FDM) are used.

- Tools: Synopsys Sentaurus, GTS Nano Device Simulator.

2.1.3.2 Circuit level simulation

- Simulate groups of transistors, diodes, resistors, etc.
- E.g., Spice, Signal integrity tools

2.1.3.3 Timing level and macro-level simulation

- Signals are analog
- Models are simplified to accommodate large netlists
- Piecewise linear elements used insated of nonlinear elements.
- Group of devices represented by a macro

2.1.3.3.4 Switch-level simulation

- Transistors are considered as switches
- Signals almost digital (one, zero)-but they have signal strength to model parasitics

2.1.3.3.5 Gate level or logic level simulation

- Transistors are modeled by Gates for the simulation.
- Signal flow is unidirectional

2.1.3.3.6 Register transfer level simulation

- Synchronous circuits-clock controls registers being assigned
- Registers store the state of the system
- Combinational logic gates compute next state of registers
- State transitions are of interest; other effects secondary

2.1.3.3.7 System level simulation

- Block of hardware specified through Verilog/VHDL
- Simulation of such blocks is the system-level simulation

2.1.3.3.8 Mixed-mode simulation

- Circuit-blocks described in different abstraction levels
- Simulators combine all levels of abstractions
- Hardware-software co-simulation
- Certain parts hardware -Certain parts software

The following table summarizes different simulation levels and tools available.

Table 1. Simulation Levels.

Simulation Level	Signal states	Application	Available tools
System	Digital/analog		SystemC
Behavioral	Digital/analog		SystemVerilog, Veriolg, VerilogA, SystemC,VerilogAMS
RTL	0, 1		Questa, VCS
Gate	0, 1, X, Z	Logic verification	Questa, VCS
Switch	0, 1, X	Logic verification	Bryant's Model
Macro-model	Analog		Cadence Verilog AMS
Circuit	Continuous signal	Analog circuit verification	Cadence Spectre
Device	Continuous signal		Synopsys Sentaurus

2.2 Digital Simulators

Digital simulators are used at system RTL, gate and switch levels. Usually, they are faster than analog simulators. In digital simulators, input and output signals have discrete states. In the following section, we describe digital simulators at the Gate level.
In this section, we limit to the gate level, but the same description is valid for other levels.

2.2.1 Gate-Level Simulation

At this simulation level, a digital signal has 02 states (0, 1), but it may have additional states such as X(Uninitialized) and U(Unknown) or a strength such as Z (high impedance) and (H) Strong High, etc.

A combinational gate may be modeled by a truth table or subroutine program or script implementing the Boolean expression of inputs to outputs. Gates delay (propagation and rise/fall delay) is modeled by a delay in computing evaluation the gate Boolean expressions. Particular delay models are 0 delay and 01-unit delay models

2.2.5 Simulation mechanisms

Two major classes of simulators' kernel that define algorithms and techniques used in the evaluation of Boolean expressions and thus computing the response to stimuli at the inputs.

2.2.5.1 Compiler Driven Simulators

By analogy to code compiling phase in any programming language, this class of digital simulators evaluate all logical expressions of all gates forming the circuit in one pass at all simulation ticks. It applies to zero-delay combinational logic. For unit delay models, it needs net values for previous points, so it used two arrays of values; simulation instant t and t-1. It can also be extended for delay models higher that unit-delay model.

Let's consider the netlist depicted in the following figure 3 [1].

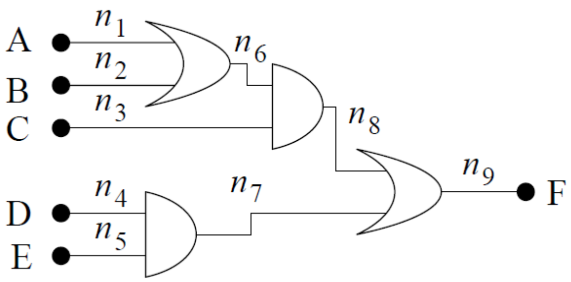


Figure 3. Combinational circuit.

The algorithm starts by assigning levels to every net starting by inputs with level 0. Then, level 1 is assigned to nets that are one gate away from inputs and same steps repeated for all nets at Level n that are n gates away from inputs until reaching outputs.

```
$n_1 \leftarrow$ A;
$n_2 \leftarrow$ B;
$n_3 \leftarrow$ C;
$n_4 \leftarrow$ D;
$n_5 \leftarrow$ E;
$n_6 \leftarrow$ OR($n_1$, $n_2$);
$n_7 \leftarrow$ AND($n_4$, $n_5$);
$n_8 \leftarrow$ AND($n_6$, $n_3$);
$n_9 \leftarrow$ OR($n_7$, $n_8$);
F $\leftarrow n_9$;
```

Logic expression evaluation is achieved by the evaluation of the nets at level 0, then other levels in ascending way. The outputs of the previous netlist in response to stimuli are depicted in figure 4.

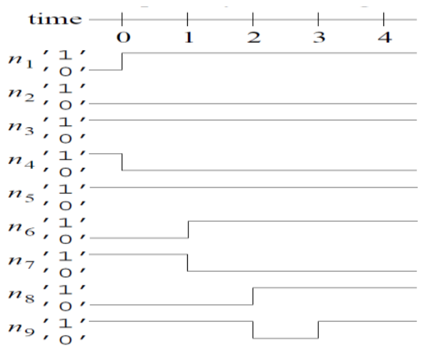


Figure 4. Waveforms of the circuit nodes.

2.2.5.2 Event Driven Simulator

Contrary to compiler-driven simulators, in event-driven mechanisms, logic expressions are evaluated only for gates or modules with input events. An event is an abstraction of a signal state change of a net. Events are stored in a queue, figure 5, and have three attributes:

- Time when an event should occur
- Nets that will change
- New value assumed by net

Change in an input is an event which triggers a change in only gates connected to it. If the outputs of these gates change, they, in turn, create another event which triggers other gates, figure 6. Such an approach might be faster than a compiler-driven simulation and can incorporate various delay models.

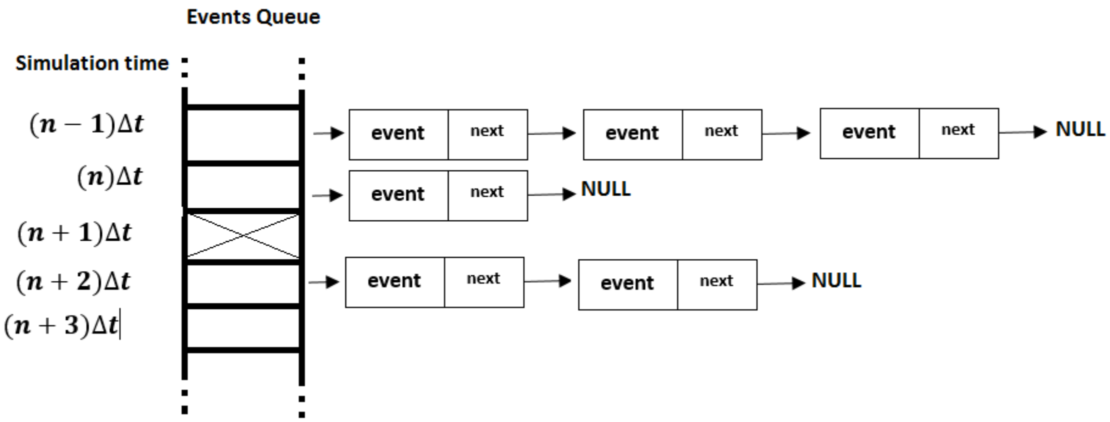


Figure 5. Digital Simulators: Event queue.

```
Event_driven_simulation ()
{
Struct event_queue * Q;
Q ← new queue();
"insert stimuli in Q";
"initialize to U: all  nodes connected to a memory  and all other nodes to 'X'";
for(t←$t_{start}$;t<$t_{end}$;)
{
Current_event←first_event(Q);
t←current_event->time;
"process the current_event and add new events to Q at time t + appropriate delay";
}
}
```

Figure 6. Event Driven Algorithm.

2.2.6 Digital Verification

Logic simulators are essential tools for design verification. Verification testing vectors and expected responses are generated (often manually) from specifications. A logic simulator is implemented using either a compiled-code or event-driven method. Per vector complexity of a logic, a simulator is approximately linear in circuit size. Modeling level guides the procedures to be used by the simulator.

2.3 Analog Simulator

Analog simulators are used to estimate analog circuits` performances at device, transistor and circuit levels. Spice-like simulators are widely used by analog designers to evaluate the performances of designs according to prior-defined metrics. For example, the metrics for an LNA are: open loop gain, closed loop gain, input and output impedances, noise figure, bandwidth, center frequency, and linearity and so on. Initially, these are given as requirements and analog designs are validated with respect to these requirements. To this end, designers use analog simulators and throughout different analysis such as DC, transient, PSS, and PNOISE, etc. … to estimate analog design performances and compare with requirements. Analog designs are pronounced valid if the simulated performances match the requirements with a certain acceptable margin of errors.

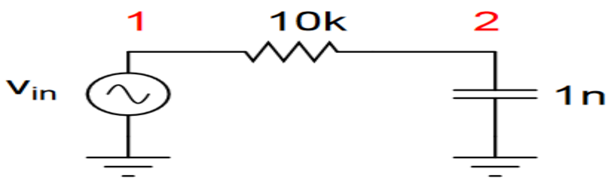
The continuous nature of analog circuits inputs and outputs leads to circuit testing based on a small space of test inputs and outputs. As the output of analog design is often a continuous function and an output value corresponds to a set of inputs. For example, usually, an output at instant t provides information on previous inputs. Knowing the cut frequency of a passband filter is enough to estimate the frequency response to all frequencies lower than the cut off frequency, this means that the cut off frequency constitutes a corner test that reduces the test bins.

Spreading over the last 30 years, spice simulator with integrated circuit emphasis has been used to simulate the operation of an analog circuit without having to build the corresponding hardware.

In this section, the Modified Nodal Analysis methods are presented and an overview of different numerical methods used in resolving the differential equations describing the electrical behavior of a circuit.

Analog simulators have similar bloc architecture as digital simulators. The significant difference is the Kernel wherein analog simulators it is numerical solvers of differential equations.

The Netlist of an analog circuit is a list of components with characteristic values and connecting nets. For the following circuit, the netlist is provided in figure 7.



Netlist

```
RC circuit
r1 1 2 10k
c1 2 0 1n
vin 1 0 ac 1 dc 0
```

Figure 7. Example of an analog circuit and spice netlist.

2.3.1 Modified Nodal Analysis

Applying KCL on the circuit depicted in figure 8, to each node (n_1, n_2) results on the following algebraic equations:

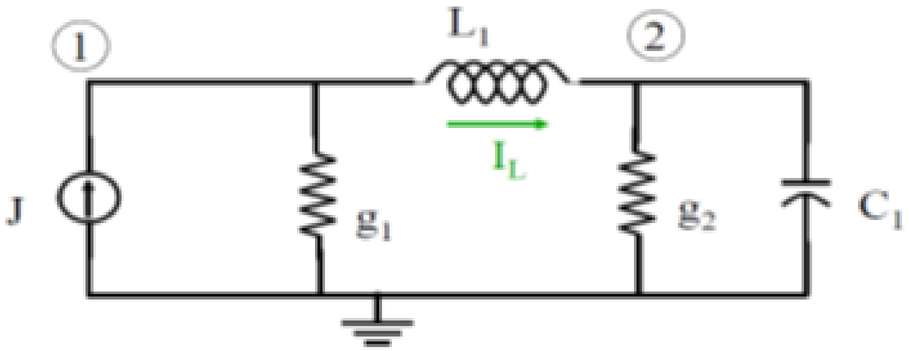


Figure 8. Example of an Analog circuit.

The only unknown variables in the above equations are signal levels at each node V1, V2. Before resolving these equations, they are transformed into matrix representations. Thus, by putting current on the right hand, and voltages as variables, the above equations are re-written in the following matrix representation.

For the circuit above, figure 8, we apply ohms law to each component, table2.

Table 2. Ohm's law for each circuit component.

Circuit componet	Time domain	Frequency domain
resistors	$i(t) = gv(t)$	$i = gv$
capacitors	$i(t) = C\dfrac{dv(t)}{dt}$	$i = sCV$
inductors	$V(t) = L\dfrac{di(t)}{dt}$	$V = sLI$

In time domain, we obtain the differential equations:

$$\begin{cases} g_1 v_1(t) + I_L(t) = J(t) \\ g_2 v_2(t) - I_L(t) = 0 \\ v_1(t) - v_2(t) = 0 \end{cases}$$

Re-writen into matrix form.

$$\left(\begin{bmatrix} g_1 & 0 & 1 \\ 0 & g_2 & -1 \\ 1 & -1 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & C_1 & 0 \\ 0 & 0 & -L_1 \end{bmatrix}\right)\begin{bmatrix} v_1(t) \\ v_2(t) \\ I_L(t) \end{bmatrix} = \begin{bmatrix} J(t) \\ 0 \\ 0 \end{bmatrix}$$

$$\boldsymbol{Gx(t)} + \boldsymbol{C\dot{x}(t)} = \boldsymbol{b(t)} \qquad \textbf{(2.1)}$$

In the frequency domain, we obtain the following matrix.

$$\left(\begin{bmatrix} g_1 & 0 & 1 \\ 0 & g_2 & -1 \\ 1 & -1 & 0 \end{bmatrix} + s\begin{bmatrix} 0 & 0 & 0 \\ 0 & C_1 & 0 \\ 0 & 0 & -L_1 \end{bmatrix}\right)\begin{bmatrix} v_1 \\ v_2 \\ I_L \end{bmatrix} = \begin{bmatrix} J \\ 0 \\ 0 \end{bmatrix}$$

$$(\boldsymbol{G} + \boldsymbol{sC})\boldsymbol{X} = \boldsymbol{b} \qquad \textbf{(2.2)}$$

2.3.2 General equation on the nonlinear MNA

Adding nonlinear elements such as a diode into the circuit analysis, the equation of general networks of figure 9.

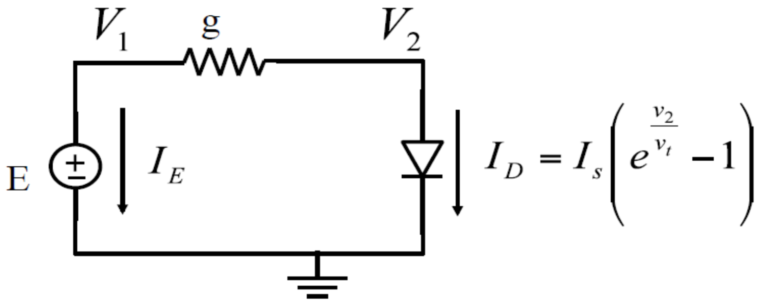


Figure 9. Example of a nonlinear circuit.

We obtain, the set of nonlinear equations:

$$\begin{cases} (v_1 - v_2)g + I_E = 0 \\ (v_2 - v_1)g + I_s\left(e^{\frac{v_2}{v_{th}}} - 1\right) = 0 \\ v_1 = E \end{cases}$$

Re-writing into matrix form:

$$\left(\begin{bmatrix} g & -g & 1 \\ -g & g & 0 \\ 1 & 0 & 0 \end{bmatrix}\begin{bmatrix} v_1 \\ v_2 \\ I_E \end{bmatrix} + \left[I_s\begin{pmatrix} 0 \\ e^{\frac{v_2}{v_{th}}} - 1 \\ 0 \end{pmatrix}\right]\right) = \begin{bmatrix} 0 \\ 0 \\ E \end{bmatrix}$$

Therefore, ***the general equation of nonlinear modified nodal analysis is:***

$$\boldsymbol{Gx(t)} + \boldsymbol{C\dot{x}(t)} + \boldsymbol{f\big(x(t)\big)} = \boldsymbol{b(t)} \tag{2.3}$$

G: matrix that represents memoryless elements.

C: matrix for energy storage elements.

$\boldsymbol{f\big(x(t)\big)}$**:** nonlinear vector representing nonlinear elements.

b(t): vector representing independent sources.

Once the matrix representation of the netlist is written, simulator kernel solves the matrix using different numerical methods. The numerical methods to apply depends on the type of analysis: DC, Transient, AC, time or frequency analysis, S-parameters, phase noise, time jitter noise and so on.

Three major class of analysis which are: DC, frequency and transient analysis.

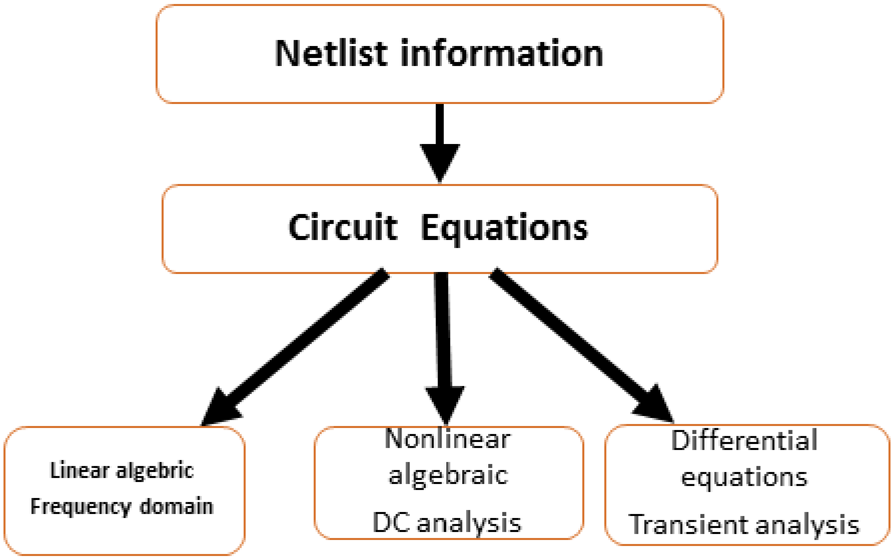


Figure 10. Three primary analysis methods [2].

Starting with a netlist, the nodal equations are found using the MNA stamping technique.

- The frequency domain solution is found by solving the resulting system of *linear* equations at each frequency point. The solution at each point is typically obtained using the LU+F/B

- DC solution involves NR iterations, with each iteration requiring LU+F/B. Convergence is a significant issue here.

- The transient analysis involves converting the differential equations at each time point to a difference equation using appropriate integration formula. The resulting nonlinear algebraic equations require solution based on NR iterations.

2.3.3 Transient Analysis

The solution of the DE is transformed into a solution of a system of linear algebraic equations at each time point. First, the nonlinear differential equations are transformed into nonlinear algebraic equations using an integration method such as Newton Raphson(NR) or Trapezoid Rule (TR).

For example, applying the Backward Euler method on the general equation

$$\boldsymbol{Gx(t) + C\dot{x}(t) + f\big(x(t)\big) = b(t)} \qquad \textbf{(2.3)}$$

Backward Euler is defined by the following 02 equations:

$$X_{n+1} \cong X_n + h\dot{X}_{n+1}$$

$$\dot{X}_{n+1} = \frac{1}{h}\,(X_{n+1} - X_n)$$

At t_{n+1} , equation (2.3), is written as:

At t_{n+1} $$\begin{cases} GX(t_{n+1}) + C\dot{X}(t_{n+1}) + f\big(X(t_{n+1})\big) = b(t_{n+1}) \\ \quad GX_{n+1} + C\dot{X}_{n+1} + f(X_{n+1}) = b_{n+1} \end{cases}$$

Applying Backward Euler equation $X_{n+1} \cong X_n + h\dot{X}_{n+1}$, we obtain:

$$GX_{n+1} + \frac{C}{h}(X_{n+1} - X_n) + f(X_{n+1}) = b_{n+1}$$

This is the Backward Euler Difference equation:

$$\boldsymbol{(G + \tfrac{C}{h})X_{n+1} + f(X_{n+1}) = b_{n+1} + \tfrac{C}{h}X_n} \qquad \textbf{(2.4)}$$

The nonlinear differential equation is transformed into the following nonlinear algebraic equations: $\boldsymbol{AX_{n+1} + f(X_{n+1}) = b}$ **(2.5)**

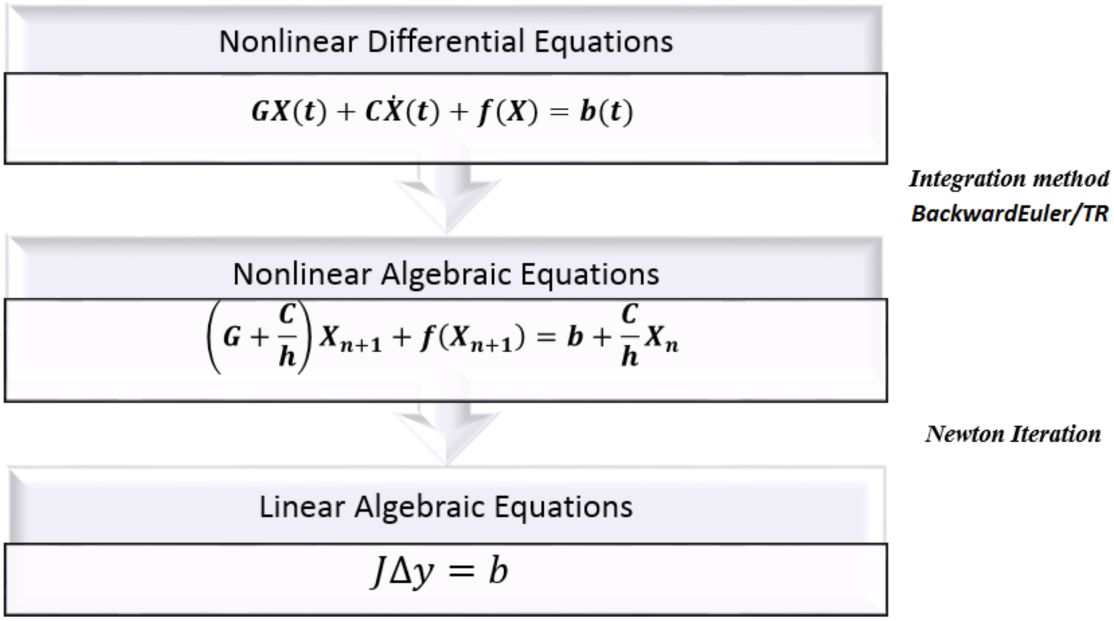

Nonlinear Differential Equations
$GX(t) + C\dot{X}(t) + f(X) = b(t)$
Integration method
BackwardEuler/TR
Nonlinear Algebraic Equations
$\left(G + \frac{C}{h}\right)X_{n+1} + f(X_{n+1}) = b + \frac{C}{h}X_n$
Newton Iteration
Linear Algebraic Equations
$J\Delta y = b$

Figure 11. Numerical methods for transient analysis.

2.3.4 DC analysis

The time domain equation of a linear circuit is given by the equation $\boldsymbol{Gx}(\boldsymbol{t}) + \boldsymbol{C\dot{x}}(\boldsymbol{t}) = \boldsymbol{b}(\boldsymbol{t})$

In DC analysis, we have constant input: $\dot{\boldsymbol{x}}(\boldsymbol{t}) = \boldsymbol{0},$ so the equation after simplification become a system of linear algebraic equation: $\boldsymbol{Gx}(\boldsymbol{t}) = \boldsymbol{b}(\boldsymbol{t})$

Two classes to solve a system of linear equations

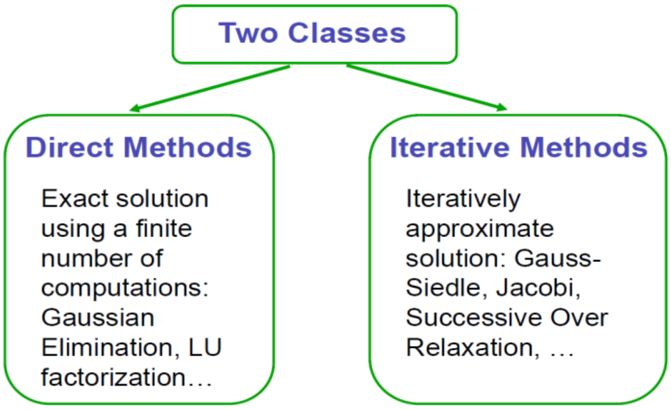

Two Classes
Direct Methods
Exact solution using a finite number of computations: Gaussian Elimination, LU factorization…
Iterative Methods
Iteratively approximate solution: Gauss-Siedle, Jacobi, Successive Over Relaxation, …

Figure 12. Numerical methods of solving linear algebraic equations.

2.3.5 Frequency Analysis

In general, we need to solve the system of linear equations: $\boldsymbol{A}(\boldsymbol{s})\boldsymbol{X}(\boldsymbol{s}) = \boldsymbol{b}(\boldsymbol{s})$ and $\boldsymbol{A} = \boldsymbol{G} + \boldsymbol{sC}$. We obtain a different matrix for each value of ω; therefore we need to solve the system of equations at each frequency using linear algebra techniques using the methods depicted in figure 12.

2.4 Conclusion

The choice of analog or digital simulators depends on the level of abstraction. Working at transistor or circuit levels requires analog simulators as input and output signals are continuous and performances metrics are computed based on signal interpolation. At logic levels such as gate and RTL, digital simulators are best suitable as signals are presented by discrete states, and the number of inputs is large. Even though theoretically, it is possible to use analog simulators at any levels, but in practice, it is not feasible as analog simulations are heavily CPU and memory usage because simulations rely on numerical solvers of nonlinear differential equations. Figure 13 and table 3 summarize differences between both types of simulators.

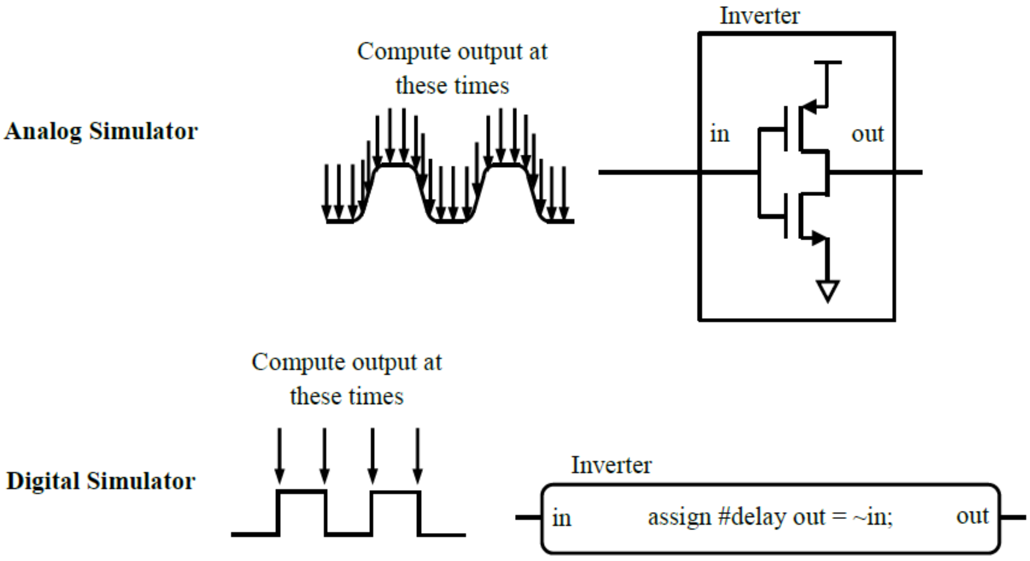


Figure 13. Simulation time in Analog and Digital Simulators.

Table 3. Analog Simulators versus Digital Simulators.

Analog Simulators	**Digital Simulators**
Accurate metrics	Functional, logic coverage
Heavy Simulations	Fast Simulations
Continuous inputs and outputs	Input and output discrete states
Bidirectional	Unidirectional

Chapter 3

Mixed Signal Verification; Overview and Challenges

Verification plays an essential role in modern SoC design where analog and digital modules are integrated into SoC. In the flow of modern designs, depicted in figure 14, verification occurs before fabrication to ensure the correctness of the design and to avoid fabrication cost due to design failures. In the ASIC process, verification is usually qualified as pre-silicon verification as it happens before sending masks to silicon foundries.

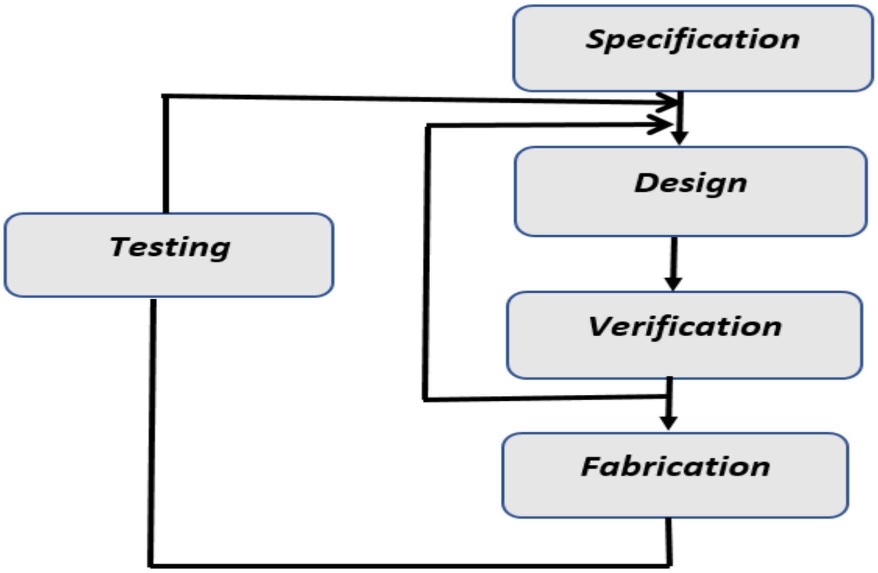


Figure 14. Modern design flow.

3.1 Verification methods

Many verification methods and methodologies have been presented in the literature [3,4,5,6] and adopted in the industry. These methods and methodologies are classified in one of the following categories.

3.1.1 Prototyping

Prototyping consists on building an abstracted version of the system on a breadboard to test –bread-boarding. Programmable devices such as FPGAs are also used to verify ASICs circuits. This approach is called rapid system prototyping; however. This class of methods is not feasible for large designs.

3.1. 2 Simulation

Simulation is defined as coding the system behavior by computer programs and run a system-level simulation to observe signal outputs to verify the functional correctness with respect to its specifications. Even though simulation is widely adopted by industry and many simulation platforms are available such as Cadence, Synopsys and Mentor Graphics.

3.1.3 Formal Verification

Instead of designing a software program or hardware prototype, the goal of formal verification is finding an algorithm or mathematical models reflecting the function of on-chip implementation or re-construct the gate-level netlist. This method does not require test vectors generation; however, modern systems are more complicated to be modeled by a formal method [4,7,8].

3.2. Analog Verification

SPICE-like Analog simulators are used to estimate analog circuits ‘performances at the transistor level. These tools such as Spectre, Pspice, fastSpice, and HSPICE are widely used by analog designers to evaluate the performances of designs according to prior-defined metrics. For example, the metrics of a Low Noise Amplifier (LNA) are: open loop gain, closed loop gain, input and output impedances, noise figure, bandwidth, center frequency, and linearity. Initially, these are given as requirements and verified throughout different simulation analysis methods such as DC, transient, PSS, PNOISE, etc. Finally, analog designs are pronounced valid if the simulated performances match the requirement with a certain acceptable margin of errors.

Despite the large variety of characteristics that can be measured of analog responses, analog outputs are smooth functions of the circuit's input. Therefore, designers are usually not concerned with enumerating every possible test stimulus such that the input space is finely sampled. Typically, a small set of test inputs is sufficient to fully characterize the analog response surface, because the result from one input case yields a great deal of information on results from similar inputs. Using this small set of test cases, analog verification can be completed for each performance metric even if each analog test case (especially those that require transient simulations) might be time-consuming. For example, knowing the cut-off frequency of passband filter is enough to estimate the frequency response to all frequencies lower than the cut off frequency, this means that the cut off frequency is a corner test that reduces the test bins.

3.3 Digital Verification

Digital blocks are generally coded using a hardware description language (HDL) like VHDL or Verilog, and simulated in a digital simulator such as VCS or ModelSim. These blocks are verified according to their logical expressions connecting inputs to outputs. The main factor of complexity is the number of digital inputs. Let consider a system of N inputs, and exhaustive testing requires test vectors generation of 2^N combinations which can be huge in case of millions of gates.

On the contrary of analog designs, a correct state of an output does not provide information on other inputs. Also, digital blocks are verified at the transistor level. The purpose of the digital simulator, hence, is to evaluate the logical function.

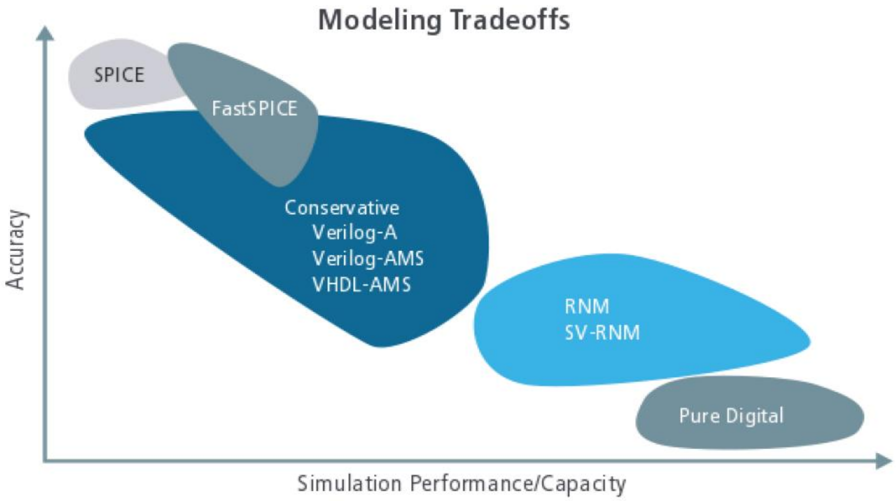


Figure 15. Simulation Performance of Analog and Digital Simulators [9].

3.4 Approaches for mixed-signal verification

Approaches to mixed-signal validation will be compared are classified into three major categories:

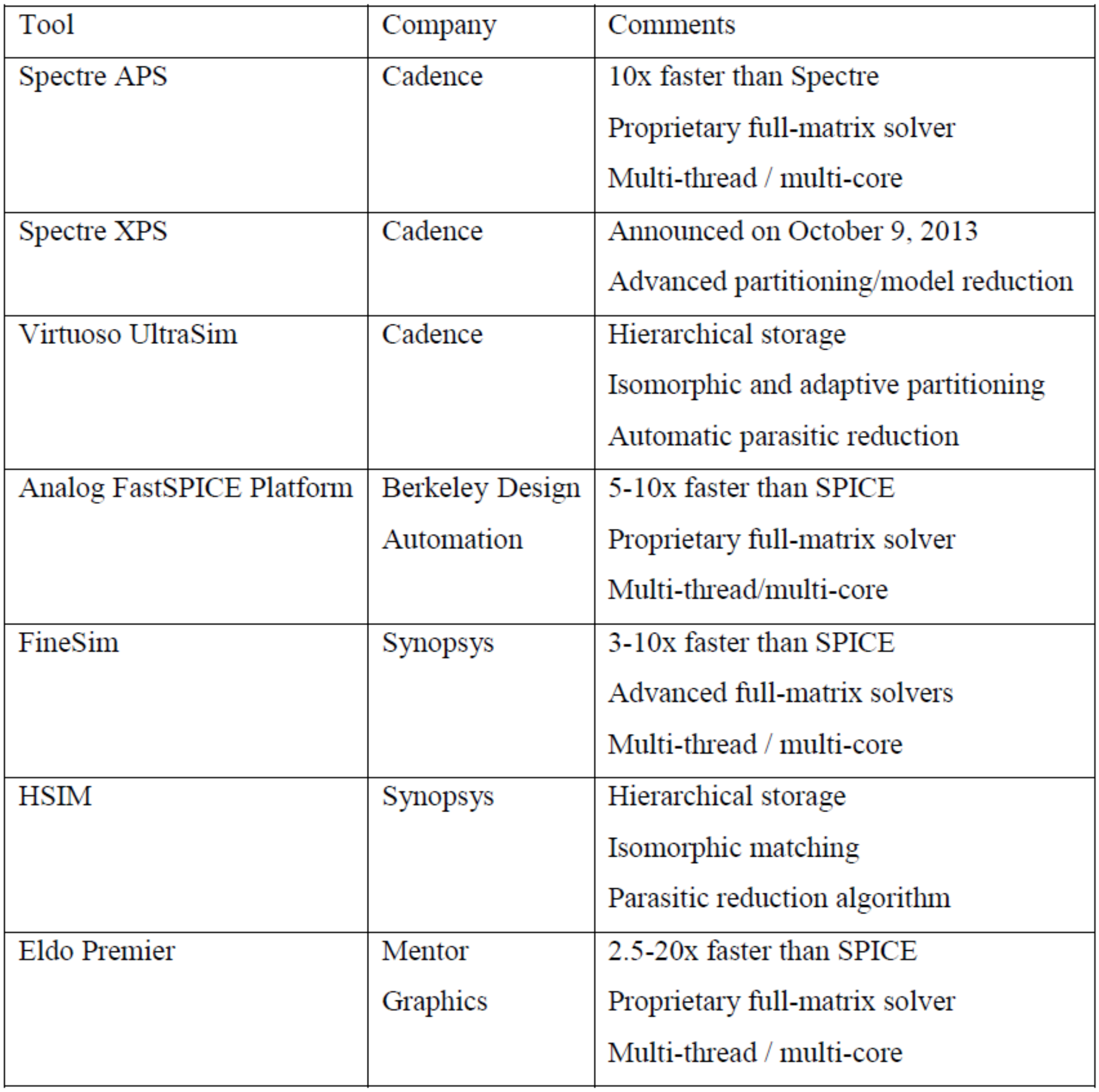
- ***Modified Simulators*** to run fast analog simulations.
- ***Macro-modeling*** to reduce model's complexity to speed up analog simulations.
- ***Behavioral modeling*** to use run digital simulations with behavioral models of analog blocks.

3.4.1 Modified simulators

The first approach to mixed-signal verification was adding extensions to existing analog simulators, table4, to support digital blocks and run simulations at transistor or/and gate levels. The two techniques used are:

- ***Fast SPICE***: matrix-based, graph-based and circuit-based, parallel computing.
- ***Piecewise-linear simulator***: piece-wise linear description of analog circuit instead of a nonlinear differential equation.

Table 4. Modified Analog Simulators to support digital designs [17].

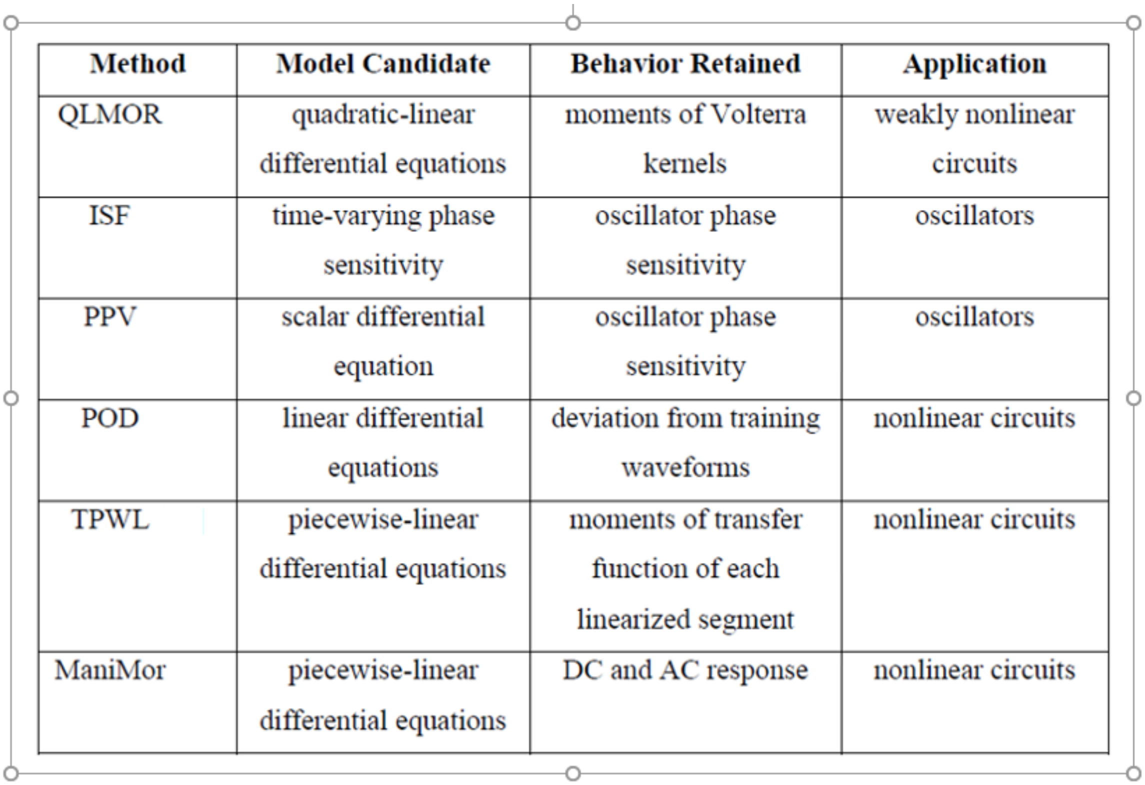


Tool	Company	Comments
Spectre APS	Cadence	10x faster than Spectre Proprietary full-matrix solver Multi-thread / multi-core
Spectre XPS	Cadence	Announced on October 9, 2013 Advanced partitioning/model reduction
Virtuoso UltraSim	Cadence	Hierarchical storage Isomorphic and adaptive partitioning Automatic parasitic reduction
Analog FastSPICE Platform	Berkeley Design Automation	5-10x faster than SPICE Proprietary full-matrix solver Multi-thread/multi-core
FineSim	Synopsys	3-10x faster than SPICE Advanced full-matrix solvers Multi-thread / multi-core
HSIM	Synopsys	Hierarchical storage Isomorphic matching Parasitic reduction algorithm
Eldo Premier	Mentor Graphics	2.5-20x faster than SPICE Proprietary full-matrix solver Multi-thread / multi-core

3.4.2 Macro-modeling

It is also called model reduction problem which consists on finding a model (linear or nonlinear) to an analog block, more straightforward, so its simulation runs faster while preserving the primary function of the block. For example, rather than using the spice model of transistors, a transfer function could be used instead. Table 6 provides a list of macro-models.

Table 5. Nonlinear macro-modeling algorithms [17].



Method	Model Candidate	Behavior Retained	Application
QLMOR	quadratic-linear differential equations	moments of Volterra kernels	weakly nonlinear circuits
ISF	time-varying phase sensitivity	oscillator phase sensitivity	oscillators
PPV	scalar differential equation	oscillator phase sensitivity	oscillators
POD	linear differential equations	deviation from training waveforms	nonlinear circuits
TPWL	piecewise-linear differential equations	moments of transfer function of each linearized segment	nonlinear circuits
ManiMor	piecewise-linear differential equations	DC and AC response	nonlinear circuits

3.4.3 Behavioral Modeling

Behavioral models are written in a high-level language and require expert knowledge. It is clear, some losses in accuracy are allowed as models are not derived directly from low-level transistor models. It is primordial to note that this research is at system-level, as depicted in figure 16.

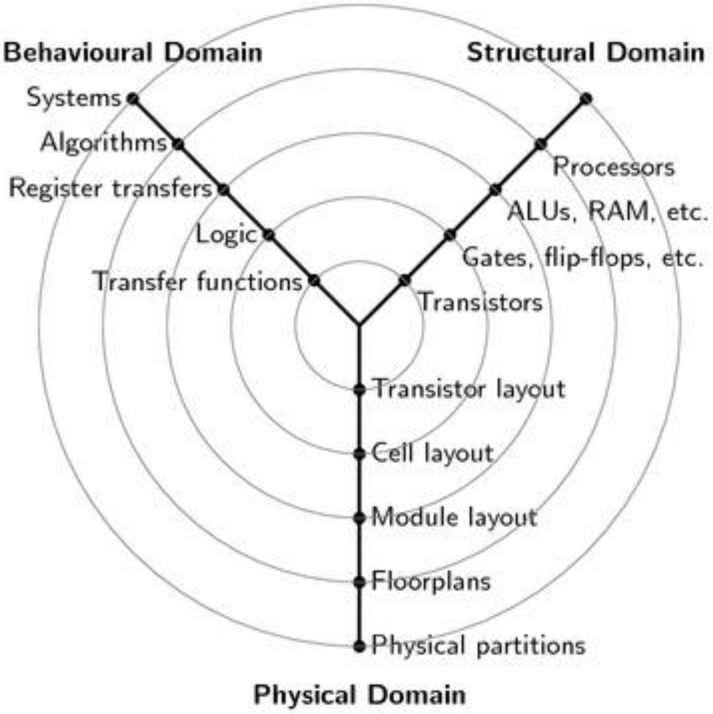


Figure 16. Gajski Kuhn Y-Chart.

The attractiveness of behavioral modeling, however, comes from the potential for 100-1000x simulation speedup [34,35,37] and at such speed, complete validation of mixed-signal SoC becomes feasible. In addition, it is possible to include checking or assertions in the behavioral models to aid the verification efforts.

The industry response to this approach is many high-level languages such as Simulink/Matlab, SystemC-AMS, Verilog-A/Verilog-AMS/VHDL-AMS [5,13,14,31, 36]. These languages have a SPICE-like analog solver. Even though it is most likely solving a much smaller problem than a full circuit Differential-Algebraic Equations, once the solver is invoked, it must execute all steps to solve differential equations. Therefore, using the continuous time analysis capabilities of these languages will result in longer simulation times compared to pure digital simulation. Consequently, other languages have been developed to allow running simulations of behavioral models with digital simulators.

We have chosen System Verilog in this research as it the industry de-facto of mixed-signal verification. It is:

- Faster than Spice
- Digital solver only
- Multi-value nets
- SV constructs and verification techniques

3.5 Challenges in Verification based on behavioral models

Verification method based on behavioral models simulated in digital simulators is the most common approach adopted in the industry for the speed of simulation. However, capturing a full system within the boundaries of a behavioral model is a tedious task because of:

- Metrics used in analog: bandwidth, gain, threshold, power, area, etc.
- Types of components available.
- Environment variations: temperature, Pressure, voltages, etc.
- Components' Size decreases, RF circuit issues become dominant in both digital and analog circuits.
- Multiple noise sources: thermal noise, Coupling noise, EMR, flicker noise, channel noise, power supply noise
- Circuit parameters, Impedance mismatches, open and closed loop gain, input and output impedances.
- Simulation cycle handles time in discrete and continuous values
- Separate simulators, working with the same set of signals
- Output analysis in Time/Frequency domains
- Simulation techniques for analog circuits work very well in linear domains, but many (most) circuits are nonlinear in the voltage or current domains.

3.6 Conclusion

Behavioral modeling is currently one approach that allows integrating analog blocks with digital ones and run simulations with digital simulators. These models are written by experts at a high level of abstraction but less accurate than the physical models. However, in the extent of tolerable error margin, these models allow mixed-signal verification at system-level and in many cases is the only solution to run simulations on large designs. In the next chapters, we present our framework with case study as proof of concept.

Chapter 4

Proposed Framework to Mixed-Signal Verification

The goal is to put forward a framework that is generic and reusable. The Generic framework will allow the development of behavioral models for analog designs in digital chains or analog chains. Reusable means the framework can be customized to speed up modeling effort. The main steps, as depicted in figure 17, to writing behavioral models for SoC verification are:

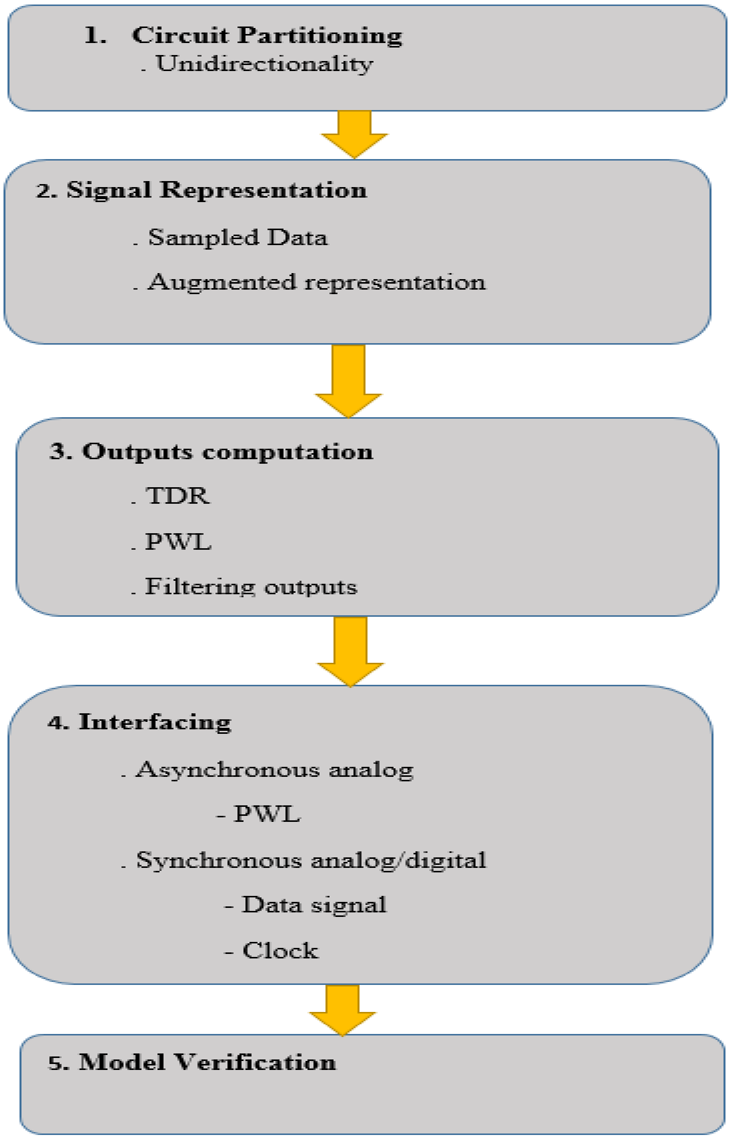


Figure 17. Proposed Framework for Mixed-Signal Verification.

4.1 Circuit partitioning

Digital simulators assume all digital gates are unidirectional; in other words, the output of one gate drives only the input of the next. Digital simulators are designed to deal with unidirectional signal propagation and uses the fact that a gate's input changes cause it to re-evaluate only that gate. On the other side, within the context of analog circuits, unidirectionality is not a restriction. Therefore, simulation of analog design using digital simulator requires that analog designs to be partitioned into modules which are unidirectional and breaks any loop or feedback in analog design. A good example is current summing nodes [17]. A current summing node, figure 18, cannot be on the boundary of a standalone module; instead, circuit blocks must be combined so that these nodes are absorbed into the interior of a module.

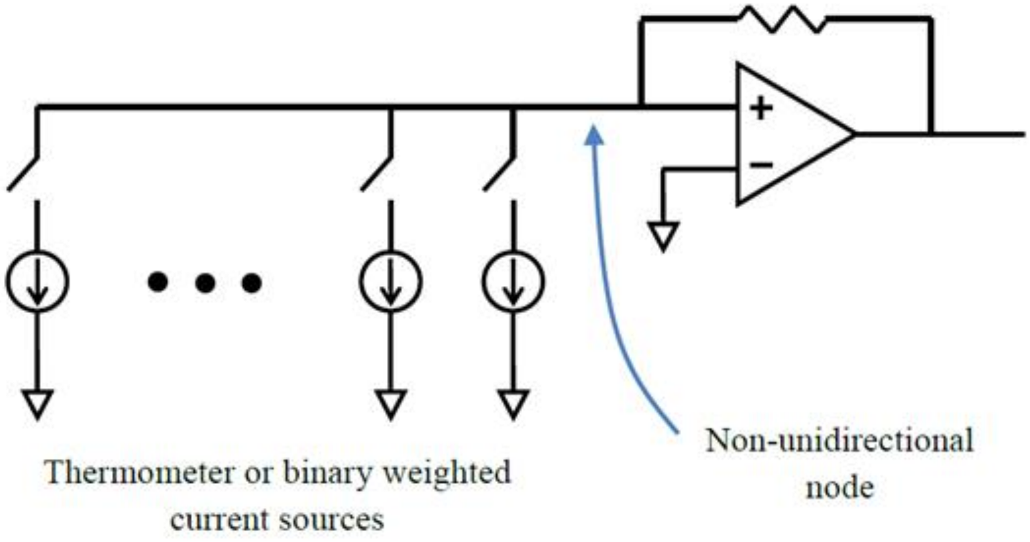


Figure 18. Summation node in analog circuits [17].

4.2 Signal Representation

After the analog circuits are properly segmented into unidirectional blocks, the next challenge to overcome is the fact that digital simulators work with Boolean values and discrete events, while analog signals are continuous. The two data representations commonly used to represent analog signals in digital simulators are:

- ***Sampled data representation*** of analog signals.
- ***Augmented Representation***: Piecewise linear PWL representation and XMODEL.

4.2.1 Sampled data representation

This is the approach used usually in "synchronous" analog systems. Unfortunately, most of analog circuits does not fall into this category. For instance, consider the example of asynchronous comparator depicted in figure 19. A comparator toggles its output from low to high as its positive input increases in value beyond its negative input. In the continuous time world, the crossing point is indicated in figure 19. When the input is represented as sampled data, the crossing point will be shifted in time. This error becomes smaller as the sample rate increases; however, this means that an analog signal requires many finely spaced samples leading to a large simulation time. In other words, the fundamental issue is the fact that digital simulators cannot interpolate between two signal updates until the arrival of the next samples.

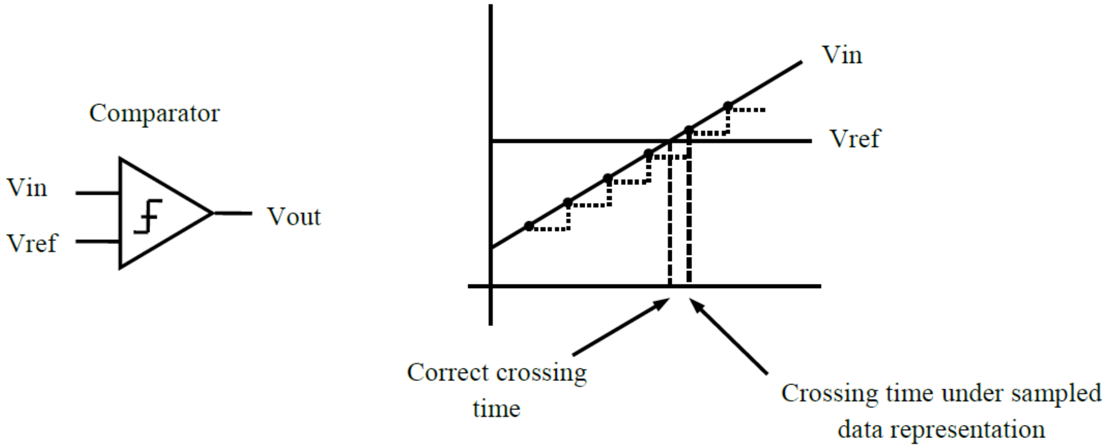


Figure 19. Sampled Data Representation of an analog comparator.

4.2.2 Augmented Representation

The drawback of sampled data representation of analog signals is the lack of information on the signal shape between sample updates. One solution is to associate (augment) more information to the samples, so that, digital simulators will process the samples and the augmented information in discrete-time.

In System Verilog, pin-accuracy can be retained by defining structures that contain more than one element and passing these structures across module ports. The following System Verilog code shows a structure containing a starting value (samples) of the signal and a slope.

```
typedef struct {
 real amplitude;
real slope;
} pwl_struct;
```

The PWL representation results in small spurs in the continuous time domain, while the sampled data representation does not introduce any distortion to synchronous systems in the discrete time domain. A signal may exhibit different rates of change throughout the simulated time frame; the update rate may be faster when the signal slope changes quickly, or it may be slower when the signal barely moves. This freedom cannot be easily afforded by models leveraging the sampled data representation and ***bilinear transforms***, since such transforms require a known and fixed sample rate.

The advantage of the PWL representation lies in its ability to allow digital simulators to ***generate asynchronous events***, thereby solving the issue with the sampled data representation.

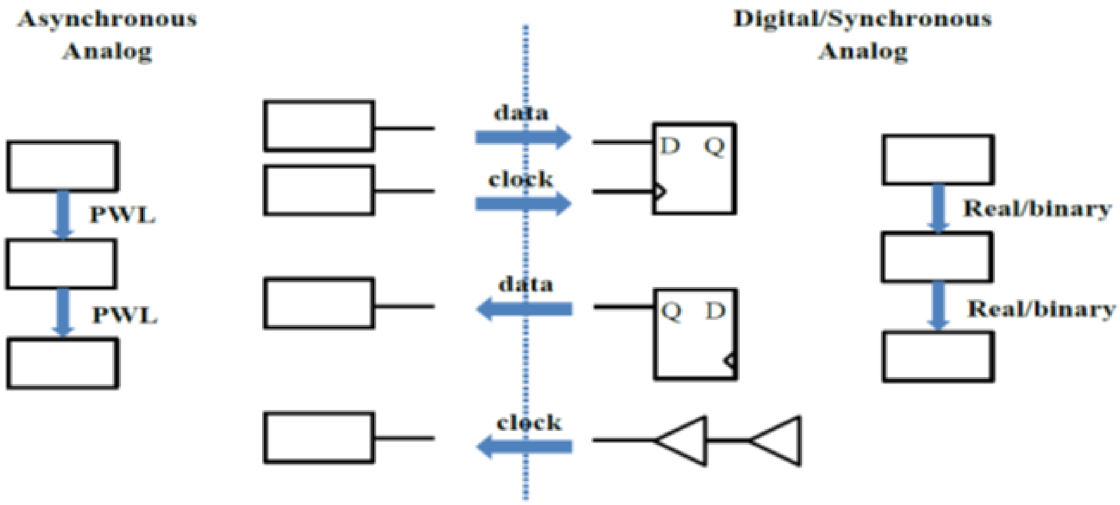


Figure 20. Communication between Analog/Digital components.

Consider a PWL signal crossing from the analog to the digital domain. The signal could cross as a data signal or a clock signal. If it crosses as a data signal, then there will usually be a digital clock that samples it, and the sampled value can be **interpolated** at the clock edge according to the slope of the signal. If the signal is a clock signal that is generated in the analog domain and supplied to the digital domain, then the timing of the clock edges is critical for the digital circuits. In this case, any delay or skew of the clock signal will need to be accounted for in the analog block that generated the signal.

Similarly, the signal flow from synchronous to asynchronous circuits can also be modeled. The procedure is the same whether it is a data or clock signal. The format of the signal needs to be first converted to the PWL structure, in which the slope element would be identically 0 in this case. If the rise or fall time is essential for modeling the receiving analog module's behavior, then slopes and delays can be attached to the staircase signal to make them trapezoidal signals.

The XMODEL seeks to write behavioral models that describe analog functions as nearly linear filters in the s-domain to place the circuit in a mostly linear domain and consequently uses the s-domain representation of analog signals. Data augmentation is achieved via the equivalence between a time domain signal and its Laplace transform. The output of a model can then be computed by multiplying the s-domain representation of the input with the s-domain transfer function of the system.

Unlike the PWL representation, XMODEL does not require steps in time; however, reconstruction (inverse Laplace transform) is necessary to view the waveforms in the time domain.

In summary, the commonly used sampled data representation of analog signals is unsuitable for modeling asynchronous analog circuits. A piecewise linear signal representation solves this issue. Providing the value and slope of the signal during each update allows digital simulators to generate asynchronous events for asynchronous circuits.

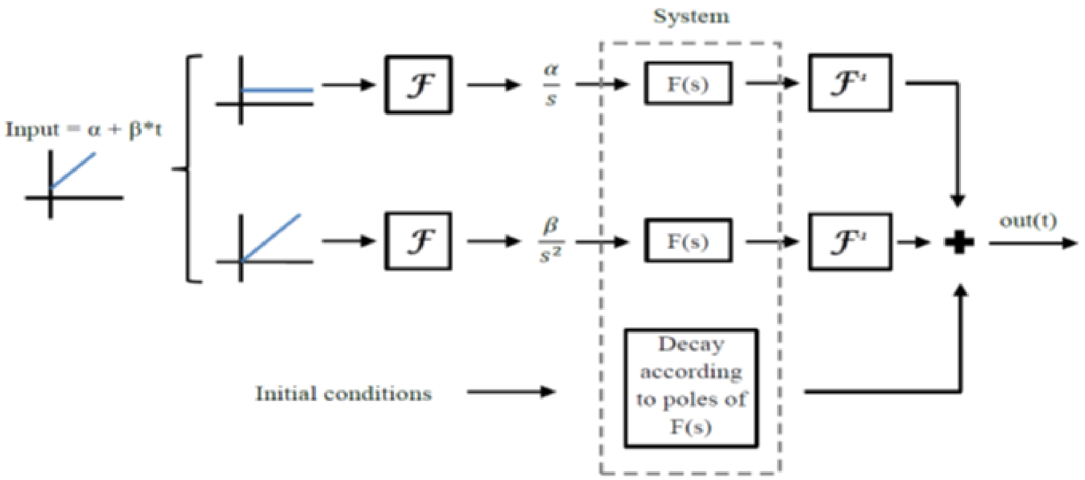
4.3 Module Output Computation

The input and output of analog circuits are a multi-element, real-valued structure. The computation of the PWL output of a module due to a PWL input can be accomplished in three steps. First, compute the continuous time domain response of a module due to a single linear segment on its input. Then based on the time constants of the system, a piecewise linear approximation is formed. Repeat step 1 and 2 with the next arriving input linear segment. Lastly, to be efficient, output updates that are within a specified error tolerance are removed.

- **Time Domain Response**

A piecewise linear input can be viewed as a series of delayed inputs; each with a different initial value and slope. The total transient response is a trajectory traced out by the evolution of the system's states when stimulated with successively delayed inputs. Since the final states of the current input segment are the initial states for the next input segment, it is sufficient to examine the behavior of the system due to a single linear segment and repeat the computation for the sequence of linear segments as they get updated.

A piecewise linear segment can be decomposed into a step with magnitude equaling to the initial value element of the PWL structure, and a ramp that starts at 0 but increases at the rate indicated by the slope element of the PWL structure. The total response of the system, then, is composed of the response to the step, the response to the ramp and the decay of the initial states of the system, figure 21.



f

Figure 21. Output computing transformation: time and frequency domains.

- **Forming Piecewise-Linear Output**

The next step is converting time output waveform stimulated by a linear ramp into piecewise linear segments. The general idea is to estimate the length of time Δt after which the output response deviates too much from a linear ramp. After that, output immediately the segment describing the signal from the current instant until that time. Then, return after the expiry of Δt to compute the output waveform again to determine the next linear segment.

- **Filtering Output Updates**

Filtering unnecessary output updates by comparing slopes of 02 consecutive outputs. Communication between blocks is through the propagation of an analog signal in the piecewise linear format through a chain of analog modules. Each input update will result in one or more output updates which will become the input updates for the next block. Contrary to logic signals where it is apparent when an output changes, with continuous time signals it is not so clear. Thus, it is necessary to limit output updates to avoid an unnecessarily large number of events.

4.4 Behavioral models' verification

To verify the accuracy of behavioral models towards physical analog blocks, one or many of the following methods can be used:

- Transfer matrix matching under linear system assumption
- Simulation trace matching
- Finite-state-machine-based micromodel generation

4.5 Conclusion

A generic and reusable framework allows fast development of behavioral models. To verify the feasibility of the framework, we presented a case study in next chapters where we present a behavioral model of an analog VCO and compare performances towards the physical model.

Chapter 5

Case Study: Design of an analog VCO

In this chapter, we present the physical design of a Voltage Controlled Oscillator (VCO). We highlight the key elements that are covered by behavioral models and prepare plots to be used during behavioral model verification. The choice of the VCO is not random, but it is justified as a VCO is a critical element of PLLs and interfaces with analog and digital blocks.

The choice of VCO as a proof of concept to validate the proposed framework is justified by the following arguments:

- VCO is present in many devices such as PLL, mixers, IQ generator, frequency synthesizers and so on.
- The intent here is not designing a VCO to exceed the actual designs performances, but rather, a proof of concept to demonstrate the applicability of the framework.

5.1 Summary of results

The performance of the designed VCO and the initial specifications are summarized in the following table 1. All simulations are realized using Cadence/Spectre.

Table 6. Summary of results

Design Parameters	**Initial Specifications**	**Calculated performances**	**Simulated performances**
Nominal Frequency f_c	8.050 GHz	8.05GHz	8.05GHz
Frequency Range	300 MHz(8.05GHz-8.35 GHz)	NA	420 MHz

Design Parameters	**Initial Specifications**	**Calculated performances**	**Simulated performances**
Output Amplitude	300 mV peak diff	300 mV peak diff	415 mV peak diff
Phase Noise	$-106.9\ dBc/Hz$	-113.44 dBc/Hz	-111 dBc/Hz
Supply Voltage	1.2 V	1.2 V	1.2V

5.2 Design approach

The design steps for the VCO, as presented in [15] are:

- First estimation of the biasing current and inductor for the tank circuit
- Transistor characterization
- Varactor characterization
- Inductor characterization
- Performances analysis:
    - ✓ Time Domain Analysis
    - ✓ Frequency Domain Analysis
    - ✓ Steady period analysis
    - ✓ Phase noise analysis
    - ✓ Noise analysis
    - ✓ Power dissipation and optimization
    - ✓ Final design and list of components

5.3 VCO topology

The main blocks of the VCO topology as depicted in figures 22 are:

- ▪ Current mirror source
- ▪ -Gm Differential Oscillator
- ▪ Buffer

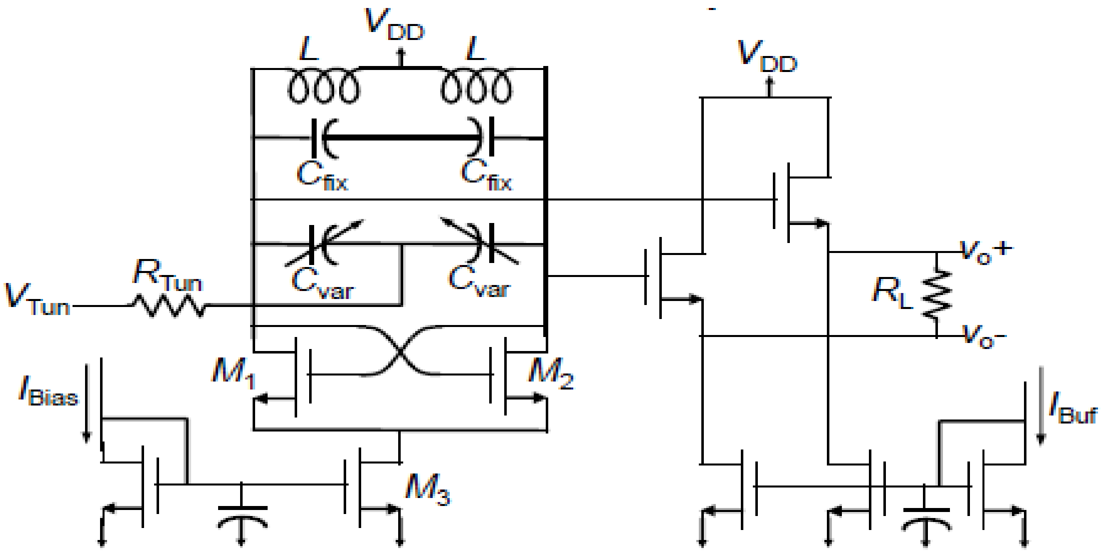


Figure 22. –Gm CMOS differential oscillator.

Advantage:

- Better phase noise and jitter performance at high frequency

Disadvantages:

- High power consumption

5.4 Design procedure

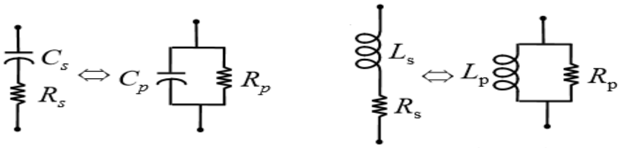
5.4.1 First estimation of biasing current and tank inductor

- Assuming that $V_{tank} = 500\ mV$ is enough to achieve 300 mV peak differential at the buffer output.
- Assuming the tank has a quality factor 15, and the varactor quality factor will be higher (Q > 30, varactor characterization).

$$V_{tank} = \frac{2}{\pi} I_B Q_T \omega L \;\; \rightarrow \; I_B L = (V_{tank} * \pi)/(2 * Q_T \omega) = (0.5 * \pi)/(2 * 15 * 2 * \pi * 8.05e9) = 1.0352e - 12\ v/rad$$

- Starting point: I have chosen $I_{bias} = 1\ mA\ or\ 0.5\ mA\ per\ side \Rightarrow$

  $L = 1.0352nH\ or\ 517.6\ pH/side$
- In the section Inductor characterization, it will be shown that this inductor is feasible at 8.05GHz and has Q of 23 and resonance frequency 45 GHz



$$R_p = R_s\left(1+Q^2\right) \qquad R_p = R_s\left(1+Q^2\right)$$

$$C_p = C_s\left(\frac{Q^2}{1+Q^2}\right) \qquad L_p = L_s\left(\frac{1+Q^2}{Q^2}\right)$$

- Applying series-parallel conversions, the parallel inductor resistance at the nominal frequency $f_{min} 8.05Ghz\ and\ \ f_{max}\ 8.35Ghz\ \ (VCO\ specification:\ 300\ Mhz\ frequency\ range)$
    - At frequency $f_{min} = 8.05Ghz$

$$Q = \frac{X_L}{R_s} \ \Rightarrow\ R_s = \frac{X_L}{Q} = \frac{f_{min}*2*pi*L}{Q} = \frac{8.05Ghz*2*pi*1.035nH}{15} = 3.4907\ Ohms$$

$$R_p = R_s(1+Q^2) = 4.3633*(1+15*15) = 788.88\ Ohms$$

-
    - $At\ frequency\ f_{max} = 8.35Ghz$

$$R_s = \frac{X_L}{Q} = \frac{f_{max}*2*pi*L}{Q} = \frac{8.35Ghz*2*pi*1.035nH}{15}$$
$$= 3.62\ Ohms$$

$$R_p = R_s(1+Q^2) = 4.5259*(1+15*15) = 818.288\ Ohms$$

-
    - As Q > 10, $L_p \cong L_s$

- The required transconductance:
  Assuming Q of capacitors >> that Q of the inductor, then $Q_{tank} \approx Q_{\,inductor}$, hence $R_T \approx R_p\ of\ the\ inductor$

$g_m > \frac{2}{R_T} = \frac{2}{788.88} =\ 2.5\ \ mA/V$. Thus, for oscillation, a transistor of gm higher than 2.5 mA/V at the bias current of 0.5mA is required.

5.4.2 Transistor Characterization

Starting by setting all VCO transistors at 20 uM transistors (10 fingers*2uM).

The characterization of this transistor shows that Gm is **5.8 mA/V** and f_T is 45 GHz at the bias current of 0.5mA.

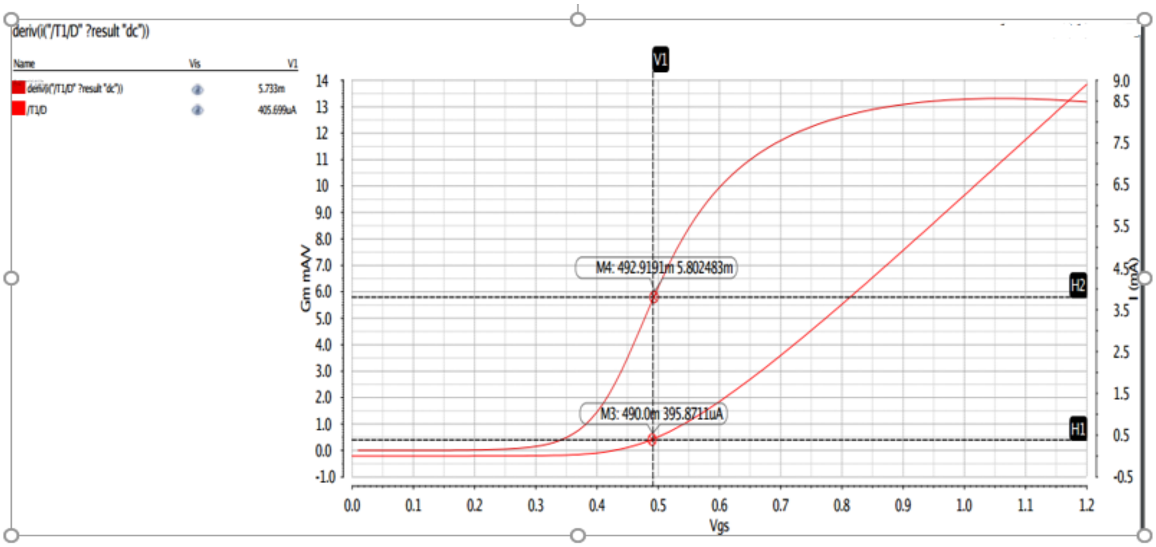


Figure 23. Transistor Characterization: gm and Ids versus Vgs.

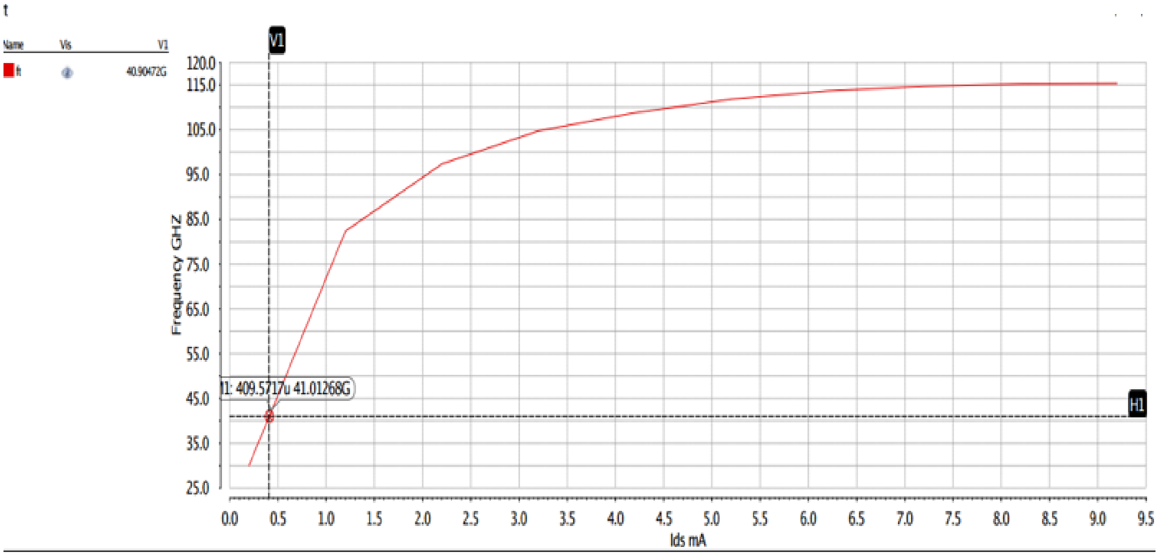


Figure 24. Transition frequency versus bias current.

5.4.3 Setting Frequency and Tuning Range

- Choosing capacitors and varactors to achieve the correct frequency and tuning range

○ $f_{min} = \frac{1}{2\pi\sqrt{LC_{min}}} \dashrightarrow C_{min} = \frac{1}{f_{min}^2 * 4 * \pi^2 L} = \frac{1}{8.05G^2 * 4 * \pi^2 * 1.035nH} = 377.59fF$

- $f_{max} = \frac{1}{2\pi\sqrt{LC_{max}}} \;\; --\rightarrow \;\; C_{max} = 350.95\,fF$
- $C_{min}\ and\ \ C_{max}\ \ are\ calculated\ using\ the\ total\ inductor, thus\ for\ each\ side\ the\ capacitances\ are\ 755.18\,fF\ and\ 701.9\,fF$

To achieve the tuning range of 300 MHZ, the required capacitance ratio is:

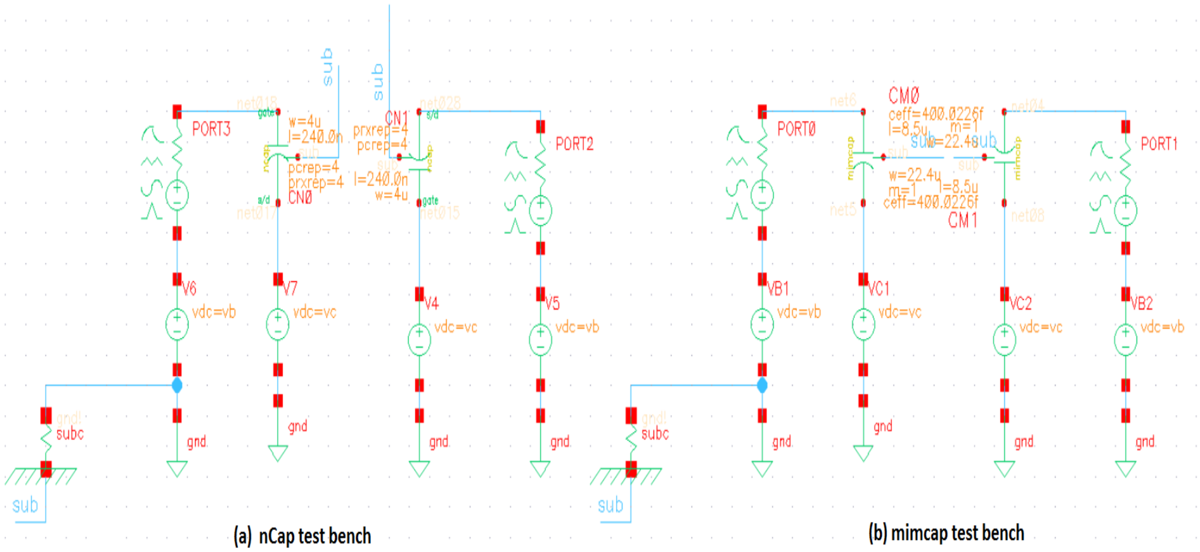
- $\frac{C_{max}}{C_{min}} = (\frac{f_{max}}{f_{min}})^2 = (\frac{8.35G}{8.05G})^2 = 1.076$

5.4.4 Varactor Characterization

- ncap capacitors used to tune the VCO
- mimcap used as a fixed capacitance

Why ncap and mimcap?

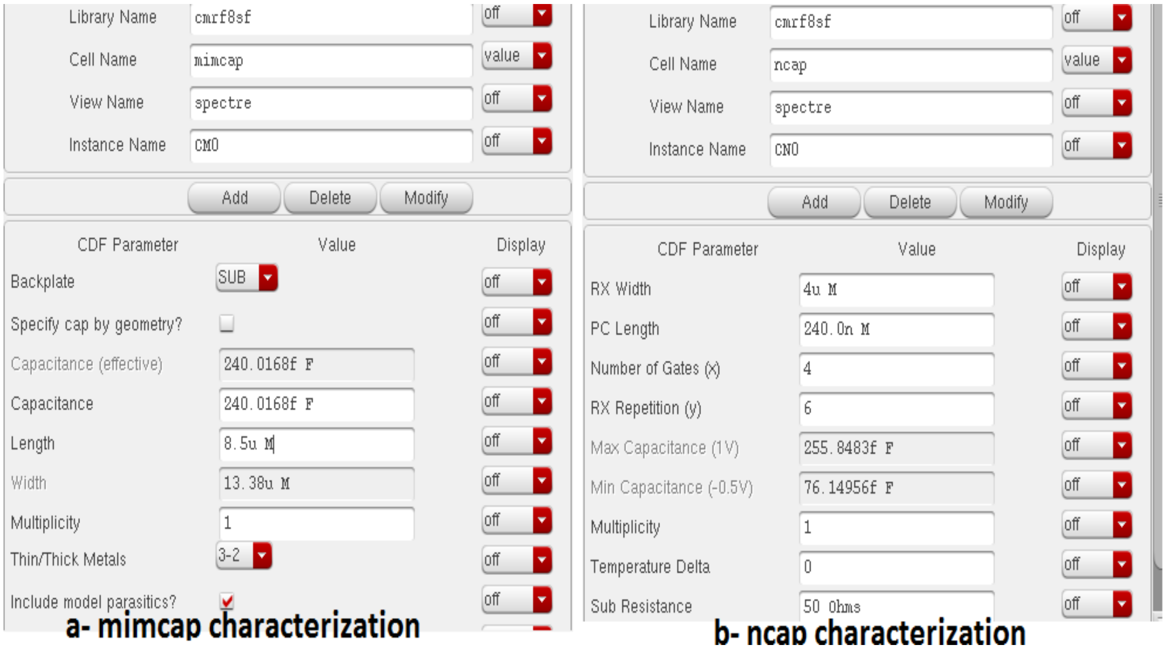
- Impedance varies with voltage in the Gate.
- ncap provides maximal capacitance near low voltage
- Capacitance and Q at the Gate are close to those at the Source.
- mimcap has stable capacitance and high Q



(a) nCap test bench

(b) mimcap test bench

Figure 25. ncap and mimcap test bench characterization circuit.



Library Name	cmrf8sf	off
Cell Name	mimcap	value
View Name	spectre	off
Instance Name	CM0	off

Add Delete Modify

CDF Parameter	Value	Display
Backplate	SUB	off
Specify cap by geometry?	□	off
Capacitance (effective)	240.0168f F	off
Capacitance	240.0168f F	off
Length	8.5u M	off
Width	13.38u M	off
Multiplicity	1	off
Thin/Thick Metals	3-2	off
Include model parasitics?	☑	off

a- mimcap characterization

Library Name	cmrf8sf	off
Cell Name	ncap	value
View Name	spectre	off
Instance Name	CN0	off

Add Delete Modify

CDF Parameter	Value	Display
RX Width	4u M	off
PC Length	240.0n M	off
Number of Gates (x)	4	off
RX Repetition (y)	6	off
Max Capacitance (1V)	255.8483f F	off
Min Capacitance (-0.5V)	76.14956f F	off
Multiplicity	1	off
Temperature Delta	0	off
Sub Resistance	50 Ohms	off

b- ncap characterization

Figure 26. mimcap and ncap parameters.

- ncap capacitance varies from 254 fF to 177.54 fF which offers wide tuning range (Cmax/Cmin ratio =1.4).
- Q varies from 37 to 57

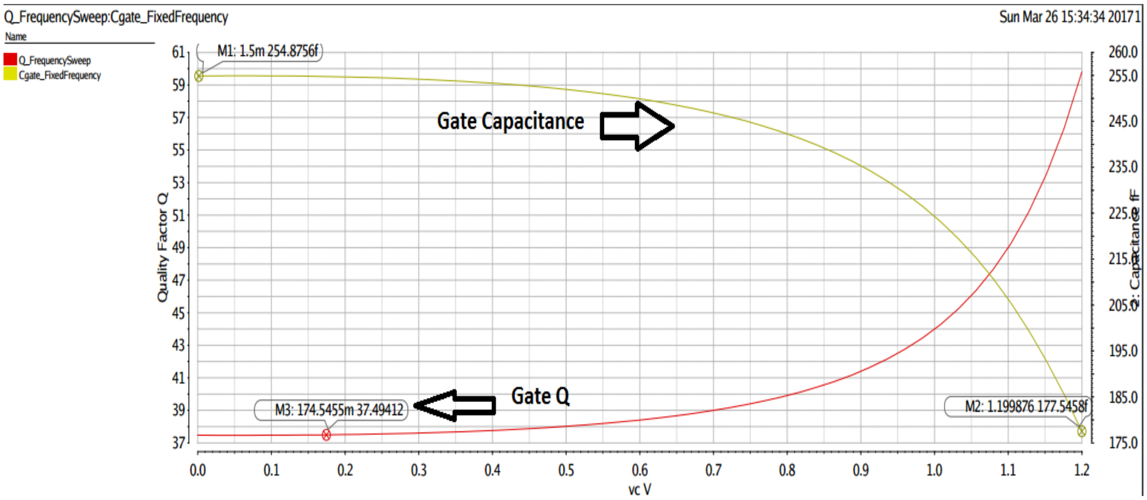


Figure 27. ncap Characterization: Gate biased at 1.2V

- mimcap has stable capacitance over the tuning voltage range
- mimcap has Q of 51

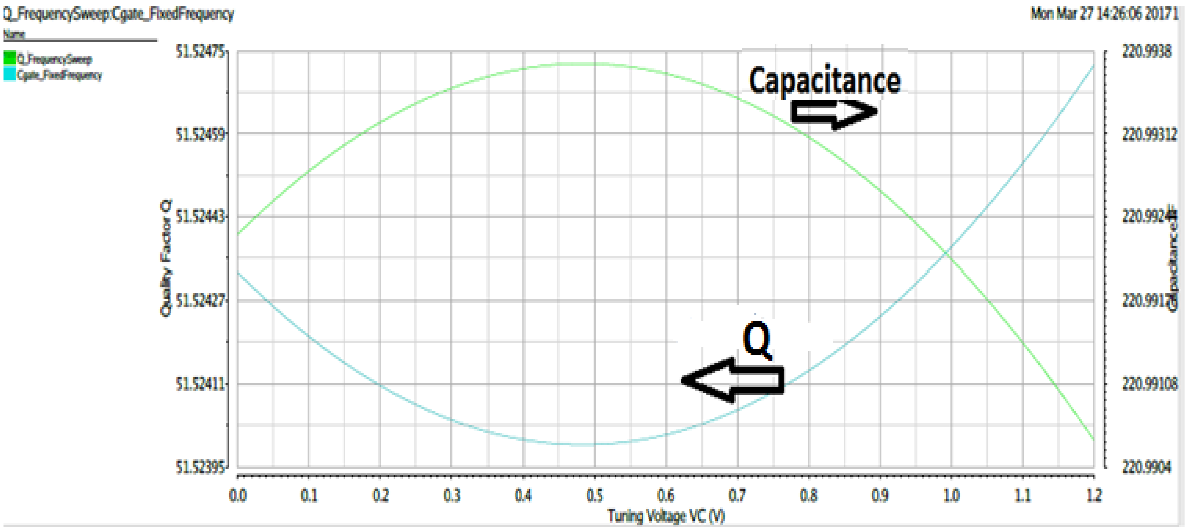


Figure 28. mimcap Characterization- biased at 1.2 V.

- After simulations and adjustment of capacitors size, the final values are:

Table 7. Tank Circuit components.

		Fmin=8.05 GHz	**Fmax=8.35 GHz**
Inductor	Inductance per side	517pH or 1.035 nH total	517 pH
	Q (initial)	15	15
	Series Resistance	4.3633 Ohms	4.5259 Ohms
	Equiv Parallel Resistance	788.88 Ohms	818.28 Ohms
Varactor	Capacitance per side	254 fF	220 fF
	Q	37	43
	Series resistance	1.1116 Ohms	1.4918 Ohms
	Equiv Parallel Resistance	3.0459 K Ohms	9.0286 K Ohms
Fixed Capacitor mimcap	Capacitance per side	240 fF	240 fF
	Q	51	51
	Series Resistance	0.6191 Ohms	0.6422 Ohms
	Equiv Parallel Resistance	3.221 k Ohms	3.341 k

Parasitic Capacitances	Buffer Transistor (for each)	Cgg=110 fF , Cgs=84 fF
		Cgd=31 fF	

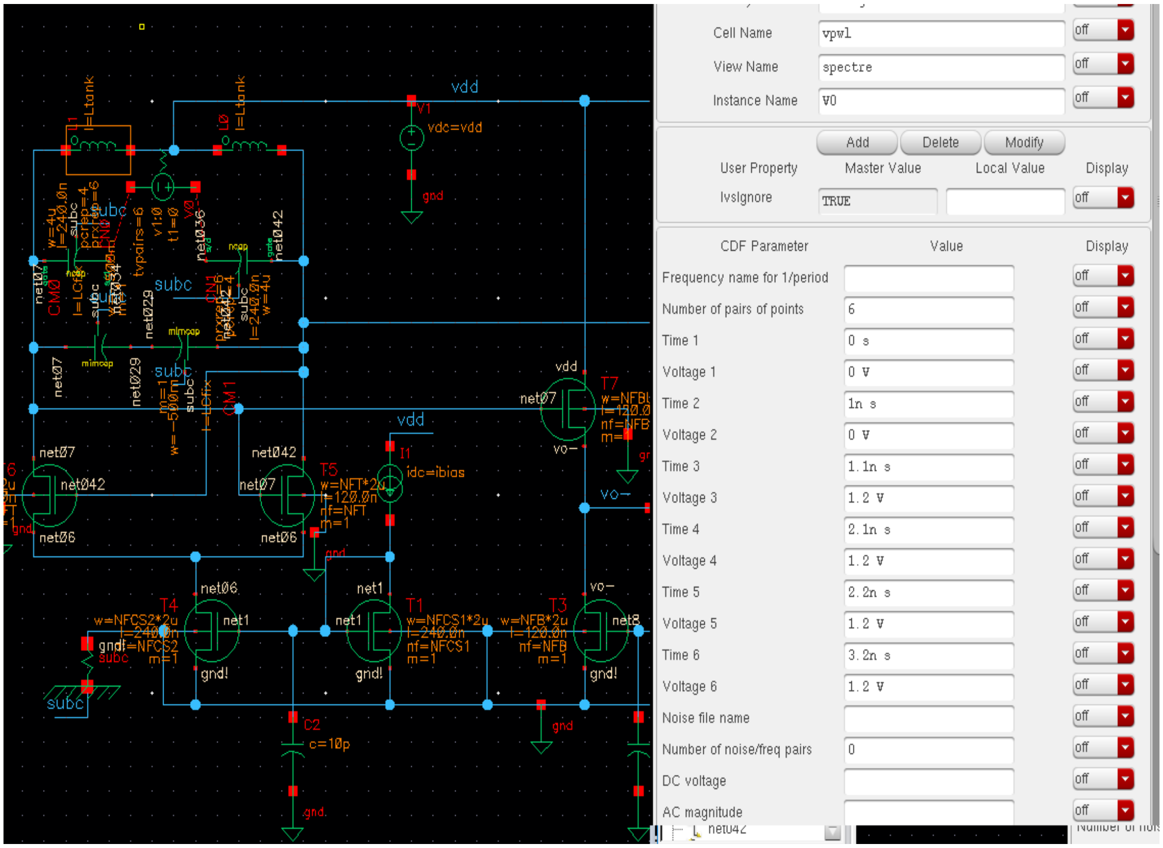
From simulation	Cross Couples Transistor (for each)	Cgg=32.87 fF ,
		Cgs=24.82fF
		Cgd=9.38 fF	

Tank Circuit	Total Parallel Resistance	744.9369 Ohms	918.774 Ohms
	Effective tank Q	13.5334	11.388

- **Estimating the capacitors:**
    - Parasitic capacitance from buffer transistor: $C_{gs} + 4C_{gd} \cong 84 + 4 * 31$ ≅208 fF added to each side
    - Parasitic capacitance from cross-coupled transistor: $C_{gs} + 4C_{gd} \cong 24 + 4 * 9$ ≅60 fF added to each side
    - Total parasitic capacitance added on each side: 208+60=268 fF
    - Capacitance required for tank circuit is: 755.18 fF at 8.05 GHz and 701.9 fF at 8.35GHZ
    - Subtracting the parasitic capacitance: 755-268=487 fF and 701-268=433 fF
    - Using varactor: 254 fF to 220 fF → Fixed capacitance $\approx 487 - 254 = 230\ fF$
    - Through simulations, the fixed capacitance is adjusted to 280fF (simulation) as also the inductor characterized has an inductance of 574 pH, see the section of inductor characterization.

5.5 Time Domain Analysis

- A Voltage piecewise linear source is used to start the oscillator in transient analysis.
- Initially, ideal inductors are used with series resistance defined in the component parameters. Later inductors from RF kits are used.



Cell Name	vpwl	off
View Name	spectre	off
Instance Name	V0	off

Add Delete Modify

User Property	Master Value	Local Value	Display
lvsIgnore	TRUE		off

CDF Parameter	Value	Display
Frequency name for 1/period		off
Number of pairs of points	6	off
Time 1	0 s	off
Voltage 1	0 V	off
Time 2	1n s	off
Voltage 2	0 V	off
Time 3	1.1n s	off
Voltage 3	1.2 V	off
Time 4	2.1n s	off
Voltage 4	1.2 V	off
Time 5	2.2n s	off
Voltage 5	1.2 V	off
Time 6	3.2n s	off
Voltage 6	1.2 V	off
Noise file name		off
Number of noise/freq pairs	0	off
DC voltage		off
AC magnitude		off

Figure 29. Startup settings of the VCO.

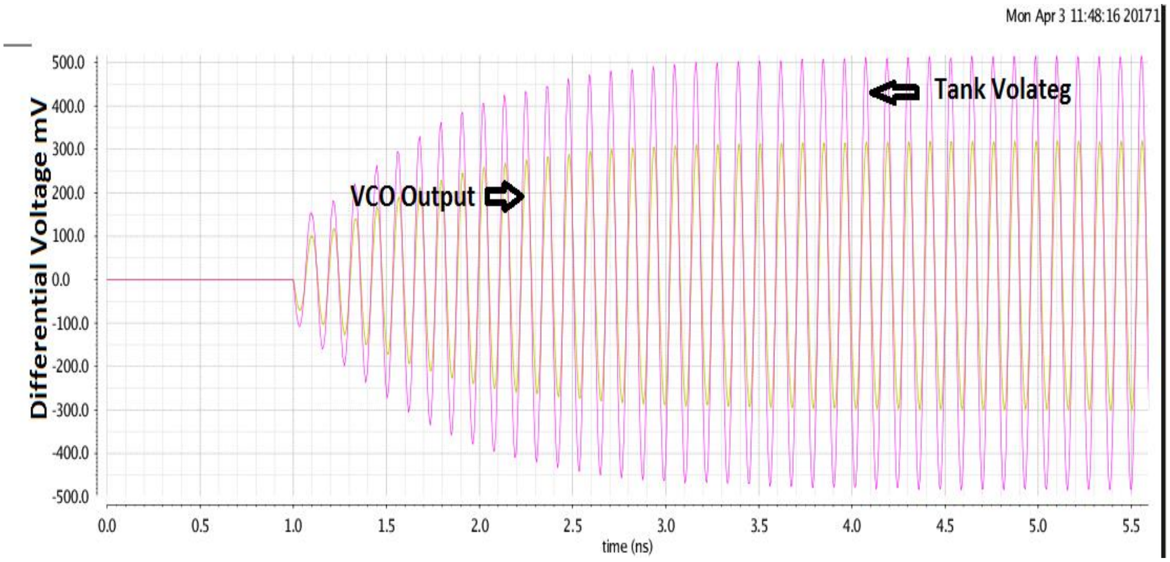


Figure 30. Transient Analysis: Tank and Output voltage.

5.7 PSS analysis

- Figure 31, shows the tuning frequencies versus the tuning voltage. The nominal frequency of 8.05 GHz is obtained with tuning voltage equals 0. V. 8.35 GHz is obtained with tuning voltage equals 1 V.
- The frequency range is 8.412 GHz-7.907 GHz =505 MHZ that is higher than specification (300 MHz) due to the ratio of $\frac{C_{max}}{C_{min}} = \frac{C_{fix}+C_{var,max}}{C_{fix}+C_{var,min}}$ (1.4) of ncap varactor.
- $K_{VCO} = \frac{8.412GHz-7.907GHz}{1.2V-0V} = 420.84\ MHz/V$

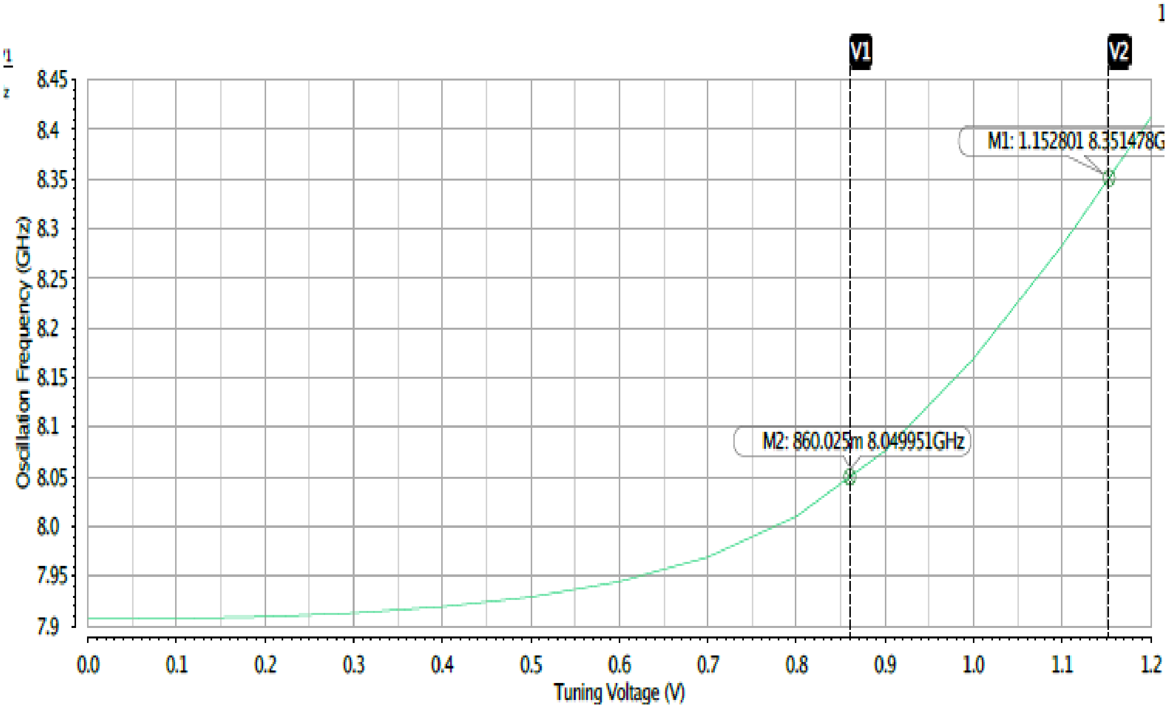


Figure 31. Frequency Range versus Tuning Voltage.

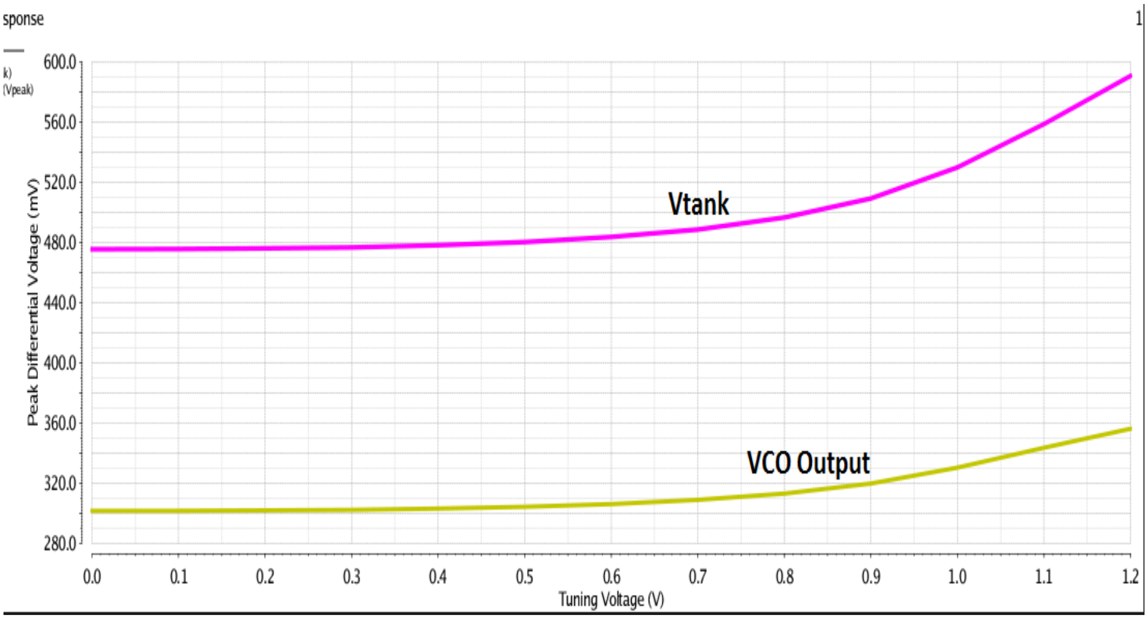


Figure 32. PSS Analysis: Tank and VCO output versus Tuning Voltage.

5.8 Phase Noise

Phase noise is a metric of merit to characterize the frequency variation or shifting of the VCO output due to the variation of resistors, capacitors, inductors, and transistors due to temperature, voltage variations. The goal here is to use Cadence simulator to simulate the phase noise of the VCO to use in the behavioral model.

5.8.1 Estimating Phase noise

- Using the equation $PN = \left(\frac{A\omega_o}{(2Q\Delta\omega)}\right)^2 \left(\frac{FkT}{2P_s}\right)$
  - $P_s = \frac{V_{tank}^2}{2R_p} = \frac{0.63^2}{2*744.9369} = \frac{0.3969}{1.4899*10^3} = 2.664*10^{-4} = 266.4\mu W$
  - Assuming that 1/f is not dominant and transition time is 20%. Then $F = 1 + 4\gamma g_m R_p(1-\rho)$=1+4*0.67*0.0058*744.9369*0.2=1+2.3159=**3.3159**

$$PN = \left(\frac{\sqrt{2}*2\pi*8.05Ghz}{(2*13.5334*2\pi*1Mhz)}\right)^2 \left(\frac{3.3159*1.38*10^{-23}*298}{2*266.4*10^{-6}}\right) =$$

$(176.9081857*10^3)2.55936*10^{-17} = 4.5277*10^{-12} = -113.44\ dBc/Hz$

- Considering phase noise due to the tuning resistance, the phase noise can be estimated by $PN = \left(\frac{V_m K_{VCO}}{2\Delta\omega}\right)^2$

  $V_m = \sqrt{4kTr} = \sqrt{4x1.38x10^{-23}x298*100Ohms} = 1.283nV/\sqrt{Hz}$

$$K_{VCO} = \frac{8.412GHz - 7.907GHz}{1.2V - 0V} = 420.84\ MHz/V$$

$$PN = \left(\frac{1.283n * 420.84M}{2 * 1M}\right)^2 \text{ =-131 dBc/Hz}$$

5.8.2 Phase noise simulation and scaling transistors size

- First simulation shows the phase noise at 1 MHz offset from the nominal frequency is -105 dBc/Hz, figure 12.
- The corner frequency or 1/f corner is at 10 kHz offset.

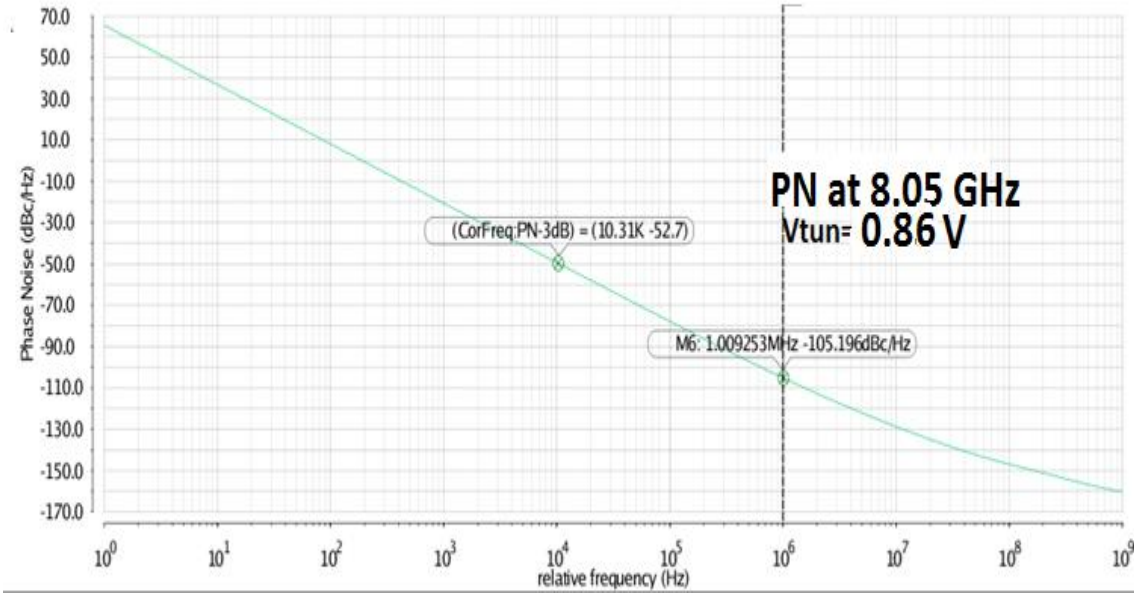
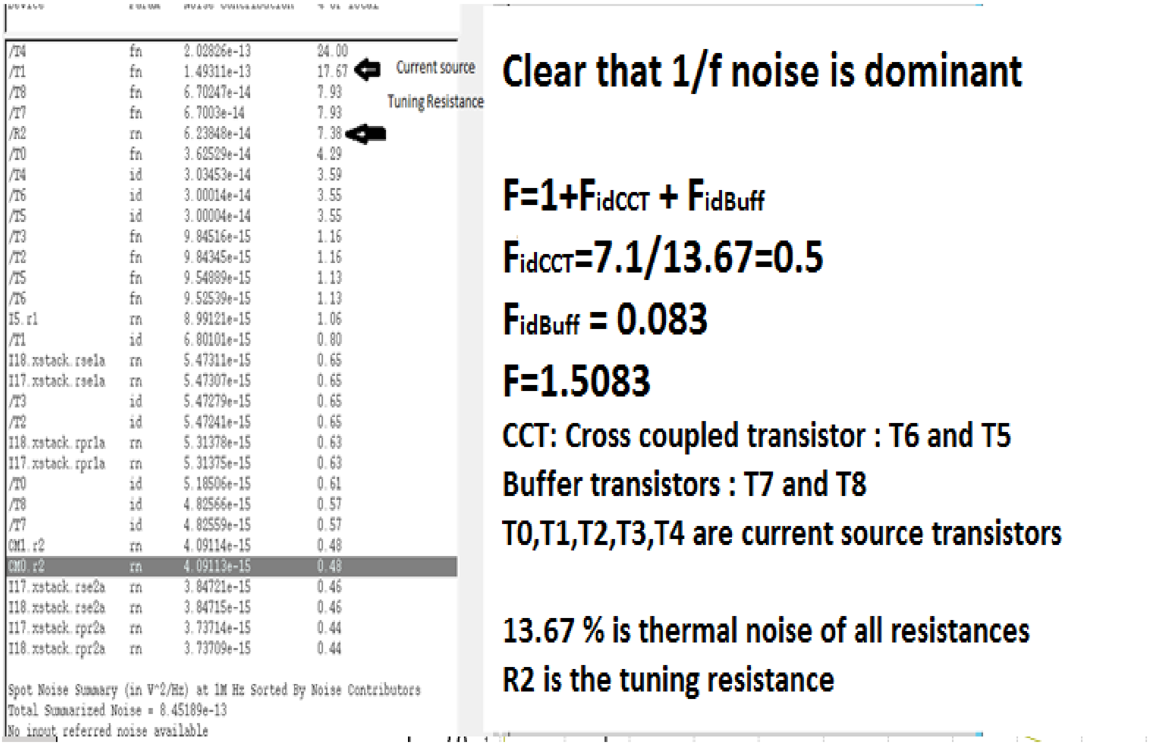


Figure 33. Initial design: Phase noise at 1 MHz offset

5.9 Noise Analysis

Noise summary to identify significant contributors to noise:



```
/T4              fn    2.02826e-13    24.00
/T1              fn    1.49311e-13    17.67
/T8              fn    6.70247e-14    7.93
/T7              fn    6.7003e-14     7.93
/R2              rn    6.23848e-14    7.38
/T0              fn    3.62529e-14    4.29
/T4              id    3.03453e-14    3.59
/T6              id    3.00014e-14    3.55
/T5              id    3.00004e-14    3.55
/T3              fn    9.84516e-15    1.16
/T2              fn    9.84345e-15    1.16
/T5              fn    9.54889e-15    1.13
/T6              fn    9.52539e-15    1.13
I5.r1            rn    8.99121e-15    1.06
/T1              id    6.80101e-15    0.80
I18.xstack.rse1a rn    5.47311e-15    0.65
I17.xstack.rse1a rn    5.47307e-15    0.65
/T3              id    5.47279e-15    0.65
/T2              id    5.47241e-15    0.65
I18.xstack.rpr1a rn    5.31378e-15    0.63
I17.xstack.rpr1a rn    5.31375e-15    0.63
/T0              id    5.18506e-15    0.61
/T8              id    4.82566e-15    0.57
/T7              id    4.82559e-15    0.57
CM1.r2           rn    4.09114e-15    0.48
CM0.r2           rn    4.09113e-15    0.48
I17.xstack.rse2a rn    3.84721e-15    0.46
I18.xstack.rse2a rn    3.84715e-15    0.46
I17.xstack.rpr2a rn    3.73714e-15    0.44
I18.xstack.rpr2a rn    3.73709e-15    0.44

Spot Noise Summary (in V^2/Hz) at 1M Hz Sorted By Noise Contributors
Total Summarized Noise = 8.45189e-13
No input referred noise available
```

Clear that 1/f noise is dominant

$F=1+F_{idCCT} + F_{idBuff}$

$F_{idCCT}=7.1/13.67=0.5$

$F_{idBuff} = 0.083$

$F=1.5083$

CCT: Cross coupled transistor : T6 and T5

Buffer transistors : T7 and T8

T0,T1,T2,T3,T4 are current source transistors

13.67 % is thermal noise of all resistances

R2 is the tuning resistance

Figure 34. Noise Summary at VC=850mV.

- **Adjustment to reduce phase noise**
  - Reduce 1/f noise by increasing the buffer transistors and current transistors to 100 uM/2uM with Length 600 nM
  - Increase the size of the cross-coupled transistors to 30 uM (15 fingers/2uM)
  - Increase the buffer current from 1.5 mA to 2 mA
- Phase noise is reduced to -111.56 dBc/Hz Specification met, figure 13.

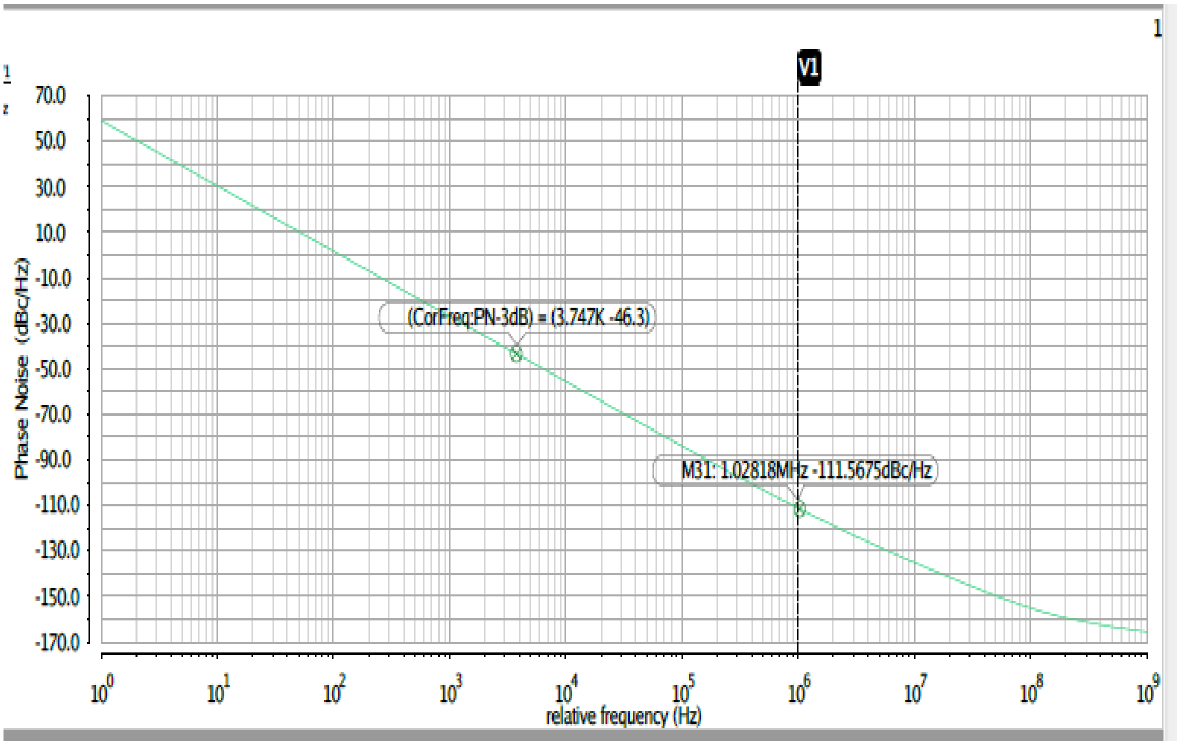


Figure 35. Phase Noise of the final design at offset 1Mhz.

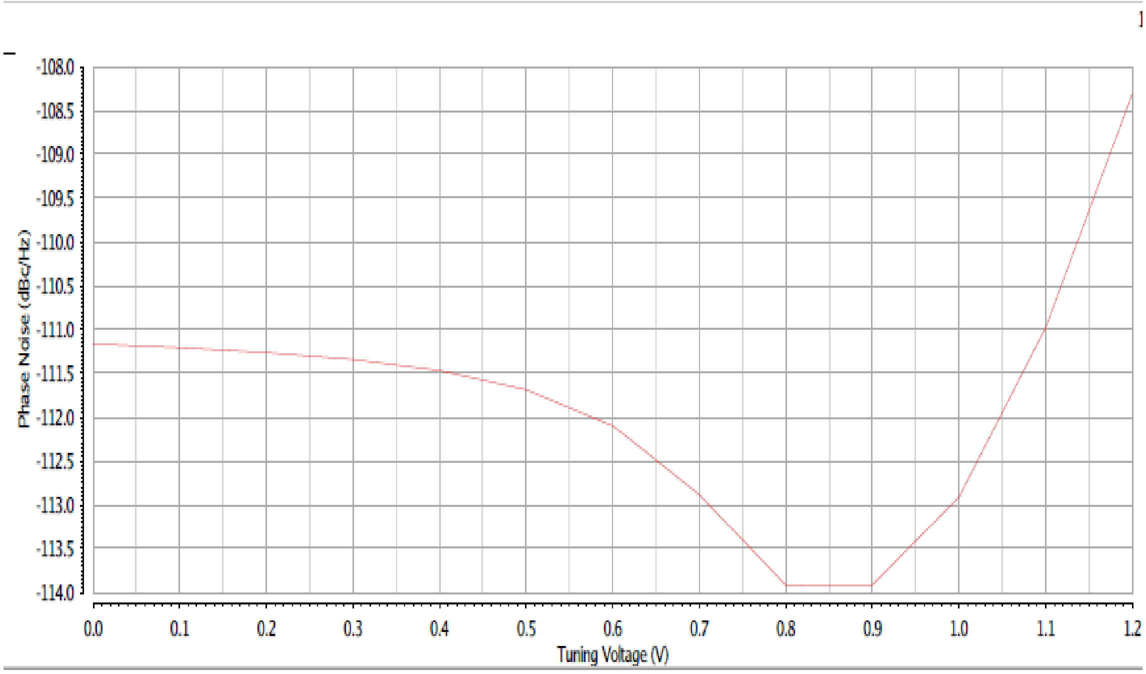


Figure 36. Phase Noise versus Tuning Voltage, tuning resistance = 100 Ohms.

- Figure 15 shows that at all tuning voltage (0 to 1.2 V), the designed VCO meets the phase specification. At 0V, PN=-111 dBc/Hz and at 1.2 V, PN=-108.3 dBc/Hz. Lower PN is at 0.8 V

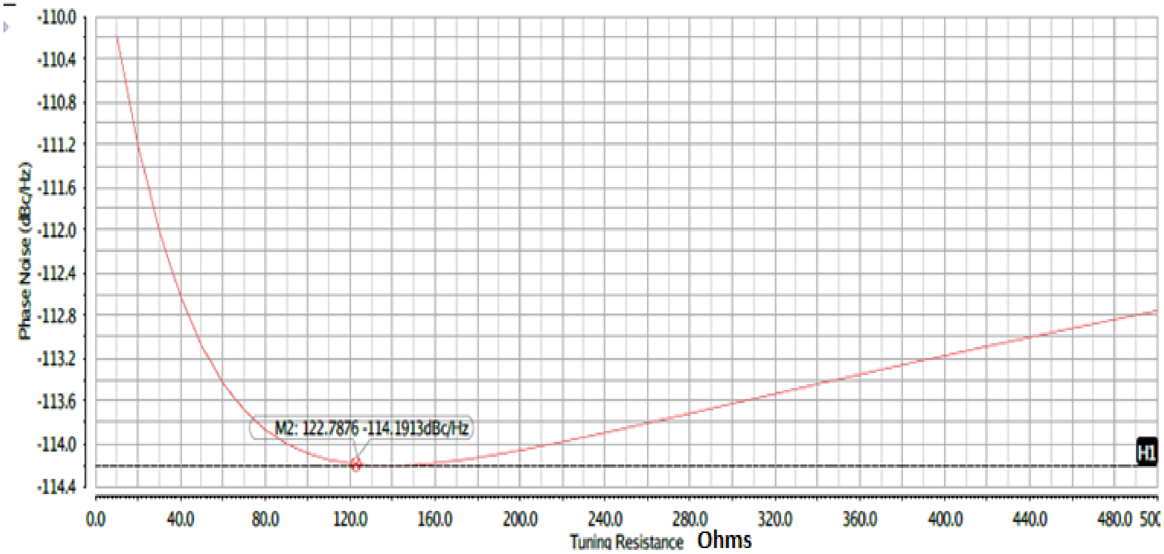


Figure 37. Phase Noise versus Tuning Resistance.

- The optimal tuning resistance for lowest phase noise is 122.7 Ohms for PN=-114.19 dBc/Hz

5.10 Inductor Characterization

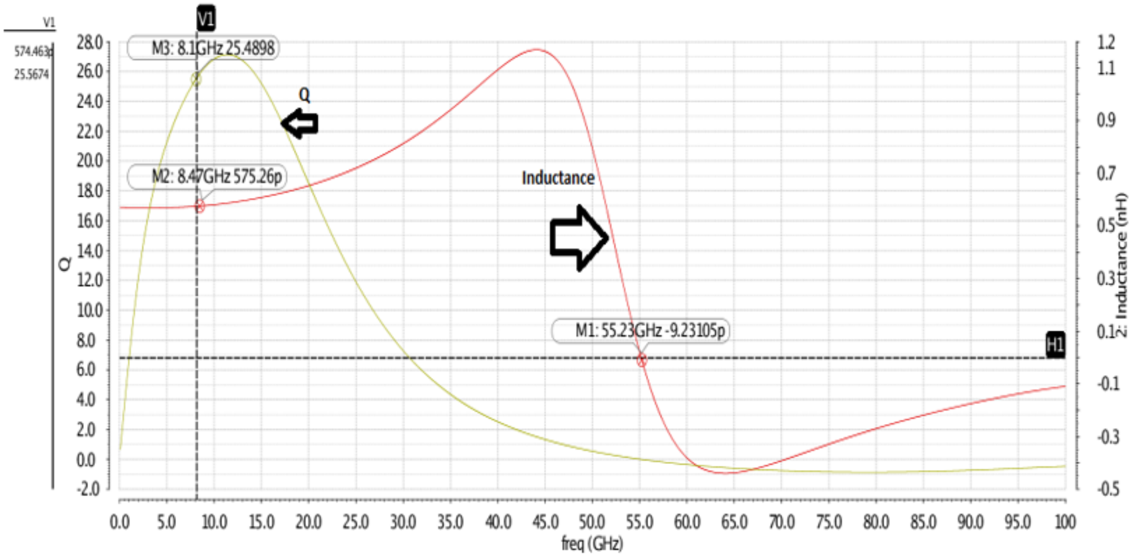
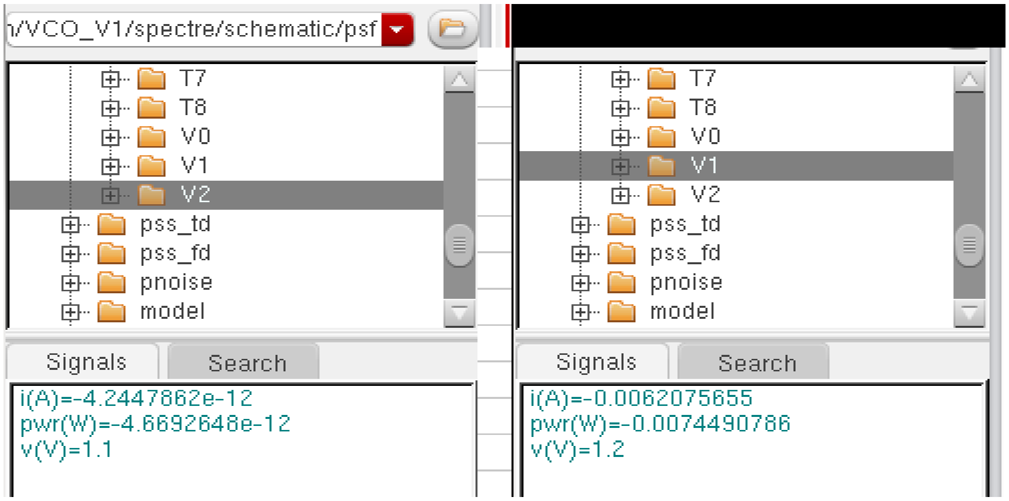


Figure 38. Inductor Characterization.

Design parameters	**Tank inductor**
Inductance	**574 pH per side**
Outer dimension	160 um
Meta width turns n	10 um
Number of turns	1.5
Space dimension s	5um
Self-resonance frequency	55 Ghz
Quality factor Q	23

5.11 Power Dissipation and Optimization

- Simulation shows the total power dissipated (static power)



```
i(A)=-4.2447862e-12
pwr(W)=-4.6692648e-12
v(V)=1.1
```

```
i(A)=-0.0062075655
pwr(W)=-0.0074490786
v(V)=1.2
```

- From the simulation, the power is 7.4490 mW (from the primary voltage source VDD).
- The power dissipation of each stage of the oscillator:
  - Buffer stage: Buffer current=1.5 mA, Bias Voltage =1.2 V $\rightarrow P = VI = 1.2 * 2mA = 2.16\ mW$ but simulation shows that the power dissipated is 0.95704 mW
  - Current source stage: I bias=1 mA $\rightarrow P = VI = 1.2 * 1mA = 1.2\ mW$, but simulation shows that the power is 0.621141 mW.
  - Tuning voltage stage: for the nominal frequency 8.05GHZ, V_{tune}=860mA but the current is in the order of pA, thus can be ignored.

Table 8. Dissipated power per component.

Stage	**Component**	**Dissipated Power (simulation result)**
Buffer Stage	T_0 T_2 T_3	0.8904 mW each
Buffer stage	T_7 T_8	0.9269 mW each
Current Mirror stage	T_4	0.48511mW
Current Mirror Stage	T_1	0.3975mW
VCO	T_5 T_6	0.240133 mW each
Tuning stage	R_{tune}	7.20728e-21 mW
Buffer Current Source	I_{buf}	0.95704 mW
Bias Current Source	I_{bias}	0.621140 mW
	Total	**7.4659 mW**

- The power dissipated by inductors and capacitors is roughly the difference between the total power and the total in the previous table: 7.4490- (7.4659-0.24-0.9269) = 1.15 mW. Taking into consideration that transistors T_5 and T_6 operate in complementary mode (if the current in T_5 is high, current in T_6 is low and vice versa, same behavior for buffer transistors T_7 and T_8).

- The dynamic power due to capacitors charging and discharging is approximately 1.15 mW.
- The power at the turning resistance is small. The Buffer stage transistors (T_0 T_2 T_3 T_7 T_8) are most component consuming power.

5.12 Final Design and List of Components

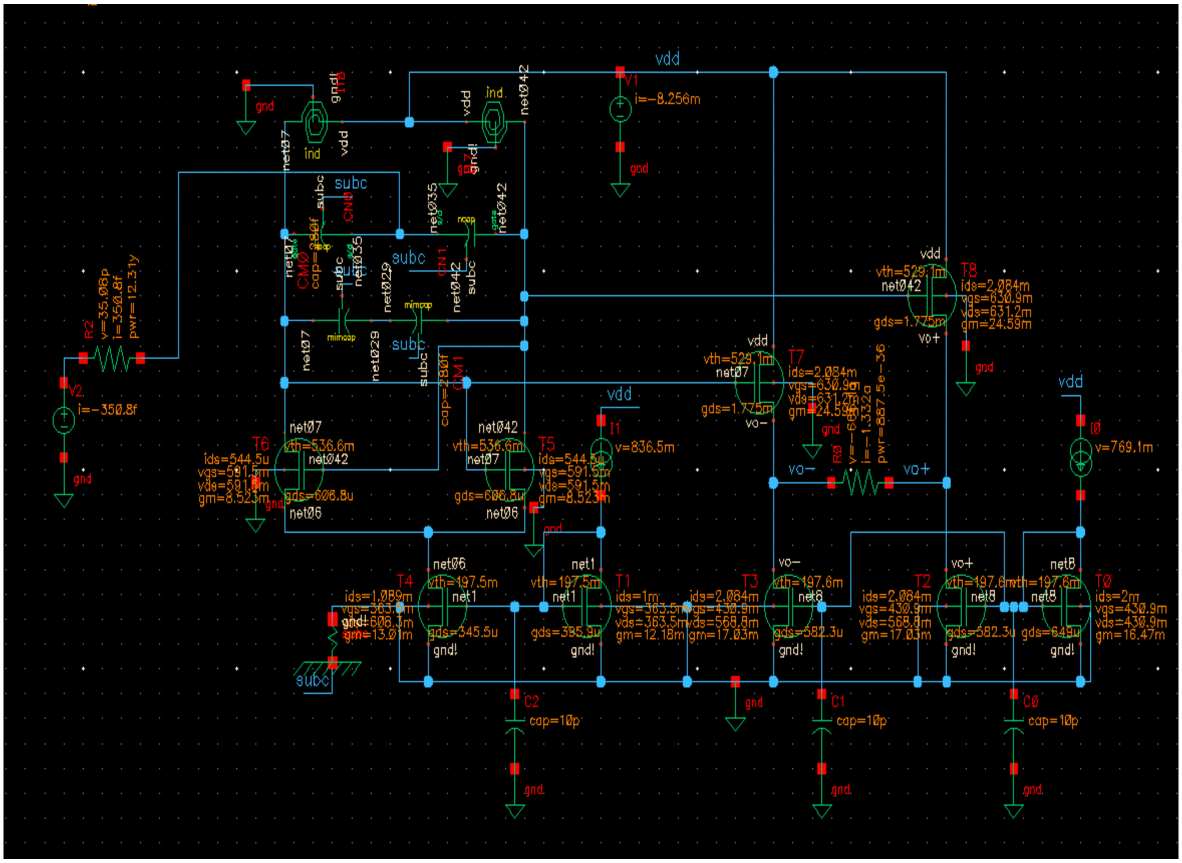


Figure 39. Final design.

- Transistors size in the final design are provided in table 9:

Table 9. Transistors ‘size.

Current source transistors	100uM (50 fingers/2uM), L=600nM
Bias Current	1 mA
Buffer Current	2mA
Cross-coupled transistors	30 uM (15 fingers/2uM)
Buffer Transistors	60 uM (30 fingers/2uM)

5.13 Conclusion

Throughout the physical design of the VCO, we have identified three independent variables that define the behavioral model which are K_{vco}, phase noise (PN) and frequency variations due to the tuning voltage and power pushing. Spice simulations of these variables will be used as bed test to validate behavioral models.

Chapter 6

Behavioral model of an analog VCO

In this chapter, we present the theoretical model and experimental results used to implement the behavioral model following the methodology presented in chapter 4. As it is going to be detailed, the behavioral model is analyzed in the time domain as well in the frequency domain.

6.1 Phase/frequency modeling

Let's start by deriving the phase equation. Angular frequency is the derivation of phase with respect to time.

$$\omega = \frac{d\emptyset}{dt} \qquad (1)$$

Then utilizing integration, the phase is given by equation 2.

$$\emptyset = \int \omega dt + \emptyset_0 \qquad (2)$$

From the analog simulation, the plot of frequencies with the variation of the tuning voltage shows a linear region around the nominal frequency. At that linear region, we derive the linear equation of the VCO, equation (3) which defines the frequency response of the VCO.

$$\omega_{out} = \omega_0 + K_{vco} V_{cont} \qquad (3)$$

Replacing equation (3) in the general sinusoidal waveform output of a VCO, equation (4) is re-written in equation (5) which describes the time response of the VCO

$$V_{out}(t) = V_m \cos(\emptyset(t)) \qquad (4)$$

$$V_{out}(t) = V_m \cos\left(\int \omega_{out} dt + \emptyset_0\right) = V_m \cos(\int \omega_0 + K_{vco} dt + \emptyset_0)$$

$$V_{out}(t) = V_m \cos(\omega_0 t + K_{vco} \int V_{cont}\, dt + \emptyset_0) \qquad (5)$$

For t_0=0 $V_{out}(t) = V_m \cos(\omega_0 t + K_{vco} V_{cont} t + \emptyset_0) = V_m \cos((\omega_0 + K_{vco} V_{cont})t + \emptyset_0)$

6.2 Noise Modeling

Noise is a random process, ie. the value of noise cannot be predicted at any time instant even the previous values are known. To this end. The concept of average power is widely used in quantifying noise as it is [16] as most sources of noise exhibit a constant average power.

The average power delivered by a periodic voltage $v(t)$ to a load resistance R_L is given by:

$P_{av} = \frac{1}{T}\int_{-T/2}^{T/2} \frac{v^2(t)}{R_L} dt$ measured in watts.

For a random signal, the average power is defined over a long-time interval and is given by:

$P_{av} = \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} \frac{x^2(t)}{R_L} dt$ where $x(t)$ is the instantaneous amplitude.

This formula is simplified to express the average power independently for the load resistance as.

$P_{av} = \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} x^2(t)\, dt$ expressed in V^2

Another figure of merits in characterizing noise the power spectral density (PSD). As such, the average power is defined regarding the frequency spectrum of noise, i.e., How much power carried at each frequency. The PSD of a noise waveform is defined as the average power carried by the noise waveform in a one-hertz bandwidth around a frequency f. PSD is measured in watts per Hertz (W/Hz).

The usual procedure to analysis the noise in a circuit as described in [16] is summarized as follows:

- Identifying the sources of noise and the spectrum of each.
- Find the transfer function from each noise source to the output.

- Utilize the theorem $S_Y(f) = S_x(f)|H(f)|^2$ where H(f) is the transfer function of a linear time-invariant system.
- Add all the output spectra, paying attention to correlated and uncorrelated sources.

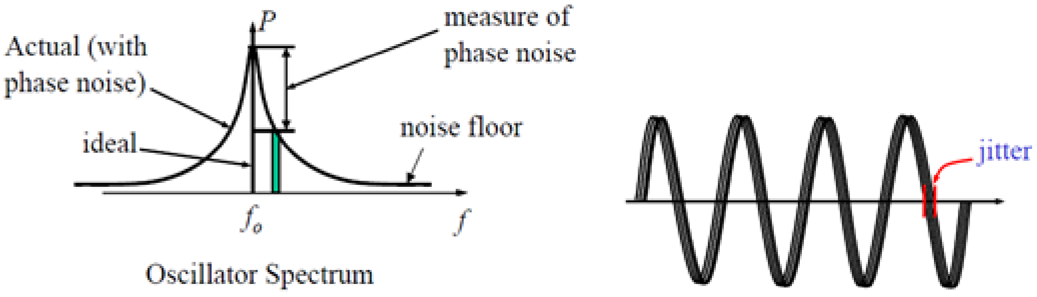
From the noise analysis in chapter 5, figure 34, the main contributors of the noise in the VCO are the flicker noise of the transistors and thermal noise of the tuning resistor and gate resistances. However, the noise analysis does not include external sources such as tuning voltage, power supply, and PVT variations.

6.3 Modeling noise approaches

There are two approaches to modeling the noise. The first approach models the effect of noise on the phase of the outputs of circuits. Phase models are linear and analyzed efficiently in the frequency domain. On the other hand, the second approach formulates the noise model in terms of voltages. Voltage noise models are in the time domain, nonlinear and can be refined for implementation.

6.3.1. Phase Noise

One of the most critical metrics of an analog VCO is the phase noise or the time jitter in the time domain.



a) Phase noise.

b) Time jitter.

Figure 40. Phase noise and Time Jitter.

The phase noise defines the variation of the output phase due to significant noise contributors:

- Variations of the tuning voltage
- Variations of the power line.
- Thermal noise of the resistors
- Process, Voltage and Temperature PVT variations.
- Parasitic elements
- Channel noise, flicker noise of the transistors

Lesson's oscillator model equation

The general equation for Phase noise due to all sources of noise: thermal noise, flicker noise, and channel noise are given by equation 6 [15]. This equation assumes a noiseless stable tuning voltage. This equation is an analytical approximation that provides an estimation of the phase noise close to simulation results provided in chapter 5.

$$\boldsymbol{PN} = \left(\frac{\boldsymbol{A\omega_0}}{\boldsymbol{2\varphi\Delta\omega}}\right)^{\mathbf{2}}\left(\frac{\boldsymbol{FKT}}{\boldsymbol{2P_s}}\right) \tag{6}$$

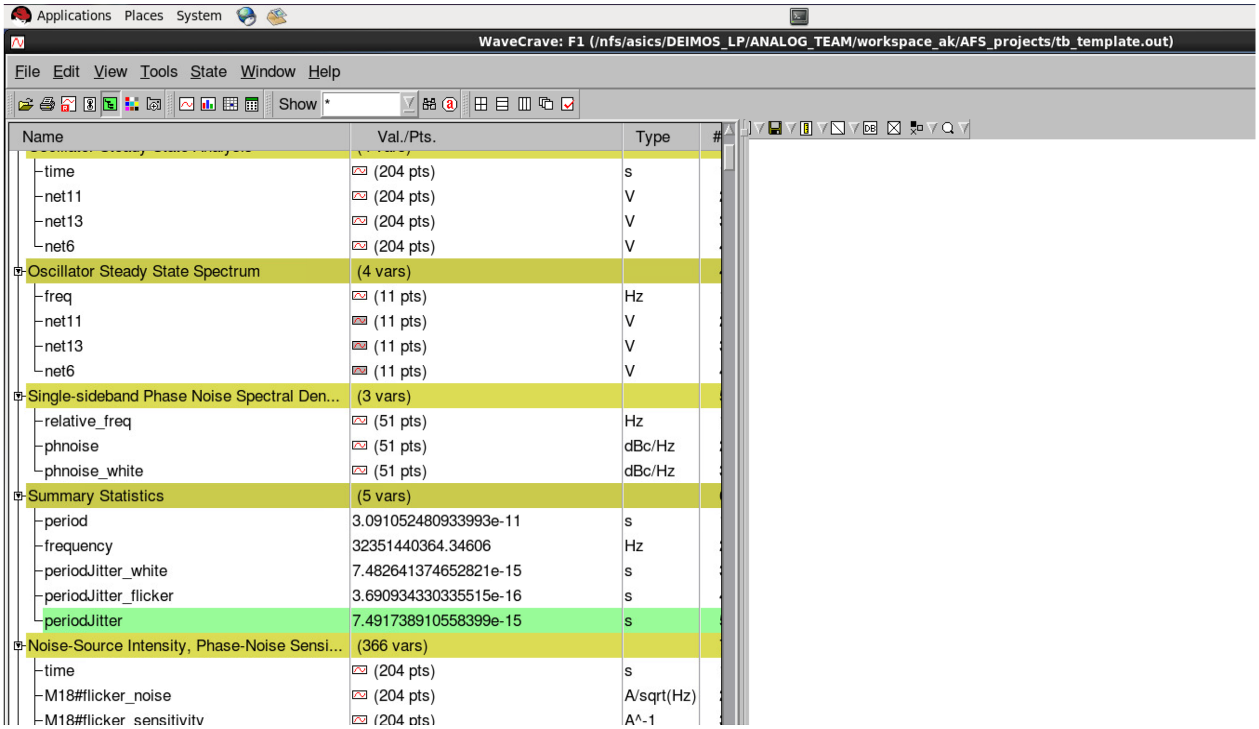
General equation of Phase noise

Another analytical approximation is the Lessons, oscillator model equation 7 [15], which this time assumes a variable tuning voltage $\mathrm{V}_{\boldsymbol{cont}} = V_m cos(\Delta\omega t)$. This equation could be used to model the PN response to VDD variations, ignoring other sources of noise.

$$PN = \left(\frac{V_m K_{vco}}{2\Delta\omega}\right)^2 \tag{7}$$

6.3.2 Time Jitter

Standards solutions for determining oscillator phase noise, timing jitter, and total noise are inaccurate because they make linear approximations and assume that the phase error is a small signal phenomenon. On the other hands, AFS tools and technique proceed to analyze phase noise at the device level and based on stochastic nonlinear engine, AFS run a nonlinear analysis without making any linear approximations. In addition, the analysis provides a full report on all noise contributors and their contributions as a time function.


Applications Places System
WaveCrave: F1 (/nfs/asics/DEIMOS_LP/ANALOG_TEAM/workspace_ak/AFS_projects/tb_template.out)
File Edit View Tools State Window Help
Show *

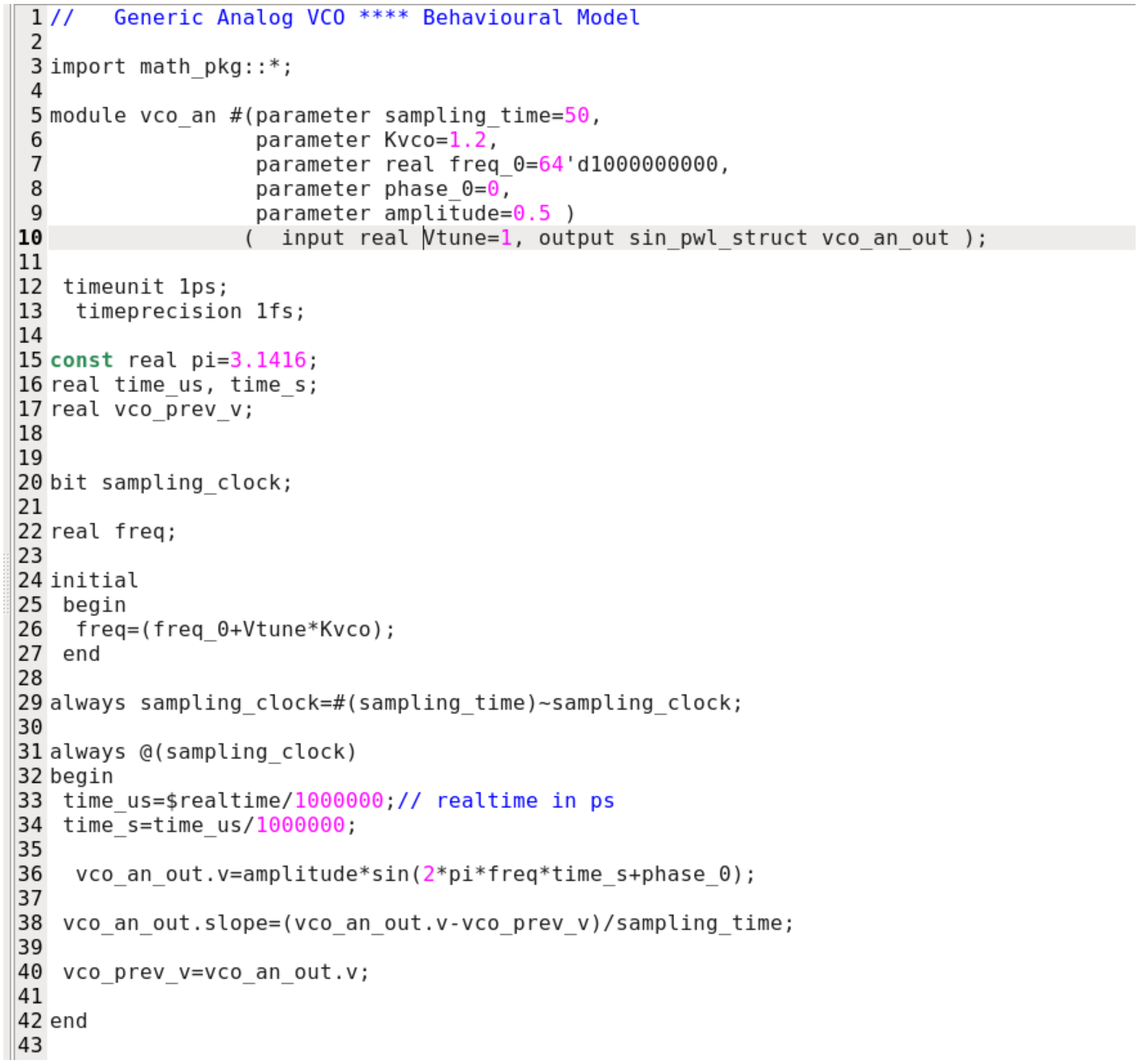
Name	Val./Pts.	Type	#
time	(204 pts)	s	
net11	(204 pts)	V	
net13	(204 pts)	V	
net6	(204 pts)	V	
Oscillator Steady State Spectrum	(4 vars)		
freq	(11 pts)	Hz	
net11	(11 pts)	V	
net13	(11 pts)	V	
net6	(11 pts)	V	
Single-sideband Phase Noise Spectral Den...	(3 vars)		
relative_freq	(51 pts)	Hz	
phnoise	(51 pts)	dBc/Hz	
phnoise_white	(51 pts)	dBc/Hz	
Summary Statistics	(5 vars)		
period	3.091052480933993e-11	s	
frequency	32351440364.34606	Hz	
periodJitter_white	7.482641374652821e-15	s	
periodJitter_flicker	3.690934330335515e-16	s	
periodJitter	7.491738910558399e-15	s	
Noise-Source Intensity, Phase-Noise Sensi...	(366 vars)		
time	(204 pts)	s	
M18#flicker_noise	(204 pts)	A/sqrt(Hz)	
M18#flicker_sensitivity	(204 pts)	A^-1	

Figure 41. Fast spice analysis of Time Jitter of the physical model.

Once the periodic time jitter is provided by the simulator Fastspice, the frequency response of the behavioral model is adjusted by introducing delays in the frequency response.

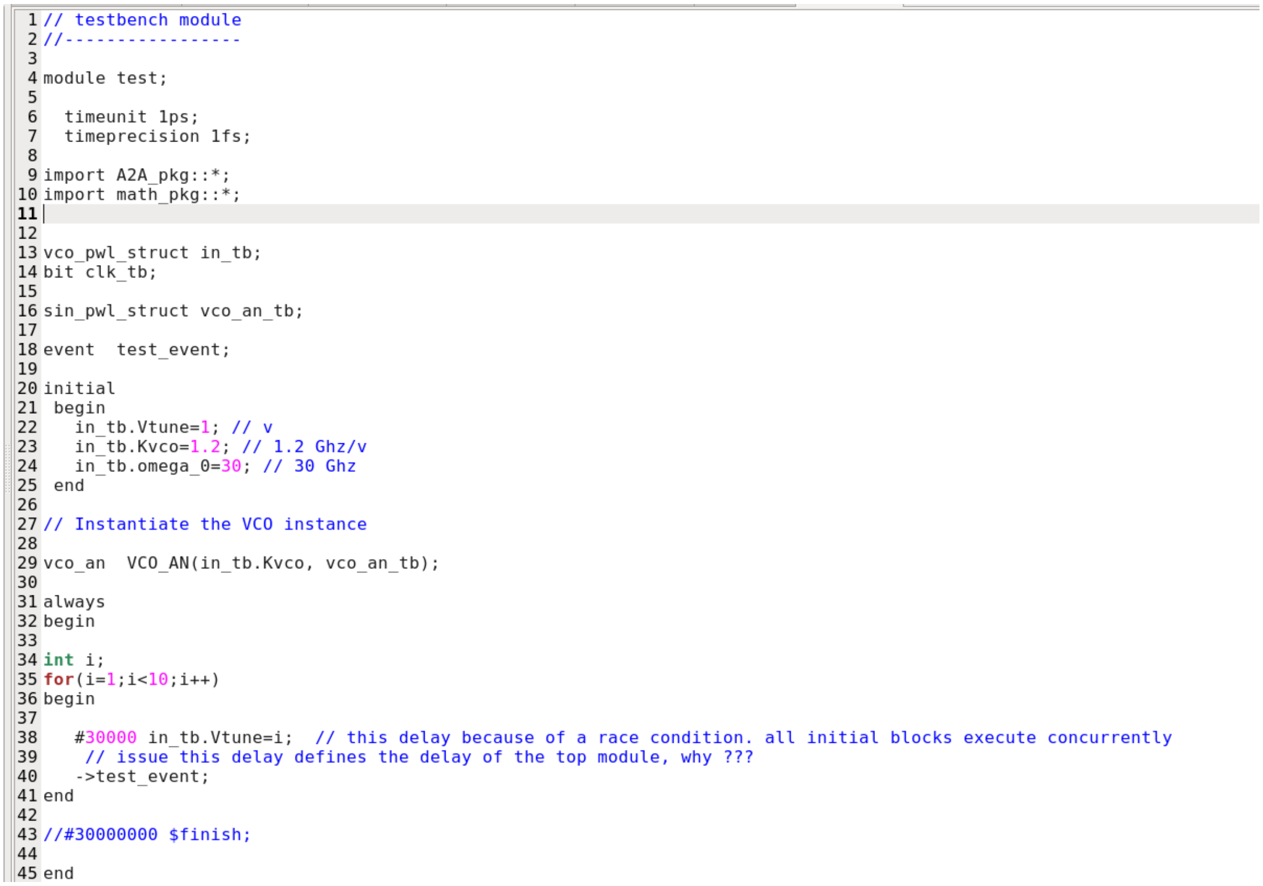
6.4 System Verilog implementation

The following System Verilog code, figure 42, is a snapshot of the HDL code implemented to simulate the frequency output of a VCO. The arguments of the code are extracted from the simulation of the physical model using Cadence/ Spectre analog simulator.



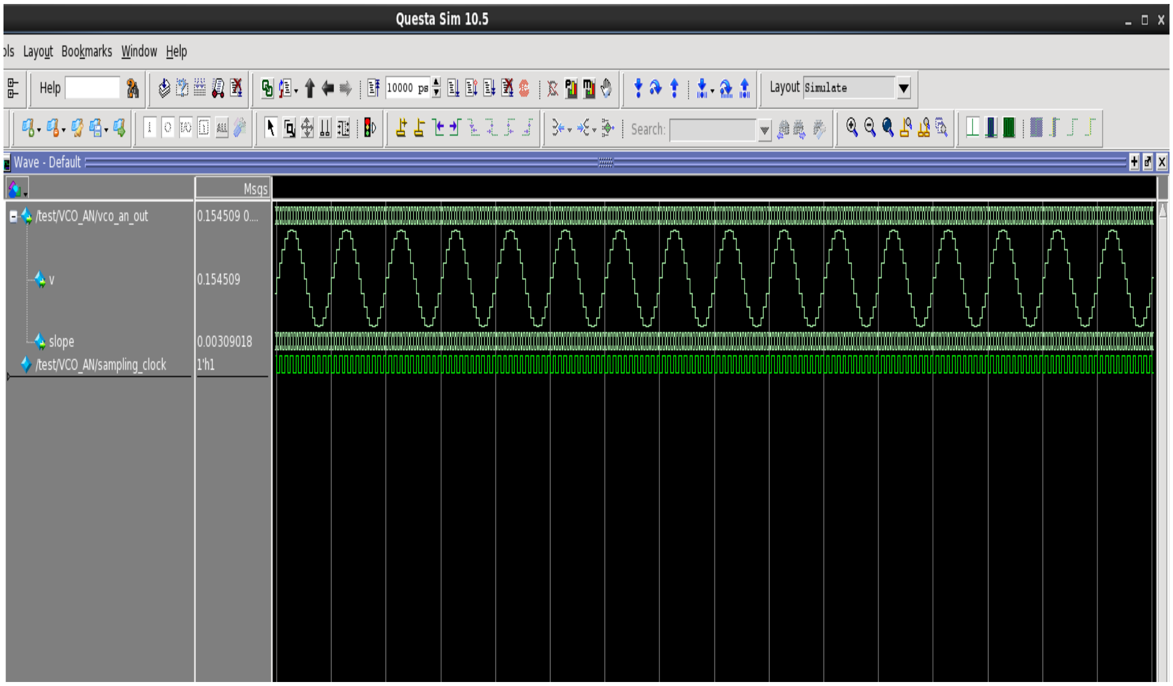
```
//   Generic Analog VCO **** Behavioural Model

import math_pkg::*;

module vco_an #(parameter sampling_time=50,
                parameter Kvco=1.2,
                parameter real freq_0=64'd1000000000,
                parameter phase_0=0,
                parameter amplitude=0.5 )
               (  input real Vtune=1, output sin_pwl_struct vco_an_out );

 timeunit 1ps;
  timeprecision 1fs;

const real pi=3.1416;
real time_us, time_s;
real vco_prev_v;


bit sampling_clock;

real freq;

initial
 begin
  freq=(freq_0+Vtune*Kvco);
 end

always sampling_clock=#(sampling_time)~sampling_clock;

always @(sampling_clock)
begin
 time_us=$realtime/1000000;// realtime in ps
 time_s=time_us/1000000;

  vco_an_out.v=amplitude*sin(2*pi*freq*time_s+phase_0);

 vco_an_out.slope=(vco_an_out.v-vco_prev_v)/sampling_time;

 vco_prev_v=vco_an_out.v;

end

```

Figure 42. VCO behavioral model.



```
// testbench module
//-----------------

module test;

  timeunit 1ps;
  timeprecision 1fs;

import A2A_pkg::*;
import math_pkg::*;


vco_pwl_struct in_tb;
bit clk_tb;

sin_pwl_struct vco_an_tb;

event  test_event;

initial
 begin
   in_tb.Vtune=1; // v
   in_tb.Kvco=1.2; // 1.2 Ghz/v
   in_tb.omega_0=30; // 30 Ghz
 end

// Instantiate the VCO instance

vco_an  VCO_AN(in_tb.Kvco, vco_an_tb);

always
begin

int i;
for(i=1;i<10;i++)
begin

   #30000 in_tb.Vtune=i;  // this delay because of a race condition. all initial blocks execute concurrently
    // issue this delay defines the delay of the top module, why ???
   ->test_event;
end

//#30000000 $finish;

end
```

Figure 43. Testbench Verilog block.



	Msgs
/test/VCO_AN/vco_an_out	0.154509 0....
v	0.154509
slope	0.00309018
/test/VCO_AN/sampling_clock	1'h1

Figure 44. Behavioral model output waveform.

6.5 Frequency spectrum of the output waveform

To analyze the frequency components of the output, we use the Laplace transformation.

Let's consider the signal output displayed in figure 42. At each sampling interval, the output constitutes of an amplitude and a slop. If we consider the constant amplitude and the slop as a ramp function. We derive the equation of the ramp shown in figure 43.

$x(t) = slope * (t - t_1) + v_1 : \ t \ varies \ from \ t_1 \ \ to \ t_2$

Laplace transformation is defined as $L(X(t)) = \int_{-\infty}^{\infty} x(t)e^{-st} \, dt$

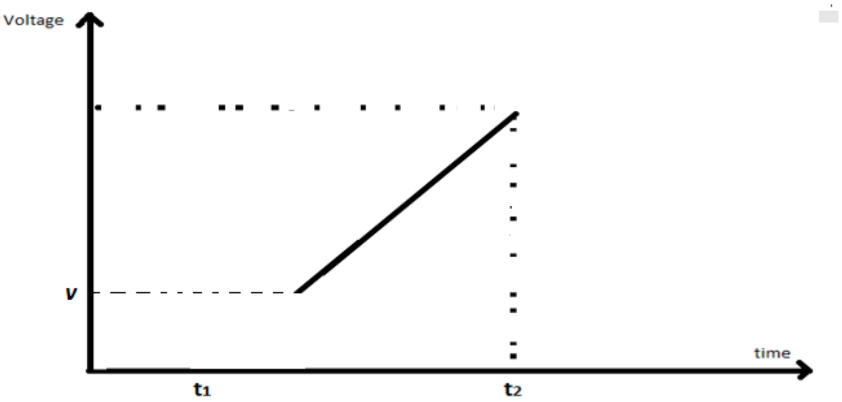


Figure 45. Step output represented as a ramp function.

$$\boldsymbol{L\big(X(t)\big) = slope * \frac{1}{s^2} e^{-t_1 s} + \frac{v_1}{s}} \qquad (8)$$

Bilinear Transformation

The bilinear transformation is commonly used in designing digital filters that preserves the frequency response and stability of their equivalent analog filters [17]. The bilinear transformation, equation, allows transforming the time continuous signal S-representation into the time discrete Z-representation.

$$\boldsymbol{s = \frac{2}{T_s} * \frac{1-z^{-1}}{1+z^{-1}}} \qquad (9)$$

6.6 Layout consideration in the behavioral model

A typical layout of a VCO is provided in figure 46. A layout is a set of masks used during the fabrication process. The only stage where PVT are considered is during Layout. In this section, we try to provide reflection on how a behavioral model would be augmented to include PVT variations and their effects on the frequency output of a physical VCO. It is important to mention that we rely mainly on our experience in doing layouts for Analog, digital, mixed signals and high-speed designs at Ciena. As we stated from the beginning of this research work, the primary goal is to provide a behavioral model that is close enough to a physical model. Mainly, from our experience, performances of Layout may diverge from performances simulated based on schematics.

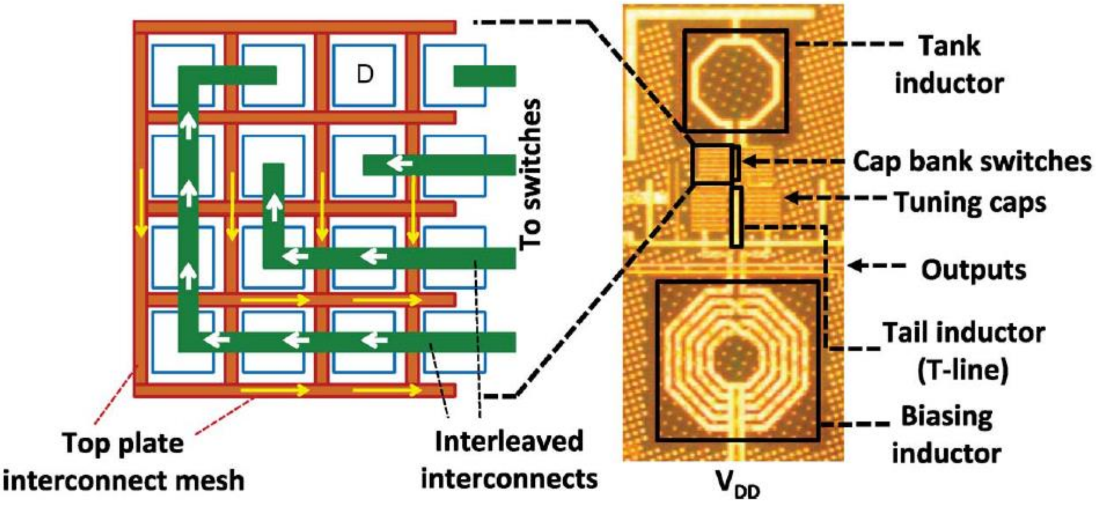


Figure 46. Typical VCO layout, adapted from [18].

6.6.1 Process variations (P)

Q. Kun in his Phd dissertation [19] provided the list of process variations impacting physical properties of devices and circuits and consequently their electrical characteristics.

- Lithographic variations

- line-edge roughness
- random dopant fluctuations
- layout-dependent
- stress variations, rapid thermal annealing (RTA)
- temperature induced variations
- well-proximity effects (WPE)
- deposition and growth processes
- chemical mechanical polishing (CMP).

The above process variations cause variations in device parameters [19] such as dimensions, oxide thickness, doping concentrations, diffusion depth, and mobility.

Corner models and Monte Carlo models are the two types of statistical device models used by designers to predict the impact of silicon manufacturing variations on device parameters.

It is critical to understand and quantify process variations and impact on designs to avoid discrepancies between designs and manufactured chips. Our goal here is throughout simulations of using both models, extract and interpolates parameters to be used to augment the behavioral model to reflect the impact of process variations of the performances of designs.

6.6.2 Voltage variations(V)

Resistors and capacitors in a CMOS process have changing values with temperature and voltage [20]. The change is expressed $ppm/°_C$ (parts per million per degree C). The $ppm/°_C$ is equivalent to a multiplier of $10^{-6}/°$ C.

The voltage dependence of resistors is expressed by the first order parameter $\mathrm{VCR} = \frac{1}{R} * \frac{dR}{dT}$, so $R(V) = R_{V_0}(1 + VCR * V)$ (10)

Similarly, the variation of a capacitor is expressed by first-order parameter $VCC = \frac{1}{C} * \frac{dC}{dV}$, so $C(V) = C_{V_0}(1 + VCC * V)$
(11)

6.6.3 Temperatures variations(T)

The variation of the resistance and capacitance are expressed respectively by first-order parameters [20] $TCR = \frac{1}{R} * \frac{dR}{dT}$, so $R(T) = R_{T_0}[1 + TCR(T - T_0) + TCR_2(T - T_0)^2]$ (12)

And $TCC = \frac{1}{C} * \frac{dC}{dT}$, so $C(T) = C_{T_0}[1 + TCC(T - T_0)]$ (13)

Unfortunately, it is not practical to use this equation in the behavioral model as resistors and capacitors are not visible at this level of abstraction; therefore, the high-level model is needed.

6.7 Behavioral model of an analog PLL

Phase-locked loops (PLL) are used in many applications such as providing clocking in digital systems like CPUs, data converters, in wireless applications such as cellular transceivers, Wi-Fi transceivers, TV tuners, and RF receivers [36].

PLL performances are stability, low-noise, a tunable signal with fast locking times, low power and low time jitter or phase noise. Figure 47, shows the block diagram for a fractional-N-PLL.

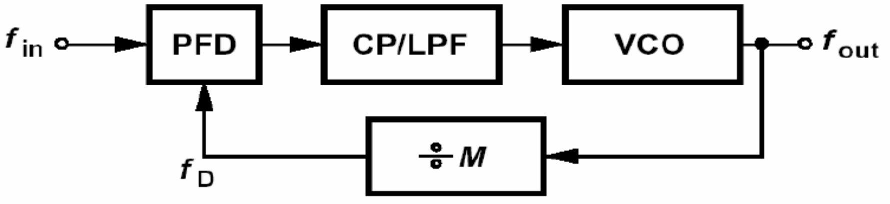


Figure 47. Analog PLL bloc diagram [36].

Each block contributes to the output jitter/phase noise as a function of its noise generation and noise transfer function to the PLL output. Output jitter/phase noise is also a function of process, voltage, and temperature (PVT) variations. Device noise, post-layout parasitic, process variability, and device mismatch significantly impact PLL performance at nanometer technology nodes.

Performing transistor-level, closed-loop PLL verification has been impossible or impractical due to traditional SPICE and RF simulator performance and capacity limitations. Thus, block-level verification and behavioral models are essential to validate the performances of PLLs. Block-Level simulations are carried to estimate the contribution of each block with respect to specific metrics. Afterword, the block-level behavioral models are used to simulate the full model of PLL.

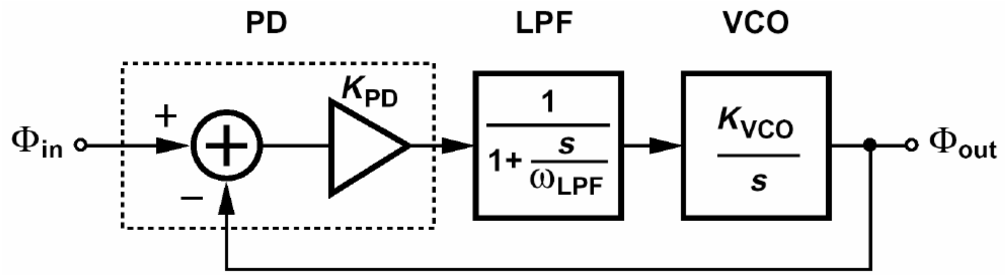


Figure 48. Contribution of each bloc to PLL phase noise [36].

6.7 Conclusion and Future Perspectives

Verification of high-precision nanometer analog and mixed-signal circuits has become very challenging because traditional SPICE simulators do not have the performance and capacity required to support the circuit complexity. As a result, traditional SPICE is not suitable for large analog and mixed-signal circuits as the runtime is too long, or the simulation is not possible, and the accuracy is compromised.

To overcome this limitation and address the need of speed and accuracy of high speed, nanometer designs, behavioral models are developed.

In this research project, a reusable and flexible framework for developing behavioral model is proposed. As a proof concept, we have designed a physical model of an analog voltage-controlled oscillator (VCO), then, developed a behavioral model following the proposed framework to show the feasibility of the proposed framework. After that, we provide insights on how to integrate the VCO behavioral models in an analog phase loop locker(PLL). Finally, based on the Layout of the VCO, we provide a short reflection on how to augment the behavioral model to include temperature, power and process variation.

In future work, integration of the framework within the standard Universal Verification Methodology (UVM) is an unavoidable step as the UVM is widely accepted and used in industry. This integration will mainly require re-organizing the System-Verilog code into an oriented objects hierarchy according to the concepts and principals of UVM.

Bibliography

[1] P. Gunupudi, Class Lecture, Topic:" *Introduction to Electronic Design Automation Algorithms and Techniques*."ELEC5402 ,Department of Electronics, Carleton University. Ottawa, Canada. [Online]. Available: http://doe.carleton.ca/~pavan/Public/Courses. [Accessed March 2018].

[2] D. A. Tannir, "*Moments Based Analysis of Intermodulation Distortion in Radio Frequency Circuits*, "Ph.D. dissertation, Department of Electrical and Computer Engineering, McGill University, Montreal, Canada, 2010.

[3] S. Liao, "Verilog Piecewise Linear Behavioral Modeling for Mixed-signal Validation, "Ph.D. Dissertation, Stanford University, 2014.

[4] D. Ghai and N. Jain, "*All-Digital Phase Locked Loop (ADPLL)*-A Review," *International Journal of Electronics and Computer Science Engineering-ISSN-2277-1956,* 2013.

[5] P. O'Riordan, "Mixed-signal verification methodology," *Mixed-signal methodology guide. San Jose: Cadence Design Systems. Google Scholar,* 2012.

[6] A. B. Mehta, "Analog/Mixed Signal (AMS) Verification," in *ASIC/SoC Functional Design Verification: A Comprehensive Guide to Technologies and Methodologies*, ed Cham: Springer International Publishing, 2018, pp. 255-271.

[7] S. Little, D. Walter, C. Myers, R. Thacker, S. Batchu, and T. Yoneda, "Verification of analog/mixed-signal circuits using labeled hybrid Petri nets," *IEEE Transactions on computer-aided design of integrated circuits and systems,* vol. 30, pp. 617-630, 2011.

[8] Y. Peng and M. Greenstreet, "Integrating SMT with Theorem Proving for Analog/Mixed-Signal Circuit Verification," Cham, 2015, pp. 310-326.

[9] S. Balasubramanian and P. Hardee, "*Solutions for mixed-signal SoC verification using real number models,*" *Cadence Design Systems,* 2013.

[10] S. Liao, "*Verilog Piecewise Linear Behavioral Modeling For Mixed-Signal Validation*, "Ph.D dissertation, Department of Electrical Engineering, Stanford University, California, USA. , May 2014.

[11] M. Horauer, D. Widhalm, S. Tauner, and S. Mirtl, "*Verification challenges of complex system-on-chip devices,*" *e & Elektrotechnik und Informationstechnik,* vol. 132, pp. 269-273, September 01 2015.

[12] Y. Li, R. Iskander, F. Javid, and M.-M. Louërat, "A Design and Verification Methodology for Mixed-Signal Systems Using SystemC-AMS," Cham, 2014, pp. 89-108.

[13] J. W. M. Rogers and C. Plett, *Radio Frequency Integrated Circuit Design*: Artech House, 2010.

[14] B. Razavi, *Design of Analog CMOS Integrated Circuits*: McGraw-Hill, 2016.

[15] V. Kratyuk, P. K. Hanumolu, U.-K. Moon, and K. Mayaram, "A design procedure for all-digital phase-locked loops based on a charge-pump phase-locked-loop analogy," *IEEE Transactions on Circuits and Systems II: Express Briefs,* vol. 54, pp. 247-251, 2007.

[16] B. Sadhu et al., *"A linearized, low-phase-noise VCO-based 25 GHz PLL with autonomic biasing*," IEEE J. Solid-State Circuits, vol. 48, no. 5, pp. 1138–1150, May 2013

[17] Qian, Kun. '*Variability modeling and statistical parameter extraction for CMOS devices*. PHD Dissertation. UC Berkeley, 2015.

[18] R. J. Baker, *CMOS: circuit design, layout, and simulation* vol. 1: John Wiley & Sons, 2008.

[19] K. Kundert. *Modeling jitter in PLL-based frequency synthesizers.* Available From www.designers-guide.org/Analysis. Accessed on September 2018.

[20] D. Ulus, A. Sen, and F. Baskaya, "*Integrating Circuit Analyses for Assertion-Based Verification of Programmable AMS Circuits*," in *Languages, Design Methods, and Tools for Electronic System Design: Selected Contributions from FDL 2013*, M.-M. Louërat and T. Maehne, Eds., ed Cham: Springer International Publishing, 2015, pp. 67-84.

[21] D. S. Vidhya and M. Ramachandra, "*A Novel Design in Formal Verification Corresponding to Mixed Signals by Differential Learning,*" Cham, 2017, pp. 367-378.

[22] T. Wen and T. Kwasniewski, "*Phase noise simulation and modeling of ADPLL by SystemVerilog,"* in *Behavioral Modeling and Simulation Workshop, 2008. BMAS 2008. IEEE International*, 2008, pp. 29-34.

[23] D. Yang, "Jitter, Phase Noise and Spurs in Frequency Multiplying Delay-locked loops: A Simple Model and Analysis, "M.A. thesis. University of California, Los Angeles, 2015.

[24] Y. Karmous, "*Modelisation Verilog des circuits analogiques et mixetes a Hautes Vitesses pour la validation pre-silicon,*" M.A. thesis, Universite de Quebec en Outaouais, Gatineau, Canada. 2017.

[25] R. Achar, Class Lecture, Topic:" Signal and Power Integrity in High-Speed Designs."ELEC5401 , Department of Electronics, Carleton University. Ottawa, Canada., Dec.2017.

[26] S. Batchu, *Automatic extraction of behavioral models from simulations of analog/mixed-signal (AMS) circuits*: M.A. thesis. University of Utah, 2011.

[27] A. N. Fisher, D. Kulkarni, and C. J. Myers, "*A New Assertion Property Language for Analog/Mixed-Signal Circuits," in Languages, Design Methods, and Tools for Electronic*

System Design: Selected Contributions from FDL 2013, M.-M. Louërat and T. Maehne, Eds., ed Cham: Springer International Publishing, 2015, pp. 45-65.

[28] A. Fürtig, G. Gläser, C. Grimm, L. Hedrich, S. Heinen, H.-S. L. Lee*, et al.*, "*Novel metrics for Analog Mixed-Signal coverage*," in *Design and Diagnostics of Electronic Circuits & Systems (DDECS), 2017 IEEE 20th International Symposium on*, 2017, pp. 97-102.

[29] A. Fürtig and L. Hedrich, "*Formal Techniques for Verification and Coverage Analysis of Analog Systems,*" in *Formal System Verification: State-of-the-Art and Future Trends*, R. Drechsler, Ed., ed Cham: Springer International Publishing, 2018, pp. 1-35.

[30] P. Gang, "*Behavioral modeling and simulation of analog/mixed-signal systems using Verilog-ams,*" in *Information, Computing and Telecommunication, 2009. YC-ICT'09. IEEE Youth Conference on*, 2009, pp. 383-386.

[31] C. Gu, "*Algorithmic nonlinear macro modeling: Challenges, solutions, and applications in Analog/Mixed-Signal validation*," in *Custom Integrated Circuits Conference (CICC), 2013 IEEE*, 2013, pp. 1-8.

[32] C. V. Kashyap and C. S. Amin, "Raven: A tool for automatic generation of analog behavioral models from schematics," *Frontiers in Analog Circuit (FAC) Synthesis and Verification,* pp. 14-15, 2011.

[33] K. Lata and S. K. Roy, "Formal Verification of Analog and Mixed-Signal Designs Using SPICE Circuit Simulation Traces," *Journal of Electronic Testing,* vol. 29, pp. 715-740, October 01 2013.

[34] B. Li and V. D. Agrawal, "Applications of Mixed-Signal Technology in Digital Testing," *Journal of Electronic Testing,* vol. 32, pp. 209-225, April 01 2016.

[35] S. Little, D. Walter, K. Jones, C. Myers, and A. Sen, "Analog/mixed-signal circuit verification using models generated from simulation traces," *International Journal of Foundations of Computer Science,* vol. 21, pp. 191-210, 2010.

[36] C.-J. R. Shi, "Mixed-signal system-on-chip verification using a Recursively-Verifying-Modeling (RVM) methodology," in *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, 2010, pp. 1432-1435.