# BEHAVIORAL MODEL EQUIVALENCE CHECKING FOR LARGE ANALOG MIXED SIGNAL SYSTEMS

A Thesis

by

AMANDEEP SINGH

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 2011

Major Subject: Computer Engineering

Behavioral Model Equivalence Checking for Large Analog Mixed Signal Systems

# BEHAVIORAL MODEL EQUIVALENCE CHECKING FOR LARGE ANALOG MIXED SIGNAL SYSTEMS

A Thesis

by

AMANDEEP SINGH

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

| | |
|---|---|
| Chair of Committee, | Peng Li |
| Committee Members, | Jose Silva Martinez |
| | Duncan M. Walker |
| Head of Department, | Costas N. Georghiades |

May 2011

Major Subject: Computer Engineering

# ABSTRACT

Behavioral Model Equivalence Checking for Large Analog Mixed Signal Systems.

(May 2011)

Amandeep Singh, B.E., Punjab Engineering College, India

Chair of Advisory Committee: Dr.Peng Li

This thesis proposes a systematic, hierarchical, optimization based semi-formal equivalence checking methodology for large analog/mixed signal systems such as phase locked loops (PLL), analog to digital convertors (ADC) and input/output (I/O) circuits. I propose to verify the equivalence between a behavioral model and its electrical implementation over a limited, but highly likely, input space defined as the Constrained Behavioral Input Space. Furthermore, I clearly distinguish between the behavioral and electrical domains and define mapping functions between the two domains to allow for calculation of deviation between the behavioral and electrical implementation. The verification problem is then formulated as an optimization problem which is solved by interfacing a sequential quadratic programming (SQP) based optimizer with commercial circuit simulation tools, such as CADENCE SPECTRE. The proposed methodology is then applied for equivalence checking of a PLL as a test case and results are shown which prove the correctness of the proposed methodology.

## DEDICATION

to my friends and family

# ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Peng Li, for giving me the opportunity to work under his guidance for this thesis. I am grateful to him for his constant encouragement, guidance and support all the way through. I would also like to thank Dr. Jose Silva Martinez and Dr. Duncan M. Walker for agreeing to be my committee members. I also thank them for their interesting courses which have helped me immensely in working on this topic.

I would also like to thank my friends and colleagues and the department faculty and staff for making my time at Texas A&M University a great experience.

Finally, I thank my parents for their constant encouragement, support and confidence in my abilities.

# NOMENCLATURE

| | |
|---|---|
| AMS | Analog and mixed signal circuits |
| BIS | Constrained behavioral input space |
| $\Omega_{\mathrm{B}}$ | Behavioral signal domain |
| $\Omega_{\mathrm{E}}$ | Electrical signal domain |
| $f_{B\text{-}E}$ | One-to-many behavioral to electrical domain mapping function |
| $g_{E\text{-}B}$ | Many-to-one electrical to behavioral domain mapping function |

# TABLE OF CONTENTS

Page

CHAPTER Page

# LIST OF FIGURES

# LIST OF TABLES

# CHAPTER I

# INTRODUCTION

The recent advances in semiconductor technology and continued transistor scaling have allowed designers to integrate increasingly more functionality on the same chip. This has resulted in development of complex mixed signal system on chip (SoC) designs. Figure 1 below shows an example of a typical mixed signal SoC.

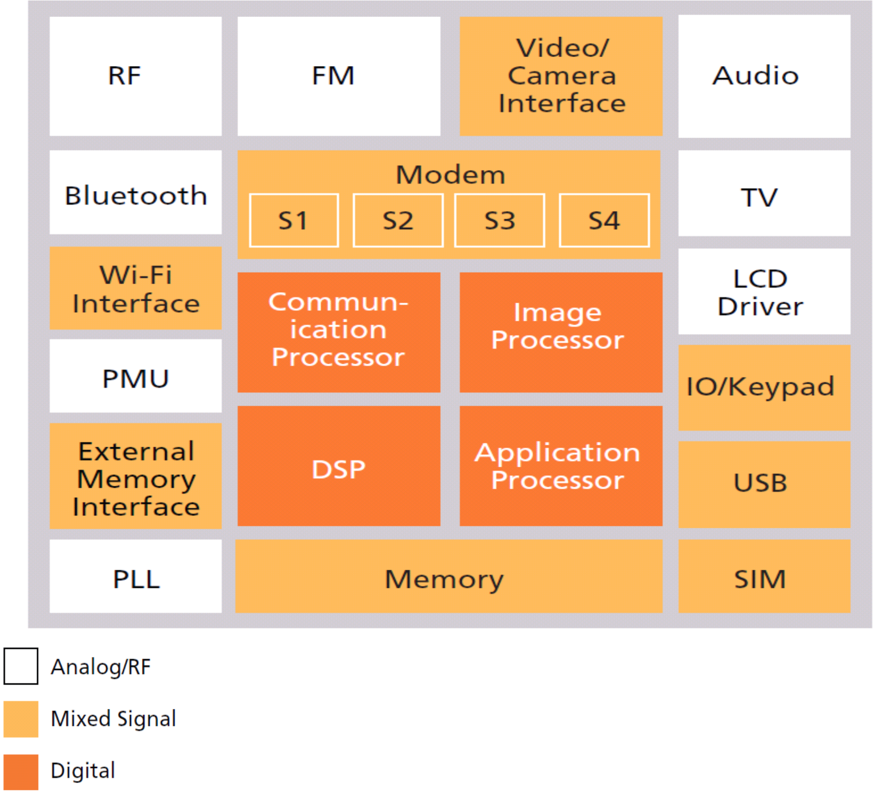


**Figure 1.** Typical mixed signal system on chip (SoC) [CADENCE 2009]

This thesis follows the style of *IEEE Transactions on Computer Aided Design.*

As shown in Figure 1, in addition to the core digital signal processors such as communication processors and image processors, a large fraction of a typical SoC consists of analog/mixed signal blocks such as phase locked loops, transceivers and I/Os for clock generation and interfacing with the external world. Further, even a typical microprocessor can no longer be assumed to be a purely digital chip as it also contains many mixed signal blocks such as phase locked loops, thermal sensors, voltage regulators and low dropout regulators (LDO). Thus, even a microprocessor IC is essentially a complex SoC.

On the one hand while the analog/mixed signal content in SoC has been increasing, the increasingly variable manufacturing processes, limited voltage headroom, and limited power budgets lead to increasingly complex analog/mixed signal circuits. Many computer-aided design (CAD) tools and methodologies have been developed in the recent past to overcome some of these design challenges. Hardware descriptive languages such as VHDL-AMS [1] and Verilog-AMS [2] have been developed to describe the behavior of analog/mixed signal circuits. Similarly, advances have also been made in analog synthesis and topology selection [3]. Also, significant advancements have been made in automatic layout generation for analog circuits. However, the increased design complexity necessitates the development of efficient verification methodologies for mixed signal systems to prevent costly design errors and reduce development time.

Typically the verification problem tries to answer the following question (Fig. 2): Given a set of specifications or a golden reference illustrating these, does the actual transistor level circuit design meet the required specifications across the entire feasible input range or not ?

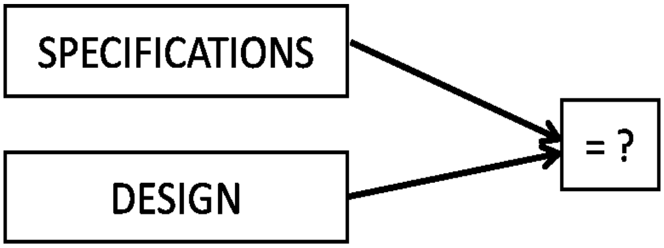


**Figure 2**. The verification problem

The current state-of-the-art verification tools and methodologies have enabled efficient verification of complex digital circuits with millions of gates; however, the same cannot be said for analog or mixed signal circuits. The verification of analog circuits is still largely done manually using SPICE level simulations and is highly dependent on the skills and intuition of the designer. This is a time consuming task. Further, this non-systematic manual verification process leaves many essential questions such as ‘which test cases to use?', ‘what is the verification coverage achieved?’ unanswered. In-fact, currently concepts such as ‘verification coverage’ are very vaguely defined for analog/mixed signal circuits. Furthermore, SPICE level verification for large systems involving a number of big mixed signal components such as phase locked loops (PLLs) and Analog-to-Digital Convertors (ADCs) involve huge computational complexity, which renders conventional simulation based manual verification methods almost impossible. The lack of formal verification for analog/mixed signal blocks often results in non-detection of functional errors in the design leading to re-spins and increase in time to market. Thus, automated design verification for analog/mixed signal systems is crucial.

Several methods have been proposed for formal verification of analog circuits [4-12]. These methods can be broadly categorized into two categories, equivalence checking and model checking. Equivalence checking compares the output of two different models for a given set of input conditions [4]. For analog circuits, the exact same magnitudes of current and voltage may not be attained, hence, an error bound is defined and the models are said to be equivalent if the error lies within this bound. In [4] the authors provide a good summary of the equivalence checking methods proposed till date. Model checking [4] involves representing the design to be verified in form of a transition system. The specifications of the design are translated to temporal logic formulas. State exploration algorithms are then used to verify if the specifications are satisfied or not. However, model checking algorithms [11] [12] have achieved limited success for formal verification of analog circuits. Most of the existing methods often require the conversion of a high-dimensional continuous state space to a large discrete equivalent so as to apply Boolean-like verification [7][8][10]. The resulting state explosion limits the application of these methods to toy circuits of very low dimensionality. Further, the inherent approximations in discretization can render these methods practically “informal”. Also, many of the proposed methods have limited practicality as they assume a linear behavior for the circuits under consideration [9].

Recently an interesting Boolean-satisfiability based approach has been proposed [3]. The methodology cleverly leverages recent advances in SAT engine for analog verification. However, it also suffers from scalability issues, as it is difficult to solve the satisfiability problem for large complex systems like phase locked loops. Further, the run time increases exponentially as the granularity of the discretized device I-V tables used to formulate the satisfiability problem decreases.

In this thesis we propose an optimization based, hierarchical behavioral model equivalence checking methodology that is not necessarily completely formal, but yet systematic and applicable to large designs such as PLLs,  ADCs and I/O's. We use behavioral modeling (e.g. Verilog AMS) as a system verification vehicle. The proposed methodology facilitates feasible behavioral model equivalence checking under the following system context. We assume that the desired system behaviors are "encoded" in a set of block-level behavioral models, or the reference system behavioral model (RSB). Hence, the desired system performance specifications are also reflected in the simulated performances of the RSB. A given detailed electrical (circuit) implementation, e.g., represented by a set of (extracted) block-level SPICE netlist, is checked (verified) against the RSB on an individual block basis. Either, the implementation is deemed as "equivalent", or the check is inconclusive due to the conservative nature of the check. In addition to the aforementioned equivalence checking against a given "golden" RSB, the proposed work also serves an intrinsically related purpose: compare an existing electrical-level design implementation against its corresponding behavioral model so as to provide guidance for behavioral modeling. The proposed methodology has several key characteristics:

- System-level behavioral simulations are used as a basis to derive a limited but sensible set of input stimuli for verification. Inherent abstraction in behavioral modeling, which contributes to the deviation of the behavioral model from its electrical counterpart, is specifically targeted in our verification; such modeling abstraction is mathematically characterized by defining two signal domains and mapping functions between them.
- Equivalence checking is formulated as a constrained optimization problem and solved by interfacing behavioral and SPICE-level simulators that contrast the behavioral model with the SPICE netlist.
- System equivalence checking is broken into individual block-level checks, and hence performed hierarchically; this makes the approach scalable for large designs.

# CHAPTER II

# PREVIOUS WORK

The increasing design complexity of analog/mixed signal system on chip together with the reduced time to market have necessitated the need for development of formal verification techniques for analog/mixed signal circuits. In this chapter few recent techniques proposed for verification of analog/mixed signal circuits are reviewed. A more thorough review of previously proposed techniques and methodologies is given in [4].

Formal verification techniques can be broadly classified into two types, theorem proving methods and automated state space exploration methods. The automated state-space exploration methods can further classified into either equivalence checking methods or model-checking methods. Each of these above techniques are reviewed in the following sections.

## II.A Theorem Proving Methods

Theorem provers prove design properties using formal deduction methods based on a set of inference rules [13]. Such methods have been widely used for verification of systems such as microprocessor design, cache coherence protocols and even for software verification. Recently, few works have been proposed to extend these techniques for use in verification of analog/mixed signal systems.

In [14] the authors use PVS theorem prover to prove the equivalence between the formal model of the structural description of a synthesized analog design extracted from the sized component netlist produced by the synthesis tool and the formal model extracted from the user given behavioral specification. However, the method is limited to formal verification of low-frequency or DC characteristics only. Further, the method is limited only for linear circuits or those whose behavior can be represented by piece-wise linear models only.

In [15] the authors extend predicate-logic based methods for specifying and verifying digital systems at analog levels. This method involves using predicates to characterize the behavior of analog components in terms of the voltages and currents at their terminals. An algorithm is proposed for checking the verification conditions and the same has been used to automatically verify simple digital gates. In [16] the authors propose a symbolic induction based verification strategy implemented using Mathematica for proving properties of analog/mixed signal designs.

In [17] a stochastic differential equation (SDE) based verification methodology using an automated theorem prover, MetiTarski, has been proposed. The proposed implementation models and verifies the analog design in the presence of noise and process variations. The proposed methodology is applied for verification of op-amp based integrator and band-gap reference circuits as test-cases. However, the proposed method has limited practical use because it requires the system of differential equations to be linear, or transformed into linear form so that closed form solutions for these differential equations can be evaluated easily.

As discussed above various theorem proving methods have been proposed recently for verification of analog/mixed signal circuits. However, these methods are still premature and considerable work needs to be done before any of these methods can be used for verifying all the different properties of analog circuits. Challenging verification issues still remain, such as verification methods for verification in the frequency domain still need to be developed.

## II.B State Space Exploration Methods

While theorem proving is a highly powerful verification technique it has achieved limited success in verification for analog/mixed signal circuits. Another class of methods which have been proposed for analog verification is the state space exploration methods. State space exploration methods can be further sub-divided into two methods: ‘Equivalence Checking’ and ‘Model Checking’.

An analog system can be represented as a system of ‘n’ nonlinear first-order differential algebraic equations using Modified Nodal Analysis (MNA) [18]. The MNA analysis approach relies upon the use of Kirchhoff’s node equations and additional device equations for special devices such as voltage sources and inductors to form these ‘n’ nonlinear differential algebraic equations. In general, an analog system can be represented as shown below.

$$\begin{aligned}
&f_1(x(t), x(\dot{t}), u(t) = \ 0 \\
&f_2(x(t), x(\dot{t}), u(t) = \ 0 \\
&\ldots \\
&f_n(x(t), x(\dot{t}), u(t) = \ 0
\end{aligned}$$

where *u(t)* represents the input, *t* the time, and vector *x(t)* represents the system variables such as node voltages and branch currents.

Most of the verification methods proposed till date convert this high-dimensional continuous time state space to a large discrete equivalent so as to apply Boolean like verification techniques. (Figure 3)

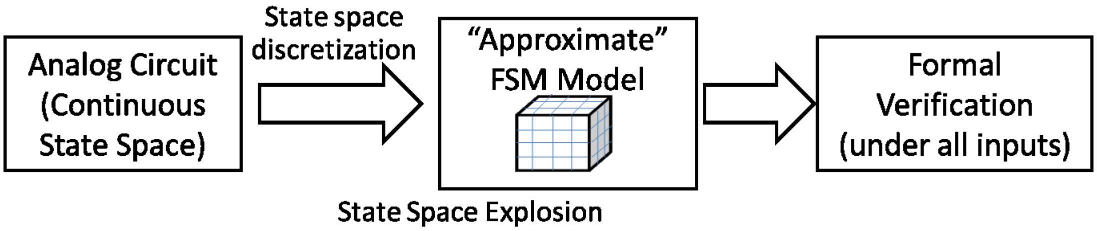

State space discretization
Analog Circuit (Continuous State Space)
"Approximate" FSM Model
Formal Verification (under all inputs)
State Space Explosion

**Figure 3.** State-space explosion problem

While such an approach facilitates the automation of the verification problem, the major disadvantage of these techniques is the state-space explosion problem for large circuits which limits most of the existing methods to small circuits only.

### II.B.1 Equivalence Checking

Equivalence checking is a problem where we are given two system models and are asked whether these systems are equivalent with respect to some notion of conformance, or functionally similar with respect to their input-output behavior [13]. The two models to be compared may be either at the same level of abstraction or at different levels of abstraction, e.g. SPICE netlist vs. SPICE netlist, SPICE netlist vs. behavioral models, or behavioral models vs. behavioral models.

Specifically, let us consider two models, model A and model B, both of which can either be transfer functions, or SPICE netlists or behavioral models. The problem of equivalence checking tries to answer the following: 'Are the two models A & B, equivalent to each other across the set of inputs and parameter variations?' i.e.

$$\forall I, \forall P\ Model\ A\ \backsim Model\ B$$

where $I$ is the set of input signals and $P$ is the set of parameters.

Equivalence checking methods have been widely used for verification of digital circuits. However, the extension of existing equivalence checking methods for analog/mixed signal circuits is not trivial. For analog circuits the exact same magnitudes of current and voltage cannot be attained and hence, an error bound is defined and the models are said to be equivalent if the error lies within this bound. This need for specification of tolerance and bounds on parameters and signals for analog/mixed signal circuits makes equivalence checking a challenging problem.

However, few equivalence checking methodologies valid for specific classes of analog circuits have been proposed in literature. In [7] the authors propose an approach for equivalence checking of transient response of linear analog circuits whose specifications are given in form of a rational transfer function. The authors propose to transform the specifications and the state-space extracted from the actual implementation from the s-domain to the z-domain by using bi-linear transformation. The discretized models are represented in terms of digital adders, multipliers, delay elements and are encoded into finite state machine (FSM) representations. Then the transient behavior of the implementation mimics that of the specification if and only if for any initial state of the specification, there exists a state in the implementation such that the FSMs representing the two circuits produce identical output sequences for all input sequences of specified length ‘K’ applicable to the specification [7]. However, this approach is fairly limited as it is difficult to generate transfer functions for non-linear circuits. Further, the approach inherently suffers from state space explosion when the discretized design is encoded as a FSM.

In [19] the authors propose a non-linear optimization based formulation to verify the frequency response of linear analog circuits whose specification is provided in the form of a transfer function. It verifies the conformance of the magnitude and phase response of the implementation with the specification over the desired frequency range by modeling the problem as a non-linear optimization problem. The results of the global optimization are then used to verify the equivalence. This work is then extended to incorporate equivalence check under parameter variations. To reduce the computational costs in transfer function modeling the authors also propose to use logarithmic transformation to obtain the transfer function as well as parameter models. This reduces the modeling problem to into a simple linear regression problem and hence reduces the computational costs.

In [20] authors combine an equivalence checker, analog simulator and term rewriting engine to form a verification methodology for verification of VHDL-AMS designs. The verification methodology partitions the design into analog, digital and convertor components. The digital components are verified using conventional boolean satisfiability (SAT) or binary decision diagram (BDD) based equivalence checkers. The

A/D and D/A convertors are matched using syntactic matching. The analog components of the design are simplified using ‘term writing’. The reduced analog architectures are then fed to comparators which are verified using simulation. While the proposed method works well for small circuits, it is difficult to apply rewriting techniques for complex analog circuits. Further, the non-linear behavior of analog circuits limits the application of the proposed methodology to higher level (behavioral or architectural) of abstraction only.

In [21] authors propose an equivalence checking methodology for the general class of non-linear dynamic circuits. The methodology compares the geometrical descriptions of state space descriptions of the models to be compared and determines whether the resulting vector and scalar fields are equal or not. The authors recognize the need for non-linear mapping of state-space descriptions to determine equivalence as the two models under consideration may not necessarily have the same internal state variables. The authors propose mapping functions to uniquely map the state variables onto virtual state variables to allow for mapping from the state space description to a canonical representation. Further, an algorithm is proposed to iteratively calculate these mapping functions. The basic idea is to linearize the system at particular sampling points and use linear mapping matrices to apply a local linear mapping. The linearization process is done in the whole state space. The authors have applied this technique for verification of analog circuits such as bandgap-references and Schmidt trigger. While an innovative approach, this approach is not scalable for big analog/mixed signal systems because of state space explosion, and associated computational complexity.

In [22] authors propose a novel technique combining formal verification and transient circuit simulation to achieve the aim of analog verification with full analog state space coverage. The proposed algorithm generates an input stimulus that covers the system's complete state space in a single transient simulation. This input is then used to simulate both the models under consideration and the resulting deviation between the outputs is used to determine equivalence between the two models. However, the input generation algorithm requires conversion of the analog circuit into a discrete graph data structure using methods similar to [23] which is difficult to implement for large analog/mixed signal systems. Thus, while an innovative approach, this approach is also restricted to small designs only and cannot be applied for big analog/mixed signal systems such as phase locked loops (PLL) and analog-to-digital convertors (ADC) in its present form.

While significant progress has been made in applying equivalence checking methods for analog circuits in recent past, most of these methods are limited to small circuits only. Thus, there is a need for equivalence checking methods that can be applied to big analog/mixed signal systems also. In this thesis we propose an equivalence checking methodology that can be applied to big analog/mixed signal systems such as PLLs and ADCs also.

### II.B.2 Model Checking

Model checking is a technique for automatic verification of finite state concurrent systems. It involves representing the design to be verified in form of a transition system. The specifications of the design are translated to temporal logic formulas. State exploration algorithms are then used to verify if the specifications are satisfied or not [4]. More formally the model checking problem can be defined as follows [4]: Given a model ‘M’ of a design and a property ‘P’ expressed in temporal logic, check $M \models P$, i.e. check if ‘P’ holds in ‘M’.

In the recent past model checking techniques have been extended for verification of hybrid and analog/mixed signal systems. While model checking techniques have been very successful for verification of complex sequential circuits and communication protocols, they have achieved limited success in verification of analog/mixed signal systems due to problem of ‘state-space explosion’ and ‘un-decidability limitations [24]’. In this section we review some of relevant prior-works in extending model checking techniques for analog/mixed signal systems.

One of the early works extending the model checking techniques to analog mixed signal systems was done by Kurshan and McMillan and is reported in [25]. In this work the authors proposed a semi-algorithmic method to extract finite state models from an analog circuit-level model by homomorphic transformations. Concepts from automata theory were then applied to these finite state models to verify digital circuits using transistor levels of abstraction. While the proposed technique maintains the desired levels of accuracy in simulation and simultaneously meets the needs for formal verification, the proposed concepts can only be applied to small circuits as the technique suffer from ‘state space explosion’ problem.

Model checking algorithms need the specification properties to be described as properties of the state space descriptions. ‘Computational Tree Logic (CTL)’ language, described by Clarke and Emerson, has been widely used in digital model checkers for this purpose. In [21] the authors extend ‘CTL’ language to ‘CTL-A’ by introducing a minimum set of operators to allow the use of the language to describe the properties of analog circuits also. The continuous variables, i.e. the time and state values, are converted into discrete state space descriptions by bounding and sub-dividing the infinite continuous state space into rectangular boxes. Further, heuristic methods are used to define the transition relation between the state space regions to get the final discrete system model. This process is similar to [25]. In [21] the proposed algorithm is applied for verification of a Schmidt trigger and a tunnel diode oscillator. However, this approach also suffers from state space explosion problem, thus, limiting its potential use. Further, only a limited set of properties can be described using CTL-A language. For example, CTL-A cannot be used for describing the frequency domain properties for analog circuits.

To reduce the computational complexity of the above model checking methods the authors in [26] propose an efficient representation of high-dimensional objects as their projection onto two dimensional sub-spaces in form of projectahedra. Further, the proposed technique is shown to be valid for both, linear and non-linear systems. While an efficient algorithm, the technique reduces the accuracy of verification.

The above technique has been extended and various tools such as d/dt [11], Checkmate [12], and PHaver [27] have been used for model checking of hybrid systems such as analog/mixed signal systems. In [11] the authors extend the reachability analysis techniques developed for hybrid control systems (using d/dt tool) to verify the time dependent properties of analog systems. The proposed technique was used to verify a second order low pass filter and sigma-delta modulators. In [12] checkmate tool was extended for verification of analog circuits. To create the finite-state abstractions of the continuous analog behavior polyhedral outer approximations to the flows of underlying continuous differential and difference equations were developed. The key advantage of this technique is that the state space is partitioned along the waveforms that the system can generate for a given initial condition and there is no need for discretization of entire state space. In [12] the authors used the above technique for verification of delta sigma modulator whose specifications were described as CTL-A formulas. Similarly, in [27] PHaver tool was extended and was applied for verification of oscillators. In this work the authors combined forward and backward reachability while iteratively refining partitions at each step.

Unlike most of the previous techniques in which the continuous analog space is divided into regions which are then represented in a Boolean manner, in [28], the authors propose techniques to model analog and mixed signal systems as timed hybrid petri nets (THPN). THPN allows modeling of continuous values such as voltage and current while still being able to model discrete events. In [28] the differential equations representing the analog circuits are first discretized and the resulting state space is then encoded into

THPN. Zone based reachability algorithms are then used to perform reachability analysis to verify the properties of the systems. In [29] authors extended this work by developing algorithms to develop THPNs directly from simulation data itself. Further, recently labeled hybrid perti nets (LHPNs) have also been proposed to allow for more effective representation of the analog/mixed signal circuit state-space. Various techniques such as reachability analysis and binary decision diagram (BDD) based algorithms are then applied for verification of the properties of the analog circuits.

As discussed above various model checking algorithms have been proposed in recent past for verification of analog/mixed signal systems. While significant efforts have been made to reduce or solve the state space explosion problem, most of the methods developed till now can only be applied for verification of small circuits. Further, till date, the use of model checking algorithms has been limited to verification of transient properties of analog/mixed signal circuits only, and techniques need to be developed to extend model checking to frequency domain also.

### II.C Other Recent Methods

In addition to the above discussed methods, few very innovative techniques for analog/mixed signal verification have been proposed very recently. In [30] authors propose a novel verification methodology for formulating SPICE level circuit simulation as a Boolean satisfiability (SAT) problem. The authors recognize that the ‘local solution’ to the set of Kirchoff’s Current Law (KCL) equations, typically provided by spice simulations, makes it difficult to answer whether the circuit obeys a particular property over the entire range of operating conditions or not. To overcome this limitation the authors propose a new circuit simulation tool, formal spice, which is based on Boolean satisfiability. This tool takes a transistor level netlist as input, and represents the I-V relationship imposed by the devices using conservative approximations in the form of tables. A SAT solver is then used to perform an exhaustive search to find all possible solutions for the simulation problem. Figure 4 below compares the SPICE and SAT based circuit simulation formulations.

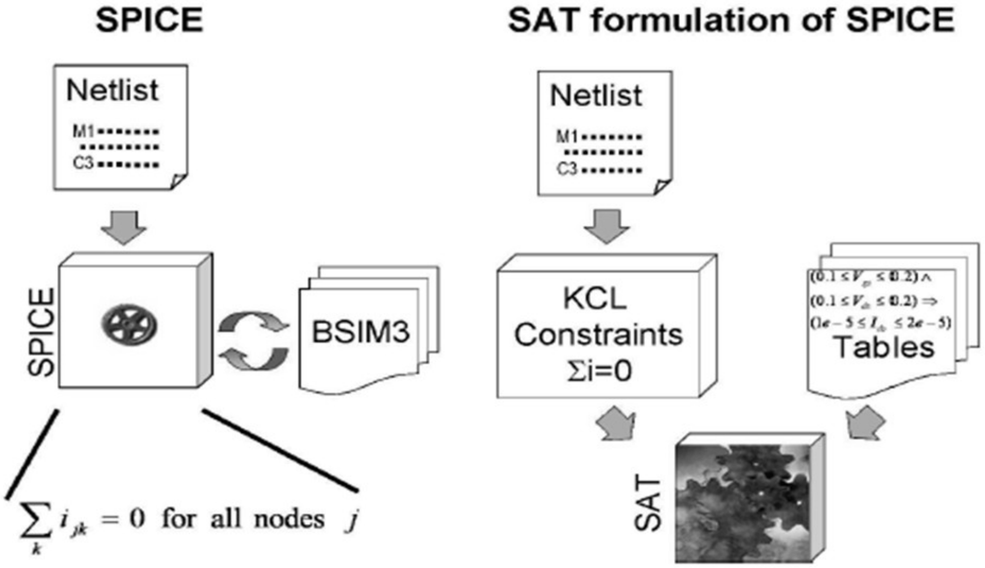

SPICE
SAT formulation of SPICE
Netlist
M1
C3
Netlist
M1
C3
SPICE
BSIM3
KCL Constraints Σi=0
$(0.1 \le V_{gs} \le 0.2) \wedge$
$(0.1 \le V_{ds} \le 0.2) \Rightarrow$
$(1e-5 \le I_{ds} \le 2e-5)$
Tables
$\sum_k i_{jk} = 0$ for all nodes $j$
SAT

**Figure 4.** SPICE simulation vs. SAT based circuit simulation [30]

In [30] the proposed method has been used for DC, transient and periodic steady state (PSS) simulations. While an innovative technique this technique also suffers from scalability issues, as it is difficult to solve the satisfiability problem for large complex systems like phase locked loops. Further, the run time increases exponentially as the granularity of the discretized device I-V tables used to formulate the satisfiability problem decreases.

In [31-32] authors propose a ‘model-first’ approach for design and verification of analog/mixed signal systems. The authors propose to create linear functional models for various analog components which can then be used during full system simulation. The linear functional models are based on the assumption that the behavior of various analog blocks is linear in some domain, and hence, variable domain translators are defined to convert signals from voltage/current domain to an apt domain. ‘Gain matrices’ are then defined from input to output to characterize the analog circuits and the deviation between the gain matrices obtained from the functional model and the circuit implementation is used as a measure of equivalence between the functional model and the actual circuit implementation. Further, the authors propose to classify various inputs/output ports into different categories such as ‘Analog I/O port’, ‘Quantized analog I/O’, ‘Analog control ports’, ‘True digital port’ to extract the linear models. While an innovative approach, the linear model assumption severely restricts the use of the technique for many practical systems.

As discussed in this chapter various techniques for analog mixed signal verification have been proposed in the recent past. However, these techniques either suffer from huge computational complexity which limits the size of the circuits to which these methods can be applied to, or they are based on linearity assumptions which are only partially valid for analog circuits. Thus, analog verification still continues to be a significant research challenge.  In this thesis we propose an optimization based, hierarchical, semi-formal behavioral model equivalence checking methodology for large analog/mixed signal designs such as PLLs, ADCs and I/O’s which is not necessarily formal, but yet, systematic and practical.

# CHAPTER III

# VERIFICATION METHODOLOGY

In this thesis we propose an optimization based, hierarchical behavioral model equivalence checking methodology for large analog/mixed signal designs such as PLLs, ADCs and I/O's. The verification methodology is not necessarily formal, but yet systematic and practical.

The proposed methodology facilitates feasible behavioral model equivalence checking under the following system context. We assume that the desired system behaviors are "encoded" in a set of block-level behavioral models, or the reference system behavioral model (RSB). Hence, the desired system performance specifications are also reflected in the simulated performances of the RSB. A given detailed electrical (circuit) implementation, e.g., represented by a set of (extracted) block-level SPICE netlist, is checked (verified) against the RSB on an individual block basis. Either, the implementation is deemed as "equivalent", or the check is inconclusive due to the conservative nature of the check. In addition to the aforementioned equivalence checking against a given "golden" RSB, the proposed methodology also serves an intrinsically related purpose: compare an existing electrical-level design implementation against its corresponding behavioral model so as to provide guidance for behavioral modeling. In this thesis the proposed methodology is used for verification of a phase locked loop as a test case.

This chapter is organized as follows, section III.A describes the preliminary definitions used in the methodology and section III.B discusses the proposed verification methodology.

## III.A Definitions and Problem Description

The proposed semi-formal, hierarchical, optimization based equivalence checking methodology aims at verifying equivalence between the system behavioral model called the 'reference system behavioral model' (RSB) against detailed electrical, i.e. transistor level implementation. The input and output signals to/from each block of the reference system behavioral model, hitherto referred to as the ***Behavioral Signals***, belong to a ***behavioral signal domain*** $\boldsymbol{\Omega_B}$. Similarly, we define an ***electrical signal domain*** $\boldsymbol{\Omega_E}$, which contains the input and output signals to/from each block of the electrical transistor level implementation.

To enable verification of large analog/mixed signal designs we also define a limited, but most likely, input behavioral signal space for the behavioral models called the ***Constrained Behavioral Input Space (BIS)***. The mechanics of generating the BIS for each block are discussed in the next section. The equivalence check is then performed not over the universe of all possible inputs in the behavioral signal space, but instead, only with respect to the chosen set of sensible input stimuli as defined by the constrained behavioral input space (*BIS*).

For each block-level behavioral model, and a given behavioral input and the resulting behavioral output, we perform equivalence check by asking the essential question: *does the corresponding block-level electrical model (spice netlist) retain the same (behavioral) input and output correspondence?*

The above question would have been trivial to answer if both models were to operate in the same signal domain. However, the fact that such equivalence check has to be conducted across two different signal domains introduces complications. As such, we define two *mapping functions* $f_{B\text{-}E}$ *and* $g_{E\text{-}B}$ to map the signals from the behavioral signal space to the electrical signal space and vice-versa. The function $f_{B\text{-}E}$ is a one to many mapping while the function $f_{E\text{-}B}$ is a many to one mapping. The generation of these mapping functions is dependent of the module being verified and is explained in Section III. D.

The obtained behavioral *BIS* are mapped to $\Omega_E$ using the mapping $f_{B\text{-}E}$ which is then used to drive the verification on an individual block basis as shown in Figure 5. Each behavioral input in the *BIS* is mapped into to a set of detailed electrical inputs which are then used to simulate the electrical transistor (Spice) level circuit.

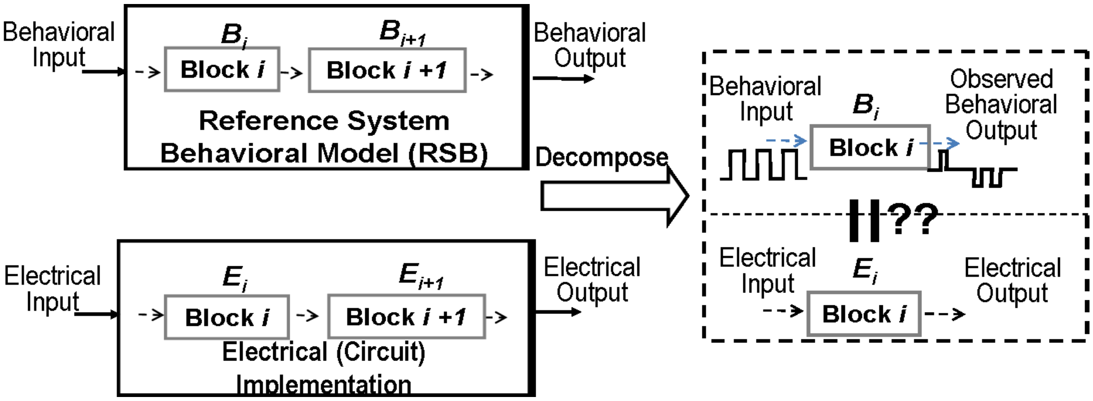


**Figure 5.** Block-level behavioral checking between behavioral and electrical implementations

The resulting electrical outputs are mapped back to the behavioral domain to compare with the reference behavioral output of the behavioral model. The maximum discrepancy of the two is used as metric to judge the equivalence.

## III.B Generation of Constrained Behavioral Space

To allow for a scalable verification methodology we recognize that the inputs to a specific circuit block are constrained by the structure of the entire design, i.e. the inputs to each block in the model cannot be any arbitrary input, and instead, only a *subset* of them (Figure 6). This constrained behavioral signal space for the behavioral model forms the constrained behavioral input space (*BIS*). For example, in a phase locked loop (PLL) the control voltage to the voltage controlled oscillator (VCO) cannot take any arbitrary shape, but instead, is constrained by the operation of the entire PLL and hence verification hence, can be done only on a selected set of inputs rather than the universe of all possible arbitrary inputs.

To generate the BIS for each block in the behavioral model, the RSB is simulated using a set of typical system-level simulation stimuli, such as the ones that are used to measure system design specs (e.g. lock-in time for PLL etc). Upon the completion of each system-level simulation, the *behavioral* input (as well as the corresponding behavioral output) is retained for each circuit block. The complete set of such behavioral inputs defines the *BIS* for the block. In this case, Equivalence Checking essentially checks the electrical implementation against the *RSB* under the typical input excitations that are employed to measure system design specs. If the equivalence check

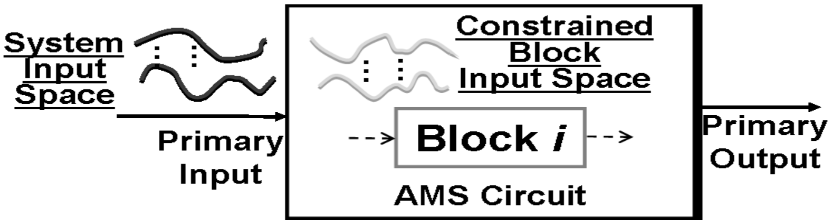


**Figure 6.** Generation of constrained behavioral space. Inputs to a circuit block are constrained by the structure of the design. A reference system behavioral model (RSB) is used to derive block-level inputs

succeeds, the corresponding design specs of the *RSB* would be deemed as reflecting those of the actual implementation. The use of the verification allows efficient determination of achieved system performance specifications without resorting to expensive flat (SPICE) simulations of the design. A more complete input space *BIS* can also be obtained by simulating the *RSB* with a more comprehensive set of system-level input stimuli and record the corresponding behavioral inputs appearing at the input to each circuit block. In practice, these system-level inputs can be obtained by using design knowledge or by introducing pseudo-random variations to typical inputs. In this case, a higher coverage in verification will be resulted as a larger set of input excitations are included in the verification process.

### III.C Behavioral vs. Electrical Domains

As described in the previous section, we use system-level behavioral simulations to generate a behavioral input set (*BIS*) for each circuit block. Then for each behavioral input $I^*_B$ (in the *BIS*) and the corresponding behavioral output of the block, $O^*_{B,}$, the electrical implementation or a SPICE-level transistor model of the block is checked

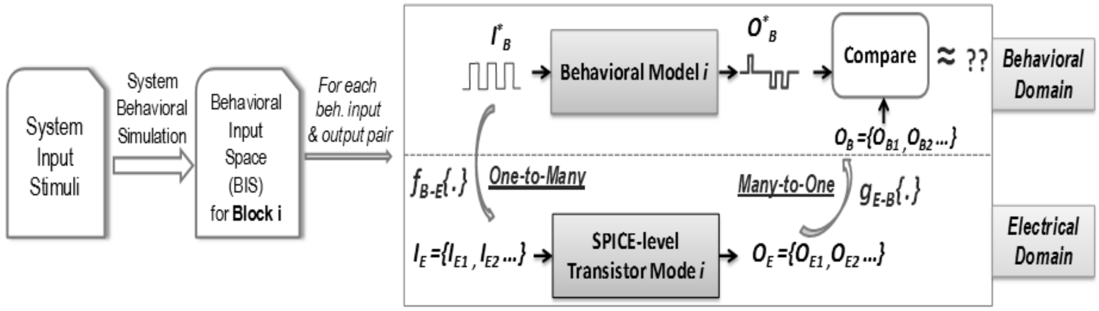


**Figure 7.** Signal mapping between behavioral and electrical domains

against the behavioral block model for equivalence. As illustrated in Figure 7, such equivalence check is performed across two different signal domains: behavioral ($\Omega_B$) vs. electrical ($\Omega_E$). In this section we highlight the key differences between the behavioral and the electrical domains. The mapping functions used to transform the signals from one domain to another are then explained in the next section.

The behavioral domain ($\Omega_B$), characterized by the behavioral signal space, is essentially an abstract form of the actual electrical domain ($\Omega_E$). The signals in the behavioral domain are abstract versions of the electrical signals and are generated by removing some details from the electrical signals. For example, let us consider two models, an electrical model and a behavioral model. We apply a sinusoidal input waveform to both the models. Further, let us also assume that the behavioral model output only depends on the frequency of the input signal and the time instants at which the waveform pulse crosses the origin. Then in principle, any signal with any arbitrary waveform shape but identical frequency and zero crossing time should produce the same behavioral output as the sinusoidal waveform. However, the same shall not be true for the electrical output. Hence, while the behavioral output for the two signals shown in

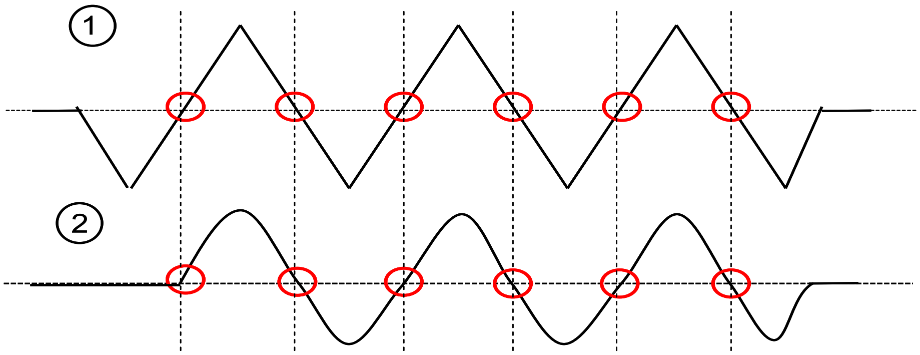


**Figure 8.** Behavioral vs. Electrical signals. Signals 1 & 2 have the same zero-crossing time instants, but different signal shapes. Therefore, while the behavioral output for the two signals, 1 & 2, will be the same, the electrical output for the two signals will be different

Figure 8 will be the same, the electrical output for these two signals will be different. This difference between the behavioral output and the electrical output comes from the fact that while the electrical input is a sinusoidal waveform the actual behavioral input signal simply abstracts away the waveform shape information while only preserving the frequency and zero-crossing times.

To further illustrate the differences between electrical and behavioral domains, especially in relation to analog/mixed signal systems, let us consider a behavioral model for a voltage controlled oscillator (VCO) as shown in Figure 9. The behavioral output of the module only depends on the time instants at which the phase changes, the low and high output voltage levels. No information about the precise waveform shape is present in the behavioral signal, whereas the same information content is present in the electrical domain output of a VCO.

```
module vco (in,out)
…..
analog begin
  freq = (V(in)-Vmin)*α+Fmin // Simple Linear model for VCO frequency
  //phase calculation
  phase = 2*’M_PI*idtmod(freq,0.0,1.0,-0.5)
  //generation of output voltage
  V(out) <+ transistion(n? Vlo: Vhi, td,tr,tf)
```

**Figure 9.** Part of a VCO behavioral model

### III.D Signal Domain Mapping Functions

To link the two domains together we define two mappings, $\boldsymbol{f_{B\text{-}E}\{\cdot\}}$: $\Omega_B \rightarrow \Omega_E$ and $\boldsymbol{g_{E\text{-}B}\{\cdot\}}$: $\Omega_E \rightarrow \Omega_B$. With the inherent abstraction in behavioral modeling, $f_{B\text{-}E}$ is one-to-many mapping and maps a behavioral signal waveform to a set of electrical realizations; $g_{E\text{-}B}$ is many-to-one mapping and abstracts away non-behavioral details from an electrical waveform. Using $f_{B\text{-}E}$ we map a single (behavioral) input $I^*_B$ to the behavioral model to a set of electrical inputs, $S_{IE} = \{I_{E1}, I_{E2}\ldots\} = f_{B\text{-}E}\{I^*_B\}$, which are used to exercise the SPICE model in $\Omega_E$ (Figure 7). The resulting multiple electrical outputs $S_{OE} = \{O_{E1}, O_{E2}\ldots\}$ are mapped back to $\Omega_B$ via $S_{OB} =\{g_{E\text{-}B}\{O_{Ei}\}\}$ to compare against the reference output of the behavioral model $O^*_B$. Note that for a single behavioral input $I^*_{B,}$, $I_E = f_{B\text{-}E}\{I^*_B\}$ defines the electrical input space over which the electrical implementation needs to be checked for equivalence. On the other hand, since the reference behavioral output, $O^*_{B,}$ is behavioral, the outputs of the electrical implementations are mapped back to the $\Omega_B$ via $g_{E\text{-}B}$ for comparison.

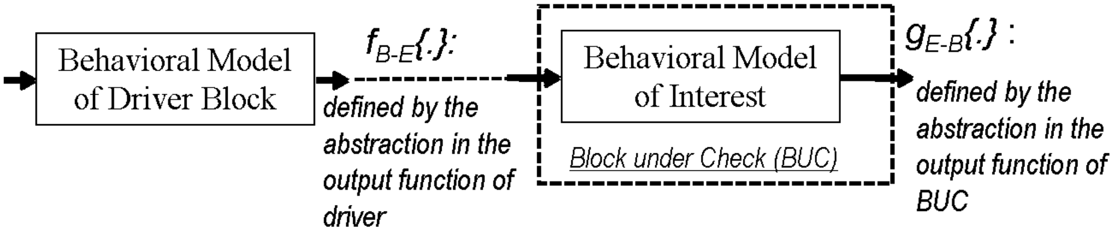

Behavioral Model of Driver Block
$f_{B\text{-}E}\{.\}$:
defined by the abstraction in the output function of driver
Behavioral Model of Interest
Block under Check (BUC)
$g_{E\text{-}B}\{.\}$ :
defined by the abstraction in the output function of BUC

**Figure 10.** Illustration for generation of mapping functions

The generation of these mapping functions is dependent on the module under verification. To illustrate how these two mapping functions are generated in practice, let us consider part of the behavioral model of a phase locked loop comprising of the charge pump and a module containing the filter and a VCO (Fig. 10). The behavioral output of the charge pump may contain only idealized current pulses which act as behavioral inputs for the filter & VCO module (Fig. 11). Note that these output signals are in the behavioral domain and only have essential modeled *behavioral characteristics* of the output signal. $f_{B\text{-}E}$ basically maps the behavioral output signal to the electrical domain by adding the un-modeled electrical details, say in this case, the rise time and the fall time of the output current pulse. Note that for each behavioral input signal multiple electrical signals are produced. Similarly, the reference behavioral output signal of the filter & VCO module, $O^*_B$, only contains the essential behavioral characteristics that are modeled in the output function of the VCO, which for a model shown in Figure 9 shall be the level crossing time points. To compare with this reference $O^*_B$, $g_{E\text{-}B}$ basically maps the detailed electrical output waveforms produced by the SPICE-level block model to the behavioral domain. In the present example, the electrical outputs of the corresponding SPICE-level VCO net-list shall be simply mapped back to the behavioral

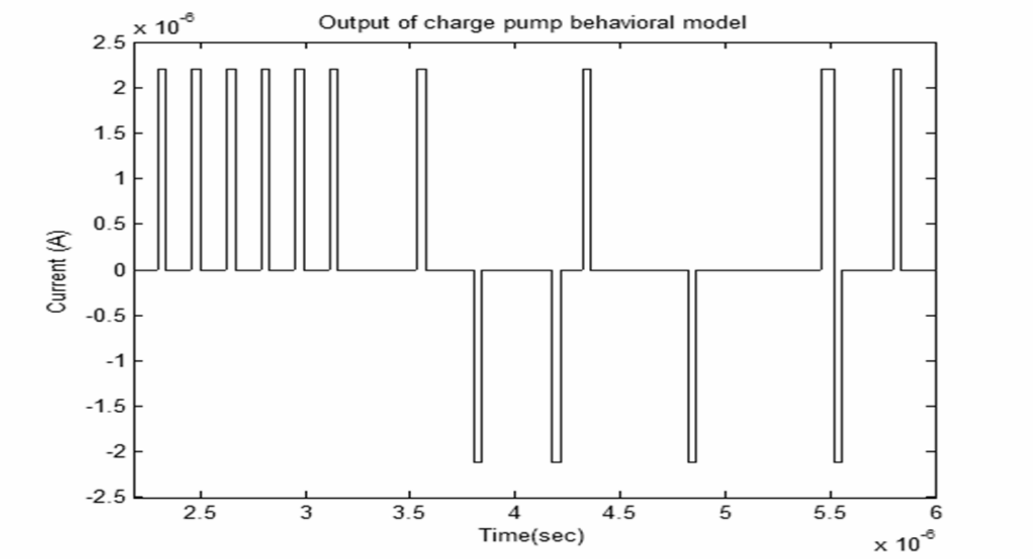


**Figure 11.** Typical behavioral output for a charge pump

domain by extracting the level-crossing time stamps. In general, $g_{E\text{-}B}$ is many-to-one since multiple electrical signals can have the same extracted behavioral features.

In principle, mapping functions $f_{B\text{-}E}$ and $g_{E\text{-}B}$ are module dependent. In particularly, as illustrated in Figure 10, $f_{B\text{-}E}$ for the block under check shall be constructed to reflect the behavioral abstraction embedded in the output function of the preceding (driver) behavioral model. On the other hand, $g_{E\text{-}B}$ effectively extracts from an electrical output the behavioral characteristics that are specified in the output of the behavioral model under check.

### III.E Optimization Based Formulation

As described in previous sections the proposed verification methodology involves generation of system level behavioral stimuli, mapping each behavioral input stimulus to a set of detailed electrical inputs which are then used to simulate the SPICE level transistor net-list. At the output, we map the set of electrical signals produced to the behavioral domain, which are then compared with the corresponding behavioral outputs from the RSB to verify equivalence between the two implementations. In this section we formulate the above comparison as a maximization problem. The optimization problem may be solved using any 'simulation based optimizer', i.e. any available optimization solver which does not necessarily require a closed form expression for calculating the objective function. In this paper we used DONLP2 [33][34], a sequential quadratic programming (SQP) based optimization engine for the same. DONLP2 was interfaced with CADENCE Spectre to allow computation of the objective function using actual spice level simulations.

For a given behavioral input $I^*_B$, the behavioral model produces $O^*_B$ at the output (Figure 7). To verify whether or not this input-output correspondence is retained in the electrical implementation, we ask the following question: *for all electrical input signals that have the behavioral characteristics specified by $I^*_B$, will the corresponding electrical outputs maintain the same behavioral characteristics specified by $O^*_B$?* For every circuit block, we perform the above equivalence check for each behavioral input in its *BIS*. An electrical implementation is deemed as equivalent to the system behavioral model if and only if all such checks are passed. We formulate the above as a maximization problem.

We parameterize the non-behavioral electrical features not modeled in a behavioral input, such as finite rise/fall times and signal shapes, by introducing additional electrical feature parameters. We denote these electrical feature parameters as

$p_E$. Such parameterization mathematically constructs the mapping function $f_{B-E}$. The mapped electrical input set $S_{IE} = \{I_{E1}, I_{E2} ...\} = f_{B-E}(I^*_{B}, \cdot)$ defines a constrained electrical input excitation space over which the SPICE net-list needs to be compared with the behavioral model. We formulate this task formally as a maximization optimization problem in $S_{IE}$ and seek to obtain the maximum deviation $\varepsilon_{\max}$ from the reference behavioral output $O^*_B$:

$$\max_{p_E} \ \varepsilon = h_{err}\left(O_B(p_E), O^*_B\right) \tag{1}$$

*Subject to:*

$$\underline{p_E} \le p_E \le \overline{p_E} \tag{2}$$

$$I_E = f_{B-E}(I^*_B, p_E) \tag{3}$$

$$O_E(p_E) = Q_{SP}(I_E(p_E)) \tag{4}$$

$$O_B(p_E) = g_{E-B}(O_E(p_E)) \tag{5}$$

Equation (1) defines the objective function, which is an error function $h_{err}$ specifying the derivation between the mapped electrical output and the reference behavioral output $O^*_B$. For instance, if $O^*_B$ and $O_B(p_E)$ are represented as vectors of sampled signal values, $L_2$ vector norm can be used to define the error function: $h_{err} = \left\| O_B(p_E) - O^*_B \right\|_2$ In practice, the definition of $h_{err}$ is model dependent and is dependent on the functionality of the block. For example, for the VCO behavior model shown in Figure 9, it may be the $L_1$ vector norm of level crossing time stamps, while for a charge pump it may be $L_2$ vector norm of

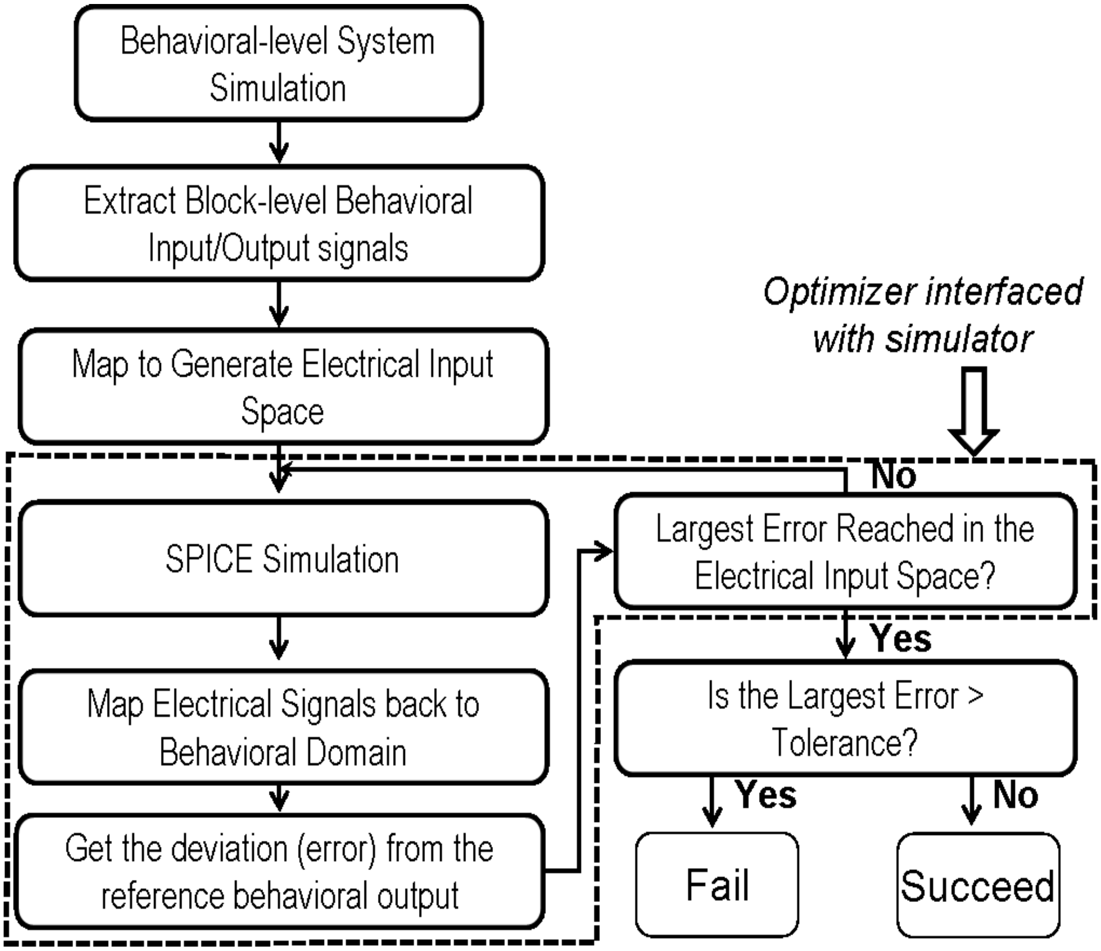

Behavioral-level System Simulation
Extract Block-level Behavioral Input/Output signals
Map to Generate Electrical Input Space
Optimizer interfaced with simulator
No
SPICE Simulation
Largest Error Reached in the Electrical Input Space?
Yes
Map Electrical Signals back to Behavioral Domain
Is the Largest Error > Tolerance?
Yes
No
Get the deviation (error) from the reference behavioral output
Fail
Succeed

**Figure 12.** Optimization based equivalence checking flow

the vector measuring the charge injected into the system at different time points. Equation (2) defines the bound on the electrical input parameters. Equation (3) maps $I^*_B$ to an electrical input signal by adding electrical features specified by $p_E$. $Q_{SP}()$ in (4) maps an electrical input applied to the SPICE-level model to the corresponding electrical output; this mapping is realized by running circuit (SPICE) simulation. Finally, (5) maps the electrical output to the behavioral domain by using $g_{E\text{-}B}$.

The proposed optimization-based equivalence-check flow is shown in Figure 12, where an optimizer (DONLP2) is employed to search for the maximum deviation $\varepsilon_{\max}$. If $\varepsilon_{\max}$ is less than a user-defined tolerance, the equivalence check is deemed as passed; otherwise, a failure is reported. At the inner loop of the optimization, the circuit simulator, CADENCE Spectre, is interfaced to provide the mapping in (4).

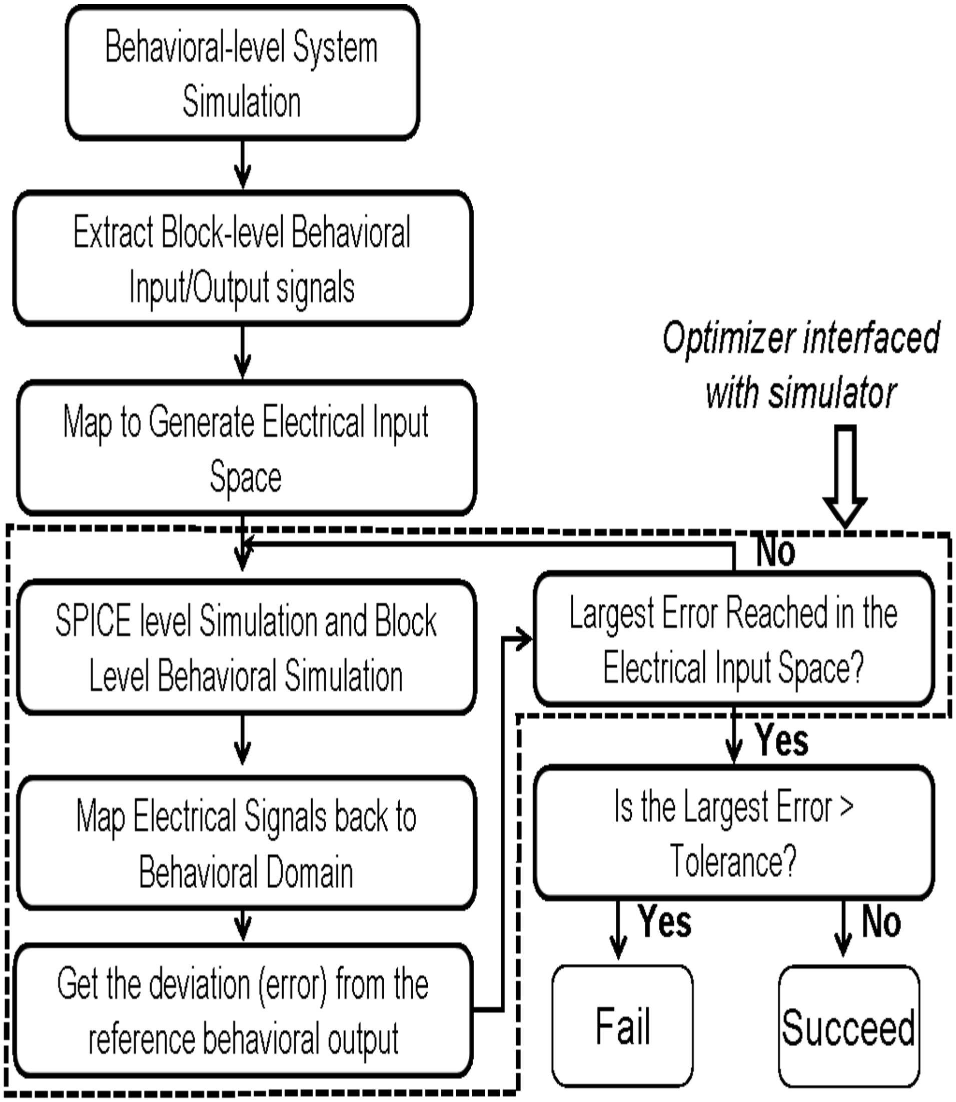

Behavioral-level System Simulation
Extract Block-level Behavioral Input/Output signals
Map to Generate Electrical Input Space
Optimizer interfaced with simulator
No
SPICE level Simulation and Block Level Behavioral Simulation
Largest Error Reached in the Electrical Input Space?
Yes
Map Electrical Signals back to Behavioral Domain
Is the Largest Error > Tolerance?
Yes
No
Get the deviation (error) from the reference behavioral output
Fail
Succeed

**Figure 13.** Modified optimization based equivalence checking flow

To make the above methodology more robust and conservative in nature, we also implemented a slightly modified form of the above flow. The modification was done to allow the proposed verification flow to merge conveniently with the existing commercial circuit simulation software like CADENCE Spectre and HSPICE. Although the behavioral output signal at any point should/does not depend on the un-modeled electrical details in the behavioral input signal, practical circuit simulators do not distinguish between the behavioral signals and the electrical signals. The circuit simulator treats the behavioral input and output in the same way as they treat the electrical signals. This anomaly in simulators may sometimes lead to unexpected results. To avoid any such occurrences and enable a conservative check we modify the above flow shown in Figure 12 slightly. The modified flow is shown in Figure 13. As shown in Figure 13, at inner loop of optimization, in addition to simulating spice level net-list we also simulate the behavioral model for the block under verification with the same electrical domain input signal. The electrical outputs are then mapped back to the behavioral domain and maximum deviation between the electrical and behavioral domain outputs are computed in the same manner as before.

# CHAPTER IV

# RESULTS – VERIFICATION OF PHASE LOCKED LOOP

This chapter discusses the performance of the above proposed verification methodology when applied towards equivalence checking of phase locked loop (PLL). The proposed methodology was implemented using C language and was applied for equivalence checking between VERILOG-AMS based behavioral model of a phase locked loop (PLL) and its electrical implementation (CADENCE Spectre net-list). The block diagram of the PLL used is shown in Figure 14. To generate the behavioral input space (*BIS*) for each block in the behavioral model, the reference behavioral model was simulated using a typical system-level simulation setup used to calculate the lock-in time of a PLL. The reference input signal was a pulse of 10.9MHz and the voltage signal 'Vcontrol' was used to modify the divider ratio of the PLL from 150 to 100 at a time instant of 3μs. The BIS can easily be expanded to verify other important PLL properties such as reference spur power by simulating the reference behavioral model with typical setups used for measuring the required property. The verification of the entire system

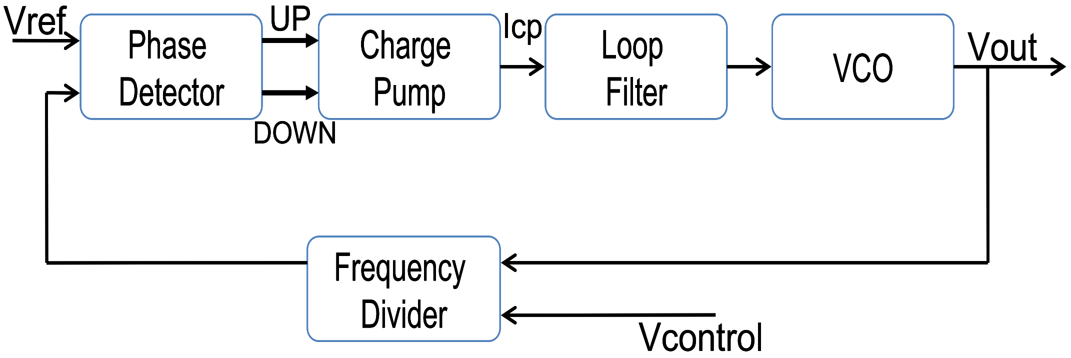


**Figure 14.** Block diagram of a phase locked loop (PLL)

was performed in a hierarchical manner by dividing the system into three modules, loop filter and voltage controlled oscillator (VCO), charge pump and phase detector.

## IV.A Verification of VCO and Loop Filter

The behavioral input to the block composed of the loop filter and the VCO consists of idealized current pulses from the charge pump. The electrical implementation of the block is shown in Figure 15. Figure 16 depicts the SPECTRE simulation results of the current pulses generated by the charge pump from 0.32µs to 0.66µs. The results depict the presence of spikes in the output current whenever the current waveform amplitude changes suddenly. To map the idealized behavioral current pulse waveforms into electrical equivalent signals four electrical feature parameters, trise ($t_r$), tfall ($t_f$), peak_pos (*p_pos*) and peak_neg (*p_neg*) were defined. trise ($t_r$) and tfall ($t_f$) represent the rise time and fall time of the current waveforms, and peak_pos (*p_pos*) and peak_neg (*p_neg*) refer to the peak amplitudes of the current spikes generated in the output waveform. The behavioral output of the block, i.e. the VCO behavioral output, is

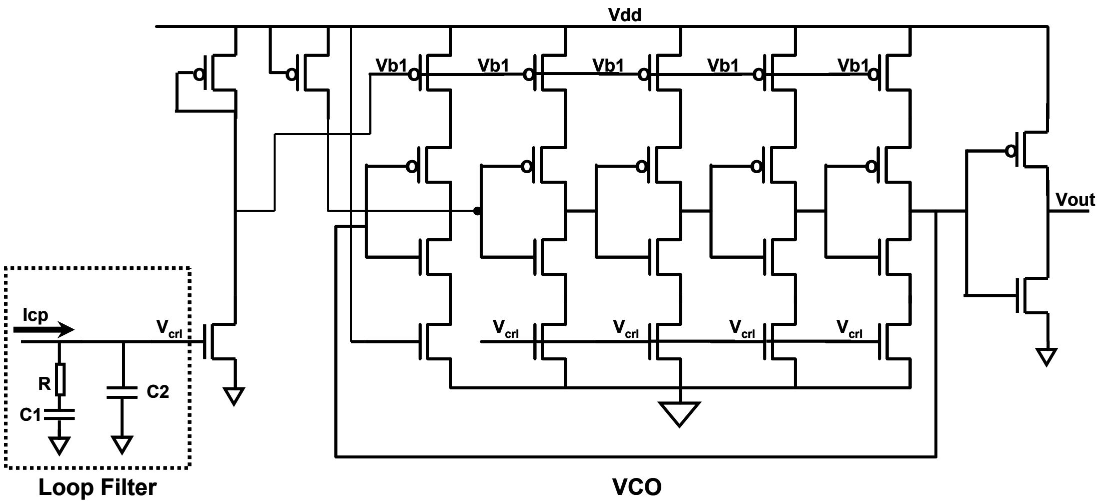


**Figure 15.** Electrical implementation of VCO and loop filter

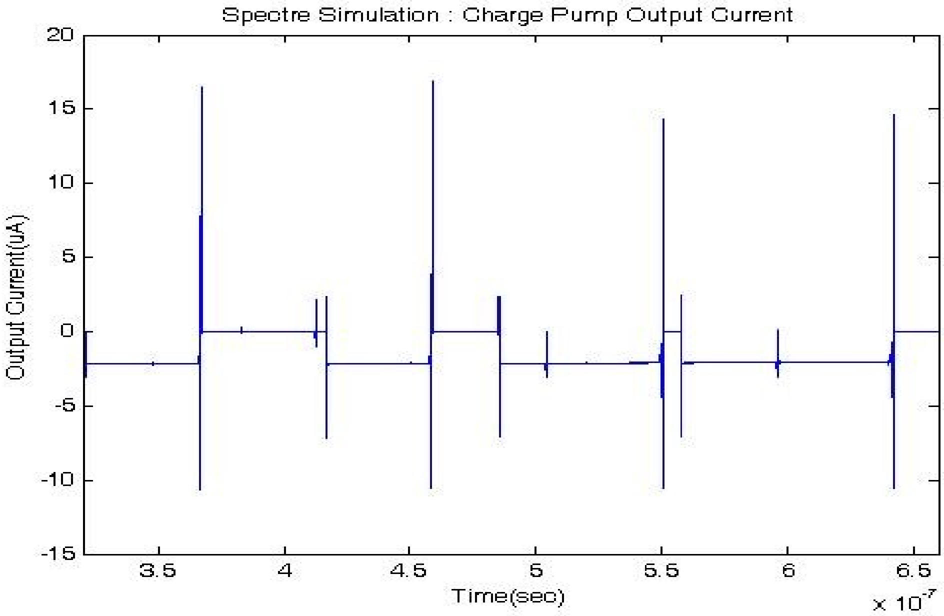


**Figure 16.** SPECTRE simulation: output of charge pump current

dependent only on the level crossing time instants (Figure 9). Thus, the electrical output signals are mapped back to behavioral domain by simply extracting the level crossing time instants. Based on the above, the optimization problem was formulated as under:

$$\max_{\substack{tr,tf,\\ p\_neg,p\_pos}} \quad \varepsilon = \| t - t_B^* \|_1^2 \tag{6}$$

*subject to:*

$$1 \le t_r \le 50ns \tag{7}$$

$$1 \le t_f \le 50ns \tag{8}$$

$$3 \le p\_neg \le 20\mu A \tag{9}$$

$$3 \le p\_pos \le 20\mu A \tag{10}$$

where $t$ and $t_B^*$ refer to the level crossing time instants obtained from the electrical and behavioral outputs respectively.

The above optimization problem was solved for three different behavioral models of the VCO. The output frequency versus control voltage plots for the three different VCO models are shown in Figure 17. Model A closely resembles the VCO characteristics across the entire control voltage range whereas models B & C are only linear approximations to the VCO output frequency characteristics. Table 1 below shows the maximum error obtained for each behavioral model, values of the electrical parameters added at that instant, equivalence decision of the methodology, and the total runtime required for optimization. As expected, the maximum error is least for model A and is the only model for which equivalence test is successful.

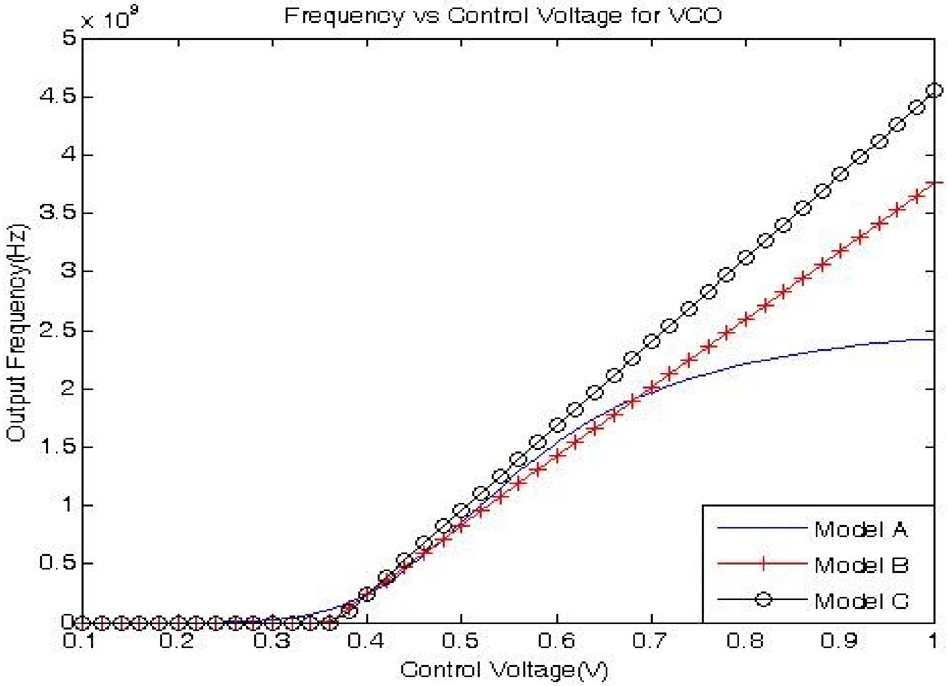


**Figure 17.** Frequency versus control voltage for the three VCO models

**Table 1: Equivalence check for VCO and filter block**

| | **Model A** | **Model B** | **Model C** |
|---|---|---|---|
| **Maximum Error** | 1.5452e+02 | 3.500e+03 | 8.03734e+03 |
| **Rise Time** | 1.999ns | 1.999ns | 50ns |
| **Fall Time** | 1.999ns | 1.999ns | 50ns |
| **Peak_pos** | 3μA | 20μA | 3μA |
| **Peak_neg** | 20μA | 3μA | 20μA |
| **Equivalence** | Yes | No | No |
| **Runtime (sec)** | 8700 | 14280 | 17690 |

In the current implementation the choice of the maximum error tolerance limit is left at the designer's discretion. One possible way to calculate the max tolerance limit could be to introduce the requisite error in the behavioral model of the block under check and find the maximum error that the block can introduce while still meeting all the required system specifications.

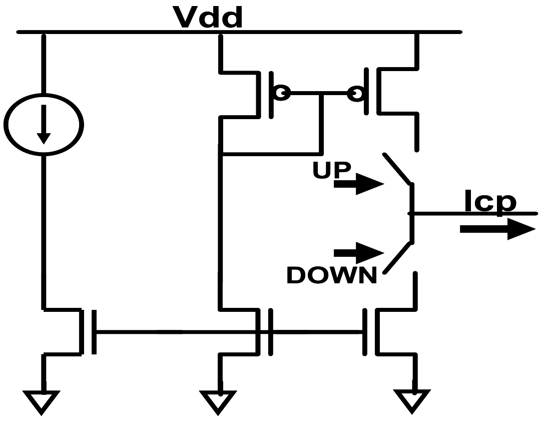
Further, in the current implementation the maximum and minimum limits of the electrical parameter variations are left at the designer's discretion. Alternatively, we can incorporate an additional outer global optimization loop to calculate the maximum error attained for different electrical parameter limits. This would reduce the number of false inconclusive checks resulting from un-reasonably large variation in electrical parameter limits and would also allow for automating the generation of electrical parameter variation limits.

### IV.B Verification of Charge Pump

The behavioral input to the charge pump (*CP*) consists of two digital like voltage pulses 'up' and 'down' controlling the output current. The electrical implementation of the charge pump is shown in Figure 18. To map the behavioral input signals into equivalent electrical waveforms, two electrical feature parameters, trise ($t_r$), and tfall ($t_f$) were defined. The electrical output of the charge pump consists of the short duration current pulses (Figure 16). Since the total charge injected into the system is the most important parameter for the charge pump, the output current was integrated to calculate the net charge introduced by the charge pump. The total charges introduced by the behavioral and electrical implementations at different instants of time were then compared to calculate the error function. Based on the above, the optimization problem was formulated as under:

$$\max_{tr,tf} \varepsilon = \| (q - q_B^*) * 1e6 \|_2^2 \tag{11}$$

*subject to:*



Charge Pump

**Figure 18.** Electrical implementation of a charge pump

$$1 \le t_r \le 50ns \tag{12}$$

$$1 \le t_f \le 50ns \tag{13}$$

where q and $q_B^*$ are vectors consisting of the total charge injected into the system by the electrical and behavioral models at different time instants. To verify the methodology for charge pumps, two different behavioral models were used, with one closer to the electrical implementation than the other (Figure 19). Behavioral model 'A' took into account the current mismatch between the 'up' and 'down' current while model 'B' simply neglected this difference and modeled both the current sources identically. The maximum deviation between the electrical and behavioral models was calculated by solving the optimization problem in (11). Table 2 below shows the maximum deviation between the electrical and behavioral signals and the equivalence decision of the proposed methodology. As expected, the maximum deviation in model 'A' is less than the maximum deviation in model 'B' and equivalence test is successful only for modelA.

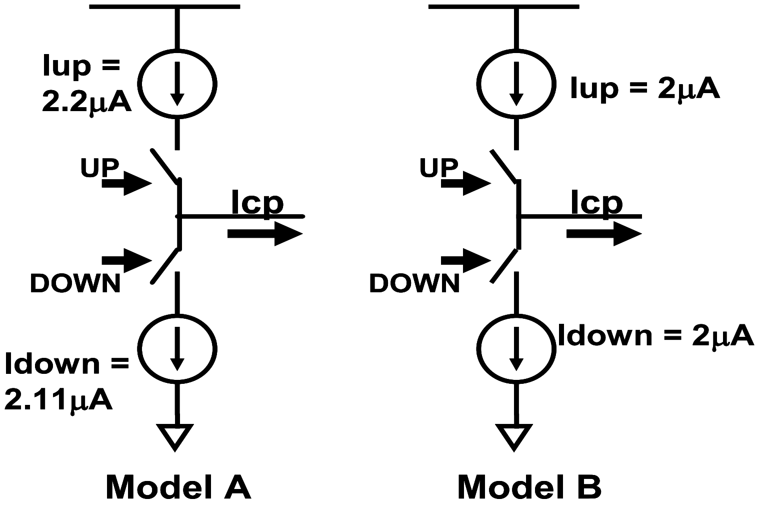


**Figure 19.** Charge pump models

**Table 2: Equivalence check for charge pump**

| Model | Maximum Error | Rise Time | Fall Time | Match | Run Time(sec) |
|---|---|---|---|---|---|
| A | 1.43297e+01 | 50ns | 2ns | Yes | 31.6 |
| B | 2.14668e+01 | 50ns | 50ns | No | 30.4 |

### IV.C Verification of Phase Detector

The behavioral input to the phase detector (*PD*) consists of input reference voltage and the output voltage waveform (behavioral) of the frequency divider. The behavioral input voltage was mapped to electrical domain by adding the rise and fall time parameters to the input behavioral waveforms. Further, similar to the VCO output waveforms, the electrical domain voltage waveforms were mapped back to the behavioral domain by simply extracting the time instants at which the output voltage crossed the Vdd/2 value. The optimization problem for the phase detector was formulated as under:

$$\max_{tr,tf} \varepsilon = \| (t_{up} - t^{*}_{up,B}) * 1e6 \|_2^2 + \| (t_{dn} - t^{*}_{dn,B}) * 1e6 \|_2^2 \tag{14}$$

*subject to:*

$$1 \le t_r \le 50ns \tag{15}$$

$$1 \le t_f \le 50ns \tag{16}$$

where $t_{u,}$, $t^{*}_{up,B}$ ,$t_{dn,}$ and $t^{*}_{dn,B}$ refer to the Vdd/2 crossing time instants obtained from the electrical and behavioral output for the 'up' and 'down' voltage respectively. The above optimization problem was solved for the phase detector and the maximum deviation of

1.48383e+01 was achieved for trise=tfall=50ns. Since, the deviation was less than the maximum threshold, the models were deemed equivalent.

### IV.D Verification of the Proposed Methodology

To verify that the proposed methodology correctly identified the equivalence between the behavioral models and their electrical implementations, system level properties of the PLL such as locking time were calculated from both the behavioral and the electrical implementations. Out of the three models available for the filter and VCO block (Table 1), model A was used to calculate the above properties as model A was deemed as equivalent by the verification methodology. Similarly, model A of the charge pump block (Table 2) was used in the above calculations. Figure 20 shows the frequency of the output signal, Vout, as obtained from the reference behavioral model and when the individual behavioral models were replaced with their electrical counterparts. A locking time of 1.03µs was achieved under all the cases. The difference in the waveforms at 'power on' represents the fact that the behavioral models do not capture the initial PLL power on process well. After 'power on' the behavioral model and transistor level implementation are equivalent to each other.

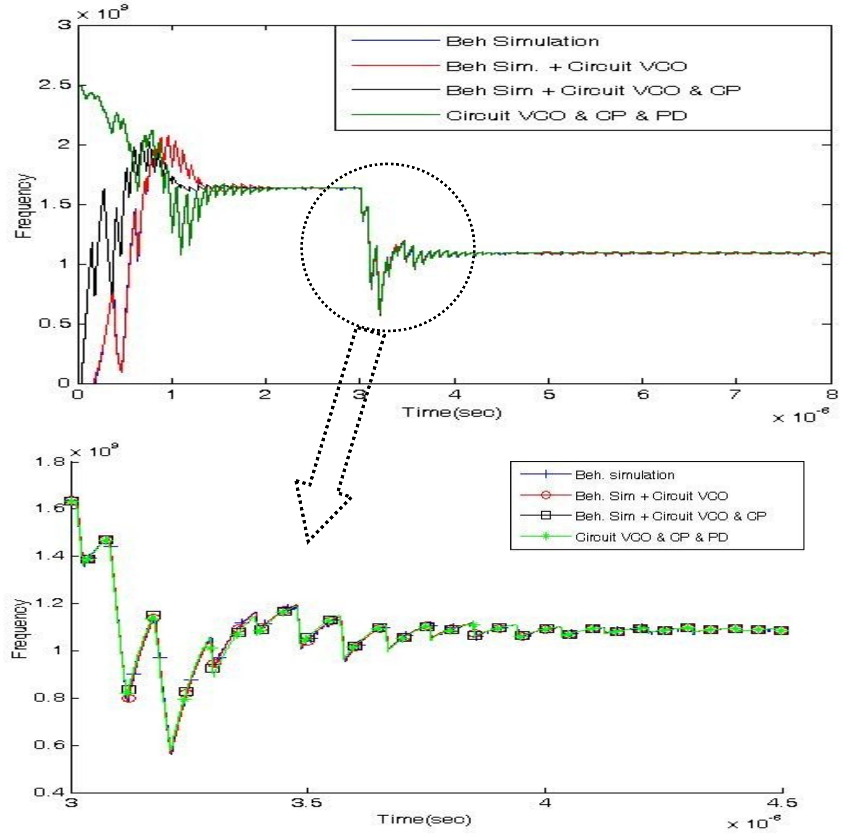


**Figure 20.** Output frequency of the PLL from behavioral and electrical simulations

# CHAPTER V

# CONCLUSION AND FUTURE WORK

## V.A Conclusion

While significant advances have been made in developing efficient verification techniques for digital circuits, verification for analog/mixed signal systems continues to be a challenge for the semiconductor industry. Recently, few verification techniques have been proposed for analog circuits also, however, most of these techniques can only be applied to small analog circuits. The primary goal of this thesis was to develop an efficient, practical, automated, hierarchical, semi-formal verification methodology for large analog/mixed signal systems such as ADCs, PLLs, and I/O's.

In the proposed methodology, we introduce the concepts of constrained behavioral input space, and clear distinction between the behavioral and electrical domain. The proposed verification methodology specifically targets the inherent abstraction in behavioral modeling, which contributes to the deviation of the behavioral model from the electrical model. Two signal domain mapping functions are also defined to map the signals from behavioral to electrical domain and vice-versa.

Subsequently, the equivalence checking between the behavioral and electrical models is formulated as a constrained optimization problem which is solved by interfacing behavioral and SPICE-level simulators with a sequential quadratic programming (SQP) based optimizer, DONLP2. Further, the proposed methodology breaks the system equivalence checking problem into individual block-level checks, which are performed hierarchically. This makes the approach scalable for large designs. Subsequently, the proposed verification methodology was used for equivalence checking of a PLL as a test case.

### V.B Future Work

While the proposed verification methodology allows for hierarchical and automated verification of large analog/mixed signal systems, few key concerns still need to be addressed to make the methodology more robust. In particular, key concerns such as 'What input set should be used for verification?', 'How to define coverage for analog/mixed signal systems?', 'How to incorporate effect of process variations in the optimization formulation', still need to be addressed. In this section, we identify some of these challenges, and propose few ideas on how the verification methodology proposed in this thesis can be extended to address these challenges.

To address the question 'What input set should be used for verification? ,' we recognize that most of specifications provided for analog/mixed signal blocks are in form of inequalities, for example, locking time of the PLL should be less than a specified time, the unity gain bandwidth of an amplifier should be greater than 'x' hertz and so on. Further, fig. 21 shows an example of how a specification may change with variation in inputs. We understand that while the variation of the specification with the input may be non-linear in nature, we may still be able to predict if the specification achieved is greater than or less than the target specification for a range of inputs by looking at only the worst case inputs, i.e. those inputs for which the difference between the target and achieved specification is the least. For example, in Figure 21, if we find the worst case

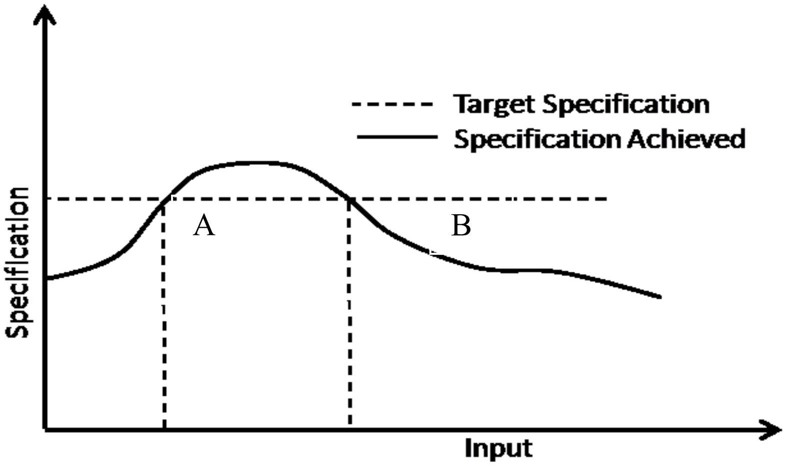


**Figure 21.** Sample non-linear variation of specifications with input

inputs corresponding to points A & B, we can find the range of inputs for which the achieved specification meets the target specification. This worst case input can be found by formulating it as an optimization problem, such as,

$$\min_{Input} |Target\ Specification - Achieved\ Specification|$$

$$subject\ to:$$

$$\underline{V} \leq Input\ \leq \overline{V}$$

where V represents the bounds on the input, such as, maximum amplitude, maximum slew rate of the input etc. Similar to the previous optimization problem shown in chapter IV, this optimization problem can also be solved by interfacing a ‘simulation based optimizer’ such as DONLP2 with circuit simulators such as cadence SPECTRE.

An example of an amplifier is taken to further illustrate the proposed method for finding the input. The schematic of the amplifier under consideration is shown below. (Figure 22)

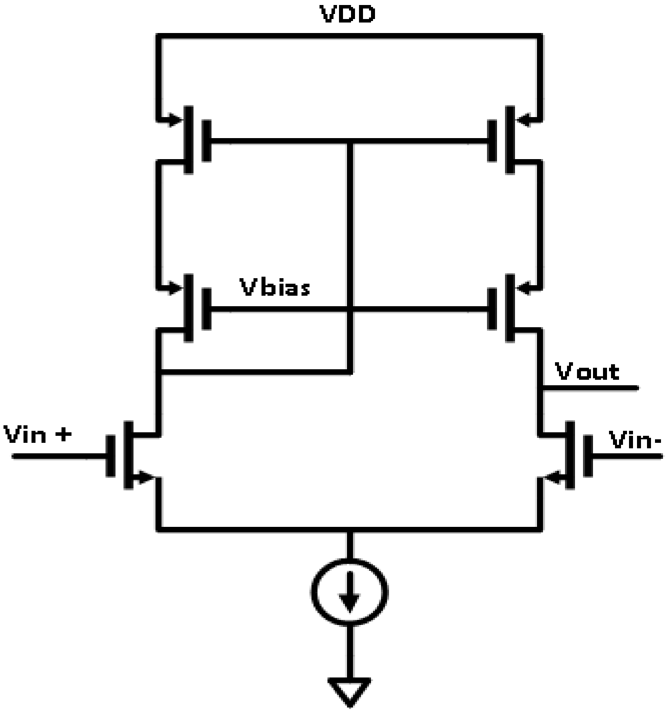


**Figure 22.** Schematic of the amplifier

Let us also assume that the specification to be verified is: 'The unity gain bandwidth of the amplifier should be greater than 4.75MHz for DC input bias voltages ranging from 1.4V to 1.9V'. The required input can be found by solving the optimization problem in (16).

$$\min_{vdc} |x - (-3)| \quad (17)$$

$$subject\ to: \quad (18)$$

$$1 \leq vdc \leq 3 \quad (19)$$

where x represents the magnitude (in dB) of the AC response of the circuit at 4.75MHz and *vdc* represents the DC input bias voltage. The above optimization problem was solved for two different initial conditions, and the worst case inputs were found to be: x=1.4883V and x= 1.943V. Subsequently, the equivalence between the behavioral and electrical implementation was verified for the target specification and the two models were found to be equivalent. Further, Figure 23 shows how ‘x’, i.e. the magnitude of the AC response at 4.75MHz, varies with the input DC bias voltage. As expected, ‘x’ > -3dB for all DC bias voltages between the worst-case inputs, and hence the UGB > 4.75MHz for all DC bias voltages lying between the worst case inputs.

The above proposed implementation solves the optimization problem on SPICE level electrical simulations to find the worst case inputs. While this works for small tiny circuits such as single stage amplifiers, the same will not work for big analog/mixed

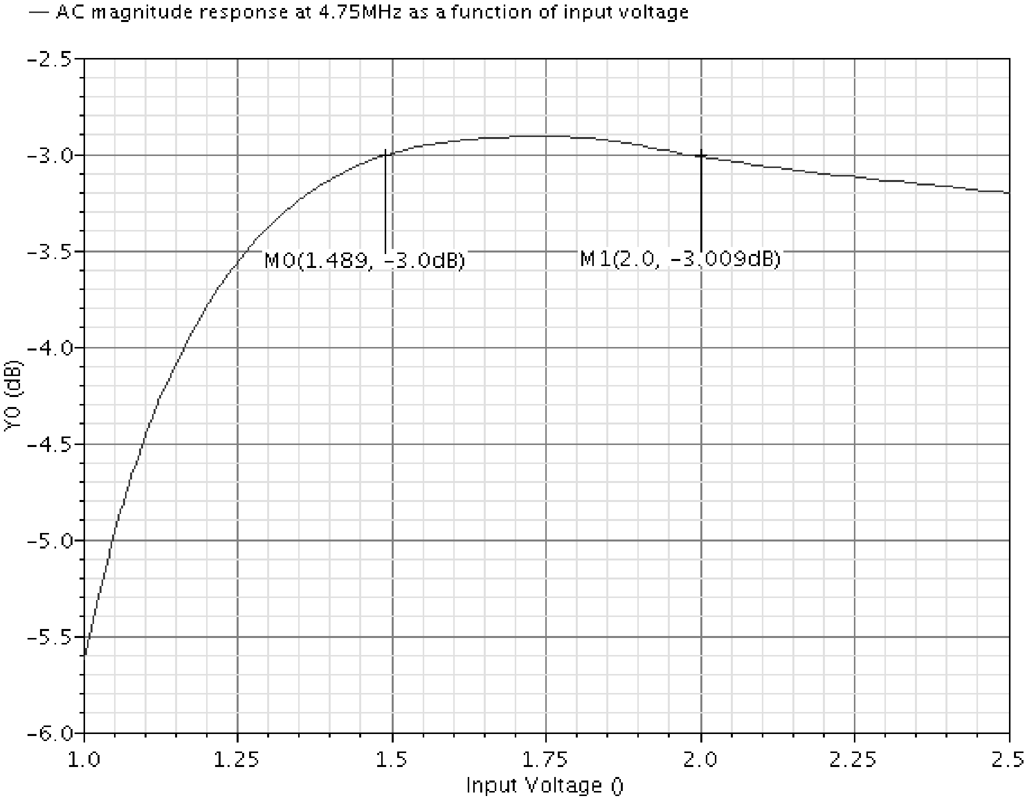


**Figure 23.** Variation of AC magnitude at 4.75MHz with DC input bias voltage

signal systems such as PLLs and ADCs because of the huge simulation time requirements. One possible alternative is to find the worst case input from the behavioral model itself. While a plausible solution, this requires a method to develop behavioral models which take into account the variation of target specifications with changing inputs. Also, it would be interesting to explore the possibility of constraining the region of operation of transistors while finding the worst case input. This would ensure that the transistors operate in the desired regions of operation even for worst case inputs. One possible way to add the region of operation in the optimization problem could be to specify valid trans-conductance to drain current ratio (gm/Id ratio) limits for key transistors.

The above proposed method for finding the input tries to provide information about coverage in an in-formal, yet systematic and practical way. However, formal methods for defining coverage in analog/mixed signal systems also need to be developed for an efficient and robust verification methodology. In [35] authors proposed a state-space guided input stimuli generation algorithm with the aim of full state-space coverage. The authors propose to represent the analog circuit as a discrete graph data-structure, and then subsequently apply a stimuli generation algorithm on this discrete graph. The input generation problem is modeled as a modified form of traveling salesperson problem, and tries to generate an input that visits every reachable state and transition of the circuit, represented by the vertices and edges of the graph.  While it is an innovative approach, the technique can only be applied to small circuits because of ‘state-space explosion’. One possible solution to this problem could be to represent the behavioral models as a graph structure, and find an input that covers all the reachable states for the behavioral model. Subsequently, these input signals from the behavioral domain can be mapped to electrical domains and equivalence checking between the two domains may be carried out using approaches similar to those proposed in this thesis. However, mapping signals from behavioral to electrical domains while still maintaining complete coverage is not trivial. Figure 24 illustrates the one-to-mapping from behavioral to electrical state-space. Any trivial mapping from behavioral to electrical domains while maintaining complete coverage will lead to a large number of input sets over which equivalence checking needs to be performed. This would lead to large simulation times. This large simulation time necessitates the need for developing innovative techniques to perform this one-to-many mapping while maintaining formal nature of the coverage without resorting to a full-blown simulation.

As described above many challenges still need to be solved before a fully robust

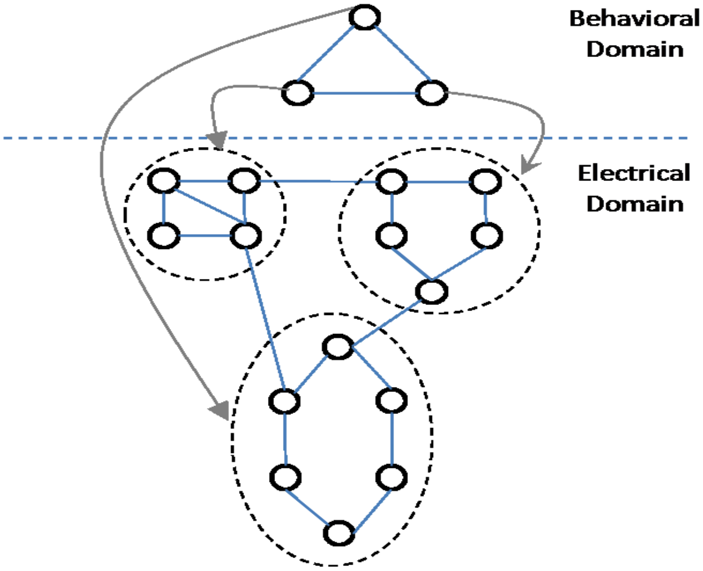


**Figure 24.** State space description for AMS circuits in behavioral and electrical domains

and efficient verification methodology for large analog/mixed signal systems can be developed. This thesis describes our small steps towards realizing this long term goal.

# REFERENCES

[1] P. J. Ashenden, G. D. Perterson, and D. A. Teegaden, *The System Designer's Guide to VHDL-AMS*, 1st ed. San Francisco, CA: Morgan Kaufmann Publishers, 2004.

[2] K. S. Kundert, *The Designer's Guide to Verliog-AMS*, 1st ed. Boston, MA: Kluwer Academic Publishers, 2004.

[3] R. A. Rutenbar, G. G. Gielen, and B. A. Antao, *Computer-Aided Design of Analog Integrated Circuits and Systems*, 1st ed. New York: IEEE Press, 2002.

[4] M. H. Zaki, S. Tahar and G. Bois "Formal verification of analog and mixed signal designs: A survey," *Microelectronics Journal*, vol. 39, pp. 1395–1404, 2008.

[5] S. Steinhorst and L. Hedrich, "Advanced methods for equivalence checking of analog circuits with strong nonlinearities," *Formal Methods in System Design*, vol. 36, no. 2, pp. 131-147, June 2010.

[6] S. Tiwary, A. Gupta, J. Phillips, C. Pinello and R. Zlatanovici, "First steps towards SAT-based formal analog verification," in *Proc. IEEE/ACM Int. Conf. CAD,* Nov. 2009, pp.1-8.

[7] A. Balivada, Y. V. Hoskote and J. A. Abraham, "Verification of transient response of linear analog circuits," in *Proc. IEEE VLSI Test Symposium*, April 1995, pp. 42-47.

[8] L. Hedrich, and E. Barke, "A formal approach to nonlinear analog circuit verification," in *Proc. IEEE/ACM Int. Conf. CAD*, Nov. 1995, pp.123-127.

[9] L. Hedrich, and E. Barke, “A formal approach to verification of linear analog circuits with parameter tolerances” in *Proc. Design Automation and Test in Europe*, Feb. 1998, pp. 649-655.

[10] T. R. Dastidar and P. P. Chakrabarti, “A verification system for transient response of analog circuits,” *ACM Trans. Design Autom. Electron. Syst.,* vol. 12, no 3, pp. 1-39, August 2007.

[11] T. Dang, A. Donze and O. Maler, “Verification of analog and mixed-signal circuits using hybrid system techniques,” *Formal Methods in Computer Aided Design, Lecture Notes in Computer Science*, vol. 3312, Berlin: Springer, 2004.

[12] S. Gupta, B. H. Krogh and R. A. Rutenbar, “Towards formal verification of analog designs,” in *Proc. IEEE/ACM Int. Conf. CAD, Nov.* 2004, pp. 210-217.

[13] T. Kropf, *Introduction to Formal Hardware Verification*, 1st ed. Berlin: Springer, 2000.

[14] A. Ghosh and R. Vermuri, “Formal verification of synthesized analog circuits,” in *Proc. IEEE/ACM Int. Conf. CAD,* Nov. 1999, pp. 40-45.

[15] K. Hanna, “Automatic verification of mixed-level logic circuits,” *Formal Methods in Computer-Aided Design, Lecture Notes in Computer Science*, vol. 1522, Berlin: Springer, 1998.

[16] G. Al Sammane, M. Zaki and S. Tahar, “A symbolic methodology for the verification of analog and mixed signal designs,” in *Proc. Design Automation and Test in Europe*, April 2007, pp. 249-254.

[17] R. Narayanan, B. Akbarpour, M. H. Zaki, S. Tahar, and L. C. Paulson, "Formal verification of analog circuits in the presence of noise and process variations," in *Proc. Design Automation and Test in Europe*, March 2010, pp.1309-1312.

[18] C. W. Ho, A. E. Ruehli, and P. A. Brennan, "The modified nodal approach to network analysis," *IEEE Trans. Circuits and Syst.,* vol. 26, no. 6, pp.504-509, June 1975.

[19] S. Seshdhari and J. A. Abraham, "Frequency response verification of analog circuits using global optimization techniques," *Journal of Electronic Testing, Theory and Applications*, vol. 17, no. 5, pp. 395-408, Oct. 2001.

[20] A. Salem, "Semi-formal verification of VHDL-AMS descriptions," *in Proc. IEEE Int. Symp. Circuits and Syst.,* May 2002, pp.333-336.

[21] W. Hartong, R. Klausen, and L. Hedrich, "Formal verification for non-linear analog systems: approaches to model and equivalence checking," *in Advanced Formal Verification*, R. Drechsler ed., Netherlands: Kluwer Academic Publishers, 2004, pp. 205-245.

[22] S. Steinhorst, and L. Hedrich, "Improving verification of analog circuit blocks by state-space guided transient simulation," *in Proc. IEEE Int. Symp. Circuits and Syst.*, June 2010, pp. 645-648.

[23] W. Hartong, L. Hedrich, and E. Barke, "Model checking algorithms for analog verification," *in Proc. IEEE/ACM Des. Autom. Conf.,* June 2002, pp. 542-547.

[24] E. M. Clarke, O. Grumberg, and D. A. Peld, *Model Checking*, Cambridge, MA: MIT Press, 2000.

[25] R. P. Kurshan and K. L. McMilla, "Analysis of digital circuits through symbolic reduction," *IEEE Trans. Computer Aided Design*, vol. 10, no. 11, pp. 1350-1371, Nov. 1991.

[26] M. R. Greenstreet and I. Mitchell, "Reachability analysis using polygonal projections*," in Hybrid Systems: Computation and Control, Lecture Notes in Computer Science,* vol. 1569, Berlin: Springer, 1999.

[27] G. Frehse, "PHaver: algorithmic verification of hybrid systems past HyTech," *Hybrid Systems: Computation and Control, Lecture Notes in Computer Science*, vol. 3414, Berlin: Springer, 2005.

[28] S. Little, D. Walter, N. Seegmiller, C. J. Miller and T. Yoneda, "Verification of analog and mixed signal circuits using timed hybrid petri nets," in *Automated Technology for Verification and Analysis, Lecture Notes in Computer Science,* vol. 3299, Berlin: Springer, 2004.

[29] S. Little, D. Walter, N. Seegmiller, C. J. Miller and T. Yoneda, "Verification of analog/mixed signal circuits using hybrid labeled petri nets," in *Proc. IEEE/ACM Int. Conf. CAD*, Nov. 2006, pp. 275-282.

[30] S. Tiwary, A. Gupta, J. Phillips, C. Pinello and R. Zlatanovici, "First steps towards SAT-based formal analog verification," in *Proc. IEEE/ACM Int. Conf. CAD,* Nov. 2009, pp.1-8.

[31] M. Horowitz, M. Jeeradit, F. Lau, S. Liao, B. C. Lim and J. Mao, "Fortifying analog models with equivalence checking and coverage analysis," *in Proc. IEEE/ACM Des. Autom. Conf., Nov.* 2010, pp. 425-430.

[32] B. C. Lim, J. Kim and M. A. Horowitz, "An efficient test vector generation for checking analog/mixed-signal functional models," *in Proc. IEEE/ACM Des. Autom. Conf.*, June 2010, pp. 767-762.

[33] P. Spellucci, "An SQP method for general nonlinear programs using only equality constrained sub-problems," *Mathematical Programming*, vol. 82, pp. 413–448, 1993.

[34] P. Spellucci, "A new technique for inconsistent problems in the SQP Method," *Mathematical Methods of Operation Research,* vol. 47, pp. 355-400, 1998.

[35] S. Steinhorst, and L. Hedrich, "Improving verification coverage of analog circuit blocks by state space-guided transient simulation," *in Proc. IEEE Int. Symp. Circuits and Syst.,* May 2010, pp.645-648.

## VITA

Name: Amandeep Singh

Address: C/O Dr. Peng Li  
Department of Electrical and Computer Engineering  
Texas A&M University  
College Station, TX 77843-3259

Email Address: amandeep.singh@tamu.edu

Education: B.E., Electrical Engineering, Punjab Engineering College, India, 2008  
M.S., Electrical & Computer Engineering, Texas A&M University, 2011