LAYOUT-ACCURATE ULTRA-FAST SYSTEM-LEVEL DESIGN EXPLORATION THROUGH VERILOG-AMS

Geng Zheng

Dissertation Prepared for the Degree of

DOCTOR OF PHILOSOPHY

UNIVERSITY OF NORTH TEXAS

May 2013

Zheng, Geng. Layout-accurate ultra-fast system-level design exploration through Verilog-AMS. Doctor of Philosophy (Computer Science and Engineering), May 2013, 103 pp., 18 tables, 35 figures, 110 numbered references.

This research addresses problems in designing analog and mixed-signal (AMS) systems by bridging the gap between system-level and circuit-level simulation by making simulations fast like system-level and accurate like circuit-level. The tools proposed include metamodel integrated Verilog-AMS based design exploration flows. The research involves design centering, metamodel generation flows for creating efficient behavioral models, and Verilog-AMS integration techniques for model realization. The core of the proposed solution is transistor-level and layout-level metamodeling and their incorporation in Verilog-AMS. Metamodeling is used to construct efficient and layout-accurate surrogate models for AMS system building blocks. Verilog-AMS, an AMS hardware description language, is employed to build surrogate model implementations that can be simulated with industrial standard simulators. The case-study circuits and systems include an operational amplifier (OP-AMP), a voltage-controlled oscillator (VCO), a charge-pump phase-locked loop (PLL), and a continuous-time delta-sigma modulator (DSM). The minimum and maximum error rates of the proposed OP-AMP model are 0.11 % and 2.86 %, respectively. The error rates for the PLL lock time and power estimation are 0.7 % and 3.0 %, respectively. The OP-AMP optimization using the proposed approach is ~17000× faster than the transistor-level model based approach. The optimization achieves a ~4× power reduction for the OP-AMP design. The PLL parasitic-aware optimization achieves a 10× speedup and a 147 µW power reduction. Thus the experimental results validate the effectiveness of the proposed solution.

ACKNOWLEDGMENTS

I would like to express my deepest appreciation to my major professor, Dr. Saraju P. Mohanty, and my co-major professor, Dr. Elias Kougianos, for their support, guidance, and invaluable advise throughout my doctoral degree program at the University of North Texas. I am also grateful to my Ph.D. committee members, Dr. Paul Tarau and Dr. Song Fu, for reviewing my progress and my dissertation, and for providing inspirational suggestions. I would like thank all my labmates at the NanoSystem Design Laboratory (NSDL, `http://nsdl.cse.unt.edu`), including, Oleg, Karo, and Mahesh for their help and collaboration. I would like to extend my appreciation to the staff of the Department of Computer Science and Engineering for their very helpful support.

Special thanks to my mother Xingying Chen and my father Wanfa Zheng. Words cannot express my gratitude for their support, help, and sacrifices. Last but not the least, I would like to thank all my friends who have supported me throughout.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

System-level design exploration is a critical step in designing integrated circuits (ICs) under stringent time-to-market and yield requirements. System-level simulations are typically behavioral in nature in which the underlying semiconductor technology is abstracted. By performing system-level design exploration, a designer can profile many aspects of a range of candidate designs and center the design at an optimal point. Thus the design time and resources spent on non-optimal design solutions can be reduced. The core part of design exploration involves estimating design performance in a design space using a design model. Evaluating a modern analog and mixed-signal system-on-chip (AMS-SoC) or their building blocks using traditional transistor-level models is computationally prohibitive [33]. Hardware description languages (HDLs) such as Verilog-AMS, VHDL-AMS, and SystemC-AMS provide different ways to behaviorally model AMS systems and their building blocks [103]. On one hand, behavioral models described in HDLs greatly accelerate the design exploration process; on the other hand, most existing behavioral models do not successfully incorporate realistic circuit block non-idealities, which leads to substantial prediction error. In this dissertation, employing metamodeling techniques to create efficient yet accurate behavioral models that capture circuit-level and layout-level non-idealities is studied. Verilog-AMS is adopted to describe the behavioral models for system-level design exploration.

1.1. Behavioral Simulation

Behavioral simulation is a technique that performs system-level or architectural level simulations of a system without resorting to computationally expensive schematic transistor-level or layout-level simulations [66]. Typically the heart of the behavioral models consist of mathematical equations, functional descriptions, algorithms, or look-up-tables (LUTs) that behaviorally describe the input or output relation and the characteristics of circuits, systems, sub-systems, or architecture components [30, 61, 60, 3]. Developing traditional behavioral models requires the understanding of the circuits and systems. Developing fast and accurate behavioral models suiting a specific simulation need requires a high degree of understanding. Researchers have been working to develop approaches to take advantages of the high-speed behavioral simulations at the same time enabling them to capture more of the underlying technology information so that behavioral simulation is as accurate as possible. Such techniques include macromodeling, metamodeling, and LUT integration, which are derived from structural information of the circuits and systems.

Metamodels are special behavioral models. The core of a metamodel is still mathematical equations [24]. However, these equations usually belong to a certain template. The input and output relation is constructed through sampling the circuit and system design space. This way the requirement on the design expertise is lessened. Also, it has the potential to achieve better speed and accuracy trade-offs.

Macromodels are models that also aim at reducing the simulation cost. Marcomodeling simplifies the original transistor-level circuits and systems by extracting the important elements and replacing them with ideal components [10]. This simplification limits the accuracy of macromodels since the accuracy directly depends on the complexity.

It may be noted that the terms macromodels and metamodels, while often used interchangeably in the literature, they are very distinct. Behavioral models including metamodels for analog and mixed-signal circuits and systems are typically described in hardware description language such as Verilog-AMS and VHDL-AMS. They can be also implemented in numeric tools such as MATLAB and its time-domain simulation environment—Simulink and Simscape. Macromodels are typically built and simulated using conventional circuit simulators such as SPICE [88].

1.2. Languages and Tools for Mixed-Signal System Modeling

Fast and accurate time-domain simulations are crucial in modern analog and mixed-signal design and verification. Mixed-signal simulators that support analog and digital behavioral simulations started to emerge at around 1990, e.g., Saber [30] and XSPICE [17]. Saber provides a modeling language that allows users to efficiently create behavioral models for complex analog circuits. XSPICE is an extension to the well-known SPICE simulator.

SPICE is a low-level simulator for circuit-level analog simulations. XSPICE introduced analog and digital co-simulation algorithms that support event-driven simulation approaches. XSPICE also came with a variety of built-in functional models such as integrators, multipliers, etc. Users can create custom behavioral models through the C programming language. All of these features greatly enhanced XSPICE's capability for system-level simulation. Nevertheless, digital circuits and systems are usually described in Verilog or VHDL [65, 1]. Saber and XSPICE are not compatible with these standard digital hardware description languages, which led to low acceptance of these two tools.

NGSPICE [87] is an open-source simulator package for mixed-signal simulations. SPICE and XSPICE are included in NGSPICE and have been reconstructed to improve efficiency and robustness. NGSPICE also includes a code generator, ADMS, for converting Verilog-A or Verilog-AMS models into C code that conforms to the SPICE simulator API. Since the resultant code is still simulated using the circuit-level simulator, SPICE, this approach does not offer good efficiency for system-level simulations. Gnucap [34] is another open-source tool for mixed-mode and mixed-signal simulations. It supports behavioral and event-driven simulations. Like Saber and XSPICE, it is incompatible with Verilog and VHDL.

SIMULINK, Simscape, Verilog-AMS, VHDL-AMS, SystemC-AMS are well-known tools that can model a system and its building blocks behaviorally. Initially, SIMULINK with extensive built-in functional block targets, was used primarily at system-level simulations [50]. Lately its capability of modeling low-level circuits and devices has been enhanced through the introduction of the Simscape physical modeling language [63]. Both SIMULINK and Verilog-AMS have their own strengths. In this dissertation the design and modeling of a CT DSM for biopotential signal acquisition using these two tools are studied. The design procedure is divided into several steps. Based on the purpose of each step, different models are needed for speed and accuracy tradeoffs. The reasons behind the choice of the tools and the modeling language for each step are discussed. The designed third-order CT DSM with feedforward loop filter has 87.3 dB signal-to-noise ratio (SNR) and 20 kHz input bandwidth.

1.3. Deficiency in System-level Languages to Capture Circuit Level Characteristics

Parasitics which are present at the layout-level or circuit level of a chip or system design greatly degrade the performance of nano-CMOS circuit designs. The parasitics may originate from both the devices (semiconductor transistors, diodea, inductors, capacitors and resistors) as well as the metal interconnects or wires which together constitute the chip design. The parasitics may include resistance (R), capacitance (C), inductance (L), mutual inductance (L_m), and mutual capacitance (C_m). They cause significant discrepancy between schematic-level and layout-level simulations of circuits and systems. To overcome the parasitic effects to accurately meet the design specifications, numerous iterations between the schematic and the layout are usually required. The design closure may consequently require great amounts of time and effort [32]. Layout verification is the major obstacle because the iteration time is mainly spent on layout modification and simulations. Behavioral models that are capable of representing circuit layouts will dramatically shorten the design cycle. Various behavioral models have been developed for AMS systems, e.g., [43, 42, 76]. Parasitic effects, however, were not discussed in these papers. Also, the circuit models in these papers were implemented as Verilog-A modules rather than Verilog-AMS modules which are generally more efficient. Thus handling modern consumer electronic device modeling (like mobile phones) which are intrinsically mixed-signal in nature in a unified fashion is not possible. Modeling techniques such as model order reduction [97] and symbolic model generation [9] have been also proposed but they only work for small circuits.

Both polynomial and nonpolynomial metamodeling techniques for nano-CMOS AMS circuits and systems have been proposed in the literature [24, 28]. The models built with this method accurately reflect the parasitic effects. In this dissertation, an accurate voltage-controlled oscillator (VCO) behavioral model is proposed based on this approach. This behavioral model is implemented using the Verilog-AMS language which enables fast simulations. Combining the metamodeling technique and the Verilog-AMS simulation, the design verification process achieves a large speedup and maintains reasonably high accuracy. In fact, not only the proposed Verilog-AMS behavioral model can help the design verification, it can also assist the design optimization and accurate design space exploration. A phase-locked loop (PLL) design with an LC VCO design using a 180 nm CMOS process is used to demonstrate the proposed metamodeling technique and the Verilog-AMS based new framework. Among different PLL architectures, the charge-pump PLL (CPPLL) has been widely used in various system due to its simplicity and effectiveness. Thus this PLL architecture is used in this dissertation.

1.4. The Novel Idea of Integration of Circuit (transistor or layout) Characteristics in System Level

In order to achieve high performance and high yield, an analog/mixed-signal circuit or system must be optimized at both system and circuit levels. For a top-down design approach, this starts with designing and optimizing the system with sub-block models at high levels of abstractions (such as system or architecture level). The specifications for each sub-system (a.k.a. sub-block, component, module) that lead to the best system performance are then obtained. Each sub-block is then designed and optimized toward these specifications. The issue with this approach is that generating an accurate model of the sub-block, if it is even feasible, takes significant effort. Therefore some characteristics of the sub-blocks are ignored at the high-level simulation. This makes the resulting sub-block specifications less reliable, and less accurate. Design exploration using such sub-blocks will lead to sub-optimal circuits and systems and consequently loss of yield.

To address the aforementioned issues in mixed-signal design exploration, a three-step design optimization flow is proposed in this dissertation using circuits like OP-AMP and PLL as case studies. The proposed flow is assisted by polynomial metamodels (POMs) to significantly reduce the design optimization cycles. In the first step, based on the specifications from high-level simulations, an optimized OP-AMP design is obtained by ultra-fast POM-assisted optimization. This optimized design serves as the starting point for constructing an OP-AMP meta-macromodel in the second step. The meta-macromodel is then integrated into a Verilog-AMS module which is used at system-level simulations and can greatly reduce the computation time. The third step performs optimization at the system level. The computation time is greatly reduced due to the use of the **Verilog-AMS po**lynomial **m**eta-macromodel (which is called **Verilog-AMS-POM** in this work). The meta-macromodel is the polynomial metamodel generated from the macromodel (i.e. transfer function or SPICE macromodel). Since the optimization is performed at system-level with the circuit-level metamodel based on the OP-AMP design, the resulting final OP-AMP design has a much higher chance of meting the system requirements and attaining much higher performance. For the optimization a customized Cuckoo Search algorithm is used for the first time for OP-AMP optimization through Verilog-AMS-POM. At the same time **Verilog-AMS-PAM** is proposed to integrate layout-level parasitics in Verilog-AMS for layout-accurate system design exploration. **P**arasitic **a**ware **m**etamodels are integrated in Verilog-AMS thus called Verilog-AMS-PAM.

1.5. Organization of this Dissertation

The organization of the rest of this dissertation is as follows. Chapter 2 reviews commonly used metamodeling techniques, two popular modeling options—Verilog-AMS and Simulink—and their uses. Chapter 3 further studies AMS system modeling with Verilog-AMS and Simulink using a DSM example. Chapter 4 proposes a Verilog-AMS integrated polynomial metamodeling approach for fast and accurate AMS design exploration. Chapter 5 presents a method for creating parasitic-aware Verilog-AMS metamodels for layout-accurate AMS system design exploration. Chapter 6 concludes this dissertation and suggests future research. Fig. 1.1 illustrates this organization.

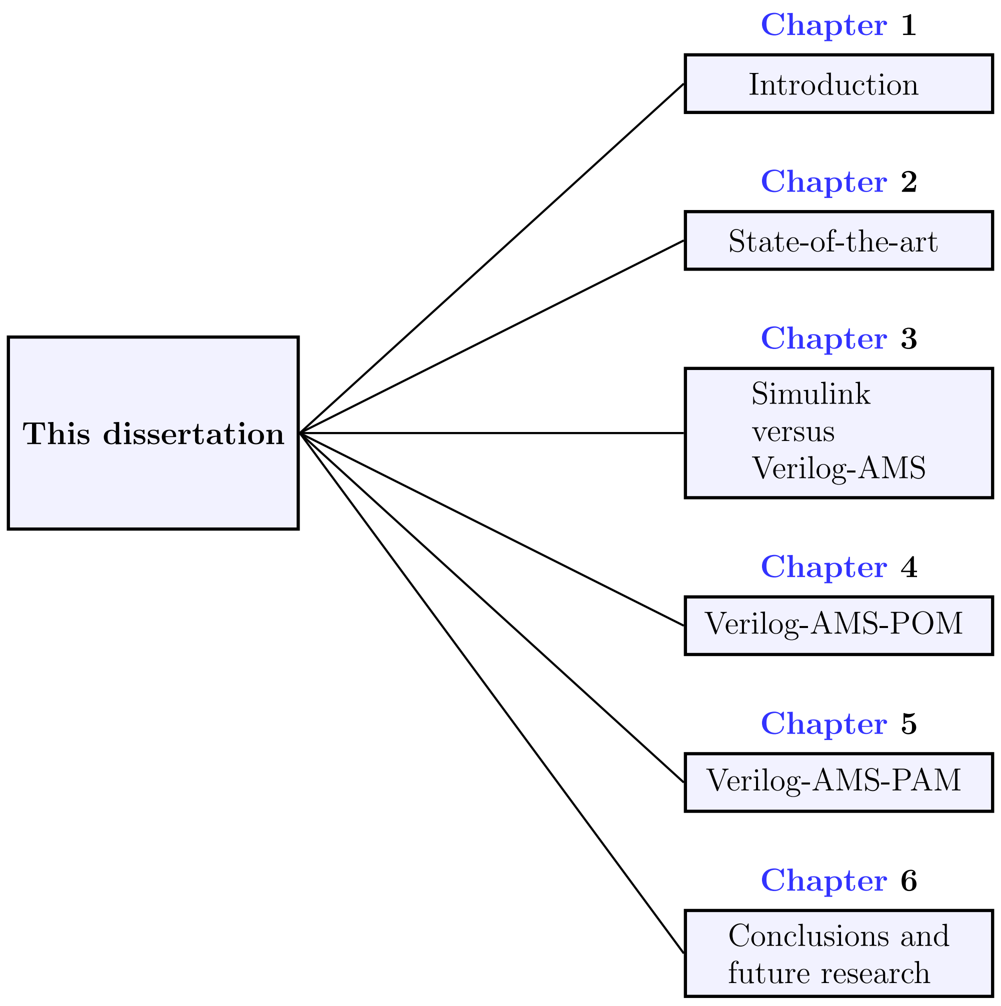


FIGURE 1.1. The organization of this dissertation.

CHAPTER 2

REVIEW OF STATE-OF-THE-ART

System-level accurate analog/mixed-signal design exploration requires significant integrated effort. While analog portions are typically custom designed, digital portions are top-down abstract driven hierarchical designs. Metamodeling leverages the creation of accurate and efficient surrogates in place of transistor-level models. Modeling languages and their simulation environments offer viable solutions to implement these surrogate models. Section 2.1 reviews the commonly used metamodeling techniques. Section 2.2 discusses popular lookup-table based surrogate models in Verilog-AMS. Section 2.3 reviews the popular modeling environment MATLAB/Simulink. Section 2.4 introduces two typical case study mixed-signal system building blocks. Section 2.5 summarizes the contributions of this dissertation to advance the state-of-the-art.

2.1. Review of Metamodels

The major types of metamodels that have been used to model electronic circuits and systems are the following:

(1) Polynomials [21, 57, 58, 25]
(2) Artificial neural networks (ANNs) [96, 44, 47, 27]
(3) Splines [90, 91, 6, 48]
(4) Support vector machines (SVMs) [18, 49, 77, 8]
(5) Kriging models [102, 36, 101, 74, 75]

Typical analog building blocks that have been used in metamodeling studies include the following designs:

(1) Operational amplifiers (OP-AMPs)
(2) Operational transconductance amplifiers (OTAs)
(3) Bandgap references (BGRs)

Typical RF building blocks that have been used in metamodeling studies include the following designs:

(1) Low-noise amplifiers (LNAs)

(2) Power amplifiers

(3) Microwave filters

The LC voltage-controlled oscillator (LC VCO) is a typical mixed-signal block and the phase-locked loop (PLL) is a typical mixed-signal subsystem. They are all common metamodeling case study circuits. In additional, metamodeling has also been extensively applied to nanometer devices such as carbon nanotube (CNT) transistors. Figure 2.1 samples publications that present these major metamodeling approaches and their model targets.

Polynomial metamodels rely on traditional polynomial regression to approximate circuit and system response surfaces. Splines are piecewise polynomial metamodels and hence can potentially be more accurate metamodels than polynomial metamodels. Artificial Neural networks (ANNs) and support vector machines (SVMs) are intelligent metamodels that are based on machine learning and have been a topic of research for large circuits. Kriging metamodels stem from geostatistics. Kriging is receiving attention recently as a metamodel method to capture correlations in device parameters and in particular nanoelectronics process variations. A comparative study of these metamodels can be found in [64]. The benchmark circuit was an OTA in 0.7 μm CMOS process with 13 selected design variables. Table 2.1 shows the model accuracy obtained in [64] for a polynomial (PO), a artificial neural network (ANN), multivariate adaptive splines (MARS), a support vector machine (SVM), and a Kriging (KG) function. The metamodel outputs being evaluated were the OTA performance metrics: low-frequency gain (A_0), phase margin (PM), offset voltage (V_{os}), unity-gain frequency (f_u), and positive and negative slew rate (SR$_p$ and SR$_n$). Table 2.2 compares the cost of metamodel construction in terms of modeling effort and time for the aforementioned approaches based on the results in [64]. The construction effort is estimated using the numbers of lines of the MATLAB code for generating the metamodels. The construction time was the CPU execution time in the computer, when the model templates were selected and

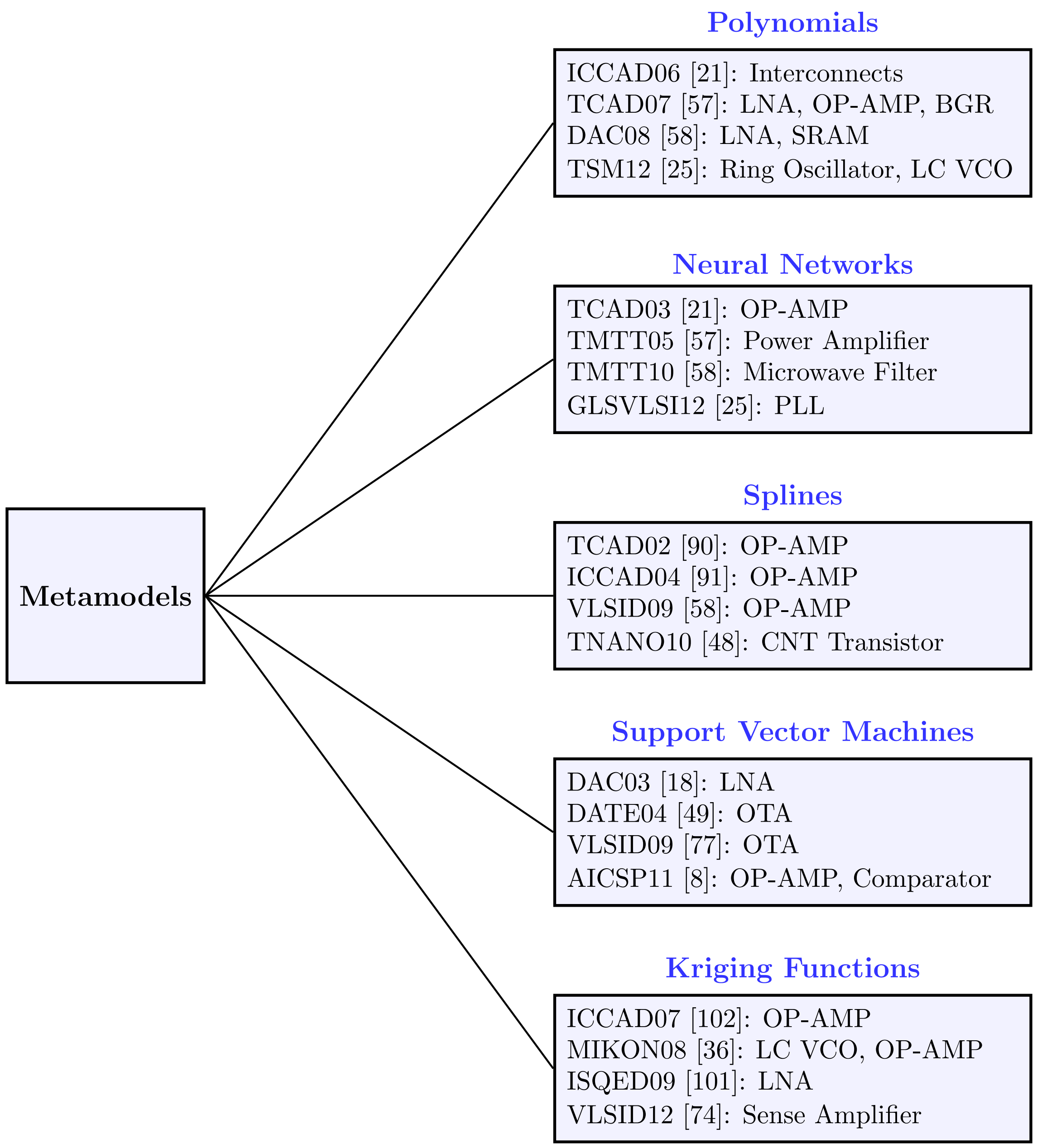


FIGURE 2.1. A sampling of major metamodeling approaches.

after all necessary sample data had been obtained, that it took to build the metamodels (i.e., to compute the model coefficients for the selected templates).

It can be seen that MARS results in low error rate compared to other models. The accuracy differences between ANN, KG, and SVM are not significant. The prediction error rate of PO is higher than other models. However, it is the most accessible metamodel since it take less effort to build. In this case, PO required 25 lines of MATLAB code and less than 1

Table 2.1. Comparison of Metamodel Prediction Error(%) of a CMOS OTA [64].

Performance	**PO**	**ANN**	**KG**	**SVM**	**MARS**
A_0	14.7	5.0	7.3	11.5	4.4
PM	11.7	6.8	3.8	5.8	1.8
V_{os}	4.3	2.9	2.2	1.8	1.2
f_u	20.8	9.3	7.3	12.7	9.4
SR$_p$	33.8	8.2	8.9	10.0	7.2
SR$_n$	16.1	9.5	5.1	4.1	5.4

Table 2.2. Comparison of the Metamodel Construction Cost in [64].

Construction	**PO**	**ANN**	**KG**	**SVM**	**MARS**
Effort (# of MATLAB lines)	25	N/A	200	N/A	500
Time	$<$ 1 min	3.7 min	5 min	5 min	5 min
Circuit: A 0.7 μm CMOS OTA with 13 selected design variables. Sampling: 243 design points from full orthogonal-hypercube sampling.					

minute construction time, compared to hundreds of lines of code and increasing construction time for other approaches. Thus polynomial metamodels are easy to create, but they may not capture all the characteristics of complex circuits. In addition, the knowledge required to properly train the intelligent metamodels such as ANN and SVM can be a hidden cost. It is worth mentioning that a modeling approach, named CAFFEINE, that is not limited to a specific model template was also evaluated in [64]. It formulates the modeling as a multi-objective optimization problem and achieves high model accuracy and efficiency. The penalty is the increased model construction effort—the OTA CAFFEINE model took 12 hours to build. This study adopts a polynomial metamodel for its accessibility. When high accuracy is needed, splines can be used instead with the least additional effort since they are essentially extensions to polynomials.

2.2. Verilog-A or Verilog-AMS Based Modeling with LookUp Tables (LUTs)

Lookup tables (LUTs) are textual files that store samples of design variable and circuit or system response values. Similar to the proposed metamodels of this dissertation, they do not carry any physical information of the circuits or systems being modeled. LUTs can be useful in parametric behavioral simulation and enable behavioral languages performing accurate simulations while maintaining simulation speed. For example, LUTs can be embedded in a circuit behavioral model to estimate certain circuit or system responses for a given set of design variable values. If the given design variable values happen to be identical to any set of values in the LUTs, the resulting circuit responses can be directly read from the tables. This saves tremendous amount of simulation time and speeds up the simulation. If the given values are not identical to any stored values, interpolation is then applied between two or multiple sampled data points to obtain estimated circuit responses. However, the computational cost of interpolation is substantially lower compared to a SPICE simulation, especially if all parasitics are included in the SPICE netlist. This is where metamodels are easier, faster, and accurate compared to LUTs. In the metamodel approach no data needs to be stored. Any value needed is obtained just from the metamodel on the fly. This calculation does not cost much as it is merely a function evaluation and much faster than the netlist simulation. LUTs are often used in the following scenarios for accurate simulation through textual description of the chip design:

(1) The physical insight for yielding an analytical or physical model is not available.
(2) The available analytical or physical models are too simple to accurately model the circuit or system behavior.
(3) The available models are too costly computation-wise for the specific task.

LUTs had been used by many researcher in behavioral models of devices, analog and mixed-signal circuits and systems. The existing literature has large collection of different types of LUT based behavioral simulation approaches. In [83], LUTs were used to model the device parameters described in Verilog-A such as drain-to-source current, gate-to-drain and gate-to-source capacitance of inter-band tunneling field-effect transistors (TFETs). In [98] a LUT in Verilog-A was used to model single-event transients (SETs). SETs have become the dominant radiation effects in microelectronics. The LUT based Verilog-A model sped up the simulation by over 18,000 $\times$ compared to the mixed-model simulation (the simulation mixes

SPICE and device-level simulators). In [37] a Verilog-AMS model with embedded LUTs was implemented for nonlinear oscillators in order to replace time-consuming SPICE models. The LUTs modeled the oscillators' phase sensitivity to perturbations and their voltage outputs. In [39] a VCO Verilog-AMS model with LUTs was created for fast mixed-mode (SPICE + Verilog-A or Verilog-AMS) PLL simulation. The LUTs generated VCO output frequency and deviation of the frequency. Compared to transistor-level simulation, speedups of 2-3 orders of magnitude were achieved. Both [109] and [2] presented system-level PLL optimization using Verilog-A building blocks with LUTs. The Pareto fronts of the building block circuit parameters were stored in LUTs that were implemented in Verilog-A. The design space for each building block was thus shrunk to the Pareto points. This increases the efficiency of the system-level optimization and design exploration.

The major limitation of LUT techniques is that for high accuracy the table sizes grow rapidly which in turn can greatly increase the required memory and computation time. Metamodels overcome this limitation. In particular metamodel-integrated Verilog-AMS explicitly has everything embedded in one text representation of the target design. Consequently LUTs are restricted to low-dimensional and weakly nonlinear modeling problems. For instance, a formal method for automatically generating multi-dimensional LUTs was proposed in [92]. The LUTs were optimized in a systematic way under an error-control algorithm. Thus the sizes of the resulting LUTs were minimized for a given accuracy requirement. Nevertheless, it was shown that the accuracy is approximately proportional to the table size up to a certain break point. Little further accuracy improvement can be gained by increasing the table size beyond the break point. Different types of metamodels like polynomial (piecewise), and ANN, are well suited to capture the nonlinearity and better represent the behavior of complex circuits and systems.

2.3. Simulink Based AMS System and Circuit Modeling

Simulink is a multi-domain simulation environment backed by the well-established data processing capability of MATLAB. While MATLAB is completely text based, Simulink provides visual blocks for better system-level understanding of the design. Simulink has a user-friendly graphical editor and provides a large collection of built-in libraries with large numbers of predefined functional blocks. Users can also create custom blocks to meet their needs and perform first step simulations of their target system to analyze its feasibility before moving to next steps of design. Using a signal-flow simulation approach, time-domain simulation can be performed in Simulink for electrical, mechanical, and many other systems. The aforementioned features make Simulink a preferable tool for rapid functional verification. This is evidenced by the suggested uses of Simulink for engineering educational projects [85, 40, 59]. The objects being modeled in these projects include a magnetic levitation device, motor controllers, and switch-model power supplies.

Simulink has been used to model analog/mixed-signal and radio frequency (RF) circuits and systems. For instance, Simulink models for thermoelectric modules (TEMs), semiconductor devices built specially for temperature control/measurement applications, were presented in [89]. In [20], Simulink behavioral simulation was used to aid the study of bipolar junction transistors' parasitic effects on a BiCMOS analog decoder. To simulate RF systems, a set of Simulink blocks for RF receiver front-ends were presented in [70]. The RF building blocks had been modeled to include diverse examples in the RF domain including low-noise amplifier (LNA), mixers, and oscillators. General nonidealities such as thermal noise, nonlinearity, as well as block-specific nonidealities such as oscillator phase noise and mixer offset were included in the Simulink models.

For Simulink modeling of mixed-signaled circuits and systems, delta-sigma modulators (DSMs) are a very good case study. A DSM is a typical mixed-signal system. It consists of pure analog blocks (for example, OP-AMP), mixed-signal blocks (for example, comparator and oscillator), and possibly digital filters. In [11], Simulink models for DSM blocks were presented to perform DSM behavioral simulation with nonidealities such as sampling jitter, and noise. Block parameters such as OP-AMP finite gain, finite bandwidth, and slew-rate were included in the models. In [80], Simulink models were used as tools for high-level DSM design synthesis. Three types of DSM implementations (switched-capacitor, switch-current, and continuous-time) were studied. Using Simulink behavioral models the DSM system was simulated. Major building blocks were constructed using the S-function blocks in Simulink. The results were validated by the results from HSPICE simulations and the measured silicon results. In [16], a statistical approach was proposed for checking mixed-signal circuit properties. A MATLAB/Simulink implementation was built to examine the stability of a DSM.

Simulink is a powerful tool for system-level simulations. It is not specialized for either transistor-level and layout-level simulations. However, the proposed metamodels can enable circuit-accurate system-level simulation in Simulink. This is a key novelty of the circuit-accurate metamodel integration in Simulink or Verilog-AMS. As compared to Verilog-AMS and VHDL-AMS, Simulink solvers are not built into standard circuit simulators. To work with conventional circuit simulation tools for low-level design implementation, Simulink provides coupler blocks to communicate with other tools to perform co-simulation. Nevertheless, careful configuration should be done to the coupler blocks and the efficiency of the conventional simulation may be degraded. This is a blessing in disguise: Simulink and EDA tools are decoupled and consequently Simulnk simulation doesn't suffer the loss of speed during the simulation. At the same time they are coupled at system level using scripting languages such as ocean in Cadence. In other ways, MATLAB scripts can be written the drive EDA tools from it.

2.4. Modeling of Important Mixed-Signal Building Blocks

2.4.1. *Voltage-Controlled Oscillator*

Voltage-controlled oscillators are typical mixed-signal blocks that are often used as case-study circuits [32]. Verilog-A behavioral modules of linear VCOs were used in [45] for PLL jitter characterization and in [108] for aiding a hierarchical CPPLL sizing method. No parasitic effects were included in the model used in that research. A quasi-sensitivity analysis approach is proposed in [54] to construct behavioral models and used a characterization mode developed in [53] to extract the circuit parameters including parasitic effects. The authors also adopted the linear VCO model. While the linear model may be sufficient for performing verification on fixed designs, it is not useful for design exploration since the VCO linearity condition is not always valid. The VCO behavioral models developed in [2] and [39] used the table-lookup approach inside Verilog-A modules, which is not efficient. An event-driven analog modeling approach was proposed in [94] which used the Verilog-AMS `wreal` data type to improve the model efficiency. However, it is not clear how the VCO gain and output frequency were modeled.

2.4.2. *Operational Amplifier*

Macromodeling is a popular technique to generate simpler circuit representations for reducing simulation time. In [95], the OP-AMP to be modeled was first divided into basic building blocks and then these blocks, based on their functionality, were replaced with appropriate simplified models composed of ideal circuit elements. A symbolic expression relating the input and output response can be generated through algebraic or graph-based methods [81]. Traditionally symbolic analysis is suitable for modeling the small-signal behaviors, not for large-signal transient responses such as the OP-AMP slewing and settling. A method was proposed in [104] to overcome this shortcoming.

In [55] and [93], Verilog-A OP-AMP behavioral models were used to estimate the specifications used in a switched-capacitor filter. In [19], the continuous-time transfer function of an OP-AMP under voltage follower configuration was discretized and modeled using VHDL. A VHDL-AMS OP-AMP behavioral model was proposed in [82]. The VHDL-AMS OP-AMP model in [5] was based on a three-stage model. In [7], a framework for extracting circuit parameters was proposed.

2.5. Research Contributions of this Dissertation

The contributions of this dissertation consist of a complete set of electronic design automation (EDA) computer-aided-design (CAD) tools for performing fast and accurate design exploration for analog/mixed-signal circuits and systems. These tools include the following: (1) approaches for circuit-level metamodel integration in Verilog-AMS, (2) Metamodel-integrated Verilog-AMS based design exploration flow for finding optimal design, and (3) metamodel generation flows for creating efficient behavioral models. The proposed exploration flows are compatible to both simple and sophisticated optimization algorithms. For example, both an exhaustive search and a novel Cuckoo search optimization algorithm are demonstrated in this dissertation. The contributions of this dissertation are summarized as follows:

(1) The comparative study of delta-sigma modulator (DSM) modeling using Verilog-AMS and Simulink.

(2) A fast and accurate analog/mixed-signal (AMS) circuit and system design exploration flow based on Verilog-AMS integrated polynomial metamodels and meta-macromodeling.

(3) An efficient and layout-accurate mixed-signal system design exploration flow based on parasitic-aware Verilog-AMS metamodels.

(4) A polynomial metamodel generation flow for AMS building blocks for Verilog-AMS integration.

(5) A metamodel-assisted exhaustive search algorithm for phase-locked loop (PLL) system optimization.

(6) A metamodel-assisted Cuckoo search algorithm for OP-AMP optimization.

(7) An OP-AMP metamodeling approach for constructing parametric Verilog-AMS model for system-level design exploration.

(8) A parasitic-included VCO metamodel for PLL system design exploration.

CHAPTER 3

MIXED-SIGNAL SYSTEM MODELING: SIMULINK VERSUS VERILOG-AMS

Fast and accurate time-domain simulations are crucial in today's analog and mixed-signal (AMS) simulation, design exploration, and verification. Verilog-AMS, VHDL-AMS, Simulink, and SystemC-AMS are few known frameworks for modeling and simulation of mixed-signal circuits and systems. Based on design experiences two of these frameworks have been selected for their analysis for easy usage, speed, and accuracy for AMS circuits and systems. SIMULINK and Verilog-AMS are two well-known tools that can model a system and its building blocks behaviorally. Both have their own strengths and weaknesses. In this dissertation study of the design and modeling of a continuous-time delta-sigma modulator (CT DSM) for biopotential signal acquisition using these tools is performed. The design procedure was divided into several steps for easy and logical understanding. Based on the purpose of each of the modeling steps, different models are needed. The reasons behind the choice of the tools and the modeling language for each step are discussed in [106]. The designed third-order CT DSM with feedforward loop filter has 87.3 dB SNR and 20 kHz input bandwidth.

3.1. A Typical Mixed-Signal System

A typical mixed-signal system consists of analog, digital and/or mixed-signal blocks. Analog-to-digital converters (ADCs) and digital-to-analog converters (DACs) are typical mixed-signal circuits. Both ADC and DAC are highly essential in modern AMS circuits and systems at least as interface circuit among the analog and digital blocks.

An analog-to-digital converter (ADC) converts signals that are continuous in time and amplitude into discrete-time and discrete-amplitude signals. Common outputs of an ADC are binary digital signals (the amplitude is either logic 1 or 0). The major architectures of the ADC include the following:

- Flash ADC
- Successive-approximation (SAR) ADC

- Pipeline ADC
- Delta-sigma ADC

The flash ADC features the simplest structure and fast conversion speed. The core components are a set of clocked comparators and digital circuitry for converting the digital output format. Poor matching of nano-CMOS devices significantly limit the resolution of flash ADCs. Other ADC architectures trade off simplicity and conversion speed for higher resolution. Analog blocks such as anti-aliasing filters, sample-and-hold (S/A) circuits, and comparators, as well as digital blocks for control logic or calibration are common in most ADCs.

In contrast to an ADC, a digital-to-analog converter (DAC) converts digital signals (discrete in time and amplitude) into analog signals. The simplest DAC is the resistor-string DAC which relies on voltage division through a string of resistors to achieve the conversion. Element matching and the required element count limit it to very low resolution applications. DACs with the $R-2R$ architecture reduce the required resistor count by incorporating binary-weighted mechanisms and op amps. Current-steering and charge-scaling ADCs employ transistors and capacitors, respectively, to translate digital signal magnitude into the analog domain. OP-AMPs are commonly used at the outputs of these ADCs. More sophisticated DACs include pipeline DACs and delta-sigma DACs with components such as S/A and OP-AMPs.

A delta-sigma modulation (DSM) is a very good example of a mixed-signal system. Usually one or multiple OP-AMPs together with passive components (capacitors and possibly resistors) form the core of a loop filter which is an essential component for noise shaping. Single-bit or multi-bit clocked comparators are employed to convert the outputs of the loop filter to digital signals. These digital signals are processed by the following digital filter to remove the noise. The unfiltered digital signals are also converted back to analog form and then subtracted from the input analog signals. This forms a mixed-signal system with feedback. The clocked comparators typically need to be driven by low-to-median jitter clock signals. Thus an oscillator is required as an integral block.

3.2. High-level Design Description Delta-Sigma Modulator (DSM)

A delta-sigma modulator (DSM) typically consists of several blocks as follows:

- Loop filter
- Analog-to-digital coveter (ADC)
- Feedback digital-to-analog converter (DAC)

High-level block diagrams of a discrete time delta-sigma modulator (DT CSM) and a continuous time delta-sigma modulator (CT DSM) are shown in Fig. 3.1. The discrete-time realization of the DT DSM and continuous-time loop filter CT DSM are represented as, $L(z)$ and $L(s)$, respectively. The ADC which is usually referred to as a quantizer, converts its input $y[n]$ to digital output code $v[n]$. The feedback DAC converts $v[n]$ back to analog form so it can be subtracted from the input $u(t)$. In this process, the signal and the quantization noise experience different transfer functions as follows:

- Signal transfer function (STF)
- Noise transfer function (NTF)

The major difference between the DT DSM and the CT DSM is that the DT DSM samples the analog input outside the loop while the CT DSM samples the signal at the loop filter (LF) output. The sampling is controlled by a clock signal ϕ with sampling frequency f_s. A CT DSM has an implicit anti-aliasing filter (AAF) and generally consumes less power than a DT DSM with similar structure [29]. These advantages make the CT DSM a better candidate over DT DSM for biomedical applications.

The quantizer that generates the digital output can be either single-bit or multibit. A multibit quantizer provides more aggressive noise-shaping, lower jitter sensitivity, and better stability. However, its implementation is much more complex than the single-bit quantizer. Also, the multibit quantizer requires complex circuitry to correct its internal element mismatch due to process variations. A single-bit quantizer is used in this dissertation for its low power consumption which is especially interesting for biomedical applications. Another important design parameter is the out-of-band gain (OBG). It determines the gain for the signal with frequency component at f_s. Higher OBG offers less in-band noise at the

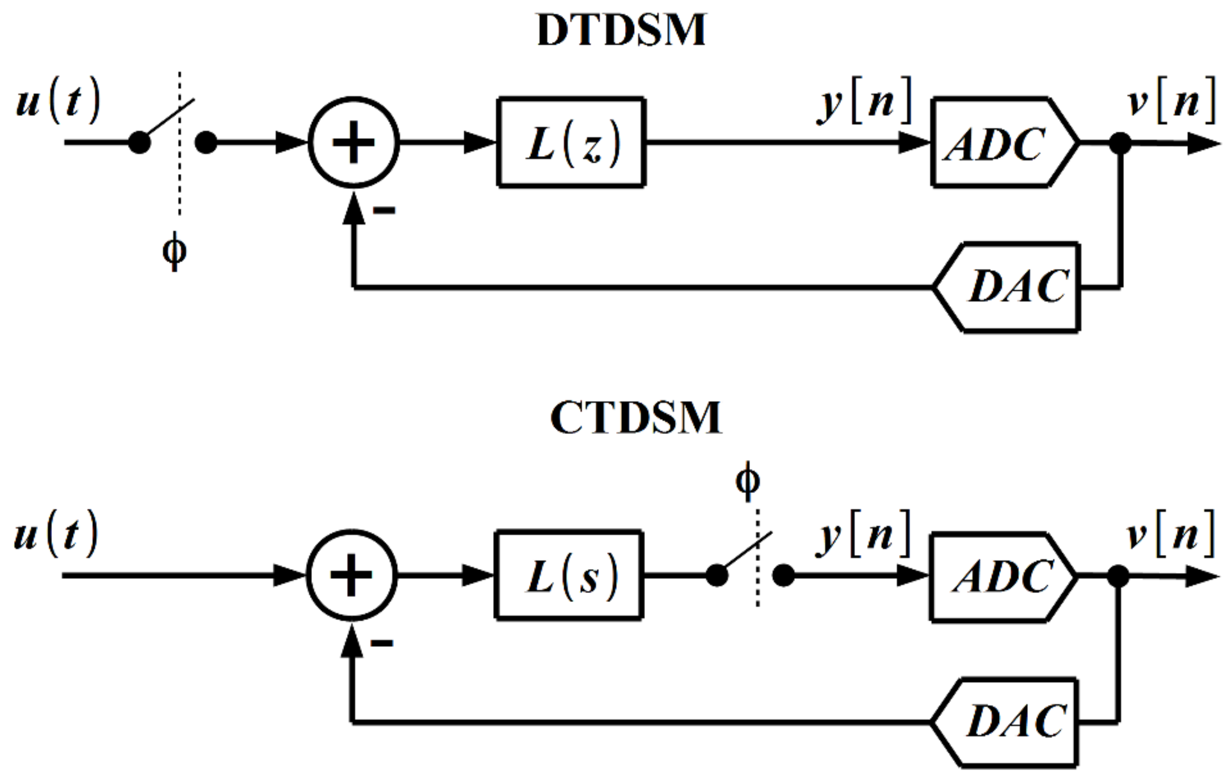


FIGURE 3.1. The DT and CT DSM system-level block diagrams.

cost of higher jitter noise and increased instability. An OBG of 1.3 is chosen for the design based on simulations.

3.3. ADC for Biopotential Signal Acquisition

The designed CT DSM is to be used in portable biomedical systems in order to monitor signals in electrocardiography (ECG), electromyography (EMG), and electroencephalography (EEG) signals. The monitoring is enabled by measuring biopotentials on the surface of living tissue. Such systems play an important role in lowering the health care cost and in simplifying clinical procedures. A simplified multi-channel biopotential signal acquisition system is depicted in Fig. 3.2. The ADC employed in such systems must process biomedical signals with a reasonable accuracy, consume extremely low power, and tolerate process variations of nanometer CMOS technology. ADCs using a CT DSM fit into such systems very well for the following reasons:

(1) They can attain high accuracy with relatively simple circuitry.

(2) The requirements for the operational amplifier (op amp) employed in such ADCs are relaxed compared to those in other architecture.

(3) Their built-in anti-aliasing filters (AAFs) simplify the analog front-end design and greatly reduce the system power requirement.

(4) With proper design, they can achieve high immunity from process variations.

The CT DSM should have at least 10-bit resolution and be able to handle input signals up to 10 kHz to meet the biomedical application requirements [100, 110].

3.4. System-level Design and Modeling of CT DSM

The theories and tools for discrete-time (DT) DSM design (e.g., [72, 84]) are quite mature compared to the CT DSM design. Thus a common way of designing a CT DSM is first obtaining a system-level DT DSM design with desired performance using the available tools. Then this design is mapped to a CT DSM topology. This dissertation follows this approach. The MATLAB toolbox provided in [84] has been widely used and is the *de facto* tool for DT DSM synthesis. It is adopted in this design flow. The CT DSM system-level design flow along with the tools and modeling languages involved in each design step are shown in Fig. 3.3. The flow starts by synthesizing a system-level DT DSM design using the MATLAB toolbox with system design attributes including the following: DSM order, oversampling ratio (OSR), and out-of-band gain (OBG. The resulting DT DSM design is then converted to a CT implementation. Necessary dynamic range scaling is done to ensure the outputs of all stages of the modulator remain bounded. Simulations are performed throughout the process to predict the design performance and thus to ensure the requirements are met. Critical non-idealities are modeled and simulated, which leads to reasonable specifications for each modulator building blocks. The following subsections detail each step in the flow and justify the decisions on the choice of the modeling language and the tool for different steps.

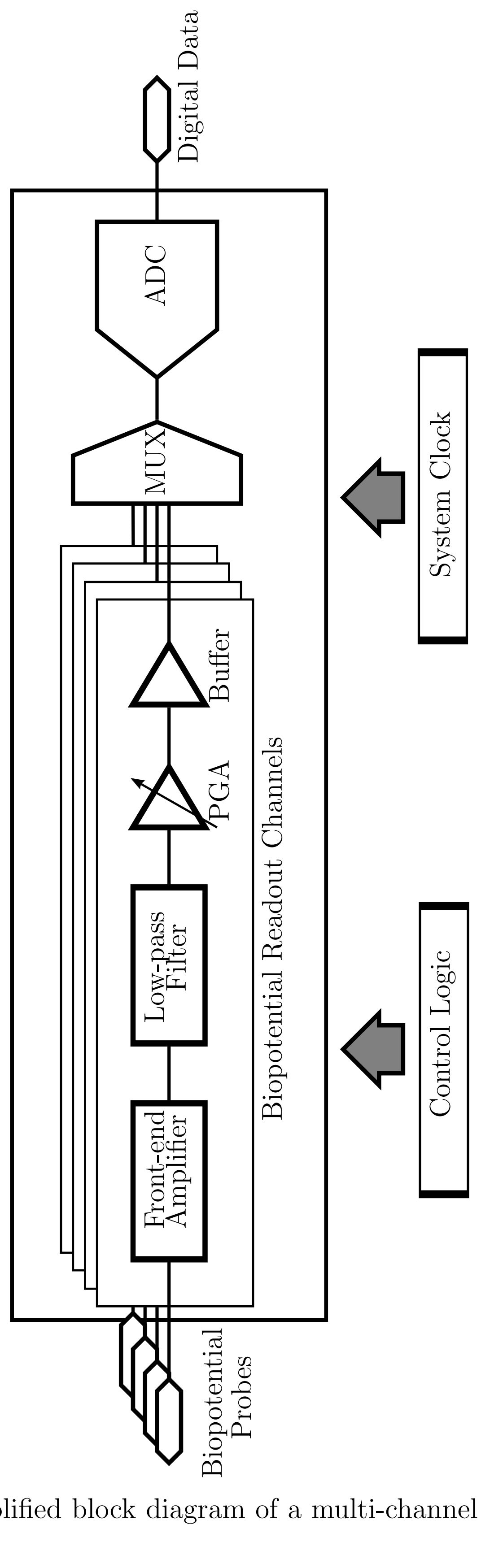


FIGURE 3.2. Simplified block diagram of a multi-channel biopotential signal acquisition system.

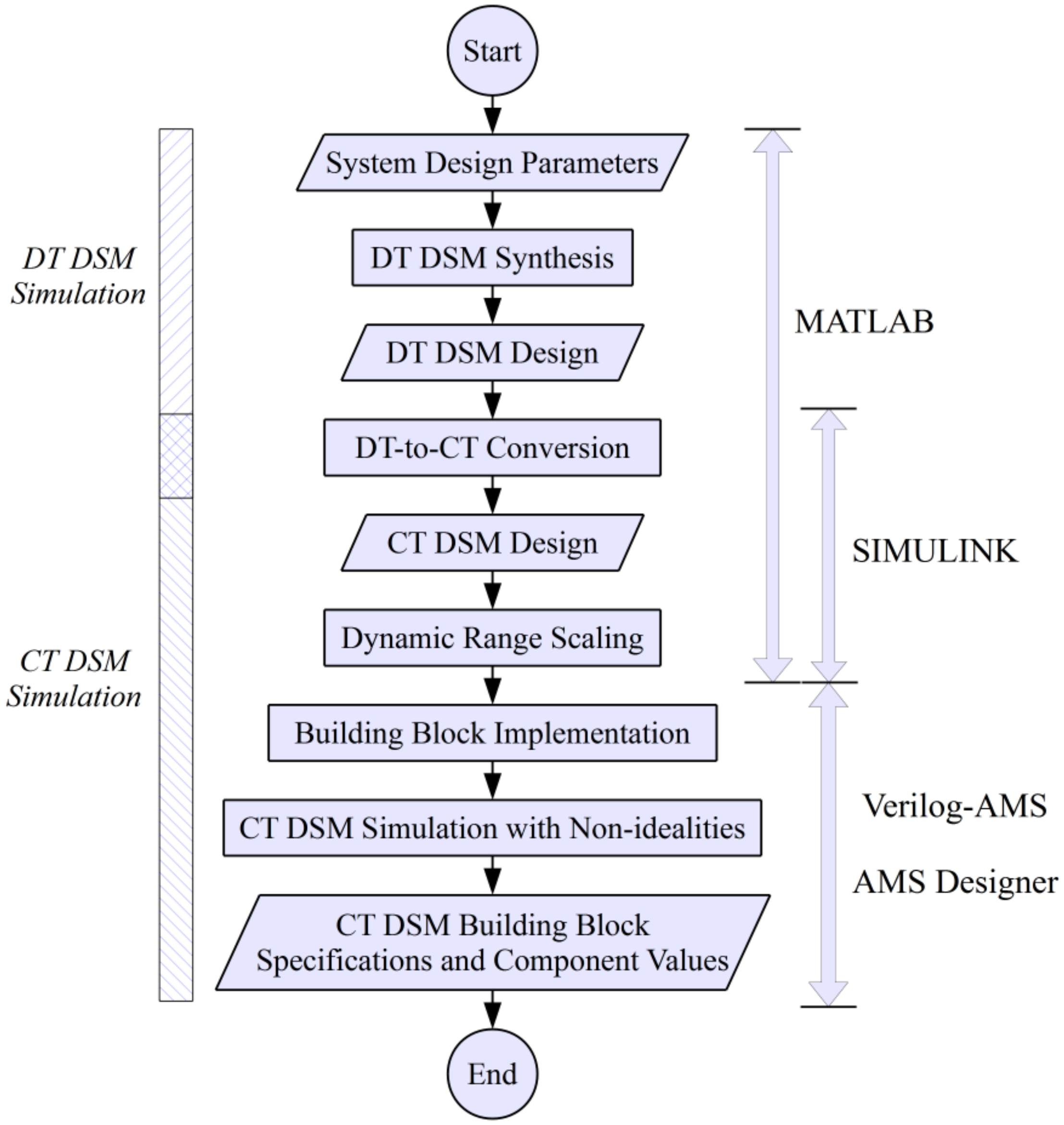


FIGURE 3.3. System-level CT DSM design flow.

3.4.1. *DT DSM Design Synthesis*

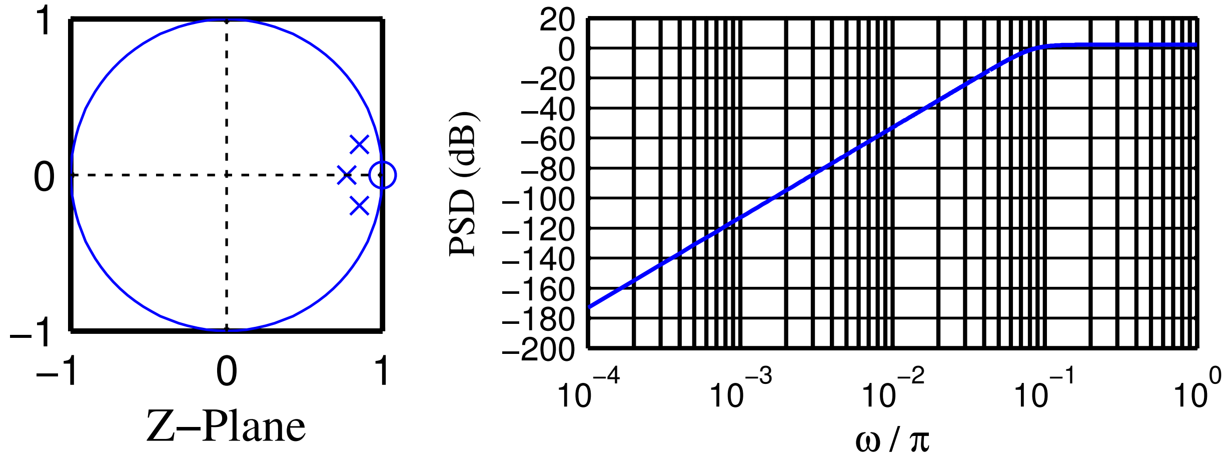
The goal of this step is to find a proper DT LF so that the DSM design satisfies the given (bandwidth and SNR) requirements. The STF is typically assumed to be unity. The NTF and LF are related as the following expression:

$$NTF(z) = \frac{1}{(1 + L(z))}. \tag{1}$$

Thus one can obtain a LF by designing the NTF first. This is done by taking into account the high-level design considerations (e.g., quantizer levels, OBG, OSR, etc.) and performing simulations iteratively until a satisfying solution is reached. This step is similar to conventional filter design. By using the functions and scripts such as `synthesizeNTF` provided in the MATLAB delta-sigma design toolbox [84], tedious manual iteration can be avoided. `synthesizeNTF` assumes the magnitude of the NTF denominator is constant in the passband and numerically determines the pole locations for the NTF. An NTF zero optimization option is provided in `synthesizeNTF` that can reduce the total noise power in the signal band. That is, by spreading the NTF zeros over the signal band instead of placing them all at DC ($z = 1$), the NTF can be improved. However, this is at the cost of circuit complexity and here the zero optimization is not enabled. The resultant NTF for this design has the following expression:

$$NTF(z) = \frac{(z-1)^3}{(z-0.770)(z^2-1.708z+0.768)}. \tag{2}$$

The NTF is evaluated in the z-plane and frequency domain as shown in Fig. 3.4 where an initial evaluation of the design stability and performance can be made.



(A) NTF poles/zeros (B) NTF Power Spectrum

FIGURE 3.4. The synthesized NTF power spectrum and its poles and zeros in the z-domain.

MATLAB simulations were run using the facility in the toolbox and the DT DSM output power spectrum density (PSD) is plotted in Fig. 3.5. It shows that this design results in an SNR of 100.5 dB which is sufficient for the ADC requirements. We can proceed to the next step with this DT DSM design. The resulting LF is shown in Eqn. (3) presented below:

$$L(z) = \frac{0.513(z^2 - 1.756z + 0.783)}{(z-1)^3}. \tag{3}$$

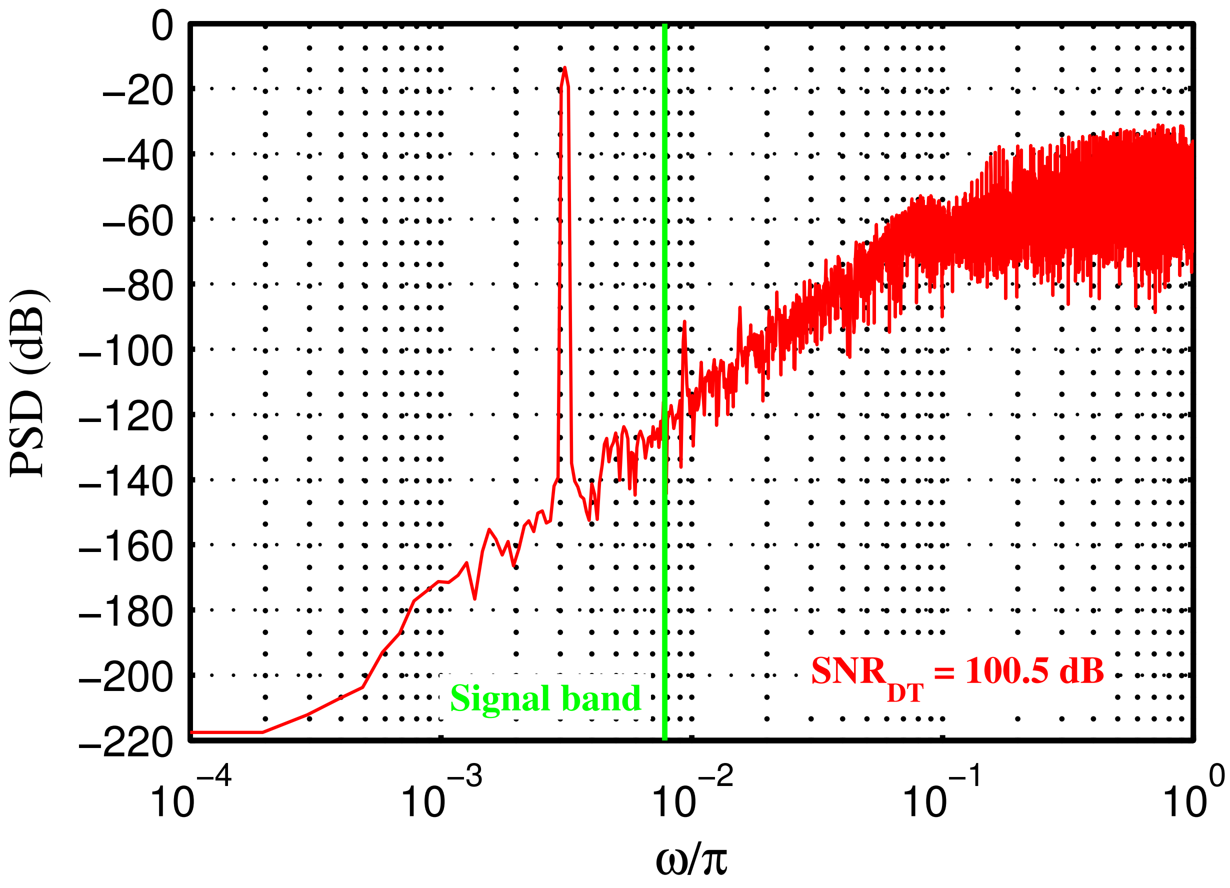


FIGURE 3.5. DT DSM output PSD from MATLAB simulations.

3.4.2. *DT-to-CT Conversion*

A practical and effective DT-to-CT conversion method was proposed in [78] and is used in this dissertation. The proposed numerical technique accounts for the LF degradation caused by non-ideality such as op-amp finite gain-bandwidth product. The objective is to find a CT equivalent of the DT DSM design presented in Section 3.4.1. The procedure is:

1) selecting a CT DSM architecture; and 2) computing coefficients for the LF of the selected architecture. A comparison of various architectures can be found in [29]. Here a third-order cascade of integrators with feedforward (CIFF) LF (shown in Fig. 3.6) is chosen.

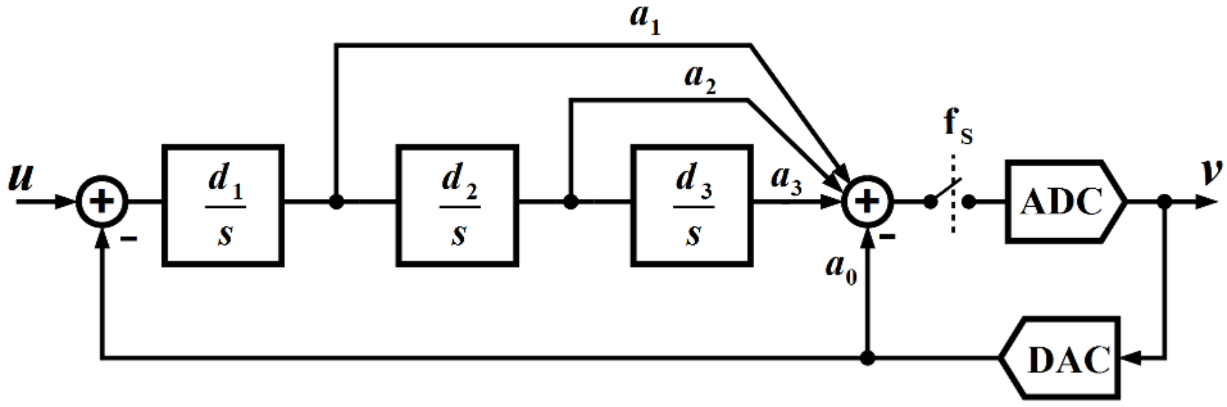


FIGURE 3.6. The selected CT DSM structure.

In order to compute the LF coefficients, the impulse responses of all integrators of the CT DSM and that of the DT DSM from time-domain simulations have to be known. A MATLAB script is used to control the simulation flow, gathering the results, and performing numerical fitting to find the LF coefficients. The CT DSM model used in time-domain simulations can be constructed using SIMULINK or Verilog-AMS. SIMULINK's built-in libraries provide a comprehensive collection of fundamental building blocks such as integrators, quantizers, summers, etc. With SIMULINK one can quickly build DSM behavioral models for time-domain simulations. In contrast, if the DSM behavioral models are built using Verilog-AMS, one has to go through basic steps such as writing code and creating symbols for the fundamental building blocks. Therefore, SIMULINK is used to create the DSM behavioral models in this step. The SIMULINK model of the CT DSM is shown in Fig. 3.7. Since dynamic range scaling presented in Section 3.4.3 altered the LF coefficients obtained in this step, the final values for the coefficients are shown in that section.

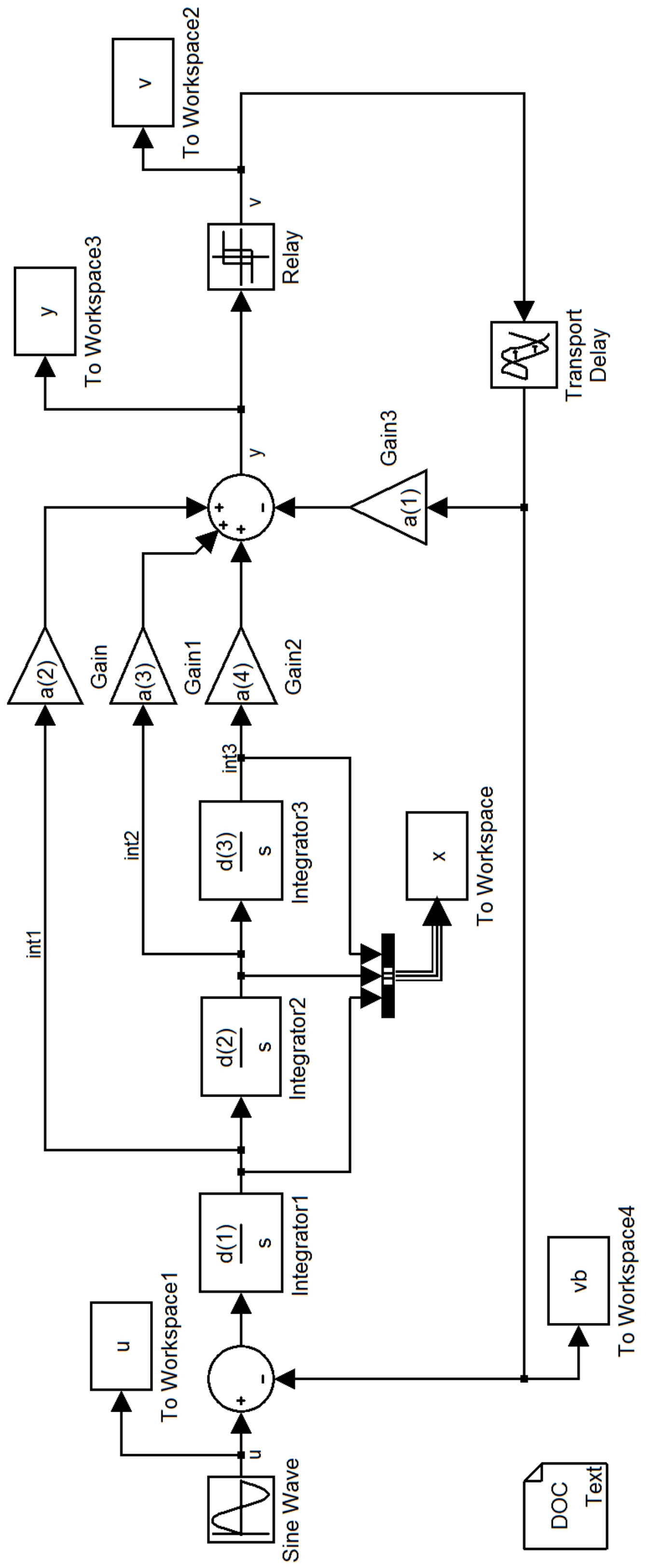


FIGURE 3.7. MATLAB/SIMULINK model of the CT DSM.

3.4.3. *Dynamic Range Scaling*

In real circuit implementations all the voltages should be bounded within the power supply range. For the CT DSM implementation shown in Fig. 3.6, the integrator outputs are scaled to the allowable range so that saturation is avoided. While previous design steps do not account for this constraint, here it can be done by adjusting the values of LF coefficients d_1–d_3. Note that the values of a_1–a_3 have to be modified accordingly [84]. This step also requires time-domain simulations. The SIMULINK model presented in Section 3.4.2 is simply reused without extra effort. The resultant LF coefficients are the following:

$$[a_1, a_2, a_3, a_4] = [0.0948, 1.2090, 0.5760, 0.8716]. \tag{4}$$

$$[d_1, d_2, d_3] = [0.4017, 0.5022, 0.0787]. \tag{5}$$

SIMULINK simulations were then run to generate CT DSM output. The PSD of the DT and the CT DSM outputs are compared in Fig. 3.8. It can be seen that their PSD plots match very well. The difference in SNR is due to the finite simulator accuracy. Thus a CT equivalent of the DT DSM has been obtained by following the steps discussed above.

3.4.4. *Building Block Implementation with Non-idealities*

With the system-level design of the CT DSM described in Sections 3.4.2 and 3.4.3, we can now assign specifications to each building blocks. The building blocks in the SIMULINK CT DSM model used in those sections are ideal blocks (i.e. the integrators have infinite dc gain, sampling clock has not jitter, etc.). For practical circuit implementation, the performance degradation caused by non-idealities should be taken into account. Thus non-idealities have to be modeled and simulated. When deciding the tool and modeling language to be used for non-ideality simulations, there are two major concerns, as follows:

(1) The modeling language should be able to describe the non-idealities and allow them to be integrated into the ideal model without a great deal of time and effort.

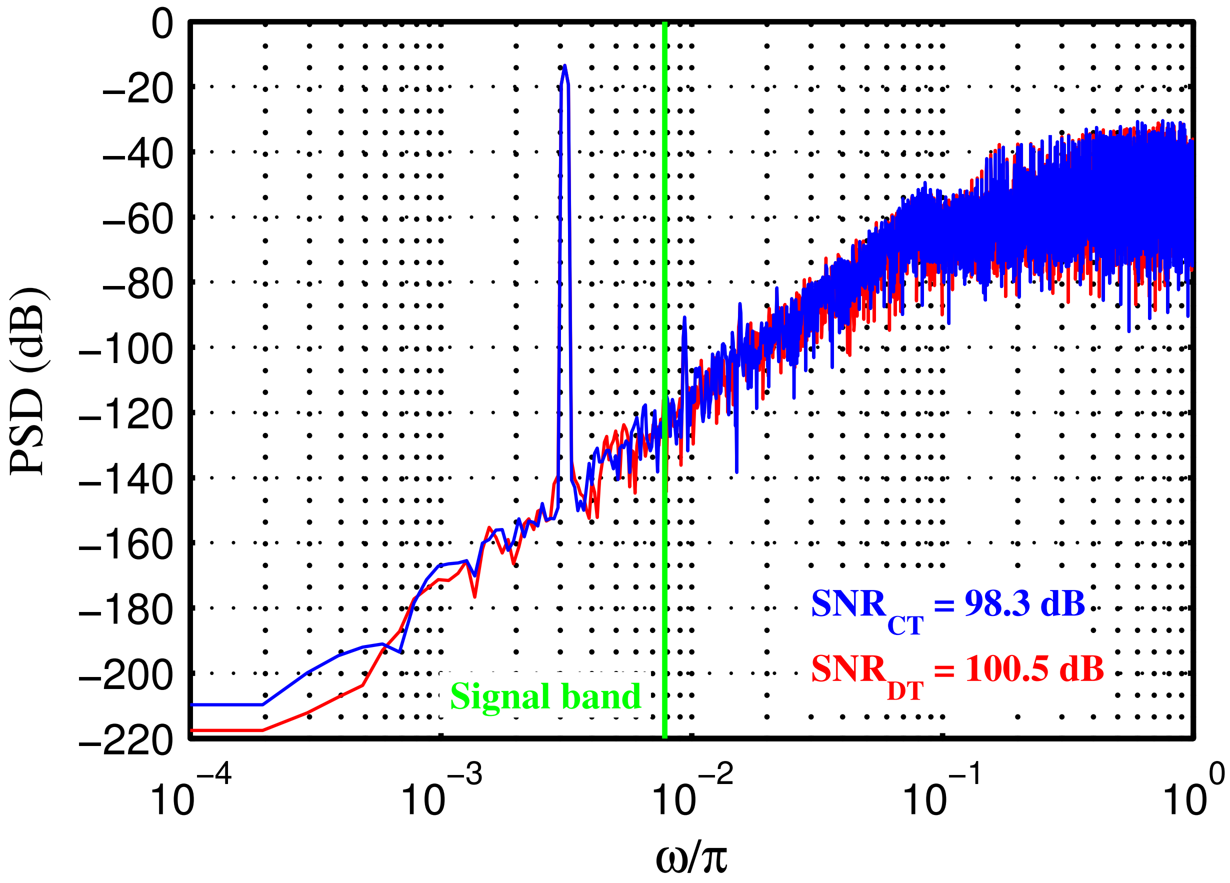


FIGURE 3.8. Comparison of the DT and CT DSM output spectra.

(2) The tool should allow the designer to switch each individual building block between ideal model, non-ideal model, and actual circuit implementation so the designer can quickly locate the source of problems during this process.

In this dissertation, two important non-idealities (finite gain-bandwidth product and clock jitter) for DSM were taken as examples to show how the modeling and simulation were done. In this dissertation Verilog-AMS and AMS Designer of CADENCE rather than MATLAB/SIMULINK are used to model the non-idealities. Since the actual circuit implementation (schematics and physical designs) were to be done in CADENCE, using AMS Designer allowed us to design and model the circuit in an unified environment. Although SIMULINK can co-simulate with AMS Designer and thus can also interact with the block of actual circuit implementation, it requires extra effort for configuration on both sides and the simulation procedure is not as convenient. Also, although in literature efforts have been made to model clock jitter in SIMULINK [41, 14, 4], it is relatively easier to model it in Verilog-AMS. The CT DSM implementation in this dissertation using active-RC integrators is shown in Fig. 3.9. A comparison of various realizations can be found in [29].

Finite Gain-Bandwidth Product. The schematic of the CT DSM shown in Fig. 3.9 includes the following components:

- three integrators,
- one summing amplifier, and
- a clocked quantizer.

The operational amplifier (OP-AMP) is the major component in the integrators and summing amplifier. Eqn. (6) and Eqn. (7) show the transfer function of an ideal op amp and that of an op amp with finite gain-bandwidth product (GBW), respectively, as shown below:

$$\frac{V_{out}(s)}{V_{in}(s)} = -\frac{1}{sRC}, \tag{6}$$

$$\frac{V_{out}(s)}{V_{in}(s)} = -\frac{1}{(RC + \frac{1}{\omega_{un}}) + s^2 \frac{RC}{\omega_{un}}}, \tag{7}$$

where $\omega_{un} = 2\pi \cdot \text{GBW}$ is the unity-gain frequency. It indicates that finite GBW introduces second-order effects. GBW $= A_{oldc} \cdot f_{3dB}$ where A_{oldc} is the OP-AMP DC gain and f_{3dB} is its bandwidth. This can be modeled using Verilog-AMS as shown in the example code in Listing 3.1. The simulated PSDs of the CT DSM output with various op-amp GBW are shown in Fig. 3.10. When the GBW is infinite, $\text{SNR}_{\text{Ideal}}$ is 97.6 dB (it is just slightly different from the SNR from SIMULINK simulation). $\text{SNR}_{\text{GBW(Lo)}} = 81.7dB$ is when the GBW is a very low value. $\text{SNR}_{\text{GBW(Optimal)}}$ corresponds to the op amp with the specifications shown in Table 3.1.

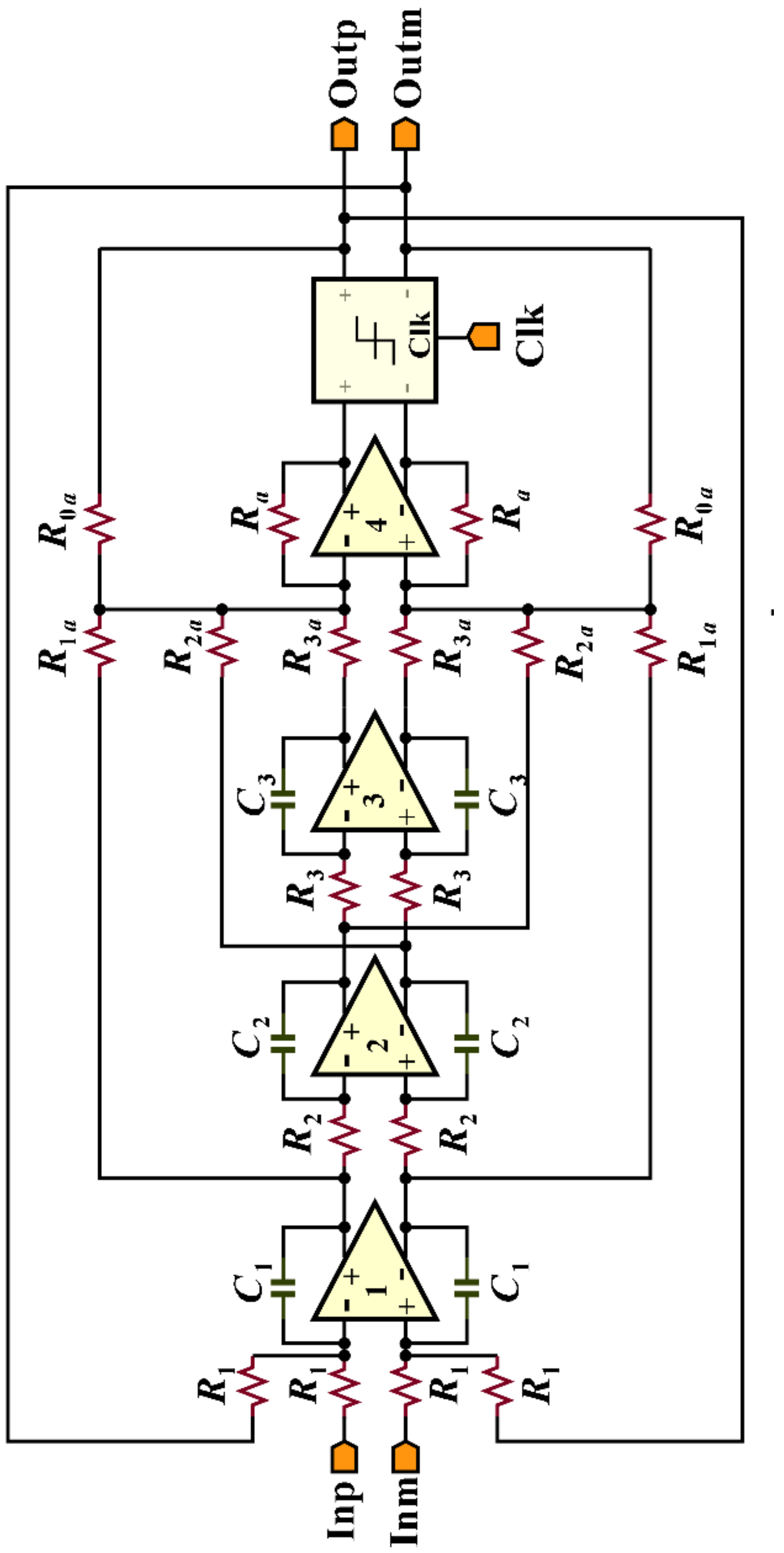


FIGURE 3.9. The CT DSM realization using active-RC integrators.

LISTING 3.1. Example Verilog-AMS source code for the integrator.

```
`include "constants.vams"
`include "disciplines.vams"
module int1_fd (outp, outm, inp, inm, fbp, fbm);
    parameter real A0 = 128.0 from (0 : inf); // Op-amp open-loop DC gain
    parameter real f0 = 5e6 from (0:inf);    // Op-amp 3dB bandwidth
    parameter real c = 1.0 from (0 : inf);   // Forward capacitor
    parameter real r = 1.0 from (0 : inf);   // Input resistor
    parameter real vdd = 1.0 from (0:10);
    parameter real gnd = 0.0;
    input inp, inm, fbp, fbm;
    output outp, outm;
    electrical inp, inm;
    electrical outp, outm, fbp, fbm, outd;
    real vcm, vout;
    real d [0:2];
    real wu;      // Unity-gain frequency
    analog begin
        @(initial_step)
        begin
            wu = A0 * 2 * `M_PI * f0;
            d[0] = 0;
            d[1] = r * c + 1 / wu;
            d[2] = r * c / wu;
            vcm = (vdd - gnd) / 2;
        end
        V(outd) <+ laplace_nd( (V(inp, inm) + V(fbp, fbm)), {1}, d );
        V(outp) <+ vcm + V(outd) / 2;
        V(outm) <+ vcm - V(outd) / 2;
    end
endmodule
```

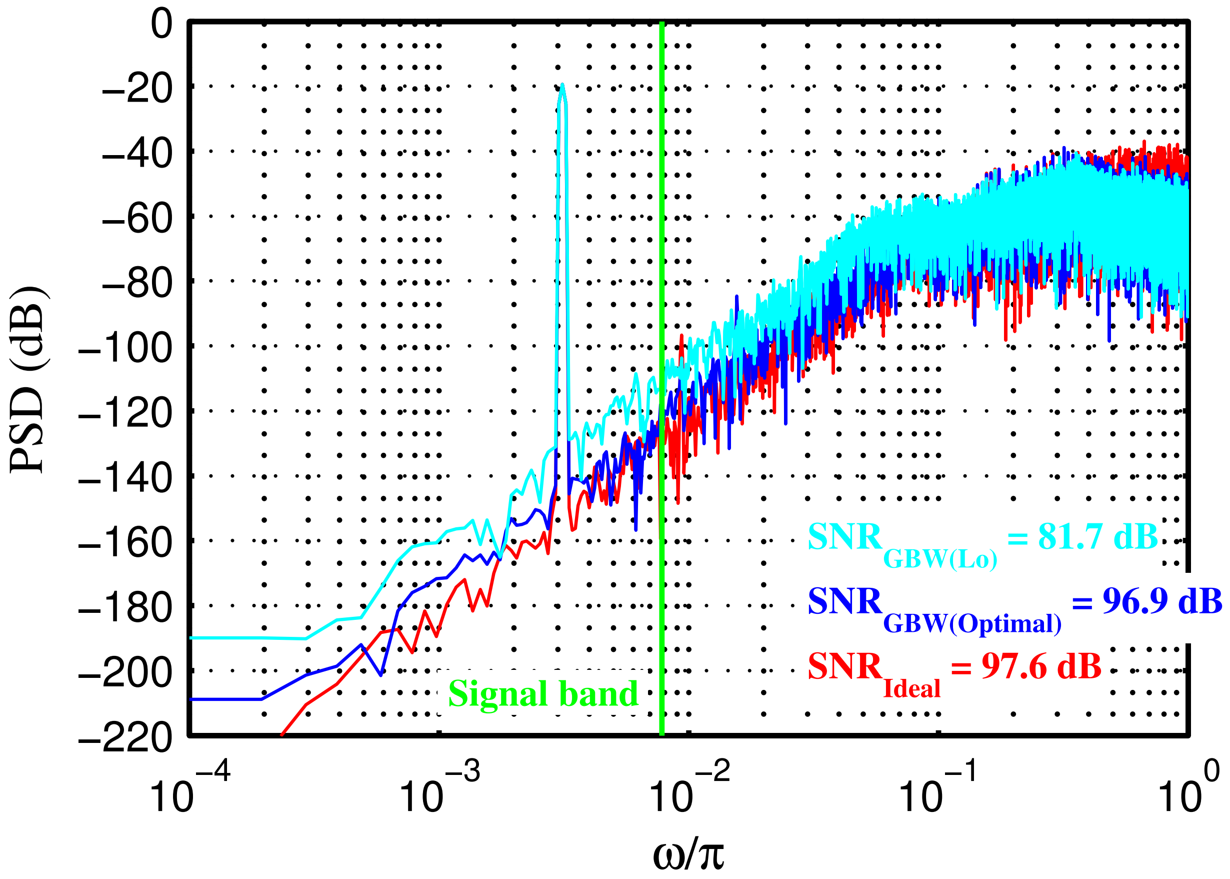


FIGURE 3.10. PSDs of the CT DSM with different GBW.

Clock Jitter. The clock signals supplied to DSMs in highly integrated systems are likely generated by on-chip VCOs. The effect of clock jitter of such sources has been studied in [13]. Here the relevant results are briefly summarized. The DSM sampling instants (e.g., each rising clock edge) can be written as follows:

$$t_n = nT_s + \beta_n, \quad n = 0, 1, 2, ..., N-1, \tag{8}$$

where T_s is the sampling period, and β_n is the clock uncertainty at each instant, or the clock jitter. For a real VCO, $\beta_n = \sum_{i=0}^{n} \beta_i$. That is, jitter is a cumulative variable. Thus (8) can be rewritten as follows:

$$t_n = nT_s + \sum_{i=0}^{n} \beta_i, \quad n = 0, 1, 2, ..., N-1. \tag{9}$$

For a VCO with center frequency f_c, the jitter effect is typically measured in terms of phase noise n_c. At a 100-kHz offset ($f_n = 100$ kHz), a typical n_c value, in dBc/Hz, is calculated as follows [56, 38]:

$$n_c = -100 + 20 \log_{10} f_c. \tag{10}$$

Through experiments and with the assumption of normal distribution [13], the relation between n_c and the variance of β_n is calculated as follows:

$$\alpha_\beta^2 \approx \left(\frac{T_s^2 \cdot f_n^2 \cdot 10^{n_c/10}}{2 f_c} \right). \tag{11}$$

Thus knowing n_c and the variance of β_n, the RMS jitter can then be calculated accordingly. The clock jitter can be modeled in Verilog-AMS using the function `$rdist_normal`. The simulated CT DSM output PSDs for different RMS jitter values (1 ps, 10 ps, and 100 ps) are shown in Fig. 3.11. It can be seen that clock jitter has a great impact on the noise-shaping performance. The noise floor in the signal band rises due to the clock jitter. The rise in the noise floor is only noticeable in the signal band and is almost flat because of the employment of a white clock jitter model and the fact that the error introduced by clock jitter only occurs at instants that the feedback DAC is making transitions. The transitions are highly correlated with the input signal pattern. This mechanism shifts the jitter noise to the signal band. Based on the simulation results, RMS jitter = 10 ps is a good tradeoff between CT DSM performance and clock generator cost.

Final Design. With the building block specifications shown in Table 3.1 and 10 ps RMS jitter, the resultant CT DSM SNR is 87.3 dB for a signal bandwidth of 20 kHz. The time-domain simulation result of the CT DSM is shown in Fig. 3.12 where v(t) is the input signal and u(t) is the modulator output.

3.5. SIMULINK versus Verilog-AMS

In this Section Simulink and Verilog-AMS are compared as media for mixed-signal system simulation in terms of various key aspects including accuracy, speed, modeling effort.

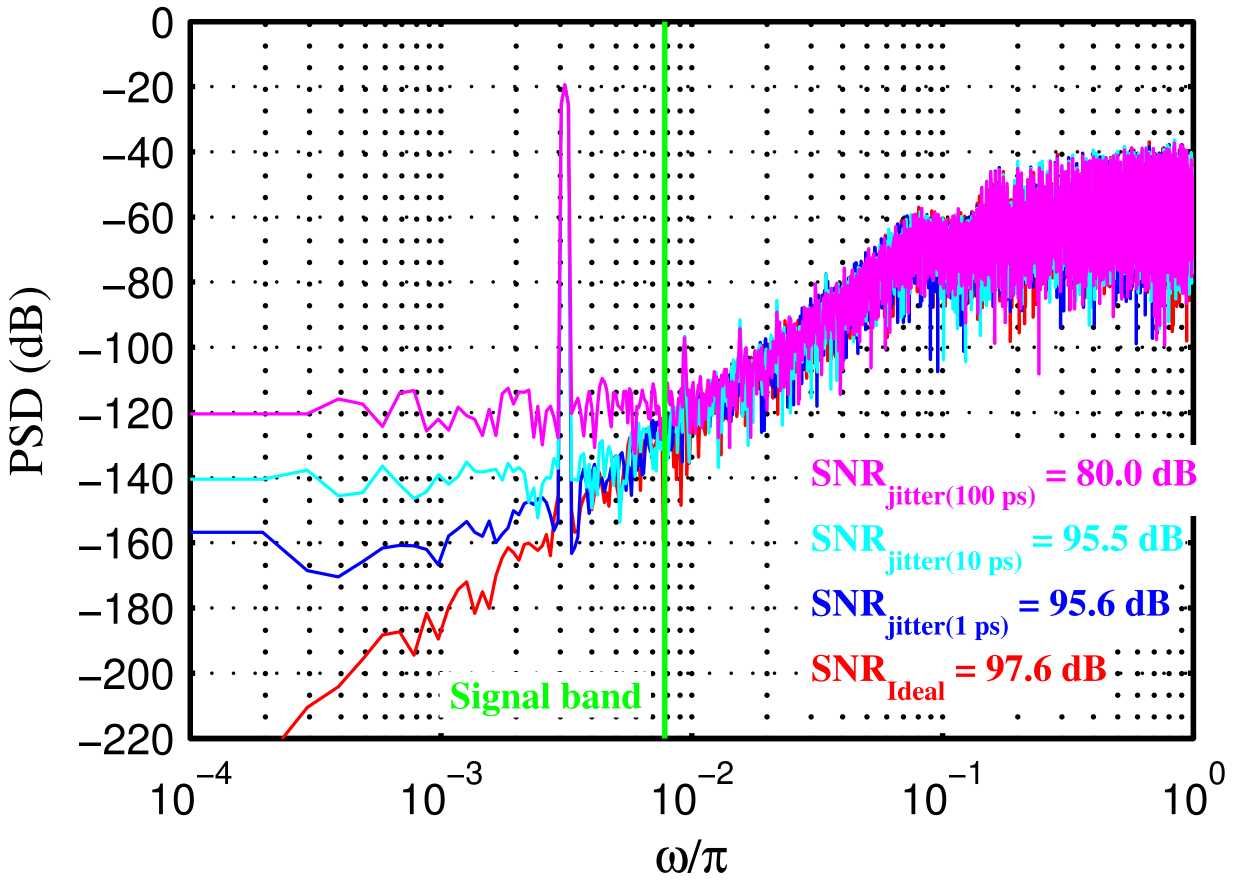


FIGURE 3.11. PSD of the CT DSM output with different RMS jitter value.

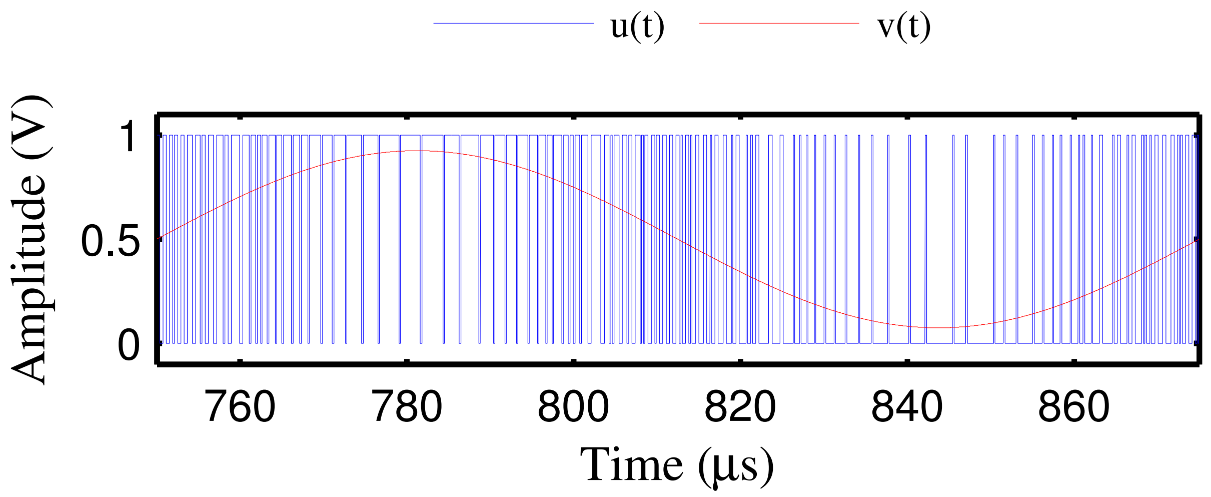


FIGURE 3.12. The time-domain simulation result.

TABLE 3.1. The required component values and building block specifications for the CT DSM.

Resistors (Ω)	$R_1 = 100$ k, $R_2 = 100$ k, $R_3 = 100$ k, $R_{1a} = 1.188$ k, $R_{2a} = 1.857$ k, $R_{3a} = 947$ $R_{0a} = 10.546$ k, $R_a = 1$ k
Capacitors (F)	$C_1 = 3.384$ p, $C_2 = 5.225$ p, $C_3 = 32.16$ p
Op amp 1	$A_{oldc} = 128$, $f_{3dB} = 12$ kHz
Op amp 2	$A_{oldc} = 128$, $f_{3dB} = 12$ kHz
Op amp 3	$A_{oldc} = 128$, $f_{3dB} = 12$ kHz
Op amp 4	$A_{oldc} = 128$, $f_{3dB} = 80$ kHz

3.5.1. *Accuracy and Speed*

As in most cases, higher simulation accuracy usually results in more computation time (i.e. lower speed). Knowing the theoretical limit of the CT DSM helps to determine a good tradeoff between accuracy and speed. Here the simulated PSD of the DT DSM output is taken as the theoretical limit, and look for the simulator settings that lead to fast simulation without sacrificing much accuracy through experiments. Tables 3.2 and 3.3 show some important simulator settings and the resultant computation time for the SIMUNLINK and Verilog-AMS simulations in this work. The computation time was for simulation time equal to 32 periods of the sine wave input to the modulator. The low-pass CT DSM Design has 87.3 dB SNR and 20 kHz input bandwidth.

TABLE 3.2. The simulator settings for SIMULINK and AMS Designer.

Simulator	Settings
SIMULINK	Analog Solver: ode23s Relative tolerance: 1×10^{-6} Absolute tolerance: 1×10^{-5} Max step size: 1×10^{-2}
AMS Designer	Analog Solver: Spectre Relative tolerance: 1×10^{-3} Voltage Absolute tolerance: 1×10^{-6} Current Absolute tolerance: 1×10^{-12}

TABLE 3.3. The computation time for SIMULINK and AMS Designer.

Simulator	Computation Time
SIMULINK	125.8 s
AMS Designer	57.0 s

In Spectre, the relative tolerance specifies the allowed maximum error relative to the signal magnitude [52]. It takes the value between 0 and 1 and is a setting that globally controls the simulation accuracy. Voltage and current absolute tolerances define the smallest interesting voltage and current, respectively, in the simulation. Voltage and current signals smaller than these values are ignored. The Simulink solver shares the same definition of the relative tolerance with Spectre, but the definition for the absolute tolerance is different [62]. In Simulink absolute tolerance specifies the maximum acceptable error when the signal approaches zero.

The SIMULINK simulation requires more than twice of the computation time as that of the Verilog-AMS simulation. This might be mainly due to the fact that the relative tolerance is set to be twice smaller than that of the AMS Designer. However, this value could not be further increased for the SIMULINK simulation and maintain comparable accuracy. There is a set of solvers available in Simulink. It is possible that a different Simulink solver can provide better trade off between error-control and speed, and thus results in the similar performance as Verilog-AMS. However, evaluation of every solver is not performed here since the goal is to find the settings that are qualified for obtaining the CT DSM performance limit. The adopted ode23s solver is a stiff solver. Compared to the default ode45 solver, a non-stiff solver, a stiff solver can better handle rapid and sharp signal transitions. A CT DSM is a stiff system in which this kind of signal behaviors are common [46].

3.5.2. *Modeling Effort and Integrability*

As discussed in Section 3.4.2 SIMULINK libraries have comprehensive fundamental building blocks. They are generally not complex models that can cover many aspects of real designs, but are usually adequate for checking whether a design can approach the theoretical limit at early design stages. Building a basic DSM SIMULINK model is as simple as picking the right blocks from the library, connecting them, and configuring simulation setting. When the design needs to be modified, the SIMULINK model takes less effort than the Verilog-AMS model does. However, modeling and simulating non-idealities such as clock jitter may not be an easy task. Also, Verilog-AMS models are easier to integrate with the actual circuit implementation to perform co-simulations.

A CT DSM design flow along with the tool and the modeling language used in each step has been presented. Based on this design practice, MATLAB/SIMULINK is suitable for high-level system design and AMS Designer/Verilog-AMS is suitable for simulations with non-idealities and for integration with low-level building block implementations in terms of modeling effort. There is no significant difference in simulation performance between these tools. The choice of the tool and the modeling language depends on the modeling object and time budget.

CHAPTER 4

VERILOG-AMS-POM: VERILOG-AMS INTEGRATED POLYNOMIAL METAMODELING USING AN OP-AMP AS A CASE STUDY

In order to achieve high performance and high yield, analog/mixed-signal (AMS) circuits and systems must be optimized at both system and circuit levels. For a top-down design approach, this starts with designing and optimizing the system with sub-block models at high levels of abstraction. The specifications for each sub-block that lead to the best system performance are then obtained. Each sub-block is then designed and optimized toward these specifications. The issue with this approach is that generating an accurate model, if it is even feasible, takes significant effort. Therefore some characteristics of the sub-blocks are ignored at the high-level simulation. This makes the obtained sub-block specifications less reliable. To address the aforementioned issues, **a three-step optimization flow is proposed in this Chapter using an OP-AMP as a case study** [107].

4.1. The Idea of Polynomial Metamodel Integrated Verilog-AMS

The proposed flow is assisted by POlynomial Metamodels (POMs) to significantly reduce the design cycle by performing fast optimization, verification, and design exploration. In the first step, based on the specifications from high-level simulations, an optimized OP-AMP design is obtained by ultra-fast POM-assisted optimization. This optimized design serves as the starting point for constructing an OP-AMP meta-macromodel in the second step. The meta-macromodel is then integrated into a Verilog-AMS module which is used at system-level simulations and can greatly reduce the computation time. The third step performs optimization at the system level. The computation time is greatly reduced due to the use of the **Verilog-AMS POlynomial M**eta-macromodel. This is is called **Verilog-AMS-POM** in this dissertation. The meta-macromodel is the polynomial metamodel generated from the macromodel. It may be noted that macromodel here refers to the transfer function or SPICE macromodel. Since the optimization is performed at system-level with the circuit-level metamodel based on the OP-AMP design, the resulting final OP-AMP design has a much higher chance of meting the system requirements and attaining much higher performance. For the optimization a customized Cuckoo Search algorithm is used for the first time for OP-AMP optimization through Verilog-AMS-POM.

Macromodeling has been used as a popular technique to generate simpler circuit representations for reducing simulation time. A macromodel usually consists of the dominant elements of the circuit and still needs the same set of electronic design automation (EDA) tools as the original model and hence is slower than metamodels. In [95], the OP-AMP to be modeled was first divided into basic building blocks and then these blocks, based on their functionality, were replaced with appropriate simplified models composed of ideal circuit elements. This method retains the circuit structure and provides some insight into the circuit. One has to understand how exactly every basic blocks affect the overall circuit characteristics and their influence on each other. This indicates large modeling effort. As the circuit size increases, a quasi-proportional increase in the simulation cost is also expected. A generic CMOS OP-AMP macromodel was presented in [35]. This macromodel is redrawn in Figure 4.1. It consists of five stages—the input stage, the common mode stage, the gain stage, the intermediate stage, and the output stage. Each stage is composed of fundamental electrical circuit elements (resistors, capacitors, and inductors) and dependent/independent sources. Signals flow through these stages sequentially.

The macromodel in Figure 4.1 assumes infinite DC input impedance, which is valid for CMOS OP-AMPs. For AC input impedance, common-mode (CM) and differential-mode (DM) impedances are considered. The CM input impedance is purely capacitive therefore it is formed by two capacitors (C_{cm}). The DM impedance is an RC circuit (R_d and C_d). These elements together create a stage with two poles at the following:

$$\boldsymbol{w_p} = \left\{0,\ \frac{C_d + C_{cm}}{R_d C_d C_{cm}}\right\}. \tag{12}$$

The OP-AMP has a zero at the following:

$$\boldsymbol{w_z} = \left\{\frac{1}{R_d C_d}\right\}. \tag{13}$$

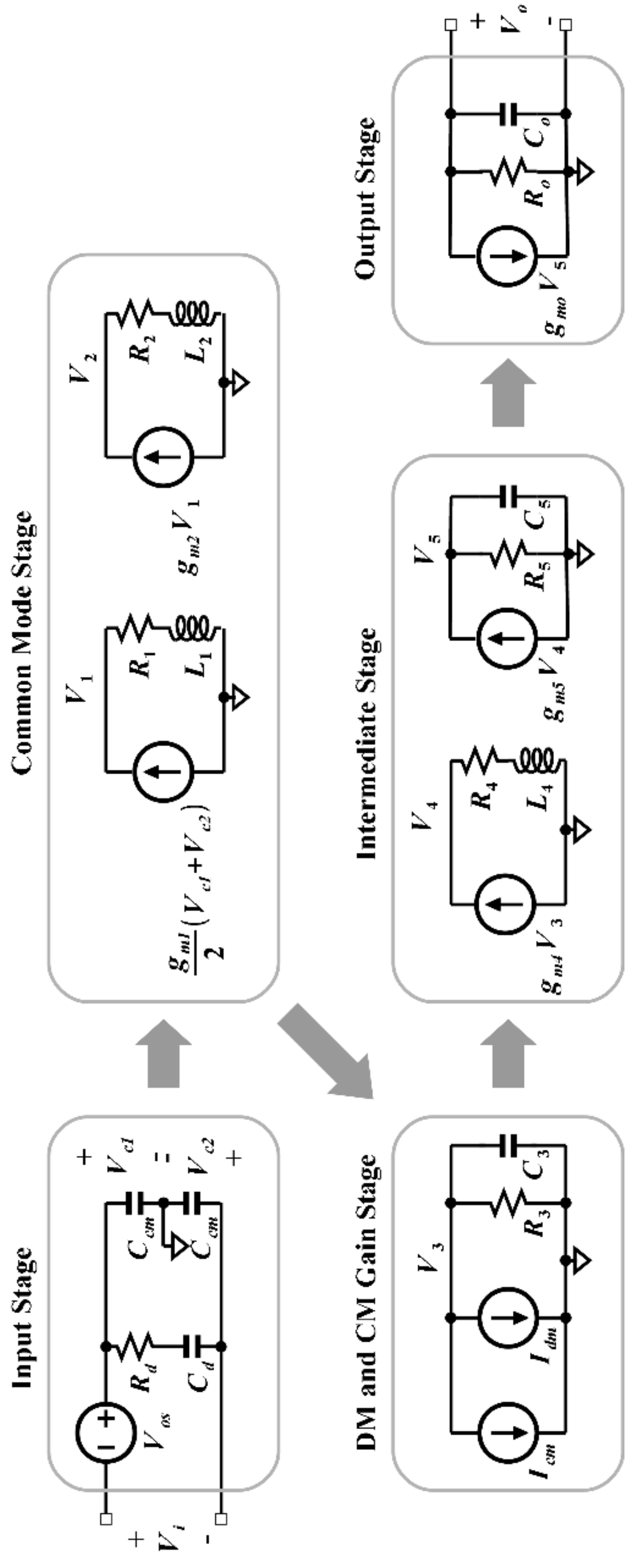


FIGURE 4.1. A generic op-amp macromodel [35].

A DC voltage source is employed in the input stage to model the input offset voltage (V_{os}). The common-mode stage models the AC CM response of the amplifier. Two zeros at the following points are created in this stage:

(14) $$\boldsymbol{w_z} = \left\{ \frac{R_1}{L_1}, \frac{R_2}{L_2} \right\}.$$

The essential input to this stage is the CM voltages, V_{c1} and V_{c2}, sensed by the CM capacitors (C_{cm}). The gain stage models the CM and DM gains of amplifier. There gains are carried by current sources I_{cm} and I_{dm}, respectively. These two gains can be represented by simple MOSFET transconductance or constructed using more complex functions that include multiple circuit parameters to model nonlinearities. The gain stage also models the op-amp pole with the second lowest frequency (the second pole):

(15) $$w_p = \frac{1}{R_3 C_3}.$$

The purpose of the intermediate stage is to provide additional poles and zeros to improve the accuracy of the modeled op-amp AC response. This is due to the fact that a real op-amp design generally has more than two poles/zeros. This stage does not change the op-amp gain since it is built to produce unity gain. The intermediate stage shown in Fig. 4.1 produces a pole at the following:

(16) $$w_p = \frac{1}{R_5 C_5}.$$

It produces a zero at the following:

(17) $$w_z = \frac{R_4}{L_4}.$$

The output stage produces the dominant pole at the following:

(18) $$w_p = \frac{1}{R_o C_o},$$

where R_o and C_o can both be nonlinear functions to account for nonlinear effects. The values for all of the variables in the macromodel can be extracted from circuit simulations.

The symbolic analysis method provides an alternative macromodeling approach. A symbolic expression relating the input and output response can be generated through algebraic or graph-based methods [81]. The traditionally symbolic analysis methods are suitable for modeling the small-signal behaviors. However, symbolic analysis methods are not for the large-transient responses such as the OP-AMP slewing and settling which is considered in this Chapter. A method was proposed in [104] to overcome this shortcoming. The authors applied symbolic pole analysis to the traditional OP-AMP two-pole model [15] in which the poles are obtained numerically. Accurate op-amp slewing and settling behavioral approximation was achieved by using two sets of poles for slewing and settling, respectively.

In [55] and [93], Verilog-A OP-AMP behavioral models were used to estimate the specifications used in a switched-capacitor filter. The OP-AMP characteristics such as gain-bandwidth (GBW), slew-rate, and phase margin were included. However, the comparison of the behavioral and SPICE simulation results are not shown to prove accuracy of such a modeling approach. In [19], the continuous-time transfer function of an OP-AMP under voltage follower configuration was discretized and modeled using VHDL. However, only a step input signal was tested. A VHDL-AMS OP-AMP behavioral model was proposed in [82]. It was validated with various load and accounts for output nonlinear behavior. However, it requires device information such as MOSFET operation region which is usually difficult to obtain for nano-CMOS circuits. The VHDL-AMS OP-AMP model in [5] was based on a three-stage model. Temperature-effect parameters were incorporated into the model so it can be used for simulating the circuits for high temperature applications. In [7], a framework for extracting circuit parameters was proposed. A similar setup is used in this dissertation for circuit parameter extraction.

4.2. An OP-AMP for Biomedical Applications

OP-AMPs are typically used in the analog front-ends of many biomedical systems. As a specific example, a typical setup for electroencephalography (EEG) recording shown in [12] is redrawn in Fig. 4.2. The measurement system consists of the following components:

- an instrumental amplifier (IA),
- a programmable gain amplifier (PGA),
- an analog-to-digital converter (ADC), and
- a digital signal processor (DSP).

As shown in the figure, the biopotential signals are first acquired by an instrumental amplifier (IA) and then boosted by a programmable gain amplifier (PGA). The ADC converts the signals into digital form which are then processed by a digital signal processor (DSP). A classical realization of the instrumental amplifier implementation using three OP-AMPs as shown in Figure 4.3 [22].

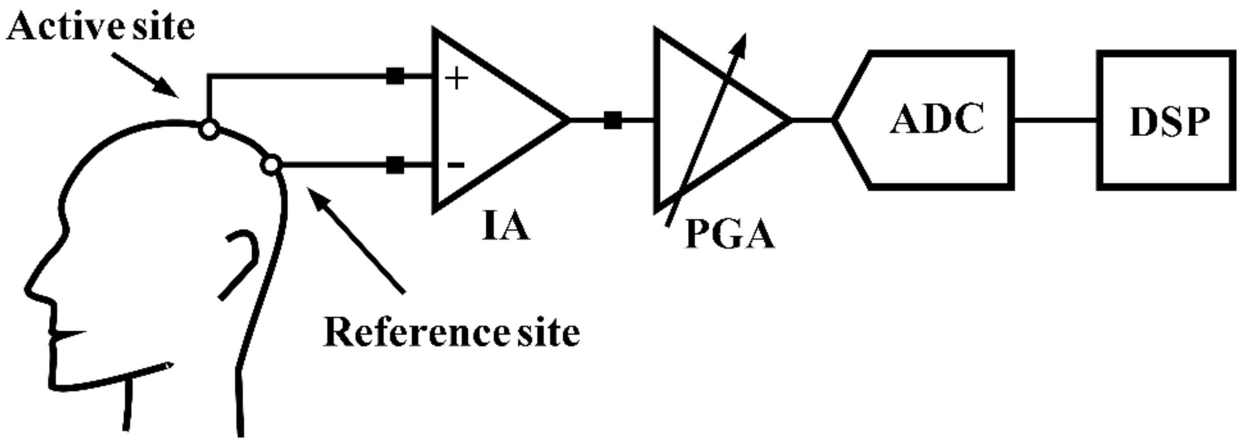


FIGURE 4.2. A typical configuration of EEG recording [12].

The transistor-level schematic of the case-study OP-AMP circuit in this Chapter is shown in Fig. 4.4. The proposed OP-AMP has the following distinct stages with different operations in the circuit:

- a folded-cascode operational transconductance amplifier (OTA),
- a common-source amplifier, and

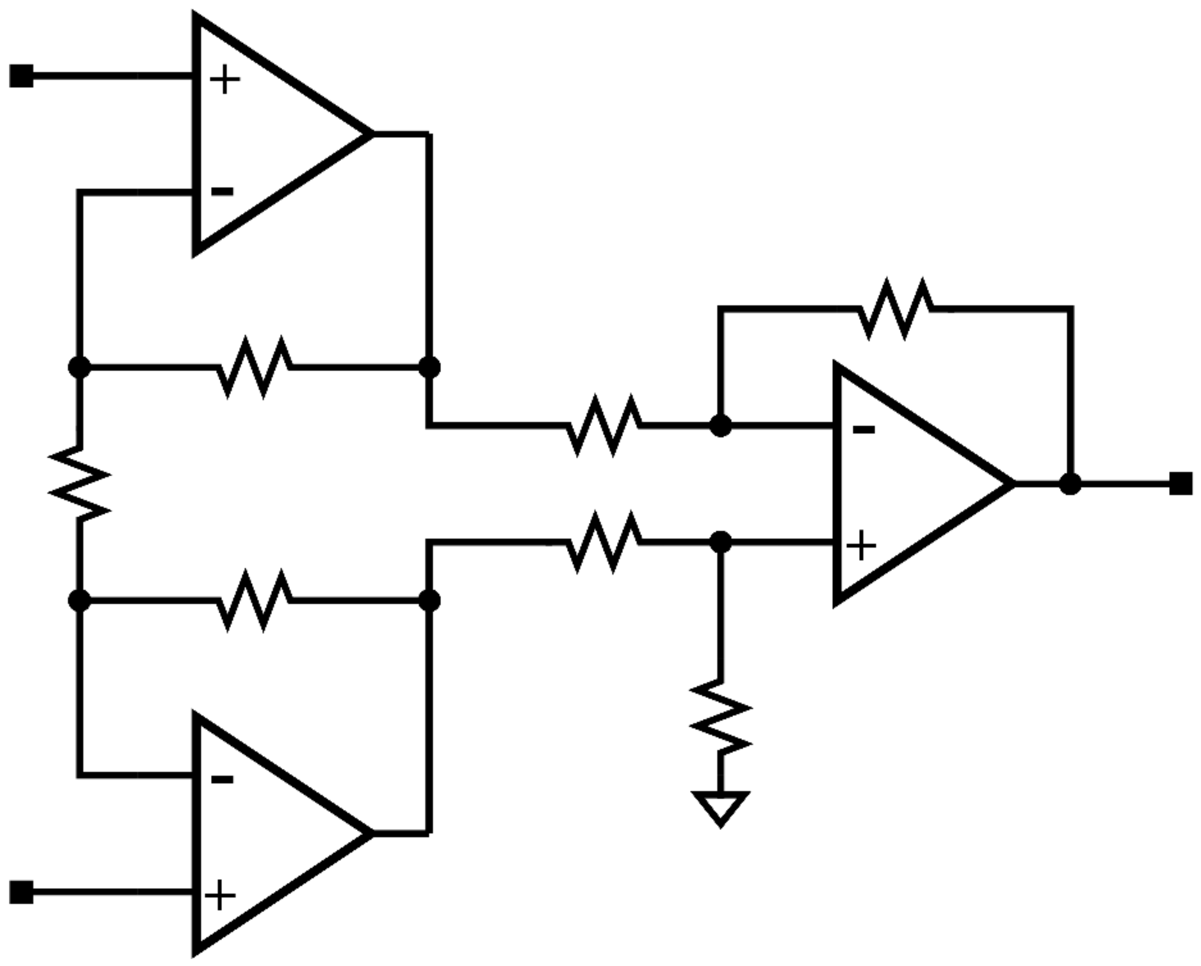


FIGURE 4.3. A classical realization of instrumental amplifier using three OP-AMPs [12, 22].

- a common-mode feedBack (CMFB) circuit.

This two-stage OP-AMP consists of a folded-cascode operational transconductance amplifier (OTA) for high gain and a common-source amplifier as output stage for low output resistance. A Common-Mode FeedBack (CMFB) circuit is used to regulate the output common-mode voltages. The two-stage OP-AMP is compensated using Miller capacitors C_1 and C_2. M_{21} and M_{22} act as resistors to remove the right-half plane zero. A fully differential implementation is used in order to suppress common-mode noise and thus the even-order harmonic distortion. A 90 nm CMOS process with 1 V power supply is used for this OP-AMP design. However, 45nm and 32nm CMOS libraries are becoming available and the same design can also be done using the new libraries. The input devices of the OTA are a pair of PMOS transistors. NMOS input devices have higher transconductance g_m than the PMOS ones for the same bias current. However, they have higher flicker noise and thus a NMOS input device is suitable for applications requiring large bandwidth but not for low-noise and low-frequency applications such as biopotential acquisition.

The test bench for the OP-AMP DC and AC analyses is shown in Figure 4.5. In the figure, DC voltage source V_0 provides the 1-V power supply, V_1 supplies a 0.5-V common-mode voltage vcm. V_2 is a `vsource` in Cadence `analogLib` library. It can be set to generate various types of voltage signals, including DC, sine, and pulse. In AC analysis V_2 produces a sinusoid small-signal voltage. The signal produced by V_2 is fed to the control terminals of two voltage-controlled voltage sources (VCVSs) E_0 and E_1 to produce differential input signals that vary around vcm. R_0 and R_1 are 50-kΩ load resistors. C_0 and C_1 are 1-pF load capacitors. The BIAS block generates global bias voltages for the op amp. vdd and vcm are also global signals. In order to test the slew rate (SR), a similar test bench with modifications to setup the op amp in unity-gain configuration is used. Transient analysis is performed using this test bench (shown in Figure 4.6). Resistors R_2–R_5 are introduced to realize an unity closed-loop gain. They are all set to be 100 kΩ.

For the target application of the OP-AMP for this Chapter, the required DC gain and bandwidth are at least 43 dB and 40 kHz, respectively. The transistor sizing is based on the (g_m/I_D) method [86] in order to maximize the current efficiency. Since the DC gain is a strong function of the MOSFET length (L), the L of the input device is determined through parametric simulation with a fixed unit MOSFET width (W). With these W and L values, a plot of (g_m/I_D) against overdrive voltage can be obtained. A high (g_m/I_D) ratio is then selected with the consideration of voltage swing. W is then scaled to provide the required current. It should be noted that if the (g_m/I_D) is too high, the W can be quite large. Hence a tradeoff between area and power efficiency should be made according to the application. The simulated OP-AMP performance together with the specifications are shown in Table 4.1. The transistor sizes for the baseline design are listed in Table 4.2. The simulated DC gain

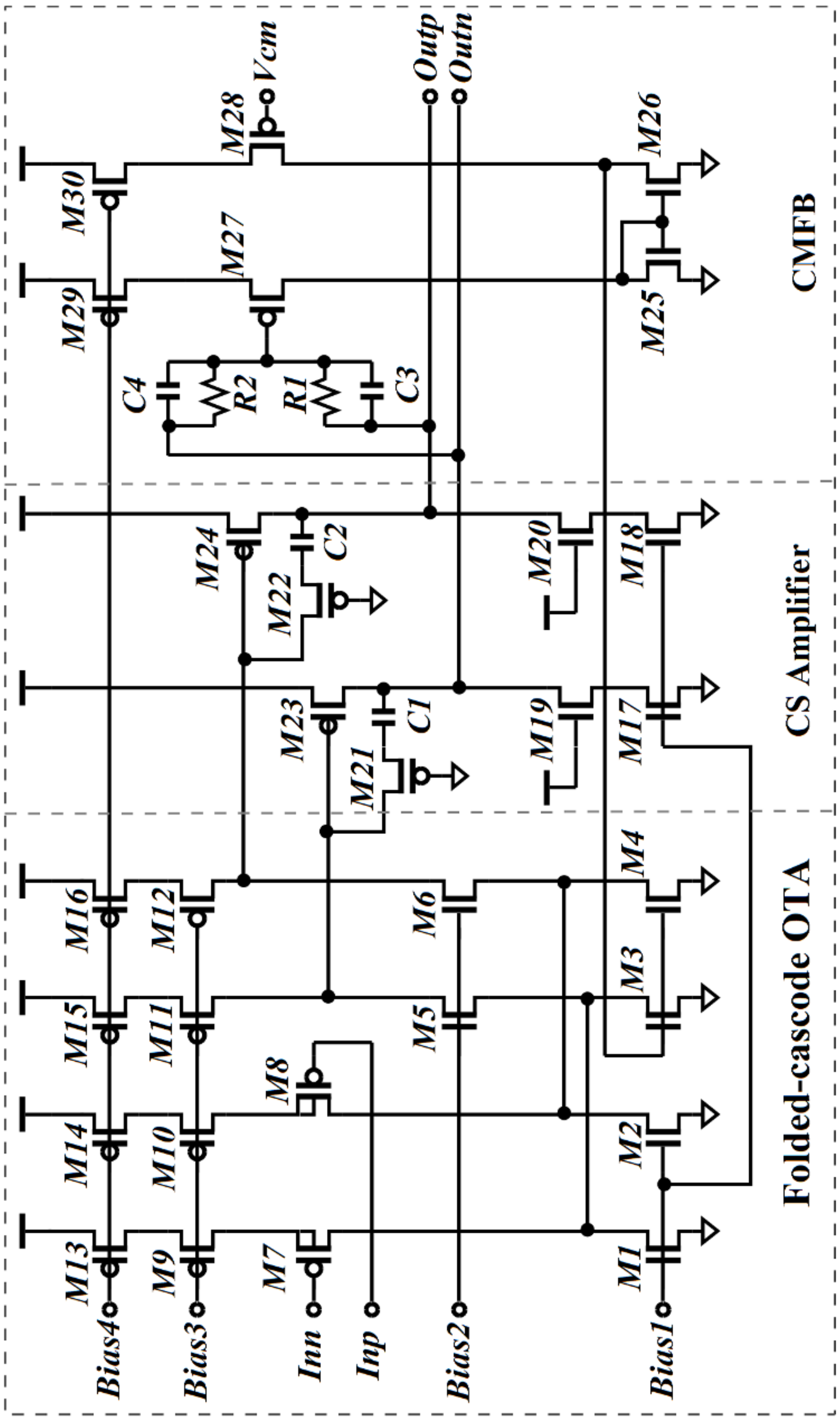


FIGURE 4.4. The schematic of the OP-AMP.

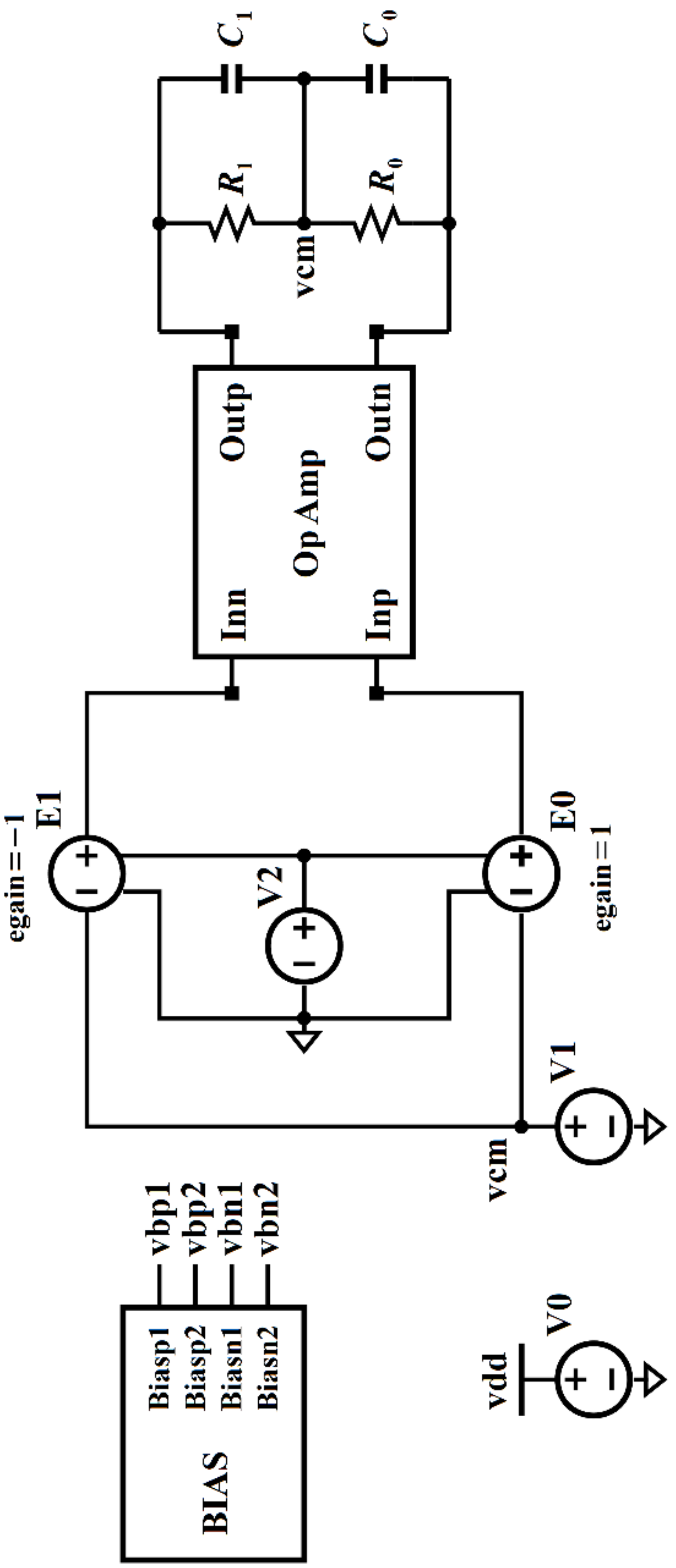


FIGURE 4.5. The test bench for the op-amp DC and AC analyses.

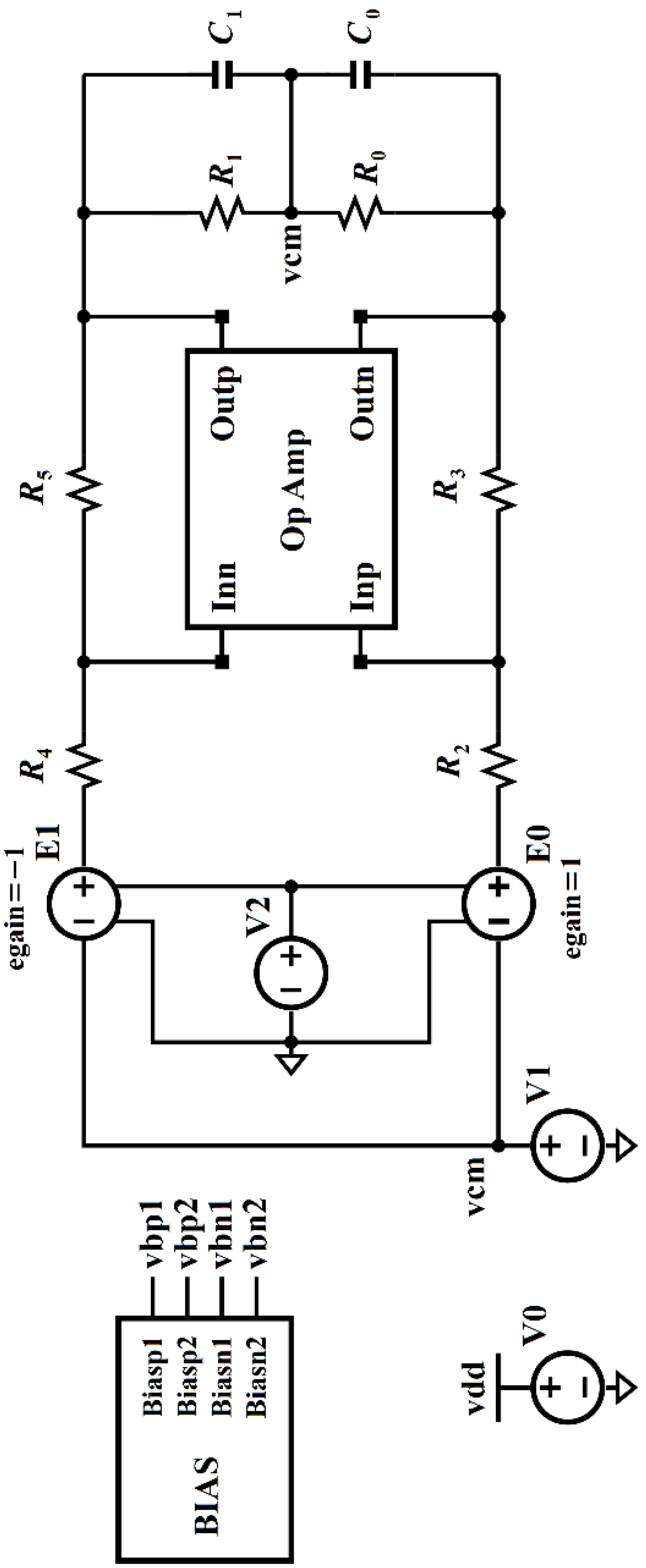


FIGURE 4.6. The test bench for the op-amp step response simulation.

is 52.3 dB, the bandwidth is 58 kHz, the phase margin is 92.5^o, and the gain margin is -25 dB. The baseline design satisfies the gain-bandwidth and stability requirements. in addition, this design requires the slew rate to be greater than 4 mV/ns.

Table 4.1. Characteristics of the OP-AMP.

Performance	**Specifications**	**Baseline**
$\boldsymbol{A_0}$ (dB)	43	52.3
BW (kHz)	40	58
PM (degree)	65	92.5
SR (mV/ns)	4	5.1
$\boldsymbol{P_D}$ (μW)	minimized	252.8

4.3. Proposed Design Optimization Flow Using Verilog-AMS-POM

The proposed design optimization flow is shown in Fig. 4.7. The compete optimization flow can be divided into three major steps as follows:

- Ultra-fast OP-AMO optimization.
- Verilog-AMS meta-macromodel generation.
- Fast AMS system optimization.

This chapter studies and implements the first two steps of the proposed overall design flow. Step 1 starts with the baseline OP-AMP design and generates an optimized OP-AMP design which serves as the starting point of Step 2 where an OP-AMP Verilog-AMS meta-macromodel is constructed. Assuming the OP-AMP is a sub-block of an AMS system, the original OP-AMP transistor-level netlist is replaced with the Verilog-AMS meta-macromodel in Step 3 to enable fast AMS simulations. The sub-blocks including the OP-AMP design can then be further optimized toward better system performance.

Metamodels are used in both Step 1 and Step 2. In Step 1, a set of metamodels are generated to predict the OP-AMP characteristics such as gain, bandwidth, phase margin, and slew rate. This allows the optimizer to process this information without conducting actual circuit simulations. The elimination of the need of performing transistor-level simulations saves a huge amount of simulation time. This is studied in Section 4.5. In Step 2, a set of OP-AMP parameter metamodels are generated which are necessary for constructing the OP-AMP meta-macromodel. The procedures of generating the OP-AMP characteristic metamodels and the OP-AMP parameter metamodels are the same and are presented in detail in Section 4.4.

4.4. Polynomial Metamodel Generation for OP-AMP

The OP-AMP characteristics and the parameters that are used in Section 4.6 are modeled using polynomial functions which are polynomial metamodels. The polynomial metamodels used have the following format:

$$f(\mathbf{x}) = \sum_{i=0}^{N_B-1} \beta_i \prod_{j=0}^{N_D-1} x_j^{p_{ij}}, \tag{19}$$

where $f(\mathbf{x})$ is the OP-AMP parameter to be modeled. N_B is the number of basis functions of this polynomial metamodel. β_i is the coefficient for the i-th basis function. N_D is the number of design variables, x_j is the j-th design variables and p_{ij} is the power term for the j-th design variable in the i-th basis function. For example, assuming that the design variables are the width and length of an NMOS and a PMOS ($\mathbf{x} := \{L_N, L_P, W_N, W_P\}$), then $N_D = 4$. As an example, the polynomial metamodel for the DC gain is the following:

$$\begin{aligned} A_0(\mathbf{x}) = &+ 4.5 \times 10^2 \cdot L_N^0 \cdot L_P^0 \cdot W_N^0 \cdot W_P^0 \\ &+ 0.8 \times 10^9 \cdot L_N^0 \cdot L_P^1 \cdot W_N^0 \cdot W_P^0 \\ &- 1.2 \times 10^{15} \cdot L_N^1 \cdot L_P^0 \cdot W_N^1 \cdot W_P^0 \\ &+ 0.3 \times 10^{16} \cdot L_N^0 \cdot L_P^0 \cdot W_N^0 \cdot W_P^2, \end{aligned} \tag{20}$$

where $A_0(\mathbf{x})$ consists of 4 basis function and is a 2nd order polynomial. The basis function coefficients are: $\beta_0 = 4.5 \times 10^2$, $\beta_1 = 0.8 \times 10^9$, $\beta_2 = -1.2 \times 10^{15}$, and $\beta_3 = 0.3 \times 10^{16}$.

The design variables in this Chapter are the following: transistor widths, lengths, and the bias current generated in the bias circuitry. There are **sixteen design variables in total** ($N_D = 16$). The design variable values and the associated devices for the baseline

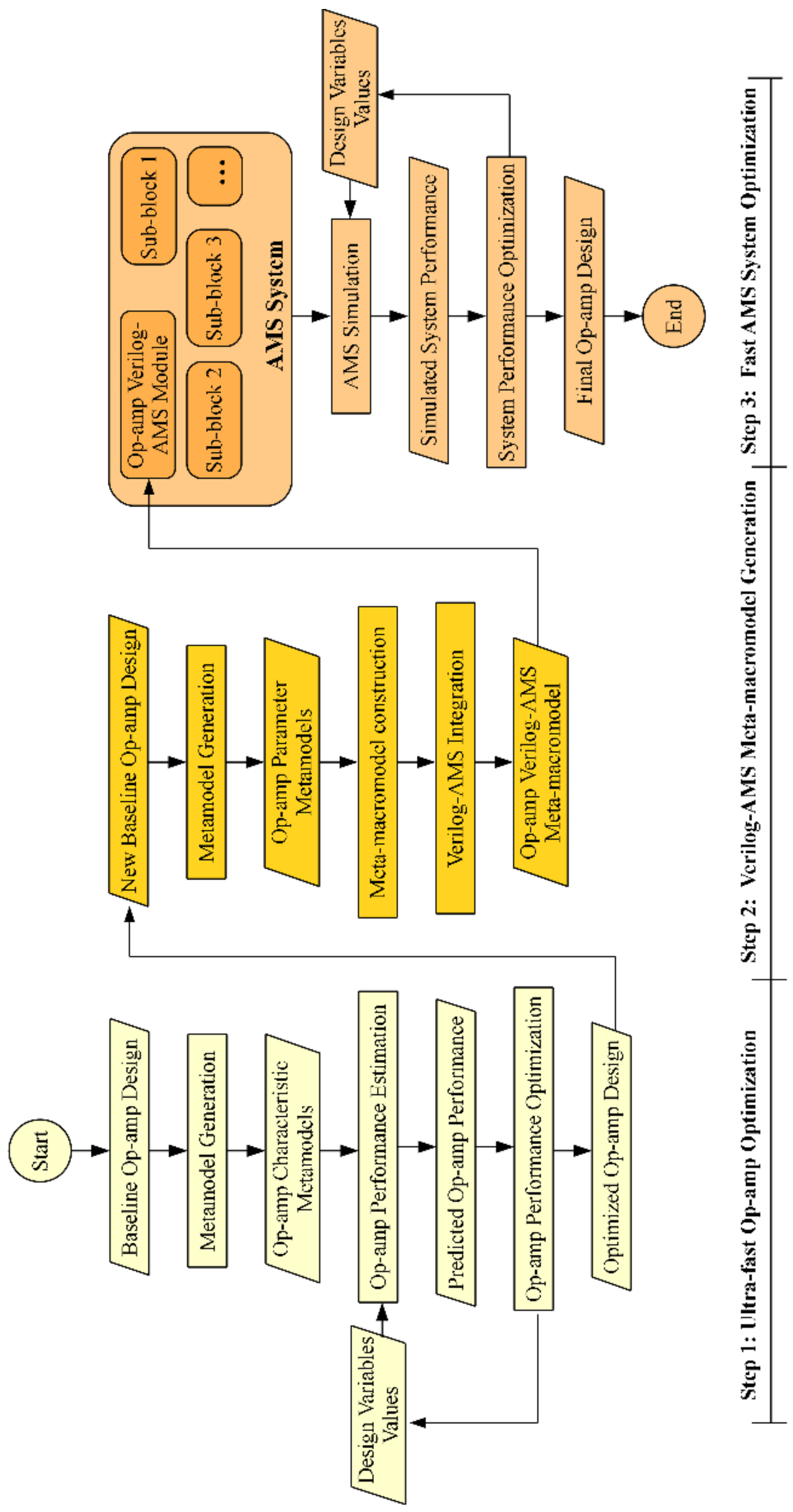

Start
Baseline Op-amp Design
Metamodel Generation
Op-amp Characteristic Metamodels
Op-amp Performance Estimation
Design Variables Values
Predicted Op-amp Performance
Op-amp Performance Optimization
Optimized Op-amp Design
New Baseline Op-amp Design
Metamodel Generation
Op-amp Parameter Metamodels
Meta-macromodel construction
Verilog-AMS Integration
Op-amp Verilog-AMS Meta-macromodel
Op-amp Verilog-AMS Module
Sub-block 1
Sub-block 2
Sub-block 3
...
AMS System
Design Variables Values
AMS Simulation
Simulated System Performance
System Performance Optimization
Final Op-amp Design
End
Step 1: Ultra-fast Op-amp Optimization
Step 2: Verilog-AMS Meta-macromodel Generation
Step 3: Fast AMS System Optimization

FIGURE 4.7. The proposed ultra-fast circuit-aware OP-AMP design optimization flow.

design and their ranges are shown in Table 4.2. This table also presents two sets of the optimized values Optimal$_{\mathrm{SCH}}$ and Optimal$_{\mathrm{POM}}$ which are discussed in Section 4.5.

TABLE 4.2. Summary of the OP-AMP Design Variables.

Design Variables	**Baseline (μm)**	**Devices**
$\boldsymbol{L_{inOTA}}$	0.180	$M_{7,8}$
$\boldsymbol{L_{npOTA}}$	0.270	$M_{1-6,9-16,25,26,29,30}$
$\boldsymbol{L_{inCS}}$	0.090	$M_{23,24}$
$\boldsymbol{L_{nCS}}$	0.090	M_{17-20}
$\boldsymbol{L_{inCMFB}}$	0.180	$M_{27,28}$
$\boldsymbol{L_{C}}$	0.180	$M_{21,22}$
$\boldsymbol{W_{inOTA}}$	49.950	$M_{7,8}$
$\boldsymbol{W_{nOTA}}$	9.000	M_{1-6}
$\boldsymbol{W_{pOTA}}$	18.000	M_{9-16}
$\boldsymbol{W_{inCS}}$	72.000	$M_{23,24}$
$\boldsymbol{W_{nCS}}$	36.000	M_{17-20}
$\boldsymbol{W_{inCMFB}}$	5.400	$M_{27,28}$
$\boldsymbol{W_{nCMFB}}$	1.800	$M_{25,26}$
$\boldsymbol{W_{pCMFB}}$	3.600	$M_{29,30}$
$\boldsymbol{W_{C}}$	0.90	$M_{21,22}$
$\boldsymbol{I_{BIAS}}$	0.500 μA	-

The metamodel generation flow with the OP-AMP parameter metamodels $A_0(\mathbf{x})$, $I_{0+}(\mathbf{x})$, $I_{0-}(\mathbf{x})$, $g_{m0}(\mathbf{x})$, $\mathbf{b}(\mathbf{x})$, and $\mathbf{a}(\mathbf{x})$ is shown in Fig. 4.8. The use of these parameter metamodels is discussed in Section 4.6. The OP-AMP characteristics metamodels are $BW(\mathbf{x})$, $PM(\mathbf{x})$, $SR(\mathbf{x})$, and $P_D(\mathbf{x})$ for the bandwidth, phase margin, slew rate, and power dissipation, respectively. They are generated using the same procedure as that of OP-AMP parameter metamodel generation. The goal is to find β_i and p_{ij} in Eqn. 19 for each OP-AMP parameter in the POM. First, N samples of design variables are generated using the Latin Hypercube Sampling (LHS) technique. For each sample, transistor-level simulations are performed and the resultant OP-AMP parameter samples are extracted. Then, given N design variable samples and N OP-AMP parameter samples, polynomial regression is done to find β_i and p_{ij}. If the resulting POM does not satisfy the accuracy requirement, the

TABLE 4.3. Summary of the OP-AMP Design Variables.

Design Variables	**Minimum** (μm)	**Maximum** (μm)	**Optimal$_{\mathbf{POM}}$** (μm)	**Optimal$_{\mathbf{SCH}}$** (μm)
$\boldsymbol{L_{inOTA}}$	0.090	0.270	0.090	0.090
$\boldsymbol{L_{npOTA}}$	0.225	0.315	0.275	0.255
$\boldsymbol{L_{inCS}}$	0.090	0.270	0.213	0.135
$\boldsymbol{L_{nCS}}$	0.090	0.180	0.151	0.147
$\boldsymbol{L_{inCMFB}}$	0.090	0.270	0.249	0.131
$\boldsymbol{L_{C}}$	0.090	0.270	0.145	0.123
$\boldsymbol{W_{inOTA}}$	25.000	75.000	59.094	33.902
$\boldsymbol{W_{nOTA}}$	4.500	13.500	13.500	13.500
$\boldsymbol{W_{pOTA}}$	9.000	27.000	9.000	9.000
$\boldsymbol{W_{inCS}}$	36.000	144.000	88.525	36.000
$\boldsymbol{W_{nCS}}$	18.000	72.000	18.000	18.000
$\boldsymbol{W_{inCMFB}}$	2.700	8.100	8.100	3.842
$\boldsymbol{W_{nCMFB}}$	0.900	2.70	2.681	0.900
$\boldsymbol{W_{pCMFB}}$	1.800	5.400	4.233	1.800
$\boldsymbol{W_{C}}$	0.450	1.350	1.282	1.350
$\boldsymbol{I_{BIAS}}$	0.100 μA	1.000 μA	0.986 μA	1.000 μA

order of the polynomial function and/or the sample number can be increased. Increasing the sample number results in longer simulation time. Increasing the polynomial order increases the complexity of the function, which results in longer initialization time when performing simulation using the POM. Experiments show that the second order POM constructed using 200 samples provides high accuracy without noticeably increasing the initialization time.

The mean (μ) and the standard deviation (σ) of the percent error and the root-mean-square error (RMSE) for each OP-AMP parameter are listed in Table 4.4. RMSE is calculated using the following expression:

$$RMSE = \sqrt{\frac{\sum_{t=1}^{T}(y_t - \hat{y}_t)^2}{T}}, \tag{21}$$

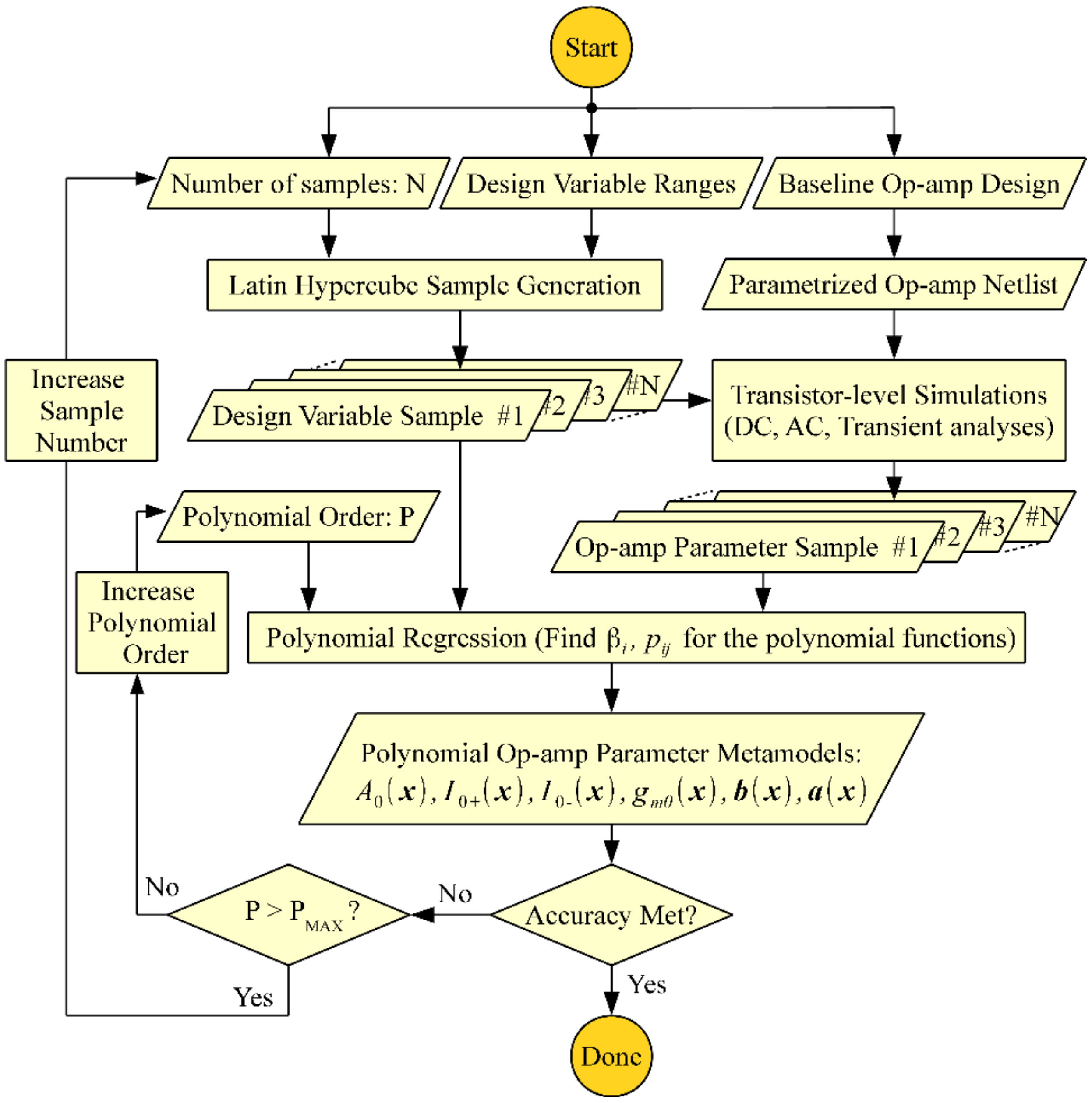

Start
Number of samples: N
Design Variable Ranges
Baseline Op-amp Design
Latin Hypercube Sample Generation
Parametrized Op-amp Netlist
Increase Sample Number
Design Variable Sample #1
#2
#3
#N
Transistor-level Simulations (DC, AC, Transient analyses)
Polynomial Order: P
Op-amp Parameter Sample #1
Increase Polynomial Order
Polynomial Regression (Find β_i, p_{ij} for the polynomial functions)
Polynomial Op-amp Parameter Metamodels:
$A_0(\boldsymbol{x}), I_{0+}(\boldsymbol{x}), I_{0-}(\boldsymbol{x}), g_{m0}(\boldsymbol{x}), \boldsymbol{b}(\boldsymbol{x}), \boldsymbol{a}(\boldsymbol{x})$
No
$P > P_{MAX}$?
No
Accuracy Met?
Yes
Yes
Done

FIGURE 4.8. The proposed flow for OP-AMP metamodel generation.

where T is the total number of test samples, y_t is the true circuit response sample, and $\hat{y}_t$ is the predicted response generated by the metamodel. The POMs for BW, PM, SR, and OP-AMP power dissipation P_D are not used for Verilog-AMS integration which will be discussed in Section 4.6.1, but they are employed in the metamodel-assisted optimization flow proposed in Section 4.5. Most POMs have a mean percent error less than 1% except PM whose RMSE is still relatively low. When the parameter samples exhibit large nonlinearity, the standard deviation of the percent error increases since polynomial regression does not handle large nonlinearity well. The standard deviations of the percent errors of PM, SR, and P_D are all over 10% but less than 15%. However, their RMSEs are all very small.

Table 4.4. Metamodel Accuracy of OP-AMP Parameters.

Op-amp Parameters	$\boldsymbol{\mu}$ **% Error**	$\boldsymbol{\sigma}$ **% Error**	RMSE
A_0	0.27	4.33	1.51 V/V
g_{m0}	0.13	1.5	0.07 μA/V
I_{0+}	0.18	1.30	6.11 nA
I_{0-}	0.13	2.86	8.05 nA
BW	0.53	5.42	40.21 Hz
PM	2.86	14.65	0.83^o
SR	0.11	10.73	0.02 mV/ns
P_D	0.77	11.74	0.70 μW

4.5. Metamodel-Assisted Verilog-AMS based Optimization of OP-AMP

Electronic devices for biomedical applications usually require ultra-low power dissipation. In this section, the OP-AMP design presented in Section 4.2 is optimized using a POM-assisted Cuckoo Search algorithm. Cuckoo Search is a metaheuristic optimization algorithm developed by Yang and Deb [99]. Their preliminary results show that Cuckoo Search can handle optimization problems with high nonlinearity and complex constraints, and can achieve better performance than particle swarm optimization [27]. Hence it is adopted for this OP-AMP design optimization problem in this Chapter. The polynomial metamodels developed using the flow presented in Section 4.4 are used to estimate the OP-AMP characteristics during each iteration for each possible solution. These iterations over the polynomial metamodels are less computationally intensive compared to the same action using a SPICE netlist and hence can lead to way faster convergence. Another optimization result using the same Cuckoo Search algorithm but with the aid of the OP-AMP schematic netlist is also presented for the purpose of comparison with the metamodel assisted approach. In this schematic netlist based optimization, the OP-AMP characteristics for the possible solutions are obtained by running transistor-level simulation. The optimized OP-AMP designs and the computation time of the two optimization approaches are compared. The polynomial-metamodel assisted Cuckoo Search optimization of the OP-AMP is shown in Algorithm 1.

Algorithm 1 POM-assisted Cuckoo Search optimization.

```
1: Initialize $n$ designs $\mathbf{x}_k$ $(k = 1, 2, ..., n)$;
2: $N_{iter} \leftarrow 0$;
3: Evaluate $P_D(\mathbf{x}_i)$ $(k = 1, 2, ..., n)$;
4: Find $P_{D,min}$ among current designs;
5: while $(P_{D,obj} < P_{D,min})$ and $(N_{iter} < N_{iter,max})$ do
6:     Get a new design $\mathbf{x}_i$ randomly by Lévy flights;
7:     Evaluate $P_D(\mathbf{x}_i)$;
8:     Choose a design among $\mathbf{x}_k$ (say $\mathbf{x}_j$) randomly;
9:     if $P_D(\mathbf{x}_i) < P_D(\mathbf{x}_j)$ then
10:        Constraint$_1 \leftarrow A_0(\mathbf{x}_i) > A_{0min}$;
11:        Constraint$_2 \leftarrow BW(\mathbf{x}_i) > BW_{min}$;
12:        Constraint$_3 \leftarrow PM(\mathbf{x}_i) > PM_{min}$;
13:        Constraint$_4 \leftarrow SR(\mathbf{x}_i) > SR_{min}$;
14:        if All constraints met then
15:            Replace $\mathbf{x}_j$ by $\mathbf{x}_j$;
16:        end if
17:    end if
18:    Evaluate $P_D(\mathbf{x}_k)$ $(k = 1, 2, ..., n)$;
19:    Rank $\mathbf{x}_k$ based on $P_D(\mathbf{x}_k)$ $(k = 1, 2, ..., n)$;
20:    Abandon a fraction $(p_a)$ of worst designs;
21:    Generate new designs randomly by Lévy flights ;
22:    Find $P_{D,min}$ among current designs;
23:    $N_{iter} \leftarrow N_{iter} + 2n$;
24: end while
```

The objective of the Cuckoo Search optimization is to **minimize the OP-AMP power dissipation** with the gain, bandwidth, phase margin, and slew rate as constraints. The objective $P_{D,min}$, the initial number of solutions n and the maximum number of iterations allowed $N_{iter,max}$ are tentatively set to be 65μW, 10, and 1200, respectively. The polynomial metamodels $A_0(\mathbf{x})$, $BW(\mathbf{x})$, $PM(\mathbf{x})$, and $SR(\mathbf{x})$ are used to estimate the OP-AMP performance and thus to determine if the constraints are met. The polynomial metamodel $P_D(\mathbf{x})$ is used as the objective function to predicate the OP-AMP power dissipation for each possible solution. The optimized designs generated using the polynomial-metamodel assisted and schematic netlist based Cuckoo Search optimization are shown in Table 4.2 which are denoted as $\text{Optimal}_{\text{POM}}$ and $\text{Optimal}_{\text{SCH}}$, respectively. With the optimized designs, the corresponding OP-AMP performances are obtained by running actual transistor-level simulations. The constraints and objective, and the performance of the optimized OP-AMP designs are summarized in Table 4.5, which shows that the OP-AMP performance has been greatly improved. Table 4.6 compares the performance of $\text{Optimal}_{\text{POM}}$ and $\text{Optimal}_{\text{SCH}}$. The power reduction reported is with respect to the baseline design. In the maximum allowed number of iterations, both optimizations achieve significant power reduction. However, $\text{Optimal}_{\text{POM}}$ completes the optimization in 2.6 seconds while the $\text{Optimal}_{\text{SCH}}$ uses more than 12 hours to finish the process. In this case, the **metamodel-assisted optimization is 17120 times faster** than the traditional method. The iterations of the Cuckoo Search algorithm optimizations are shown in Figure 4.9.

TABLE 4.5. Optimization Results for the OP-AMP Design.

Performance	**Constraint**	**$\text{Optimal}_{\text{POM}}$**	**$\text{Optimal}_{\text{SCH}}$**
$\boldsymbol{A_0}$ (dB)	> 43	56.4	52.8
BW (kHz)	> 50	58.9	85.5
PM (degree)	> 70	84.4	87.7
SR (mV/ns)	> 5	7.1	8
	Objective		
$\boldsymbol{P_D}$ (μW)	~ 65	65.5	68.1

TABLE 4.6. Comparison of Op-amp Optimization

Performance	**$\text{Optimal}_{\text{SCH}}$**	**$\text{Optimal}_{\text{POM}}$**
Power Reduction	$\times 3.71$	$\times 3.86$
Number of iterations	1200	1200
Computation Time	12.5 h	2.6 s
Normalized Speed	1	$\times 17120$

4.6. Meta-macromodeling of the OP-AMP

A complete OP-AMP behavioral model should not only model its small-signal behavior but also the large-signal behavior. The signal transfer function of an OP-AMP obtained

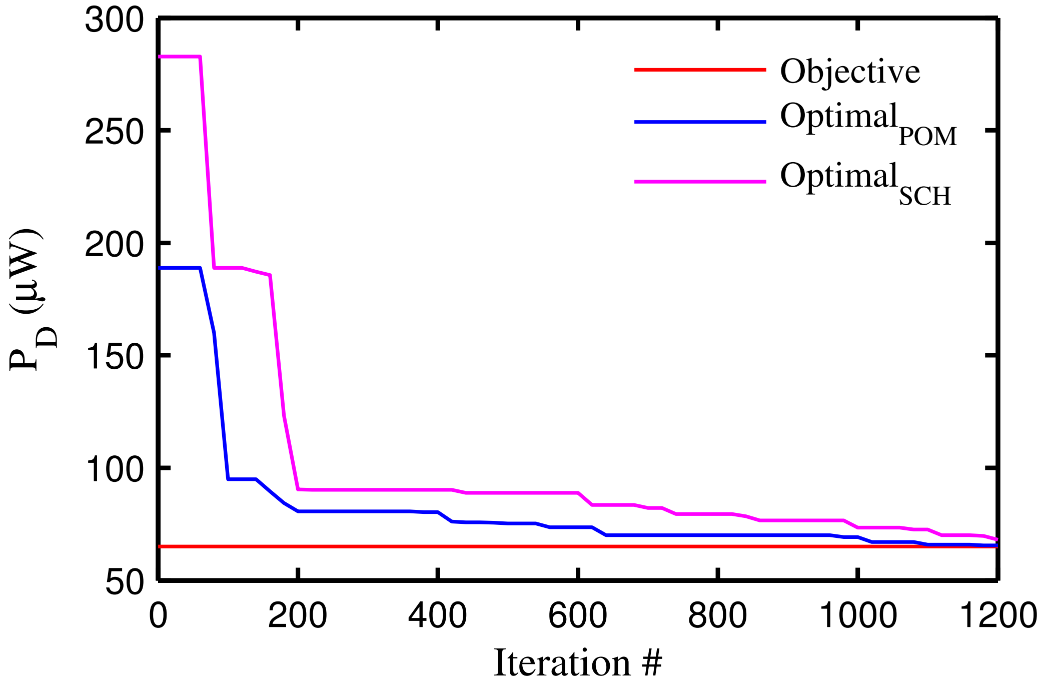
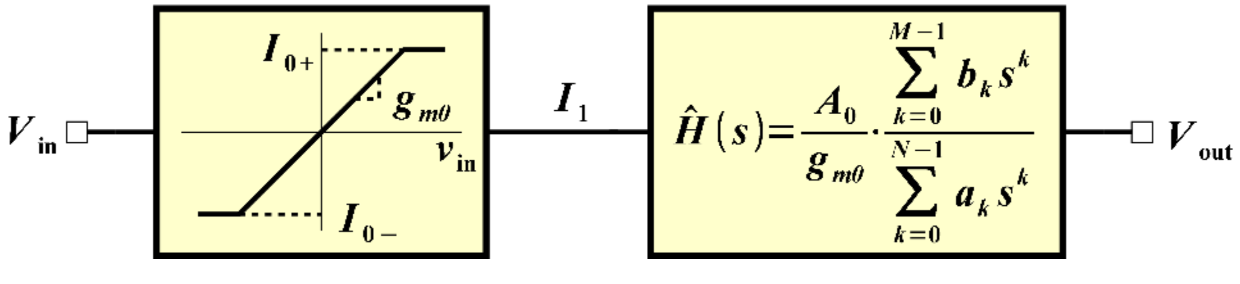


FIGURE 4.9. The iteration of the Cuckoo Search optimization.

from AC analysis can describe the small-signal behavior, but it does not account for the large-signal characteristics including the slewing and settling behaviors. In [15], an OP-AMP macromodel was proposed in order to analyze these behaviors. The model is redrawn in Figure 4.10. The model is customized to be used as the baseline of the OP-AMP meta-macromodel in this Chapter.



V_{in}
I_{0+}
g_{m0}
v_{in}
I_{0-}
I_1
$\hat{H}(s)=\frac{A_0}{g_{m0}}\cdot\frac{\sum_{k=0}^{M-1} b_k s^k}{\sum_{k=0}^{N-1} a_k s^k}$
V_{out}

FIGURE 4.10. The OP-AMP macromodel for transient analysis.

The OP-AMP model consists of two stages as follows:

- Transfer characteristic of the OP-AMP input stage.
- OP-AMP small-signal transfer function.

The first stage describes the transfer characteristic of the OP-AMP input stage due to the limited maximum available positive and negative currents I_{0+} and I_{0-}. A_0 is the open-loop DC gain of the OP-AMP. g_{m0} is the transconductance of the OP-AMP input stage. The second stage is the OP-AMP small-signal transfer function. In [15], the circuit was assumed to be properly designed and thus no zero and only two poles were presented to form the transfer function. This assumption is most likely invalid when performing design exploration. The numbers of poles and zeros are not limited in order to attain high fidelity. For simplicity of notation, the following are defined:

$$\begin{aligned} \mathbf{b} &:= \{b_0, b_1, b_2, ..., b_{M-1}\}, \\ \mathbf{a} &:= \{a_0, a_1, a_2, ..., a_{N-1}\}. \end{aligned} \tag{22}$$

The OP-AMP circuit parameters A_0, I_{0+}, I_{0-}, g_{m0}, $\mathbf{b}$, and $\mathbf{a}$ directly affect the model accuracy. In traditional symbolic macromodeling, these parameters are extracted from transistor-level simulations. When exploring the design space, the extraction has to be redone whenever the values of the design variables are updated. This approach is inefficient. With the proposed meta-macromodeling technique, no extraction is required during design exploration. Prior to design exploration, the OP-AMP parameters are sampled and their metamodels are generated. In metamodel-assisted design exploration, when the optimization algorithm updates the design variable value $\mathbf{x}$, a new set of OP-AMP parameters will be computed using the parameter metamodels without performing circuit simulations and extraction. The new set of OP-AMP parameters can be directly used in the macromodel and thus greatly improves the optimization flow.

4.6.1. *Verilog-AMS-POM*

Transient analyses for complex circuits at transistor-level may take long time. For example, simulating an analog-to-digital converter with high-order delta-sigma modulator may takes hours or days if layout parasitics are included. Thus, it is desired to replace transistor-level blocks with behavioral models as much as possible to reduce the simulation time. Thus, the construction of a Verilog-AMS module for the developed polynomial metamacromodel (Verilog-AMS-POM) of the OP-AMP is needed. The designed module reads a text files storing the β_i and p_{ij} for each OP-AMP parameter POM and the given design variable values. The OP-AMP parameters A_0, I_{0+}, I_{0-}, g_{m0}, $\mathbf{b}$, and $\mathbf{a}$ are then computed. An `analog` process implements the models seen in Fig. 4.10. The Verilog-AMS-POM can be easily scaled for different numbers of design variables and/or polynomial orders.

The AC analysis results of transistor-level schematic, the 1st and the 2nd order Verilog-AMS-POMs for the baseline OP-AMP are shown in Figure 4.11. It can be seen that both the 1st and 2nd order POM results match those of the schematic quite well at lower frequencies. At higher frequencies ($\sim$ 200 MHz) the 1st order polynomial metamodel exhibits noticeable errors while the 2nd order polynomial metamodel still attains good match. The larger error in the 1st order polynomial metamodel is likely due to that linear polynomial metamodel is not sufficiently accurate to capture the nonlinearity that determines the high-frequency poles and zeros. Nevertheless, since this noticeable error resides at the frequency higher than the unity-gain frequency it will only cause a slight error in circuit simulation. The 2nd order polynomial metamodel is adopted in this Chapter. The OP-AMP response for step inputs under the unity-gain configuration is shown in Figure 4.12. The mismatch seen in the step responses is due to the fact that the large input signal causes the poles and zeros of the OP-AMP to deviate from their original location. One solution is, as suggested in [104], to extract two sets of poles and zeros under different bias conditions of interest. Two transfer functions can thus be constructed from different signal levels. However, this is out of the scope of this research.

In order to decide whether to use the OP-AMP Verilog-AMS-POM or the traditional macromodel in the optimization of a large-scale mixed-signal system that requires long transient analysis times, it is necessary to compare their computation time. Assuming that the Verilog-AMS-POM is constructed using 200 samples, that the optimization takes $N_i = 1200$ iterations, and that extracting the OP-AMP parameters for each design takes 60 seconds, the computation time reduction by using the polynomial metamodel based technique is

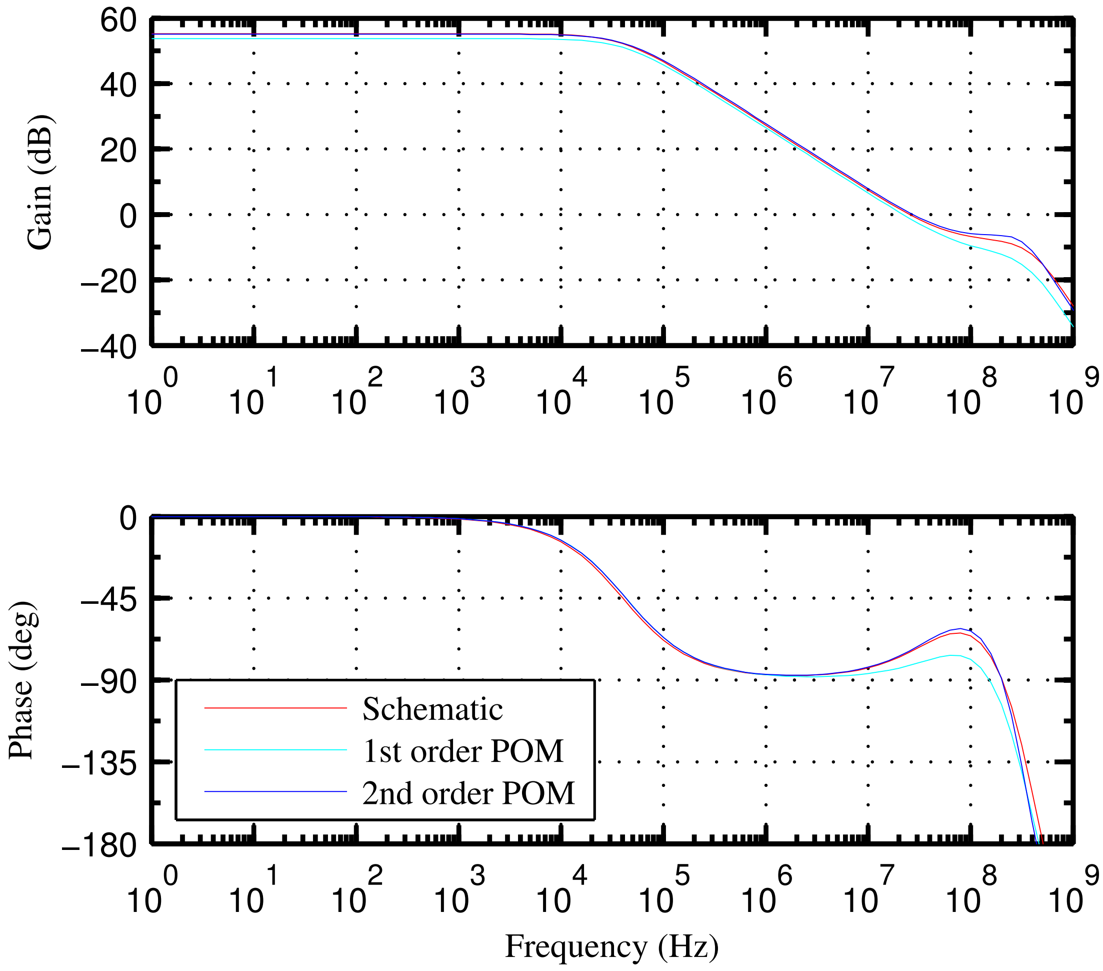


FIGURE 4.11. AC analysis of the OP-AMP.

$t_D \approx 16.7$ hours. This is significant reduction in design cycle and can lead to reduction in the non-recurrent chip cost.

In summary, a proposed polynomial-metamodel assisted OP-AMP optimization flow has been presented. The OP-AMP characteristic polynomial metamodels can accurately predict the op-amp performance with ultra-high speed. The OP-AMP meta-macromodeling technique has been discussed and the Verilog-AMS integration approach has been presented for time-domain simulations. The customized Cuckoo Search algorithm shows promising optimization results. Future research includes studying system-level optimization using the developed OP-AMP meta-macromodel.

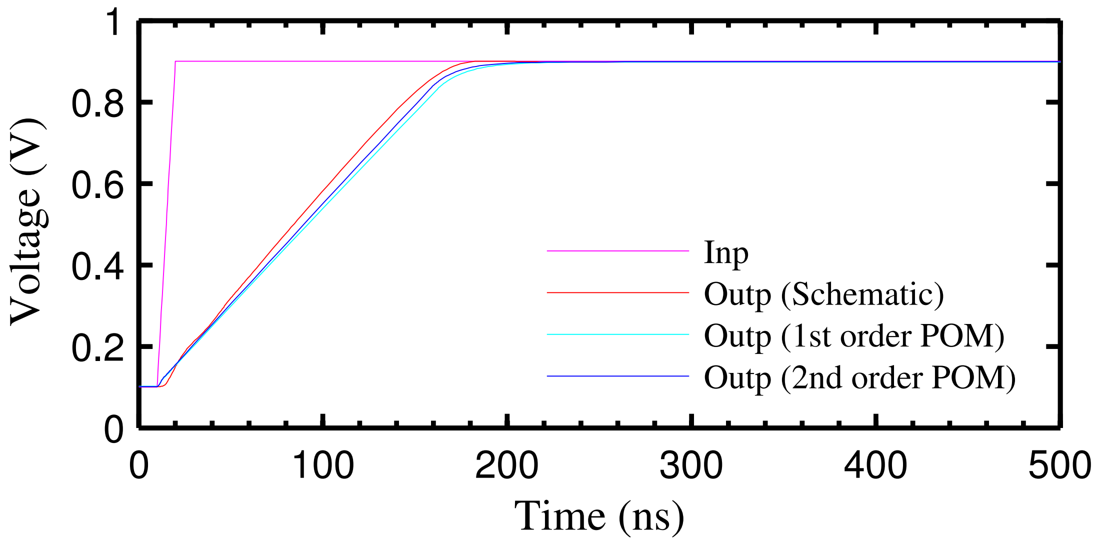


Figure 4.12. Transient simulation of OP-AMP with a step input.

CHAPTER 5

VERILOG-AMS-PAM: VERILOG-AMS INTEGRATED PARASITIC-AWARE METAMODELING USING A PLL AS A CASE STUDY

Parasitics greatly degrade the performance of nano-CMOS circuit designs. They cause significant mismatch between schematic and layout circuit simulations. To account for the parasitic effects, numerous iterations between the schematic and the layout are usually required [32, 31]. This kind of design cycle requires great amounts of time and effort. Layout verification is the major obstacle because the iteration time is mainly spent on layout modification and simulations. Behavioral models that are capable of representing circuit layouts will dramatically shorten the design cycle. Various behavior models have been developed for analog/mixed-signal (AMS) systems [43, 42, 76]. Parasitic effects, however, were not discussed in these papers. Also, the circuit models in these papers were implemented as Verilog-A modules rather than Verilog-AMS modules which are generally more efficient. Modeling techniques such as model order reduction [97] and symbolic model generation [9] have been also proposed but they only work for small circuits.

A metamodeling technique for nano-CMOS AMS circuits was proposed in [24, 26]. The models built with this method accurately reflect parasitic effects. In this dissertation, an accurate VCO behavioral model is proposed based on this approach. This behavioral model is implemented using the Verilog-AMS language which enables fast simulations. Combining the metamodeling technique and the Verilog-AMS simulation, the design verification process achieves a large speedup and maintains reasonably high accuracy. In fact, not only the proposed Verilog-AMS behavioral model can help the design verification, it can also assist the design optimization. A phase-locked loop (PLL) design with an LC VCO design using 180 nm CMOS process is used to demonstrate the modeling technique and the implementation method. Among different PLL architectures, the charge-pump PLL (CPPLL) has been widely used in various system due to its simplicity and effectiveness. Thus this PLL architecture is used in this dissertation [105].

Verilog-A behavioral modules of linear VCOs were used in [45] for PLL jitter characterization and in [108] for aiding a hierarchical CPPLL sizing method. No parasitic effects were included in this model. A quasi-sensitivity analysis approach is proposed in [54] to construct behavioral models and used a characterization mode developed in [53] to extract the circuit parameters including parasitic effects. The authors also adopted the linear VCO model. While the linear model may be sufficient for performing verification on fixed designs, it is not useful for design exploration since the VCO linearity condition is not always valid. The VCO behavioral models developed in [2] and [39] used the table-lookup approach inside Verilog-A modules, which is not efficient. An event-driven analog modeling approach was proposed in [94] which used the Verilog-AMS `wreal` data type to improve the model efficiency. However, it is not clear how the VCO gain and output frequency were modeled.

5.1. High-level Description of the CPPLL

A typical CPPLL consists of several distinct components as follows:

- a phase/frequency detector (PFD),
- a charge-pump (CP),
- a loop filter (LF), and
- a voltage controlled oscillator (VCO).

If the PLL needs to perform frequency synthesis, a frequency divider (FD) will also be employed. The system level topology of a CPPLL is shown in Fig. 5.1. The PFD compares the phase and frequency of the input clock ϕ_{in} and the feedback clock ϕ_{fb} generated by the FD. Both PFD and FD are digital circuits. The CP reads the digital output of the PFD and charges/discharges its output node accordingly. This node also connects the LF and the VCO input. The voltage at this node will be translated by the VCO to the corresponding frequency of it output ϕ_{out}. A CPPLL is a very good mixed-signal system and is used in this dissertation as a case study. The CP, LF, and VCO directly deal with the analog signal therefore they are the most critical parts of a CPPLL. More comprehensive PLL analysis can be found in [79]. In this work, developing a VCO behavioral model that can accurately mimic the VCO physical design is the focus. The model is constructed using the Verilog-AMS language to enable fast design exploration. The other parts of the PLL are also modeled behaviorally with hardware description languages in order to simulate the whole PLL system.

5.2. CPPLL Verilog-AMS Behavioral Model

Mixed-signal systems such as CPPLLs can be simulated using mixed-signal simulators which have two kernels as follows [51]:

- Event-driven digital kernel
- Continuous-time analog kernel

Calling the analog kernel in a mixed-signal simulation is generally far more computationally expensive than calling the digital kernel. Thus for fast design verification with given accuracy requirements, models that will cause unnecessary analog events should be avoided. However, at the same time, the analog kernel is much more accurate compared to the digital kernel.

Figure 5.1 illustrates the CPPLL configuration in this dissertation. The LF consists of three simple passive components R_1, C_1, and C_2. Modeling the LF behaviorally does not improve the simulation efficiency noticeably. Therefore the SPICE model is used for the LF and it is implemented in a schematic view. The PFD and FD are pure digital circuits. The frequency of the FD output ϕ_{fb} is $1/N$ of that of the VCO output ϕ_{out}, where N is the FD division ratio. The PFD activates its output Up or Dn to vary the VCO output until ϕ_{fb} and ϕ_{in} are aligned and have the same frequency. They introduce non-idealities to the system via their signal delay, and the rise/fall time. These non-idealities can be easily described in the digital domain. Thus the behavior of these two blocks are implemented using the Verilog language. The CP has digital inputs and analog output so it is implemented as a Verilog-AMS module. Listing 5.1 shows example source code for the PFD and the CP.

Listing 5.1. Example HDL source code for the PFD and the CP.

```
// PFD Verilog code
`timescale 10ps / 1ps
module pfd (qinc, qdec, ref, clk);
    output qinc, qdec;
    input clk, ref;
```

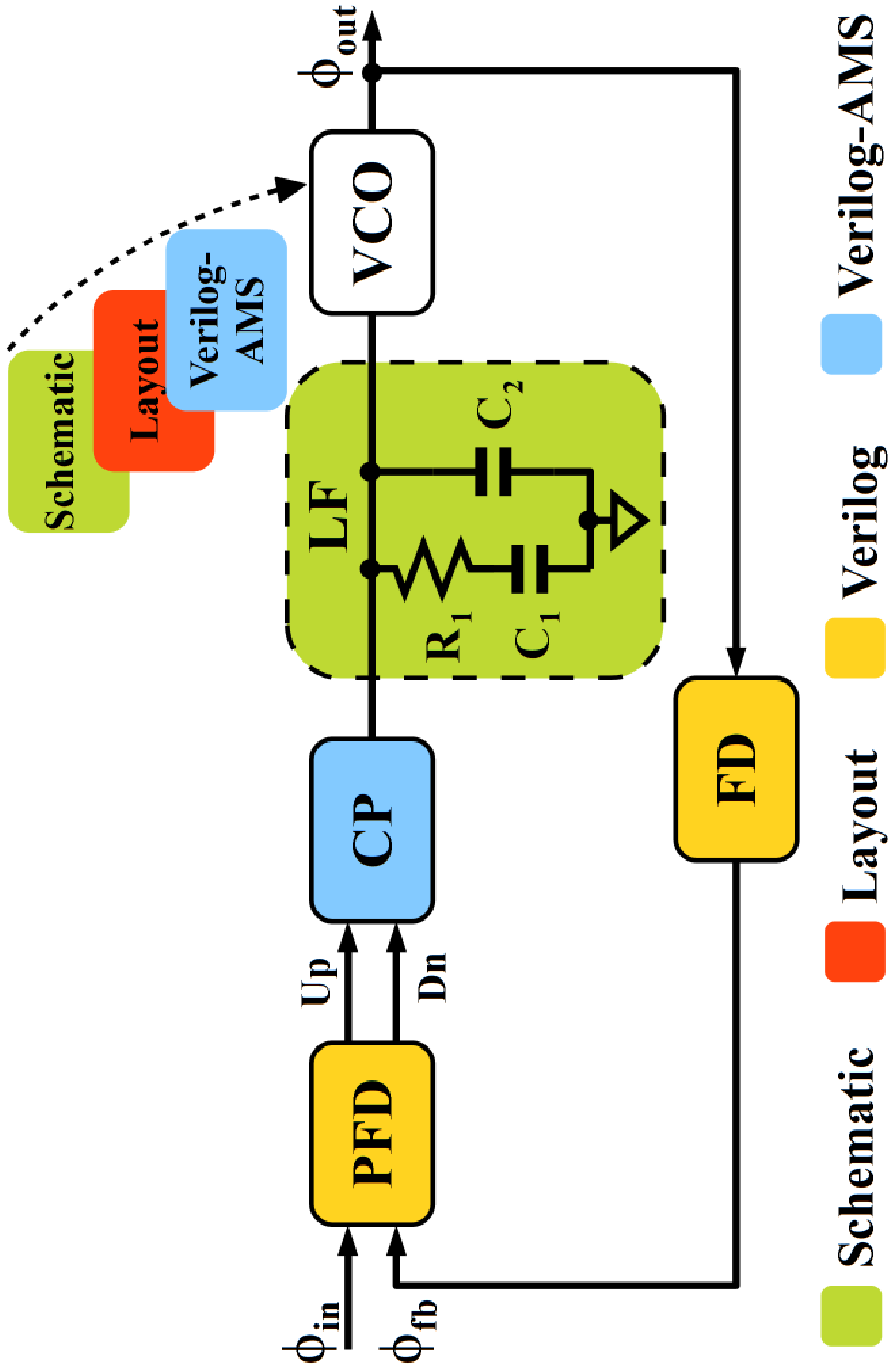


FIGURE 5.1. The CPPLL configuration in this dissertation.

```
    reg qinc, qdec;
    wire reset;
    assign #1 reset = qinc && qdec;
    always @(posedge ref or posedge reset)
        begin
            if (reset) qinc <= 1'b0;
            else qinc <= 1'b1;
        end
    always @(posedge clk or posedge reset)
        begin
            if (reset) qdec <= 1'b0;
            else qdec <= 1'b1;
        end
endmodule

// CP Verilog-AMS code
`include "disciplines.vams"
`timescale 10ps / 1ps
module cp0 (out, inc, dec);
    parameter real cur = 50u;   // output current (A)
    input inc, dec;
    electrical out;
    real iout;
    parameter real vdd = 1.2;
    parameter real gnd = 0;
    analog begin
        @(initial_step) iout = 0.0;
        if (dec && !inc && (V(out) > gnd))
            iout = -cur;
        else if (!dec && inc && (V(out) < vdd))
            iout = cur;
        else iout = 0;
```

```
            I(out) <+ -transition(iout, 0.0, 2p, 2p);
        end
endmodule
```

Three different views have been implemented for the VCO as follows:

- schematic,
- layout, and
- Verilog-AMS.

In this dissertation, an LC VCO design is used to demonstrate the use of metamodel and Verilog-AMS for design exploration. Figs. 5.2 and 5.3 show the schematic and layout views of the LC VCO design. Both schematic and layout views use SPICE models for simulations. While the layout view includes the parasitic elements and therefore takes longer to simulate, it results in much better estimate of the real silicon performance. Table 5.1 lists the number of elements in the schematic view and parasitic extracted layout view. The parasitics are extracted with RCLK extraction.

TABLE 5.1. Element Counts for The LC VCO Schematic and Layout Views

	Schematic	**Layout**
Transistor	4	4
Inductor	1	10
Capacitor	2	38
Resistor	0	560
Total	7	612

The Verilog-AMS view implements an accurate behavioral model. The modeling approach is detailed in Section 5.3. The commercial tools AMS Designer and Hierarchical Editor provided by Cadence Design Systems, allows the user to simulate a system whose sub-blocks are implemented in various types of views (mixed-mode simulation). The view of each sub-block can be conveniently replaced by another type of view. With these tools, we are able to switch the view of the VCO module, as shown in Fig. 5.1, and to compare their simulation results.

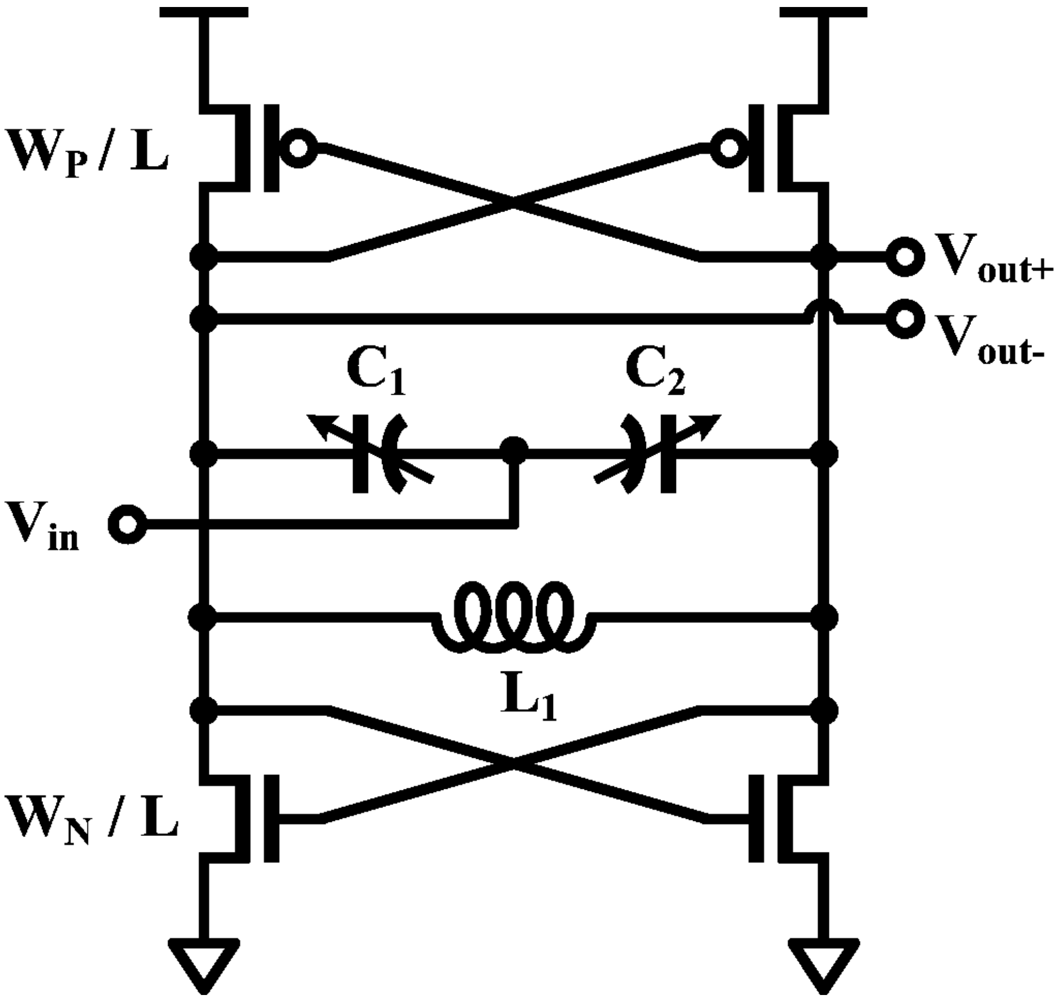


FIGURE 5.2. The LC VCO schematic.

5.3. VCO Polynomial Metamodeling

The VCO behavior is mainly determined by its voltage-to-frequency transfer curve. Thus the VCO behavioral model can be obtained by constructing this curve. A common way to model a VCO is assuming the VCO is perfectly linear and modeling it with the following equation:

$$f_{osc} = f_0 + K_{VCO}V_C, \tag{23}$$

where f_{osc} is the VCO oscillation frequency, f_0 is the center frequency, K_{VCO} is the gain, V_C is the control voltage at the VCO input. This linear model can be implemented by sampling two data points on the VCO transfer curve. When performing design exploration, however,

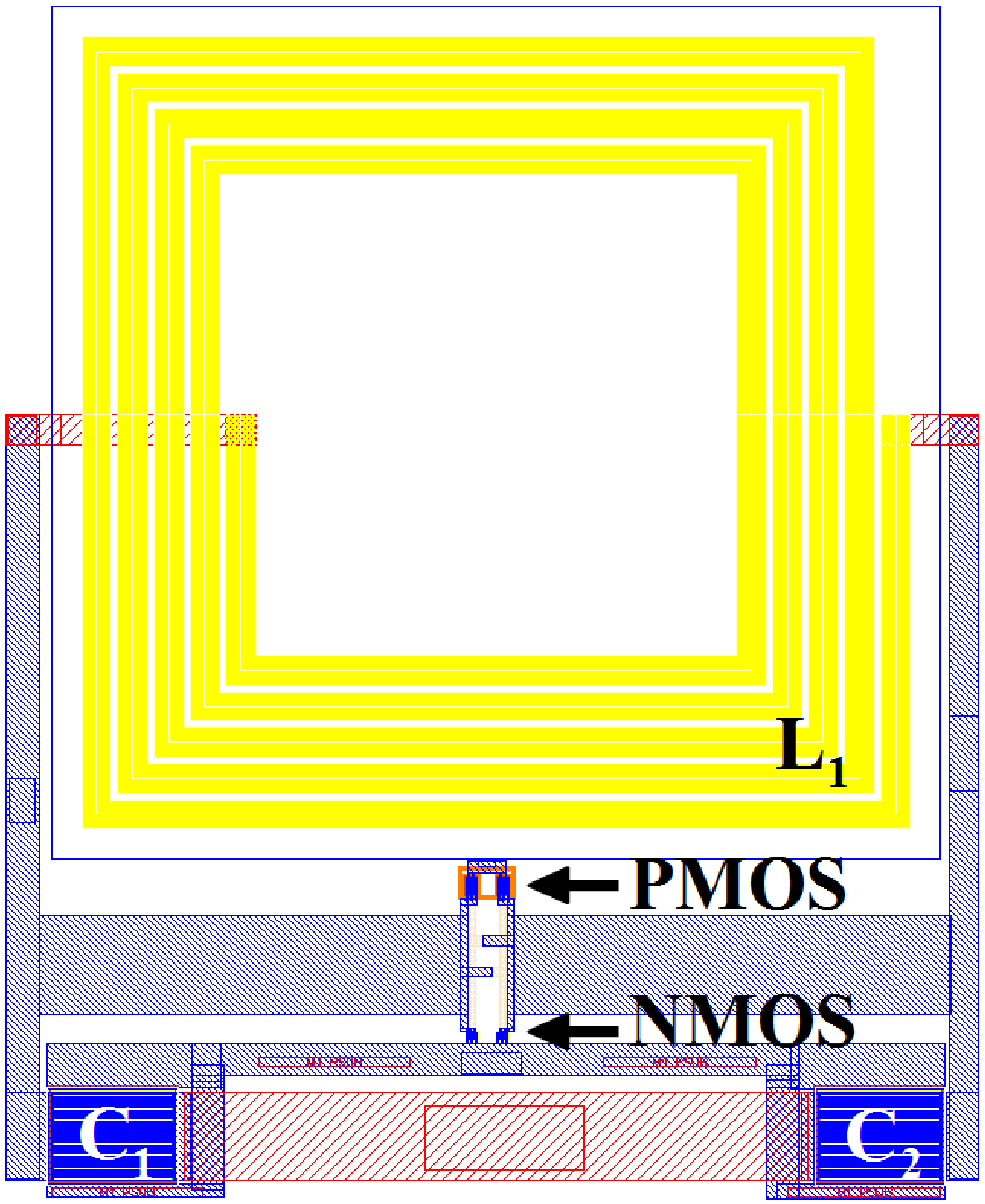


FIGURE 5.3. The LC VCO layout.

the linearity is not guaranteed, which leads to invalid simulation results. Also, parasitic effects from layout extraction further degrade the accuracy of this modeling approach. To account for the non-linearity and layout parasitics, the metamodeling approach suggested in [24, 26] is used.

We chose to implement polynomial metamodels because they have the following advantages:

- They are simple closed form equations which are easy to implement.
- Their form is flexible so that one can quickly examine and compare the accuracy of polynomial models with different degree.
- They have been widely used and their properties have been very well understood.

The polynomial metamodel used here is the following:

$$f(\mathbf{x}) = \sum_{i=0}^{K-1} \beta_i {x_1}^{p_{1i}} {x_2}^{p_{2i}} {x_3}^{p_{3i}}, \tag{24}$$

where x_1, x_2, and x_3 are three input variables corresponding to W_P, W_N, and V_C in this work, respectively. K is the number of basis functions this model has and β_i is the coefficient for the basis function. $f(\mathbf{x})$ is the output to approximate the true model. In order to construct the metamodel for a given VCO design, for each basis functions the coefficient β_i and the power terms p_{1i}, p_{2i}, and p_{3i} for each input variables need to be obtained. This is done in three steps: first, a set of input variables $[x_1\ x_2\ x_3]$ is generated using the Lain hypercube sampling (LHS) technique; second, circuit simulations are performed and the outputs for each set of inputs are saved; third, with the inputs and outputs from previous steps, the coefficients and the power terms that lead to a model with good fit are computed. In order to incorporate the parasitic effects into the model without redoing the layout for each simulation, the netlist for the extracted layout view is parameterized for W_P and W_N. Note that the inductor and capacitors were not parameterized because they have a first order impact on the parasitics and varying them may lead to potential inaccuracy. The parameterization is done by replacing the original transistor models in the netlist with parameterized ones. Listing 5.2 shows a portion of the parameterized layout netlist where the original PMOS model has been replaced with a parameterized one. The netlist is in the form of Verilog-AMS and thus can be accepted by the AMS simulator.

Listing 5.2. A portion of the parameterized netlist for the VCO layout view.

```
inductor #(.l(8.244e-11)) l1_291 (\291:RLJUNC_J1 , \25:Voutn );
inductor #(.l(8.797e-11)) l1_2 (\2:RLJUNC_J1 , \5:Voutp );
pmos1 #(.w(((cds_globals.Wp_VCO) / (4))), .l(cds_globals.L),
      .as((((cds_globals.Wp_VCO) / (4)) < 599.5n) ? (((((200n > (((400n) -
          200n) + 400n)) ? 200n : (((400n) - 200n) + 400n)) * 600n) + (((
          cds_globals.Wp_VCO) / (4)) * 200n)) + (floor(((4) - 1) / 2.0) *
          (((((400n) - 200n) + 400n) * 600n) + (((cds_globals.Wp_VCO) / (4))
           * 400n))) + ((((4) / 2) - floor((4) / 2) == 0) ? ((((200n >
          (((400n) - 200n) + 400n)) ? 200n : (((400n) - 200n) + 400n)) * 600
          n) + (((cds_globals.Wp_VCO) / (4)) * 200n)) : 0)) / 4 : ((((400n >
           (((400n) - 200n) + 400n)) ? 400n : (((400n) - 200n) + 400n)) * ((
          cds_globals.Wp_VCO) / (4))) + (floor(((4) - 1) / 2.0) * ((((400n)
          - 200n) + 400n) * ((cds_globals.Wp_VCO) / (4)))) + ((((4) / 2) -
          floor((4) / 2) == 0) ? (((400n > (((400n) - 200n) + 400n)) ? 400n
          : (((400n) - 200n) + 400n)) * ((cds_globals.Wp_VCO) / (4))) : 0))
          / 4),
      .ad((((cds_globals.Wp_VCO) / (4)) < 599.5n) ? ((floor((4) / 2.0) *
          (((((400n) - 200n) + 400n) * 600n) + (((cds_globals.Wp_VCO) / (4))
           * 400n))) + ((((4) / 2) - floor((4) / 2) != 0) ? ((((200n >
          (((400n) - 200n) + 400n)) ? 200n : (((400n) - 200n) + 400n)) * 600
          n) + (((cds_globals.Wp_VCO) / (4)) * 200n)) : 0)) / 4 : ((floor
          ((4) / 2.0) * ((((400n) - 200n) + 400n) * ((cds_globals.Wp_VCO) /
          (4)))) + ((((4) / 2) - floor((4) / 2) != 0) ? (((400n > (((400n) -
           200n) + 400n)) ? 400n : (((400n) - 200n) + 400n)) * ((cds_globals
          .Wp_VCO) / (4))) : 0)) / 4),
    .ps((((cds_globals.Wp_VCO) / (4)) < 599.5n) ? (((2 * ((200n > (((400n)
        - 200n) + 400n)) ? 200n : (((400n) - 200n) + 400n))) + 1.6u) + (
        floor(((4) - 1) / 2.0) * ((2 * (((400n) - 200n) + 400n)) + 2u)) +
        ((((4) / 2) - floor((4) / 2) == 0) ? ((2 * ((200n > (((400n) - 200
        n) + 400n)) ? 200n : (((400n) - 200n) + 400n))) + 1.6u) : 0)) / 4
        : (((2 * ((400n > (((400n) - 200n) + 400n)) ? 400n : (((400n) -
        200n) + 400n))) + (2 * ((cds_globals.Wp_VCO) / (4)))) + (floor
        (((4) - 1) / 2.0) * ((2 * (((400n) - 200n) + 400n)) + (2 * ((
        cds_globals.Wp_VCO) / (4))))) + ((((4) / 2) - floor((4) / 2) == 0)
         ? ((2 * ((400n > (((400n) - 200n) + 400n)) ? 400n : (((400n) -
        200n) + 400n))) + (2 * ((cds_globals.Wp_VCO) / (4)))) : 0)) / 4),
    .pd((((cds_globals.Wp_VCO) / (4)) < 599.5n) ? ((floor((4) / 2.0) * ((2
         * (((400n) - 200n) + 400n)) + 2u)) + ((((4) / 2) - floor((4) / 2)
         != 0) ? ((2 * ((200n > (((400n) - 200n) + 400n)) ? 200n : (((400n
        ) - 200n) + 400n))) + 1.6u) : 0)) / 4 : ((floor((4) / 2.0) * ((2 *
         (((400n) - 200n) + 400n)) + (2 * ((cds_globals.Wp_VCO) / (4)))))
        + ((((4) / 2) - floor((4) / 2) != 0) ? ((2 * ((400n > (((400n) -
        200n) + 400n)) ? 400n : (((400n) - 200n) + 400n))) + (2 * ((
        cds_globals.Wp_VCO) / (4)))) : 0)) / 4)
    , .m(”(1)*(4)”))
(* integer passed_mfactor = ”m”; *)
PM1 (Voutp, Voutn, cds_globals.\vdd! , cds_globals.\vdd! );
```

In this dissertation, the VCO output frequency and its power consumption are of interest. Therefore two metamodels are built for them, respectively. They share the same power terms for the input variables, while the coefficient β_i in the two models are different. After these values are computed, they are written into a text file which will be read by the VCO Verilog-AMS module to implement the model. A quadratic polynomial metamodel with first order interaction has been implemented. Table 5.2 shows the layout of the text file storing the values for the power terms and the coefficients for this model are obtained from 100 samples. Because of the use of the parameterized netlist, the sampling process took less then ten minutes. In the table, $\beta_{i,f}$ and $\beta_{i,p}$ are the coefficients for the frequency and power consumption models, respectively. These values are read into the Verilog-AMS module during the initialization process.

TABLE 5.2. Layout of The Text File Storing The Power Terms and The Coefficients for The VCO Quadratic Polynomial Metamodel

i	p_{1i}	p_{2i}	p_{3i}	$\beta_{i,f}$	$\beta_{i,p}$
0	0,	0,	0,	2.113e+009,	1.385e-005
1	1,	0,	0,	-3.214e+012,	44.459e+000
2	2,	0,	0,	3.456e+016,	-2.804e+005
3	0,	1,	0,	6.869e+012,	39.729e+000
4	1,	1,	0,	-1.021e+017,	2.911e+005
5	0,	2,	0,	-2.071e+017,	-1.080e+006
6	0,	0,	1,	3.513e+008,	-8.271e-004
7	1,	0,	1,	-2.565e+012,	-31.282e+000
8	0,	1,	1,	-5.331e+012,	-11.392e+000
9	0,	0,	2,	0.000e+000,	1.041e-003

Listing 5.3 shows the example source code of the VCO Verilog-AMS module. The part of the basis function related to the input variables W_P and W_N is constructed in the `initial` block. The remainder of the basis function is constructed in the `always` block since the third variable V_C needs to be updated continuously during the simulation. The output signal of this module is implemented to be digital logic type to reduce the computation cost. As in the PFD and FD modules, the non-idealities associated with this output signal can be modeled in the digital domain.

LISTING 5.3. Verilog-AMS code of the polynomial VCO metamodel.

```
`timescale 10ps / 1ps
`include "disciplines.vams"
module vco_metamodel (out, in);
    parameter  metafile = "metamodel.csv";      // Metamodel file
    parameter integer nb = 3; // Number of variables of the metamodel
    parameter  powerfile = "vco_power_meta.csv";  // File to store power
    parameter real vdd = 1.2 from (0:10);
    parameter real gnd = 0;
    parameter real wn = 10u from [0.2u : 20u];
    parameter real wp = 20u from [0.4u : 40u];
    input in;
    output out;
    electrical in;
    logic out;
    reg out;
    reg clk;    // clock for strobing and generating results
    real vin;
    integer i = 1;
    real d[1:5];
    real bf[1:nb], bp[1:nb];
    real pvin[1:nb];
    real freq, power;
    integer metaf, readfile, presult;
    initial begin  // Read metamodels
        out = 0;   // Initialize vco digital output
        clk = 0;
        metaf = $fopen({"path_to_metamodel_file", metafile}, "r");
        while (!$feof(metaf))
        begin
            readfile = $fscanf(metaf, "%e,%e,%e,%e,%e\n",
                                    d[1], d[2], d[3], d[4], d[5]);
            bf[i] = pow(wp, d[1]) * pow(wn, d[2]) * d[4];
            bp[i] = pow(wp, d[1]) * pow(wn, d[2]) * d[5];
            pvin[i] = d[3];
            i = i + 1;
        end
        $fclose(metaf);
        metaf = $fopen({"path_to_save_power", powerfile}, "w");
    end
    always begin
        vin = V(in);
        if (vin > vdd) vin = vdd;
        else if (vin < gnd) vin = gnd;
        freq = 0;
        power = 0;
        for (i = 1; i <= nb; i = i + 1)
        begin
            freq = freq + bf[i] * pow(vin, pvin[i]);
            power = power + bp[i] * pow(vin, pvin[i]);
        end
        if (freq < 0) freq = 1G;
        else if (freq > 10G) freq = 10G;
        #(0.5 / freq / 10p)   // 10p is the unit time in the timescale
        out = ~out;
    end
    always #5 clk = ~clk;
    always @(posedge clk) begin
        $fstrobe(metaf, "%e, %e", $abstime, power);
    end
endmodule
```

This Verilog-AMS module can be easily reconfigured for metamodels with different degrees by changing the parameter `K`. In Fig. 5.4, the simulation results of the VCO transfer curves for the design in Figs. 5.2 and 5.3 are shown. The parasitics cause a large difference between the schematic and layout results both in the VCO center frequency and the gain. Metamodel 1 is the Verilog-AMS module with the quadratic model from 100 samples. Metamodel 2 is the module with a 5-th degree polynomial model from 500 samples. Metamodel 2 does not demonstrate significant improvements over Metamodel 1. Thus Metamodel 1 is used in the PLL simulations shown in Sections 5.4 and 5.5. Difference between the transfer curves of layout and metamodel Verilog-AMS view can still be observed, which means a better metamodel may be used to further improve the accuracy. However, as will be seen in Section 5.4, this polynomial metamodel is sufficient for system level PLL verification to simulate lock time and average power dissipation.

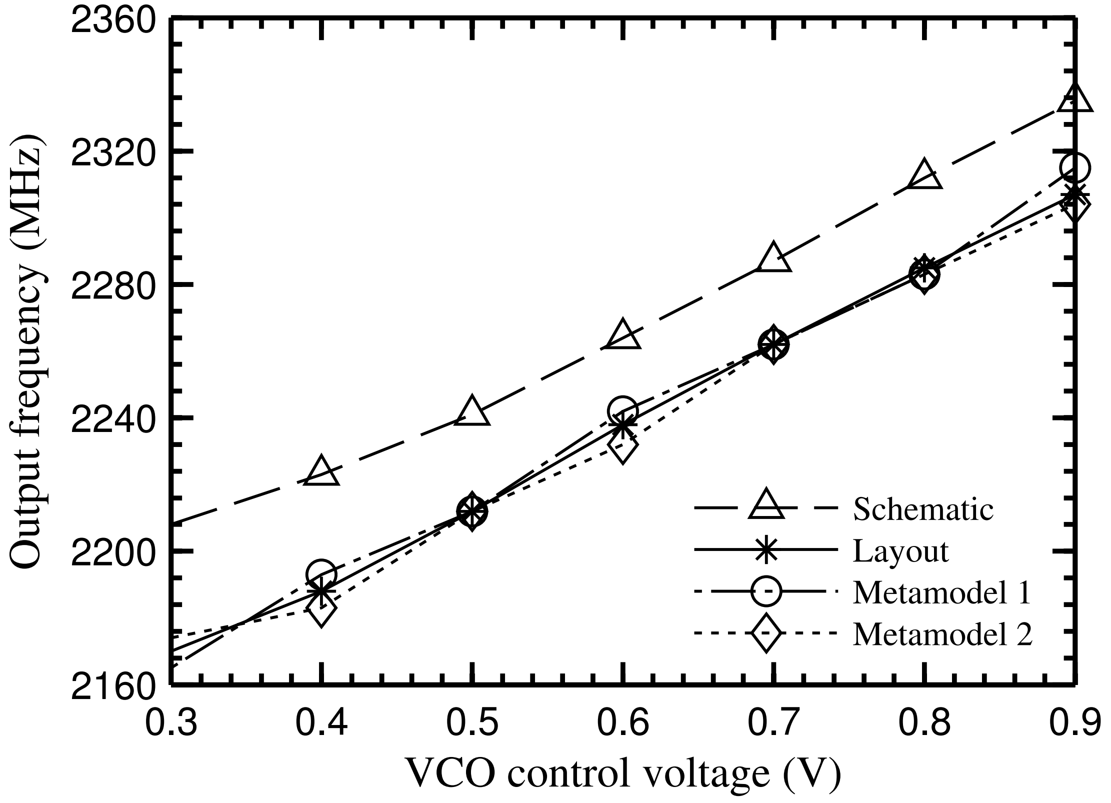


FIGURE 5.4. VCO transfer curves for threes different views.

5.4. Metamodel-Integrated Verilog-AMS PLL Simulation

In this section, PLL simulations with the VCO design shown in Figs. 5.2 and 5.3 are demonstrated. The PLL configuration shown in Fig. 5.1 is used. The PFD and FD are in Verilog view. The CP is in Verilog-AMS view and the LF is in schematic view. The views for the aforementioned blocks were not changed throughout all simulation runs. The VCO view was changed from schematic, to layout, and then to Verilog-AMS views. Two Verilog-AMS views have been implemented, as proposed in Section 5.3:

- one for the linear model and

- one for the quadratic metamodel.

The results for using different VCO views are compared in terms of speed and accuracy.

A 550 MHz input clock ϕ_{in} is assigned to the PLL input. The FD has a division ration of 4. Thus the desired frequency for the PLL output clock ϕ_{out} is 2200 MHz. Fig. 5.5 shows the ϕ_{out} frequencies from 500 ns transient simulations with different VCO views. Although the PLLs with the different VCO views are all able to lock to the same correct frequency, the one with the schematic view shows quite different locking behavior compared to the one with the layout views. This mismatch is caused by the parasitic effects which greatly change the VCO characteristics. The one with the linear model shows improvements over the the schematic one since the parasitics have been taken into account. However, it still has large errors, for example in the lock time. The PLL with the metamodel Verilog-AMS view offers the best approximation of the true model and accurately estimated the lock time. To further understand the behavior of the PLL with different VCO views, the critical analog signal V_C was inspected.

Fig. 5.6 compares the V_C waveform from the four simulations. Again, the metamodel Verilog-AMS view does the best job of approximating the layout view behavior. The PLL with the schematic VCO view can just barely lock to 2200 MHz since V_C is approaching the NMOS threshold. This shows that the center frequency and the gain of the schematic VCO view largely differ from the layout one. These further confirm the VCO transfer curves plotted in Fig. 5.4. It is worth noting that compared to the linear model, the metamodel performs much better in approximating the circuit behaviors in both the startup transient (from 0 μs to 0.32 μs) and steady-state (starting at 0.32 μs) phases. Not only the metamodel output accurately match the transition rate of the true output but also closely tracks the signal amplitude of the true output.

The Verilog-AMS metamodel also facilitates the estimation of the power consumption. Fig. 5.7 shows the average VCO power consumption per fifty cycles in the four simulations.

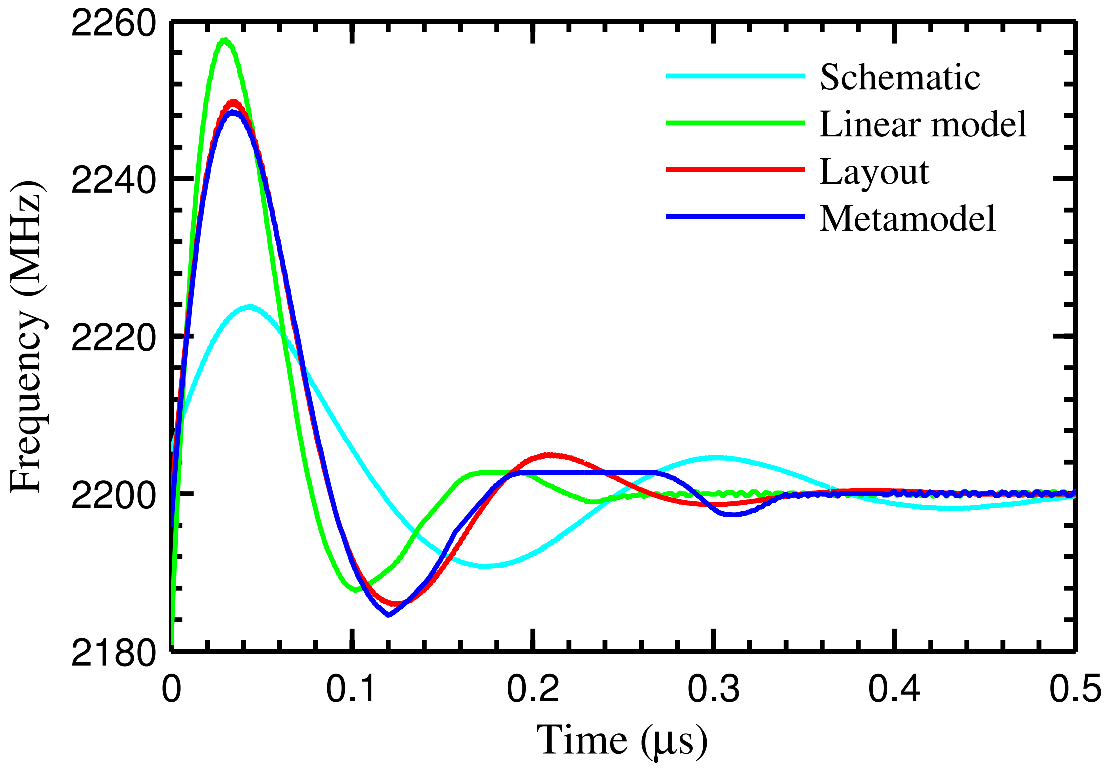


FIGURE 5.5. PLL output frequency from AMS simulation with three different VCO views.

It once again confirms that the Verilog-AMS metamodel can better model the layout counterpart. Table 5.3 summarizes the PLL simulation results and compares the accuracy of the linear model and the proposed metamodel.

In Table 5.3, the estimated PLL lock time is listed. The one from the simulation with the VCO layout view serves as the true model. The errors resulting from the other two models are computed. The metamodel achieves a very low error rate of 0.7 %, while the linear model causes a large error of 31.7 %. $\mathbf{f_{Locked}}$ is the PLL output frequency when it is locked. $\mathbf{P_{Locked}}$ is the average VCO power consumption when the PLL is locked. Again, the metamodel resulted in a good estimate of the true power dissipation. The $\mathbf{V_C}$ root-mean-square error (RMSE) of the models for the 500 ns simulations are also listed.

Table 5.4 compares the runtime for the PLL transient simulations with the layout, the schematic, and the Verilog-AMS metamodel VCO views. The Verilog-AMS metamodel achieves roughly a 10× speedup compared to the layout. Note that in practice the VCO

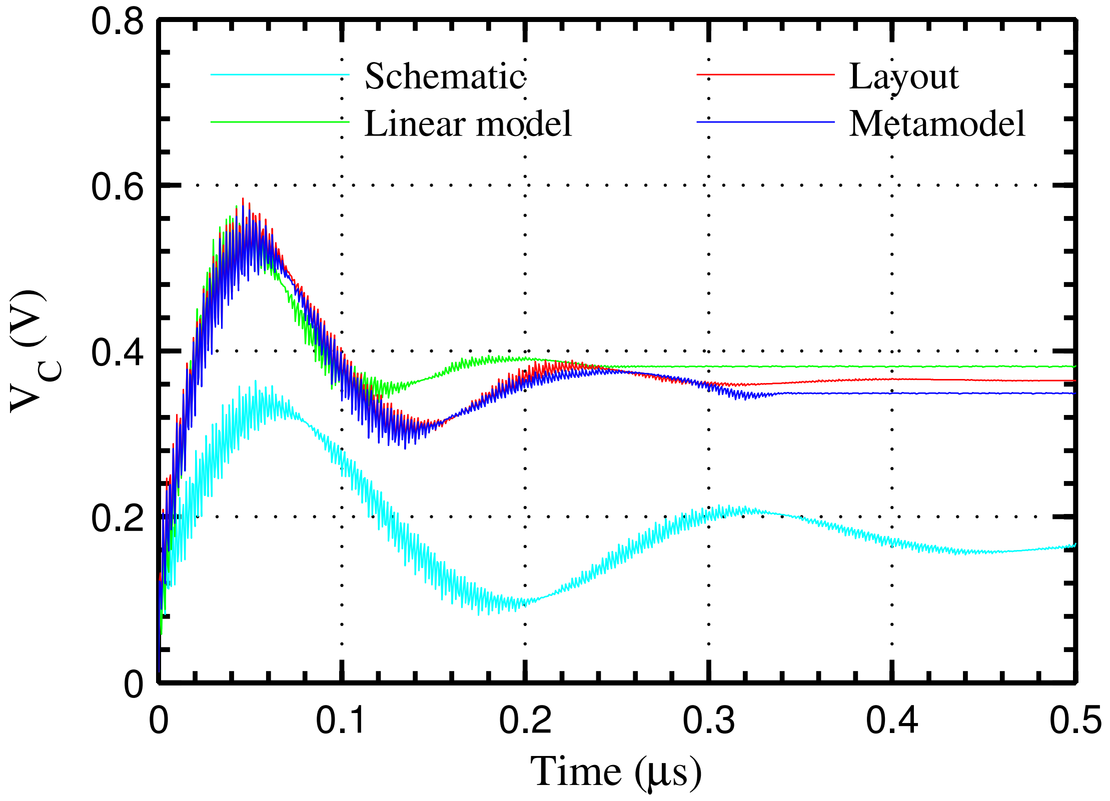


FIGURE 5.6. VCO control voltage waveforms from PLL simulations.

TABLE 5.3. Comparison of The PLL Simulations with Different VCO Modules

	Layout	**Linear Model**	**Metamodel**
Lock time (ns)	335.4	229.1	332.9
Error %	0.0 %	31.7 %	0.7 %
$\mathbf{f_{Locked}}$ (MHz)	2199.99	2199.99	2199.99
Error %	0.0 %	0.0 %	0.0 %
$\mathbf{P_{Locked}}$ (μW)	602	560	620
Error %	0.0 %	7.0 %	3.0 %
$\mathbf{V_C}$ RMSE (mV)	0	33.508	10.889

design may contain more complex circuitry which leads to longer runtimes for a simulation run. The runtime for simulation with the Verilog-AMS module will not be different. Thus the speedup will be more significant in that case. Also note that the Verilog-AMS language along with the AMS simulator allow us to model and simulate the rest of blocks in the form of HDL, which is a great advantage over the full transistor simulation.

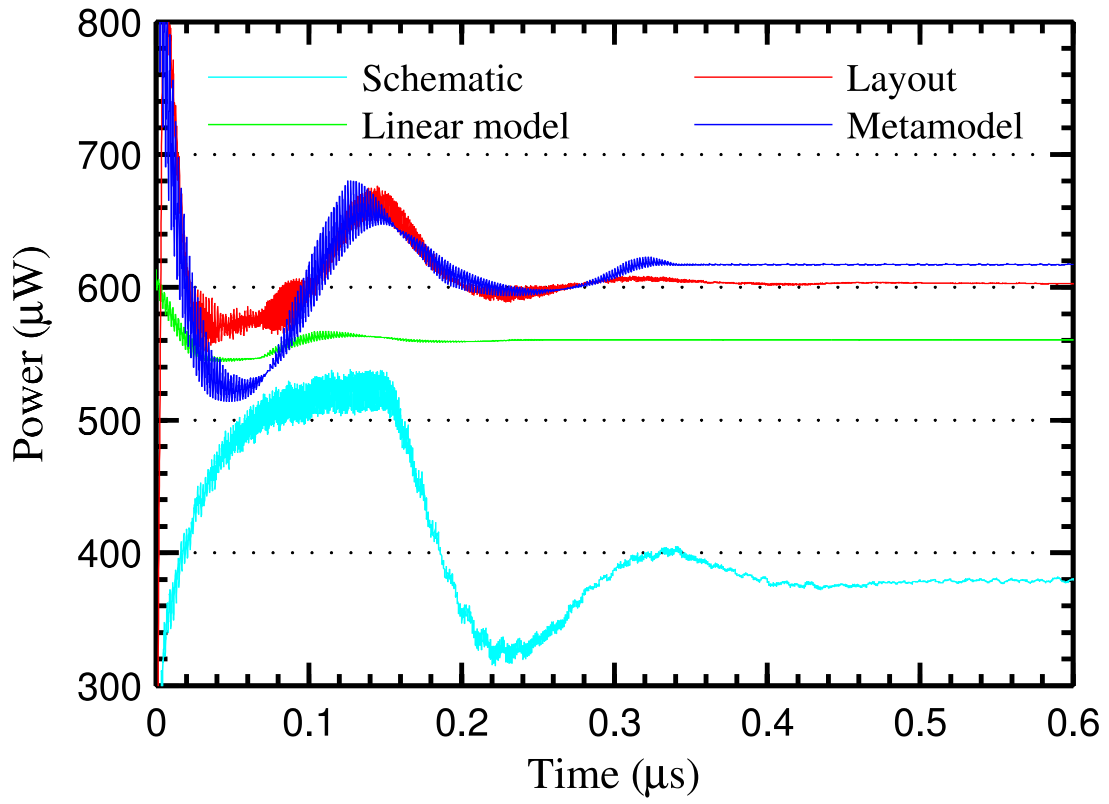


FIGURE 5.7. Average VCO power consumption per 50 cycles

TABLE 5.4. Comparison of The Speed of The PLL Simulations with Different VCO Modules

	Layout	**Schematic**	**Metamodel**
Runtime	80.5 s	40.3 s	8.7 s
Normalized speed	1×	∼ 2×	∼ 10×

5.5. Metamodel Assisted PLL Optimization

Not only the metamodel and its Verilog-AMS implementation can speedup design verifications, they can also accelerate design optimization. In this section, applying the metamodel and metamodel integrated Verilog-AMS to assist the PLL optimization is demonstrated. Nowadays low-power devices have been used in many applications; for example smart mobile phones, tablet and other embedding systems are available every where [67, 68, 69, 71]. The wakeup time for these devices is crucial, which requires short lock time if PLLs are employed. The goal of this optimization methodology is to find designs with minimized lock time and low power consumption. The transistor sizes W_P and W_N of the LC VCO are chosen as the design variables to be optimized. A simple optimization flow is investigated to highlight the use of the metamodel and the Verilog-AMS implementation. In practice, more sophisticated flows can be used to handle problems of larger sizes. Table 5.5 summarizes the optimization flow. A large set of optimization algorithms which are presented in literature can also be used for the purpose of optimization [28]. For example, artificial bee colony, stochastic gradient, particle swarm optimization algorithms are possible options for intelligent algorithm. The important point is that the optimization over metamodels instead of the SPICE netlists allows the use of sophisticated algorithms for searching optimal solutions [28, 73, 23].

TABLE 5.5. Summary of The Optimization Flow

Step #	**Action**		**Design Space** (total design counts)
1	Define Design space	$\rightarrow$	961
2	Shrink Design space with tuning range constraint	$\rightarrow$	320
3	Run AMS simulation to obtain design choices with minimized lock time	$\rightarrow$	5
4	Select optimal design with low-power consideration	$\rightarrow$	1
5	Verify the final design with layout simulation	$\rightarrow$	Done

In the first step the ranges of the design variables W_P and W_N are defined to be 10–30 μm and 5–15 μm, respectively. These two design variables re used as an example. However, other design variables and additional number of variables can be used. The metamodel assisted approach allows this easily. Within each range, 31 values are evenly selected, which results in a total of 961 possible designs. This again is a specific instance; other values can also be chosen. The larger the number of values, the finer is the search and of course the slower is the convergence process. However, with the use of metamodels very finer searches are possible. The design space is then reduced by applying the tuning range constraint. We define the desired VCO frequency tuning range to be 2180–2300 MHz. However, the target frequency will change depending on the target application of the PLL. In addition other characteristics of the PLL such as phase noise can be used in the optimization as objective as well as constraints. A metamodel is used in this step to calculate the VCO tuning range for each design without performing circuit simulations. Only 320 designs are left after this step. Verilog-AMS simulations are then run to obtain the PLL lock time for these designs. The simulations only took 30.36 minutes to complete due to the use of the Verilog-AMS module. The top five designs with the minimum lock time are saved. These designs are listed in Table 5.6 along with their average power consumption when the PLL is locked. The table will be much more complex when additional design variables and optimization objectives and constraints are used. The table shows five different design choices from the possible solutions. These choices relate to a target application of the PLL.

TABLE 5.6. Comparison of The Choices for Optimal Designs

Choice #	W_P (μm)	W_N (μm)	**Lock Time** (ns)	$\mathbf{P_{Locked}}$ (μW)
1	23.2	5	328.6	504
2	21.4	5	328.7	486
3	21.4	5.3	330.4	494
4	22	5	330.4	492
5	22.6	5.3	330.4	506

For low power applications such as mobile computing, choice 2 from Table 5.6 is selected as the final design for its lowest power consumption. Similarly, for shortest locking time choice 1 can be selected from the table. Fig. 5.8 shows the top five design candidates in the design space of 961 designs; these are marked as red circles in the surface plot. Although the lock time can be further minimized the resultant designs would violate the tuning range requirements. Table 5.7 compares the original LC VCO design (baseline) shown in Fig. 5.2 and Fig. 5.3 and the optimal design. The optimization reduces both the lock time and the power consumption. Fig. 5.9 shows the PLL simulation with the VCO layout view of the optimal design relocks from 2180 MHz to 2300 MHz. This simulation again performs the functional verification of the optimal design.

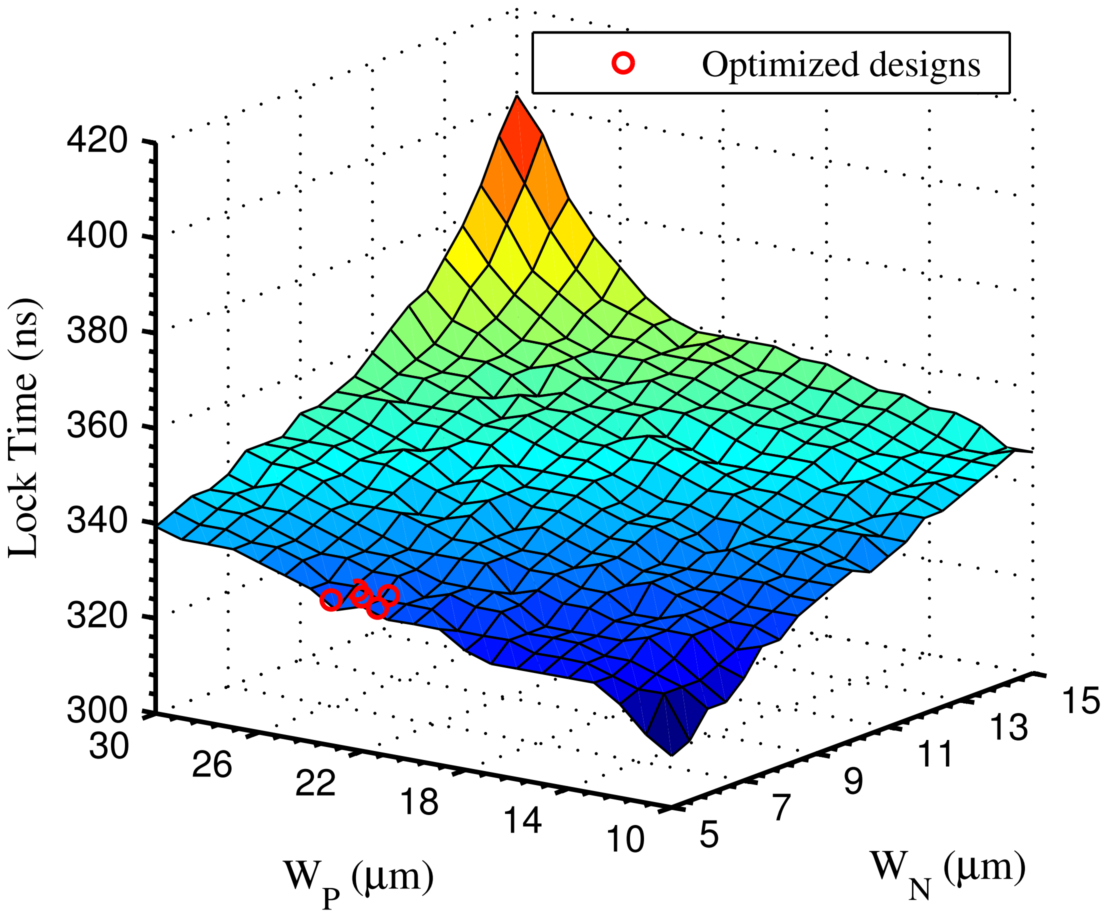


FIGURE 5.8. Sizing the transistors for Lock time.

In summary, a Verilog-AMS behavioral model based on quadratic polynomial metamodeling for a 180 nm LC VCO has been proposed. Other types of metamodels including artificial neural networks and splines are being investigated for integration in Verilog-AMS. With this behavioral model, fast and accurate PLL design verification and optimization has

TABLE 5.7. Comparison of The Baseline and The Optimized Designs

	Baseline	Optimal	Reduction
W_P/W_N (μm/μm)	20 / 10	21.4 / 5	–
Lock time (ns)	335.4	320.4	15.0
$\mathbf{P_{Locked}}$ (μW)	602	455	147
Tuning Range (MHz)	2170–2304	2173–2321	–

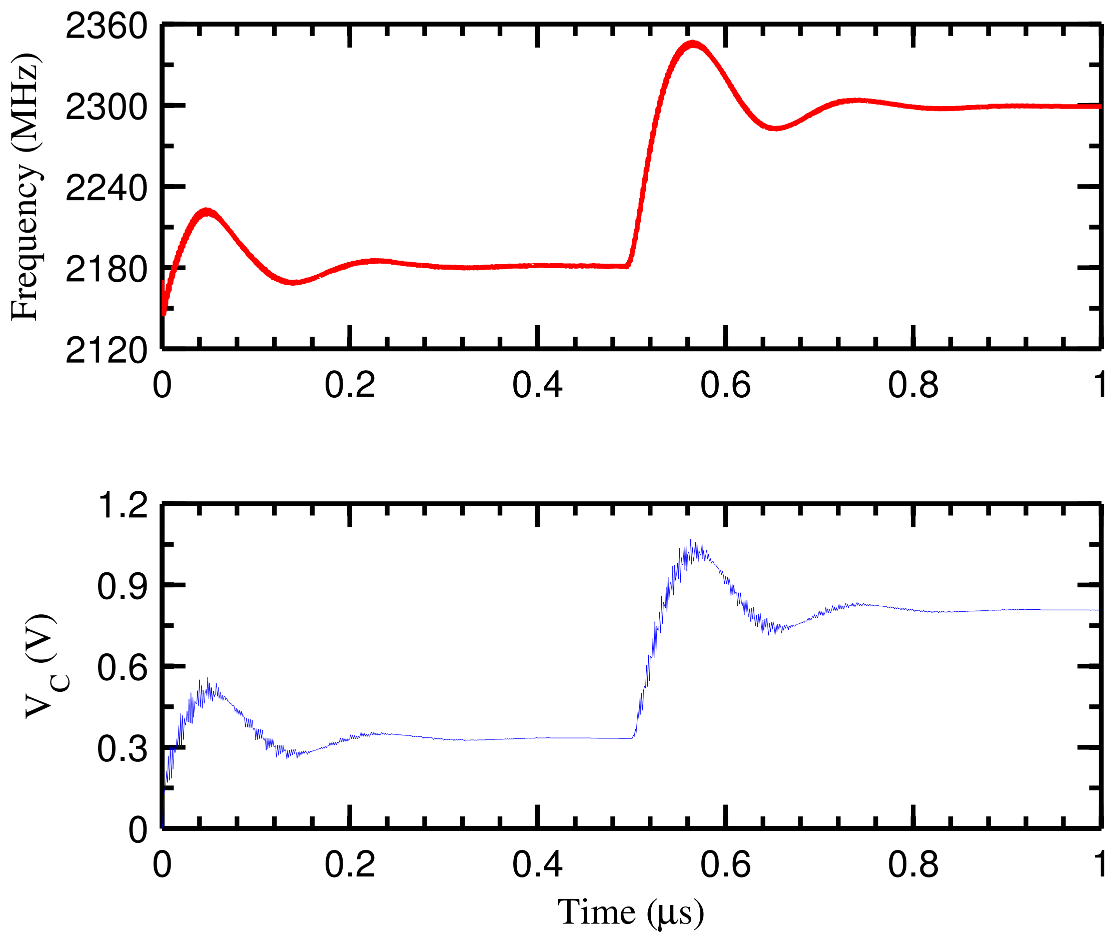


FIGURE 5.9. Simulation showing that the PLL first locked to 2180 MHz and then relocked to 2300 MHz.

been demonstrated. The behavioral model can be further improved for other characteristics such as phase noise, but is sufficient for lock time and average power estimation. Future research includes developing behavioral models incorporating the parasitics for the rest of the PLL building blocks and studying different metamodeling methods.

CHAPTER 6

CONCLUSIONS AND FUTURE RESEARCH

6.1. Summary

With multiple efforts to address the need of fast and accurate AMS system-level design space exploration, this dissertation starts with reviewing existing tools and modeling languages. The difficulty faced by existing system-level languages arises from the parasitic effects inherent in AMS physical designs and the lack of a viable way to seamlessly organize and communicate between different design abstraction levels. The proposed solution is based on metamodeling that has been reviewed in Chapter 2. Chapter 3 looks at the Simulink and Verilog-AMS frameworks as media for system-level modeling. The choice of the tool and the modeling language depends on the modeling object and time budget. Metamodeling techniques combined with the behavioral modeling language Verilog-AMS form the basis of this doctoral dissertation. Chapter 4 and Chapter 5 develop techniques that harness the power of metamodeling and hardware description languages. With these proposed techniques, the efficiency and accuracy of AMS design space exploration are significantly improved. Three circuits are used as case studies: delta-sigma modulator, OP-AMP, and Phase-Locked Loop (PLL) to examine the applicability and efficiency of the proposed methodologies for fast design exploration of analog/mixed-signal circuits and systems. The case study circuits and systems are carefully chosen to represent diverse classes of applications.

6.2. Conclusions

As shown in Chapter 3, the strength of Verilog-AMS is its compatibility with conventional SPICE circuit models and the ability to conveniently include low-level block non-idealities. In contrast, Simulink is more suitable for high-level exploration at early design phases. Simulink has the major advantage of a graphical user interface whereas the design exploration using Verilog-AMS is more textual. The method proposed in this dissertation is essentially a bottom-up approach that complements the existing top-down design approach to greatly improve the efficiency and accuracy of system-level design exploration. Hence Verilog-AMS is selected as the modeling language to leverage the proposed metamodels.

A major obstacle in mixed-signal circuits and systems design lies in the modeling of analog blocks with numerous performance specifications and the associated small error margin. Using a 90 nm CMOS OP-AMP as a case study, a three-step Verilog-AMS integrated polynomial metamodel assisted (Verilog-AMS-POM) analog block optimization approach has been proposed in Chapter 4. The polynomial metamodels model the OP-AMP characteristics and circuit parameters. They are then used to perform block-level optimization and construct parameterized macromodels for system-level design space exploration. A customized Cuckoo Search algorithm has been developed to optimize the case-study OP-AMP circuit. The generated OP-AMPS polynomial metamodels have low error rates and small absolute errors. The minimum and maximum mean error rates are 0.11 % and 2.86 %, respectively. The proposed polynomial metamodel assisted Cuckoo Search optimization algorithm is 17120× faster than the SPICE based optimization approach. The polynomial metamodel-assisted optimization achieves 3.86× OP-AMP power reduction. The accuracy values are comparable to that of the SPICE based optimization.

In order to account for the parasitic effects associated with AMS blocks, Verilog-AMS-PAM is proposed in Chapter 5 using a 180 nm CMOS PLL case study. Verilog/Verilog-AMS modules are created for the PLL building blocks to facilitate fast design simulation. Specifically, parasitic-aware metamodels are created for the LC VCO. Simulations validate that these models outperform the commonly used linear model and provide a much better approximation of the physical design behavior. The results show that the PLL lock time and power estimations based on the PAMs have low error rates of 0.7 % and 3.0 %, respectively. The PLL optimization using Verilog-AMS-PAMs achieves a 10× speedup.

6.3. Future Direction of the Proposed Research

The research presented in this dissertation integrated the metamodels in Verilog-AMS for fast mixed-signal design exploration. The idea can be used for pure analog as well as pure digital circuits and systems. Selected future possible directions for the proposed research are discussed in the following portion of this Section.

6.3.1. *Electronic Design Automation*

The proposed techniques can be implemented as a software package to accelerate the AMS modeling and optimization process. This package should include scripts or executables to intelligently parameterize AMS block netlists. A metamodel generator with user-specified error tolerance is needed to automate and to globally control the metamodel generation process. Thus the existing electronic design automation can be advanced by the proposed fast mixed-signal methodologies.

6.3.2. *Scalability*

The scalability of the proposed techniques can be improved in different ways. Firstly, different AMS blocks exhibit different characteristics and behaviors. Adjustments are possibly needed when applying the proposed techniques to other AMS blocks. By studying and evaluating the metamodels for those circuits, the reliability of the proposed techniques can be improved. Secondly, modern AMS systems can be extremely complex and therefore have to divided into multiple subsystems. While constructing metamodels for each subsystem by sampling its design space with SPICE simulation can be time consuming, the sampling can be done with Verilog-AMS metamodels. Once the hierarchical effectiveness is validated, the proposed techniques can be scaled to handle the AMS systems with ever-increasing complexity. What is the speed and accuracy tradeoffs when VHDL-AMS and SystemC-AMS are used needs research investigation.

6.3.3. *Variability*

Variability greatly impacts the chip yield in deep nanometer era. The semiconductor houses come up with ever smaller technologies. For example the 14nm technology node is rapidly becoming commodity. Effectively predicting and including system variability in early design phases can significantly help to avoid unnecessary costs. As semiconductor circuits and systems become more and more complex, the models highly relying on obtaining design structure knowledge will be harder to build and more expensive to evaluate. Metamodeling based techniques appear to be less vulnerable to the increasing circuit complexity. Thus modifying the proposed approach to include variability and statistical metamodeling should be a major future research direction.

BIBLIOGRAPHY

[1] O. Adamo, S.P. Mohanty, E. Kougianos, M. Varanasi, and W. Cai, *Vlsi architecture and fpga prototyping of a digital camera for image security and authentication*, Region 5 Conference, 2006 IEEE, IEEE, 2006, pp. 154–158. 3

[2] S. Ali, Li Ke, R. Wilcock, and P. Wilson, *Improved performance and variation modelling for hierarchical-based optimisation of analogue integrated circuits*, Proc. DATE '09. Design, Automation & Test in Europe Conf. & Exhibition, 2009, pp. 712–717. 13, 16, 66

[3] B. A. A. Antao and A. J. Brodersen, *Behavioral simulation for analog system design verification*, Proc. Custom Integrated Circuits Conf. the IEEE 1994, 1994, pp. 449–452. 1

[4] A. Ashry and H. Aboushady, *Fast and accurate jitter simulation technique for continuous-time $\Delta\Sigma$ modulators*, Electronics Letters 45 (2009), no. 24, 1218–1219. 31

[5] S. Baccar, T. Levi, D. Dallet, V. Shitikov, and F. Barbara, *A behavioral and temperature measurements-based modeling of an operational amplifier using VHDL-AMS*, Proc. 17th IEEE Int Electronics, Circuits, and Systems (ICECS) Conf, 2010, pp. 343–346. 16, 44

[6] S. Basu, B. Kommineni, and R. Vemuri, *Variation-aware macromodeling and synthesis of analog circuits using spline center and range method and dynamically reduced design space*, Proc. Int. Conf. VLSI Des., 2009, pp. 433–438. 8

[7] P. Benabes and C.-A. Tugui, *Effective modeling of CT functions for fast simulations using MATLAB-Simulink and VHDL-AMS applied to sigma-delta architectures*, Proc. IEEE Int Circuits and Systems (ISCAS) Symp, 2011, pp. 2269–2272. 16, 44

[8] D. Boolchandani, A. Ahmed, and V. Sahula, *Efficient kernel functions for support vector machine regression model for analog circuits performance evaluation*, Analog Integrated Circuits and Signal Processing 66 (2011), no. 1, 117–128. 8, 10

[9] C. Borchers, *Symbolic behavioral model generation of nonlinear analog circuits*, IEEE Trans. Circuits Syst. II 45 (1998), no. 10, 1362–1371. 4, 65

[10] G. R. Boyle, D. O. Pederson, B. M. Cohn, and J. E. Solomon, *Macromodeling of integrated circuit operational amplifiers*, IEEE J. Solid-State Circuits 9 (1974), no. 6, 353–364. 2

[11] S. Brigati, F. Francesconi, P. Malcovati, D. Tonietto, A. Baschirotto, and F. Maloberti, *Modeling sigma-delta modulator non-idealities in SIMULINK*, Proc. IEEE Int. Symp. Circuits and Systems ISCAS '99, vol. 2, 1999, pp. 384–387. 14

[12] P. K. Chan, K. A. Ng, and X. L. Zhang, *A CMOS chopper-stabilized differential difference amplifier for biomedical integrated circuits*, Proc. 47th Midwest Symp. Circuits and Systems MWSCAS '04, vol. 3, 2004. vii, 45, 46

[13] J.A. Cherry, *Theory, practice, and fundamental performance limits of high-speed data conversion using continuous-time delta-sigma modulators*, Ph.D. thesis, Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Doctor of Philosophy Ottawa-Carleton Institute for Electrical and Computer Engineering, Department of Electronics, Carleton University, 1998. 34, 35

[14] Philip M. Chopp and Anas A. Hamoui, *Analysis of clock-jitter effects in continuous-time $\Delta\Sigma$ modulators using discrete-time models*, Trans. Cir. Sys. Part I 56 (2009), no. 6, 1134–1145. 31

[15] C. T. Chuang, *Analysis of the settling behavior of an operational amplifier*, IEEE J. Solid-State Circuits 17 (1982), no. 1, 74–80. 44, 60, 61

[16] E. Clarke, A. Donzé, and A. Legay, *Statistical model checking of mixed-analog circuits with an application to a third order Δ-Σ modulator*, Hardware and Software: Verification and Testing (2009), 149–163. 15

[17] F. L. Cox, III, W. B. Kuhn, J. P. Murray, and S. D. Tynor, *Code-level modeling in XSPICE*, Proc. Symp. IEEE Int Circuits and Systems ISCAS '92, vol. 2, 1992, pp. 871–874. 2

[18] F. De Bernardinis, M. I. Jordan, and A. SangiovanniVincentelli, *Support vector machines for analog circuit performance representation*, Proc. Design Automation Conf, 2003, pp. 964–969. 8, 10

[19] G. Di Cataldo, R. Mita, G. Palumbo, and M. Pennisi, *Modeling of feedback analog circuits with VHDL*, Proc. 13th IEEE Int. Conf. Electronics, Circuits and Systems ICECS '06, 2006, pp. 882–885. 16, 44

[20] N. Duchaux, C. Lahuec, M. Arzel, and F. Seguin, *Analog decoder performance degradation due to BJTs' parasitic elements*, IEEE Trans. Circuits Syst. I 56 (2009), no. 11, 2402–2410. 14

[21] Zhuo Feng and Peng Li, *Performance-oriented statistical parameter reduction of parameterized systems via reduced rank regression*, Proc. IEEE/ACM Int. Conf. Computer-Aided Design ICCAD '06, 2006, pp. 868–875. 8, 10

[22] S. Franco, *Design with operational amplifiers and analog integrated circuits*, McGraw-Hill series in electrical and computer engineering, McGraw-Hill, 2002. vii, 45, 46

[23] O. Garitselov, S. P. Mohanty, and E. Kougianos, *Fast optimization of nano-CMOS mixed-signal circuits through accurate metamodeling*, Proc. 12th Int Quality Electronic Design (ISQED) Symp, 2011, pp. 1–6. 84

[24] O. Garitselov, S. P. Mohanty, and E. Kougianos, *A comparative study of metamodels for fast and accurate simulation of nano-CMOS circuits*, IEEE Transactions on Semiconductor Manufacturing (2012). 2, 4, 65, 72

[25] O. Garitselov, S. P. Mohanty, and E. Kougianos, *A comparative study of metamodels for fast and accurate simulation of nano-CMOS circuits*, IEEE Trans. Semicond. Manuf. 25 (2012), no. 1, 26–36. 8, 10

[26] O. Garitselov, S.P. Mohanty, and E. Kougianos, *Fast optimization of nano-cmos mixed-signal circuits through accurate metamodeling*, Quality Electronic Design (ISQED), 2011 12th International Symposium on, IEEE, 2011, pp. 1–6. 65, 72

[27] Oleg Garitselov, Saraju Mohanty, Elias Kougianos, and Geng Zheng, *Particle swarm optimization over non-polynomial metamodels for fast process variation resilient design*

of nano-CMOS PLL, Proceedings of the great lakes symposium on VLSI (New York, NY, USA), GLSVLSI '12, ACM, 2012, pp. 255–258. 8, 57

[28] Oleg Garitselov, Saraju P. Mohanty, and Elias Kougianos, *Accurate polynomial metamodeling-based ultra-fast bee colony optimization of a nano-cmos phase-locked loop*, J. Low Power Electronics 8 (2012), no. 3, 317–328. 4, 84

[29] Friedel Gerfers and Maurits Ortmanns, *Continuous-Time Sigma-Delta A/D conversion: Fundamentals, performance limits and robust implementations*, 1 ed., Springer, December 2005. 20, 27, 31

[30] I. E. Getreu, *Behavioral modeling of analog blocks using the saber simulator*, Proc. 32nd Midwest Symp. Circuits and Systems, 1989, pp. 977–980. 1, 2

[31] D. Ghai, S. P. Mohanty, and E. Kougianos, *Parasitic aware process variation tolerant voltage controlled oscillator (vco) design*, Quality Electronic Design, 2008. ISQED 2008. 9th International Symposium on, IEEE, 2008, pp. 330–333. 65

[32] ______, *Design of parasitic and process-variation aware nano-CMOS RF circuits: A VCO case study*, Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 17 (2009), no. 9, 1339–1342. 4, 15, 65

[33] G. G. E. Gielen and R. A. Rutenbar, *Computer-aided design of analog and mixed-signal integrated circuits*, Proceedings of the IEEE 88 (2000), no. 12, 1825–1854. 1

[34] Gnucap, http://www.gnu.org/software/gnucap/. 3

[35] G. J. Gomez, S. H. K. Embabi, E. Sanchez-Sinencio, and M. Lefebvre, *A nonlinear macromodel for CMOS OTAs*, Proc. IEEE Int Circuits and Systems ISCAS '95. Symp, vol. 2, 1995, pp. 920–923. vii, 41, 42

[36] D. Gorissen, L. De Tommasi, W. Hendrickx, J. Croon, and T. Dhaene, *RF circuit block modeling via Kriging surrogates*, Proc. 17th Int. Conf. Microwaves, Radar and Wireless Communications MIKON 2008, 2008, pp. 1–4. 8, 10

[37] B. Gu, K. K. Gullapalli, S. Hamm, B. Mulvaney, X. Lai, and J. Roychowdhury, *Implementing nonlinear oscillator macromodels using Verilog-AMS for accurate prediction*

of injection locking behaviors of oscillators, Proc. IEEE Int. Behavioral Modeling and Simulation Workshop BMAS 2005, 2005, pp. 43–47. 13

[38] A. Hajimiri and T. H. Lee, *A general theory of phase noise in electrical oscillators*, IEEE J. Solid-State Circuits 33 (1998), no. 2, 179–194. 35

[39] I. Harasymiv, M. Dietrich, and U. Knochel, *Fast mixed-mode PLL simulation using behavioral baseband models of voltage-controlled oscillators and frequency dividers*, Proc. XIth Int Symbolic and Numerical Methods, Modeling and Applications to Circuit Design (SM2ACD) Workshop, 2010, pp. 1–6. 13, 16, 66

[40] D. Hercog, M. Curkovic, and K. Jezernik, *DSP based rapid control prototyping systems for engineering education and research*, Proc. IEEE Computer Aided Control System Design IEEE Int. Conf. Control Applications IEEE Int. Symp. Intelligent Control, 2006, pp. 2292–2297. 14

[41] L. Hernandez, A. Wiesbauer, S. Paton, and A. Di Giandomencio, *Modelling and optimization of low pass continuous-time sigma delta modulators for clock jitter noise reduction*, Circuits and Systems, 2004. ISCAS '04. Proceedings of the 2004 International Symposium on, vol. 1, may 2004, pp. I – 1072–5 Vol.1. 31

[42] Mingta Hsieh and G. E. Sobelman, *Modeling and verification of high-speed wired links with Verilog-AMS*, Proc. IEEE Int. Symp. Circuits and Systems ISCAS 2006, 2006. 4, 65

[43] ChihDe Hung, WenShen Wuen, MeiFen Chou, and K.A. Wen, *A unified behavior model of low noise amplifier for system-level simulation*, Proc. 9th European Conf. Wireless Technology, 2006, pp. 139–142. 4, 65

[44] M. Isaksson, D. Wisell, and D. Ronnow, *Wide-band dynamic modeling of power amplifiers using radial-basis function neural networks*, IEEE Trans. Microw. Theory Tech. 53 (2005), no. 11, 3422–3428. 8

[45] Qi Jing, T. Riad, and SeeMei Chan, *Characterizing PLL jitter from power supply fluctuation using mixed-signal simulations*, Proc. 2nd Asia Symp. Quality Electronic Design (ASQED), 2010, pp. 112–117. 15, 66

[46] R. Kaald, I. Lokken, B. Hernes, and T. Saether, *High-level continuous-time sigma delta design in MATLAB/Simulink*, Proc. NORCHIP, 2009, pp. 1–6. 38

[47] H. Kabir, Ying Wang, Ming Yu, and Qi-Jun Zhang, *High-dimensional neural-network technique and applications to microwave filter modeling*, IEEE Trans. Microw. Theory Tech. 58 (2010), no. 1, 145–156. 8

[48] T. J. Kazmierski, Dafeng Zhou, B. M. Al-Hashimi, and P. Ashburn, *Numerically efficient modeling of CNT transistors with ballistic and nonballistic effects for circuit simulation*, IEEE Trans. Nanotechnol. 9 (2010), no. 1, 99–107. 8, 10

[49] T. Kiely and G. Gielen, *Performance modeling of analog integrated circuits using least-squares support vector machines*, Proc. Design, Automation and Test in Europe Conf. and Exhibition, vol. 1, 2004, pp. 448–453. 8, 10

[50] E. Kougianos, S.P. Mohanty, and D.K. Pradhan, *Simulink based architecture prototyping of compressed domain mpeg-4 watermarking*, Proceedings of the 12th IEEE International Conference on Information Technology (ICIT) (2009), 10–16. 3

[51] Ken Kundert and Olaf Zinke, *The designer's guide to Verilog-AMS*, 1st ed., Kluwer Academic Publishers, Boston, May 2004. 67

[52] K.S. Kundert and P. Foreword By-Gray, *The designer's guide to SPICE and SPECTRE*, Kluwer Academic Publishers, 1995. 38

[53] ChinCheng Kuo, *An efficient approach to build accurate behavioral models of PLL designs*, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences E89-A (2006), no. 2, 391–398. 15, 66

[54] ChinCheng Kuo, MengJung Lee, ChienNan Liu, and ChingJi Huang, *Fast statistical analysis of process variation effects using accurate PLL behavioral models*, IEEE Trans. Circuits Syst. I 56 (2009), no. 6, 1160–1172. 15, 66

[55] E. Lauwers, K. Lampaert, P. Miliozzi, and G. Gielen, *High-level design case of a switched-capacitor low-pass filter using Verilog-A*, Proc. IEEE/ACM Int Behavioral Modeling and Simulation Workshop, 2000, pp. 16–21. 16, 44

[56] D. B. Leeson, *A simple model of feedback oscillator noise spectrum*, Proc. IEEE 54 (1966), no. 2, 329–330. 35

[57] X. Li, P. Gopalakrishnan, Y. Xu, and L. T. Pileggi, *Robust Analog/RF circuit design with projection-based performance modeling*, IEEE Trans. Comput.-Aided Design Integr. Circuits Syst. 26 (2007), no. 1, 2–15. 8, 10

[58] Xin Li and Hongzhou Liu, *Statistical regression for efficient high-dimensional modeling of analog and mixed-signal performance variations*, Proc. 45th ACM/IEEE Design Automation Conf. DAC 2008, 2008, pp. 38–43. 8, 10

[59] Wei-Hsin Liao, Shun-Chung Wang, and Yi-Hua Liu, *Generalized simulation model for a switched-mode power supply design course using MATLAB/Simulink*, IEEE Trans. Educ. 55 (2012), no. 1, 36–47. 14

[60] E. Liu, G. Gielen, H. Chang, A. L. Sangiovanni-Vincentelli, and P. R. Gray, *Behavioral modeling and simulation of data converters*, Proc. Symp. IEEE Int Circuits and Systems ISCAS '92, vol. 5, 1992, pp. 2144–2147. 1

[61] H. A. Mantooth and P. E. Allen, *Behavioral simulation of a 3-bit flash ADC*, Proc. IEEE Int Circuits and Systems Symp, 1990, pp. 1356–1359. 1

[62] MATLAB, *Simulink® user's guide (R2011b)*, The MathWorks Inc., Natick, Massachusetts, 2011. 38

[63] MATLAB/Simscape, http://www.mathworks.com/products/simscape/. 3

[64] T. McConaghy and G. Gielen, *Analysis of simulation-driven numerical performance modeling techniques for application to analog circuit optimization*, Proc. IEEE Int. Symp. Circuits and Systems ISCAS 2005, 2005, pp. 1298–1301. vi, 9, 11

[65] S. P. Mohanty, *A secure digital camera architecture for integrated real-time digital rights management*, Journal of Systems Architecture 55 (2009), no. 10, 468–480. 3

[66] S.P. Mohanty and E. Kougianos, *Simultaneous power fluctuation and average power minimization during nano-cmos behavioral synthesis*, Proceedings of the 20th International Conference on VLSI Design held jointly with 6th International Conference: Embedded Systems (2007), 577–582. 1

[67] S.P. Mohanty and N. Ranganathan, *Simultaneous peak and average power minimization during datapath scheduling*, Circuits and Systems I: Regular Papers, IEEE Transactions on 52 (2005), no. 6, 1157–1165. 83

[68] S.P. Mohanty, N. Ranganathan, and S.K. Chappidi, *Peak power minimization through datapath scheduling*, VLSI, 2003. Proceedings. IEEE Computer Society Annual Symposium on, IEEE, 2003, pp. 121–126. 83

[69] ______, *ILP models for energy and transient power minimization during behavioral synthesis*, VLSI Design, 2004. Proceedings. 17th International Conference on, IEEE, 2004, pp. 745–748. 83

[70] A. Morgado, V.J. Rivas, R. del Ro, R. Castro-Lpez, F.V. Fernndez, and J.M. de la Rosa, *Behavioral modeling, simulation and synthesis of multi-standard wireless receivers in MATLAB/SIMULINK*, Integration, the VLSI Journal 41 (2008), no. 2, 269–280. 14

[71] V. Mukherjee, S.P. Mohanty, and E. Kougianos, *A dual dielectric approach for performance aware gate tunneling reduction in combinational circuits*, Computer Design: VLSI in Computers and Processors, 2005. ICCD 2005. Proceedings. 2005 IEEE International Conference on, IEEE, 2005, pp. 431–436. 83

[72] Steven R. Norsworthy, Richard Schreier, and Gabor C. Temes (eds.), *Delta-sigma data converters: Theory, design, and simulation*, 1 ed., Wiley-IEEE Press, October 1996. 22

[73] O. Okobiah, S. P. Mohanty, and E. Kougianos, *Ordinary Kriging metamodel-assisted ant colony algorithm for fast analog design optimization*, Proc. 13th Int Quality Electronic Design (ISQED) Symp, 2012, pp. 458–463. 84

[74] O. Okobiah, S. P. Mohanty, E. Kougianos, and O. Garitselov, *Kriging-assisted ultra-fast simulated-annealing optimization of a clamped bitline sense amplifier*, Proc. 25th Int VLSI Design (VLSID) Conf, 2012, pp. 310–315. 8, 10

[75] O. Okobiah, S. P. Mohanty, E. Kougianos, O. Garitselov, and Geng Zheng, *Stochastic gradient descent optimization for low power nano-CMOS thermal sensor design*, Proc. IEEE Computer Society Annual Symp. VLSI (ISVLSI), 2012, pp. 285–290. 8

[76] S. Pam, A. K. Bhattacharya, and S. Mukhopadhyay, *An efficient method for bottom-up extraction of analog behavioral model parameters*, Proc. 23rd Int. Conf. VLSI Design VLSID '10, 2010, pp. 363–368. 4, 65

[77] S. Pandit, C. Mandal, and A. Patra, *Systematic methodology for high-level performance modeling of analog systems*, Proc. 22nd Int VLSI Design Conf, 2009, pp. 361–366. 8, 10

[78] S. Pavan, *Systematic design centering of continuous time oversampling converters*, IEEE Trans. Circuits Syst. II 57 (2010), no. 3, 158–162. 26

[79] Behzad Razavi, *Monolithic Phase-Locked loops and clock recovery circuits: Theory and design*, 1 ed., Wiley-IEEE Press, April 1996. 66

[80] J. Ruiz-Amaya, J. M. de la Rosa, F. V. Fernandez, F. Medeiro, R. del Rio, B. Perez-Verdu, and A. Rodriguez-Vazquez, *High-level synthesis of switched-capacitor, switched-current and continuous-time $\Sigma\Delta$ modulators using SIMULINK-based time-domain behavioral models*, IEEE Trans. Circuits Syst. I 52 (2005), no. 9, 1795–1810. 14

[81] Rob A. Rutenbar, Georges G. E. Gielen, and Brian A. Antao, *Computer-aided design of analog integrated circuits and systems*, John Wiley & Sons, Inc., New York, NY, USA, 2002. 16, 44

[82] K. K. Sabet and T. Riad, *A generic vhdl-ams behavioral model physically accounting for typical analog non-linear output behavior*, Proc. IEEE Int. Behavioral Modeling and Simulation Workshop BMAS 2007, 2007, pp. 105–109. 16, 44

[83] V. Saripalli, A. Mishra, S. Datta, and V. Narayanan, *An energy-efficient heterogeneous CMP based on hybrid TFET-CMOS cores*, Proc. 48th ACM/EDAC/IEEE Design Automation Conf. (DAC), 2011, pp. 729–734. 12

[84] Richard Schreier and Gabor C. Temes, *Understanding delta-sigma data converters*, 1 ed., Wiley-IEEE Press, November 2004. 22, 25, 29

[85] P. S. Shiakolas and D. Piyabongkarn, *Development of a real-time digital control system with a hardware-in-the-loop magnetic levitation device for reinforcement of controls education*, IEEE Trans. Educ. 46 (2003), no. 1, 79–87. 14

[86] F. Silveira, D. Flandre, and P. G. A. Jespers, *A gm/ID based methodology for the design of cmos analog circuits and its application to the synthesis of a silicon-on-insulator micropower OTA*, IEEE J. Solid-State Circuits 31 (1996), no. 9, 1314–1319. 47

[87] NGSPICE Simulator, http://ngspice.sourceforge.net/index.html. 3

[88] SPICE Simulator, http://cutler.eecs.berkeley.edu/classes/icbook/spice/. 2

[89] H.L. Tsai and J.M. Lin, *Model building and simulation of thermoelectric module using Matlab/Simulink*, Journal of Electronic Materials 39 (2010), no. 9, 2105–2111. 14

[90] P. N. Variyam, S. Cherubal, and A. Chatterjee, *Prediction of analog performance parameters using fast transient testing*, IEEE Trans. Comput.-Aided Design Integr. Circuits Syst. 21 (2002), no. 3, 349–361. 8, 10

[91] R. Vemuri and G. Wolfe, *Adaptive sampling and modeling of analog circuit performance parameters with pseudo-cubic splines*, Proc. ICCAD-2004 Computer Aided Design IEEE/ACM Int. Conf, 2004, pp. 931–938. 8, 10

[92] Bo Wan and C.-J. R. Shi, *Hierarchical multi-dimensional table lookup for model compiler based circuit simulation*, Proc. Design, Automation and Test in Europe Conf. and Exhibition, vol. 2, 2004, pp. 1310–1315. 13

[93] Yi Wang, Yikai Wang, and Lenian He, *Behavioral modeling for operational amplifier in sigma-delta modulators with verilog-a*, Proc. IEEE Asia Pacific Conf. Circuits and Systems APCCAS 2008, 2008, pp. 1612–1615. 16, 44

[94] Yifan Wang, C. Van-Meersbergen, H.-W. Groh, and S. Heinen, *Event driven analog modeling for the verification of PLL frequency synthesizers*, Proc. IEEE Behavioral Modeling and Simulation Workshop BMAS 2009, 2009, pp. 25–30. 16, 66

[95] Ying Wei and A. Doboli, *Systematic development of analog circuit structural macromodels through behavioral model decoupling*, Proc. 42nd Design Automation Conf, 2005, pp. 57–62. 16, 41

[96] G. Wolfe and R. Vemuri, *Extraction and use of neural network models in automated synthesis of operational amplifiers*, IEEE Trans. Comput.-Aided Design Integr. Circuits Syst. 22 (2003), no. 2, 198–212. 8

[97] J. Wood, D. E. Root, and N. B. Tufillaro, *A behavioral modeling approach to nonlinear model-order reduction for RF/microwave ICs and systems*, IEEE Trans. Microw. Theory Tech. 52 (2004), no. 9, 2274–2284. 4, 65

[98] Zhao Xueqian, Zhao Zhenyu, Zhang Minxuan, and Li Shaoqing, *Verilog-A based implementation for coupled model of single event transients in look-up table technique*, Proc. IEEE 8th Int. Conf. ASIC ASICON '09, 2009, pp. 666–669. 12

[99] Xin-She Yang and Suash Deb, *Engineering optimisation by Cuckoo Search*, International Journal of Mathematical Modelling and Numerical Optimisation 1 (2010), no. 4, 330–343. 57

[100] R. F. Yazicioglu, Sunyoung Kim, T. Torfs, Hyejung Kim, and C. Van Hoof, *A 30 μW analog signal processor asic for portable biopotential signal monitoring*, IEEE J. Solid-State Circuits 46 (2011), no. 1, 209–223. 22

[101] Hailong You, Maofeng Yang, Dan Wang, and Xinzhang Jia, *Kriging model combined with latin hypercube sampling for surrogate modeling of analog integrated circuit performance*, Proc. Quality Electronic Design Quality of Electronic Design ISQED 2009, 2009, pp. 554–558. 8, 10

[102] Guo Yu and Peng Li, *Yield-aware analog integrated circuit optimization using geostatistics motivated performance modeling*, Proc. IEEE/ACM Int. Conf. Computer-Aided Design ICCAD 2007, 2007, pp. 464–469. 8, 10

[103] D. Zaum, S. Hoelldampf, M. Olbrich, E. Barke, and I. Neumann, *Systemc mixed-signal and mixed-level simulation using an accelerated analog simulation approach*, Symbolic

and Numerical Methods, Modeling and Applications to Circuit Design (SM2ACD), 2010 XIth International Workshop on, oct. 2010, pp. 1–4. 1

[104] He Zhang and Guoyong Shi, *Symbolic behavioral modeling for slew and settling analysis of operational amplifiers*, Proc. IEEE 54th Int Circuits and Systems (MWSCAS) Midwest Symp, 2011, pp. 1–4. 16, 44, 62

[105] G. Zheng, S. P. Mohanty, E. Kougianos, and O. Garitselov, *Verilog-AMS-PAM: Verilog-AMS integrated with Parasitic-Aware Metamodels for Ultra-Fast and Layout-Accurate Mixed-Signal Design Exploration*, Proceedings of the 21st ACM/IEEE Great Lakes Symposium on VLSI (GLSVLSI), 2012. 65

[106] Geng Zheng, Saraju P. Mohanty, and Elias Kougianos, *Design and modeling of a continuous-time delta-sigma modulator for biopotential signal acquisition: Simulink vs. Verilog-AMS perspective*, Proc. Third Int Computing Communication & Networking Technologies (ICCCNT) Conf, 2012, pp. 1–6. 18

[107] Geng Zheng, Saraju P. Mohanty, and Elias Kougianos, *Metamodel-assisted fast and accurate optimization of an op-amp for biomedical applications*, Proc. IEEE Computer Society Annual Symp. VLSI (ISVLSI), 2012, pp. 273–278. 40

[108] J. Zou, D. Mueller, H. Graeb, U. Schlichtmann, E. Hennig, and R. Sommer, *Fast automatic sizing of a charge pump phase-locked loop based on behavioral models*, Proc. IEEE Int. Behavioral Modeling and Simulation Workshop BMAS 2005, 2005, pp. 100–105. 15, 66

[109] Jun Zou, D. Mueller, H. Graeb, and U. Schlichtmann, *A CPPLL hierarchical optimization methodology considering jitter, power and locking time*, Proc. 43rd ACM/IEEE Design Automation Conf, 2006, pp. 19–24. 13

[110] Xiaodan Zou, Xiaoyuan Xu, Libin Yao, and Yong Lian, *A 1-V 450-nW fully integrated programmable biomedical sensor interface chip*, IEEE J. Solid-State Circuits 44 (2009), no. 4, 1067–1077. 22