Design Methodology for High-performance Circuits Based on Automatic Optimization Methods.

Catalin Adrian Tugui

Design Methodology for High-performance Circuits Based on Automatic Optimization Methods

CATALIN-ADRIAN TUGUI

Department of Signal Processing & Electronic Systems (SSE)

École supérieure d'électricité (SUPELEC), Gif-sur-Yvette

PhD Committee:

Prof. Souhil MEGHERBI	Member
Prof. Patrick LOUMEAU	Reviewer
Prof. Iulian NASTAC	Reviewer
Prof. Philippe BENABES	Supervisor
Prof. Filipe VINCI dos SANTOS	Member
Prof. Alexandre REINHARDT	Member

A thesis submitted for the degree of

PhilosophiæDoctor (PhD)

2013 January 14

1. Supervisor: Philippe BENABES, professor (Dr.-Ing.), SSE department, SUPELEC, France. Email: Philippe.Benabes@supelec.fr

2. Reviewer: Patrick LOUMEAU, professor (Dr.-Ing.), Telecom department, TELECOM ParisTech, France. Email: Patrick.Loumeau@telecom-paristech.fr

3. Reviewer: Dumitru Iulian NASTAC, Associate professor (Dr.-Ing), University Politehnica Bucharest, Romania. Email: nastac@ieee.org

Day of the defense:

Signature from head of PhD committee:

To my parents,

Ioan** and **Ecaterina

and my little brother,

Dimitrie

Abstract

The heterogeneity of novel electronic systems implies the evolution towards nanometer-scale CMOS processes along with the increasing development of systems-on-a-chip (SoCs) and systems-in-a-package (SiPs) including mixed analog and digital circuitry combined with micro-electro-mechanical systems (MEMS).

The conception of these systems requires a very difficult design effort, one of the main drawbacks residing in a gap of the current analog design methodologies, which cannot assure an optimal, time-effective system level solution when the circuits are implemented at technological level.

De facto, the complex analog parts should be recursively optimized based not only on system-level requirements but also on process limitations and imperfections. Furthermore, high-level behavioral models used for chip-level simulations, which can be re-adjusted to changes at the transistor level, should be employed.

In this context, the aim of this thesis is to establish an efficient analog design methodology, the algorithms and the corresponding design tools which can be employed in the dynamic conception of weakly non-linear continuous-time (CT) functions, assuring that the performance figures for a complete system can be rapidly investigated, but with comparable accuracy to the low-level evaluations. This methodology can be easily interfaced with the existing digital, mixed-signal and multi-domain conception paradigms, for a complete system characterization.

In a first part of this work, we present the novel design methodology based on the automatic optimization process of transistor-level cells using a modified Bayesian Kriging approach and the synthesis of robust high-level analog behavioral models in environments like Mathworks – Simulink, VHDL-AMS or Verilog-A.

The macro-model extraction process involves a complete set of analyses (DC, AC, transient, parametric, Harmonic Balance) which are performed on the analog schematics implemented on a specific technology process. Then, the extraction and calculus of a multitude of figures of merit assures that the models include the low-level characteristics and can be directly regenerated during the optimization process.

The optimization algorithm uses a Bayesian method, where the evaluation space is created by the means of a Kriging surrogate model, and the selection is effectuated by using the expected improvement (EI) criterion subject to constraints. The objectives are considered along

with 2 types of constraints: strong constraints (nonlinearities, distortions) and normal constraints (offsets, impedances, gains, bandwidths), which are adequate for analog characterization, ensuring robust optimal solutions on a limited number of evaluations.

A conception tool was developed (SIMECT), which was integrated as a Matlab toolbox, including all the macro-models extraction and automatic optimization techniques.

The developed methodology and tools were applied to the conception and optimization of two CT Sigma-Delta modulator architectures using micro-mechanical resonators of type Lamb Wave Resonator (LWR). Realistic design conditions based on a 350nm AMS technology process and measured responses for the LWRs provided by CEA-LETI allowed precisely characterizing the feasibility and the performances of the resulted Sigma-Delta structures.

Acknowledgements

I would like to take this opportunity to remember all those who gave me the possibility to develop, to learn, to know myself better and finally to make life a memorable and unforgettable experience during this PhD thesis and beyond. I will try to express all my gratitude to those persons, even though words are not enough to do this.

First of all, I would like to gratefully thank my PhD supervisor, Prof. Philippe Bénabès. Thank you for your entire support, for the professional work, for your intelligent guidance, valuable comments and for your availability when I needed it most. Without your continuous support and dedication, this achievement would not have existed.

I would like to thank Mr. Stéphane Font, Head of the Department of Signal Processing & Electronic Systems and Mr. Gilles Fleury, Director of Research & Industry Partnerships SUPÉLEC and former Head of the department for their warm welcome in the institution and their support during this thesis.

I am particularly grateful to Prof. Patrick Loumeau from Télécom ParisTech and Prof. Dumitru Iulian Năstac from the University Politehnica of Bucharest for reporting on this thesis. I would like to thank also Prof. Souhil Megherbi from IUT de Cachan, Prof. Filipe Vinci dos Santos form the Thales-SUPÉLEC Chair and Prof. Alexandre Reinhardt from CEA-Leti for the honor of having them as members of my PhD Committee.

My thanks also go to all the former and present professors and staff of the Department of Signal Processing & Electronic Systems and particularly to Prof. Jacques Oksman, former Director of Research & Industry Partnerships SUPÉLEC for accepting the initial directorship of this PhD, Prof. Richard Kielbasa for his encouragement in the beginning of this thesis, Prof. Jerome Juillard for his valuable comments, Prof. Caroline Lelandais-Perrault and Prof. Émilie Avignon-Meseldzija for their interest and support.

I want furthermore to thank the technical staff of the department, Mrs. Karine Bernard and Mrs. Alexandra Siebert for their help and support in all the situations, Mr. Francis Trelin for his unforgettable kindness and advice, Mr. Luc Batalie for his entire help and availability, Mr. Huu Hung Vuong for his advice and delicious recipes and Mr. Julien Blancard for helping me with technical problems.

I am deeply indebted and I want to take this opportunity to thank my previous professors and teachers from Romania, especially Prof. Teodor Petrescu, Prof. Teofil Teodorescu, Prof. Carmen Andronachi and Prof. Mircea Goreac.

I express my gratitude to my PhD collegues and trainees in the Department of Signal Processing & Electronic Systems who supported me in my research work and maintained a pleasant working atmosphere. Especially I am obliged to Zhiguo Song, Jean-Michel Akré, Antoine Canu, Kian Jafari, Romain Benassi, Jingwen Wu and Ștefan Apostol.

I wish to express my sincere thanks to all of my friends here in France, in Romania and everywhere and particularly to Ionuț Tolea, Mihai Ilie, Silviu Doagă, Ionela Prodan, Florin Stoican, Bogdan and Raluca Liacu, Marius and Ioana Olteanu, Ștefan Teodorescu, Dorin Panaitopol, Valentin Tanasă, Emanuel Dogaru, Cristina Vlad, Iolanda Saviuc, Gabriel Donici, Cornelia Chițan, Hammou Merzouk, Fanqi Meng, Charles and Caroline Cavaille. I want to thank them for all their help, support, interest and motivation throughout the years.

My most grateful thanks are reserved for my parents, Ioan and Ecaterina, for my brother Dimitrie, my grandparents and my relatives: Thank you so much for your patience, encouragement, love and support, for believing in me and having made all of this possible!

My special thanks go to my girlfriend, Raisa Țăruș: Thank you for all your love, for your trust, encouragement and for being close to me even if we are far away!

Finally, I would like to dedicate my work also in the memory of our grandparents and ancestors who were denied freedom, suffered and died in prisons, deportations, forced labor camps or other conditions during the past communist regime in Eastern Europe. Their sacrifice made possible our today freedom.

Thank You, God for all the blessings in my life!

Glossary

ADC	Analog-to-digital converter
AMS	Analog and Mixed-signal
ASIC	Application-specific Integrated Circuit
ASSP	Application-specific Standard Part
BAW	Bulk Acoustic Wave
BiCMOS	Bipolar and CMOS
BST	Barium Strontium Titanate
CAD	Computer Aided Design
CCCS	Current-controlled-current-source
CCII	Second-generation Current Conveyer
CCVS	Current-controlled-voltage-source
CMOS	Complementary Metal Oxide Semiconductor
CT	Continuous-time
DAC	Digital-to-analog converter
DAE	Differential Algebraic Equation
DSP	Digital Signal Processing
DT	Discrete Time
EDA	Electronic Design Automation
EGO	Efficient Global Optimization
EI	Expected Improvement
FBAR	Film Bulk Acoustic Resonator
Gm	Trans-conductance Amplifier
HB	Harmonic Balance
HDL	Hardware Description Language
LHS	Latin Hypercube Sampling
LPGP	Low-Power General-Purpose
LSB	Least Significant Bit
LWR	Lamb Wave Resonator
MEMS	Micro-electro-mechanical systems
MIMO	Multiple-input-multiple-output
MNA	Modified Nodal Analysis
MSE	Mean Squared Error
NTF	Noise Transfer Function
OSR	Over Sampling Ratio
PAC	Periodic AC

PDE	Partial Differential Equation
PLL	Phase-locked Loop
PSD	Power Spectral Density
PSS	Periodic Steady State
RF	Radio Frequency
RTL	Register Transfer Level
SAW	Surface Acoustic Wave
SIMECT	SImulation and Macro-model extraction for CT functions
SiP	System-in-a-package
SMR	Solidly Mounted Resonator
SNR	Signal-to-noise Ratio
SoC	System-on-a-chip
STF	Signal Transfer Function
TIA	Trans-impedance Amplifier
VCCS	Voltage-controlled-current-source
VCVS	Voltage-controlled-voltage-source
VHDL-AMS	Very-high-speed Integrated Circuits Hardware Description Language with Analog and Mixed-Signal Extensions
ΣΔ	Sigma-Delta

Contents

List of Figures

List of Tables

Résumé

Motivations

Le développement de systèmes électroniques actuels va dans deux directions principales : l'une conduit à une complexité croissante, l'autre conduit à une variabilité fonctionnelle importante.

Le premier développement, suivant la fameuse loi de Moore, consiste à migrer vers des géométries de processus technologiques de plus en plus petits, nous nous attendons donc aujourd'hui à voir plus de 12 milliards de transistors par puce lorsqu'on utilise une technologie 20nm standard [1]. Le nombre de transistors a déjà atteint 7,1 milliards dans les unités de traitement graphique (GPU) disponibles dans le commerce [2] et 2,6 milliards dans les microprocesseurs à usage général [3].

Le deuxième développement tente d'intégrer un nombre de plus en plus important de fonctions diverses et complexes avec des performances très exigeantes. Les applications spécifiques à circuits intégrés (ASIC) (nombre de transistors > 200 millions / puce), et les applications spécifiques à des pièces standard (ASSP) (nombre de transistors > 100 millions / puce) ont tendance à être remplacées par des systèmes complètement intégrés sur puce (SoC) et des systèmes en boîtier (SiP) contenant des parties analogiques, numériques et radiofréquence (RF) avec en plus des capteurs et des actionneurs, conçus sur la base de nouveaux paradigmes de réutilisabilité. En pratique, 70 à 80% des circuits intégrés actuels sont des systèmes-sur-puce [4], allant des applications ciblées (par exemple la télévision sur une seule puce ou la caméra vidéo sur une seule puce [5]) aux nouveaux microprocesseurs et systèmes intégrés de télécommunications.

Un tel progrès permet d'obtenir un degré de complexité jamais vu auparavant dans un système artificiel (mécanique, électrique, thermique, architecture, etc.) et ne peut être comparée qu'avec les systèmes biologiques (par exemple, le cerveau humain reste pour l'instant plus complexe, avec 80 -100 milliards de neurones et un degré très élevé d'inter-connectivité entre les cellules [6]).

Un problème important en ce qui concerne la conception des nouveaux systèmes électroniques réside dans un décalage toujours plus important entre les méthodes de conception actuelles et la mise à l'échelle, à quoi s'ajoutent les exigences fonctionnelles des circuits [7]. Et avec l'avènement d'applications mixtes, avec des parties analogiques liées entre elles et

contrôlées par les parties numériques et les blocs numériques de traitement du signal (DSP), le problème devient encore plus complexe.

90% de toutes les applications SoC contient des circuits analogiques, et le contenu analogique de ces systèmes occupe environ 20% de la surface du circuit. Par exemple, un SoC multi-noyau peut incorporer plus de 30 systèmes analogiques complexes et indépendants [8] (des PLL multi-domaines d'horloge, la gestion thermique et de puissance, les interfaces d'entrées-sorties à haute vitesse, les générateurs d'horloge à faible gigue).

Les parties analogiques sont mises en œuvre dans des procédés CMOS ou BiCMOS et sont relativement complexes, avec des centaines et souvent des milliers de signaux de commande numériques [9]. Malgré la tendance à remplacer les circuits analogiques par des fonctions numériques (par exemple, le traitement du signal numérique à la place du filtrage analogique), il y a quelques fonctions typiques qui restent toujours analogiques, celles qui sont nécessaires à l'interface entre le système électronique et le monde « réel » [10] (par exemple en entrée : amplificateurs à faible bruit, amplificateurs à gain variable, filtres, des oscillateurs, etc., afin de traiter les signaux provenant des capteurs, des microphones, antennes ; en sortie: les pilotes, les filtres pour interfacer à l'extérieur des charges ; des convertisseurs analogique-numérique et numérique-analogique. Enfin, les plus grands circuits analogiques sont aujourd'hui les circuits hautes performances numériques - les microprocesseurs, qui sont en grande partie conçus comme les circuits analogiques, afin de repousser les limites de vitesse ou de puissance).

Sans une méthodologie bien établie pour la conception et la vérification, la complexité de ces circuits se traduit par un nombre croissant d'erreurs fonctionnelles, pouvant conduire à des défaillances et de multiples re-conceptions jusqu'à l'obtention d'un système fonctionnel.

Malheureusement, lorsqu'on les compare aux méthodes de conception numériques - déjà automatisés, efficaces et maturées en industrie - les méthodes analogiques n'ont pas évolué beaucoup depuis les années 80, se basant sur des techniques de simulation SPICE et de dessin de masque dans des environnements avec quelques outils d'accompagnement (par exemple, des capacités d'optimisation limitées dans le simulateur, ou des outils de vérification des masques).

Ainsi, le cycle de conception des circuits intégrés analogiques et à signaux mixtes reste long et sujet aux erreurs. La conception des circuits analogiques est souvent le goulot d'étranglement dans les systèmes à signaux mixtes, à la fois dans le temps et l'effort de conception ainsi que le coût des tests, et ils sont souvent responsables d'erreurs de conception et des re-conceptions coûteuses [7].

Dans ce contexte, il y a un fort besoin de méthodes efficaces de conception analogique qui peuvent être corrélées avec les techniques hautes performances existantes pour le numérique et potentiellement avec d'autres méthodes multi-domaines (par exemple les méthodologies de conception de systèmes micro-électro-mécaniques - MEMS - déjà intégrable avec les CMOS)

afin d'obtenir un modèle de conception généralement structuré pour ces systèmes très complexes.

On a proposé plusieurs méthodologies de conception analogique employant des approches descendantes et des techniques de modélisation comportementales, lesquels sont fondés sur des langages de description de matériel (HDL). Cependant, le plus grand problème reste le faible nombre de méthodes systématiques nécessaire à la modélisation du comportement analogique pouvant être exploité au niveau système. On remarque en plus le manque d'outils efficaces pour automatiser ces méthodes et le processus d'extraction de modèles [11].

En conclusion, la mise en place de méthodologies de conception analogique et des outils correspondants d'automatisation et de conception électronique demeure toujours un domaine de recherche prolifique, qui répond aux besoins croissants de l'industrie des semi-conducteurs et, en définitive, des marchés de la technologie au sens large.

Objectifs et travaux de thèse

On identifie des classes différentes de fonctions analogiques : les fonctions radiofréquence (RF), qui sont étudiées à partir du formalisme des paramètres en *S*, les fonctions analogiques basse fréquence qui peuvent être des fonctions non linéaires (par exemple les interrupteurs, les comparateurs, les redresseurs, etc.) et des fonctions linéaires (par exemple les différentes catégories d'amplificateurs et de filtres linéaires, etc.).

Dans ce contexte, notre travail de thèse se situe dans la classe des fonctions analogiques basse fréquence linéaires, c'est à dire le traitement de fonctions d'amplification, des résonateurs et des filtres linéaires pour lesquels une linéarité importante est exigée tout au long du processus de conception.

Le premier objectif de cette thèse vise à établir une méthodologie de conception analogique efficace, de détailler ses méthodes, algorithmes et de mettre en œuvre ses outils correspondants pour la conception assiste par ordinateur (CAO), qui peuvent être utilisés pour la conception complexe de fonctions linéaires à temps continu (CT).

Un deuxième objectif est de valider la méthodologie proposée sur des structures existantes de circuits haute performance (par exemple des modulateurs Sigma-Delta à temps continu), assurant que les critères de performance pour un système complet peuvent être rapidement examinés, mais avec une précision comparable à celle de bas niveau (par exemple au niveau transistor).

Enfin, on utilise les paradigmes de conception qui ont été développés dans l'exploration de nouvelles architectures afin d'étudier leur faisabilité, les résultats attendus et les limitations éventuelles quand on emploie des solutions technologiques spécifiques.

On présente ici une méthodologie de conception originale. Cette méthode part d'un schéma analogique existant pour lequel les composants (transistors, composants passifs, sources

de tension / courant, etc.) doivent être dimensionnés et optimisés. Cette méthodologie est basée sur l'optimisation automatique des cellules analogiques au niveau transistor en utilisant une méthode bayésienne modifiée et la synthèse de macro-modèles analogiques de type Mathworks Matlab-Simulink ou HDL analogiques et à signaux mixtes (par exemple, VHDL-AMS, Verilog-A).

Notre méthodologie sera facilement interfacée avec les méthodologies existantes pour le numérique et la conception à signaux mixtes et multi-domaine, en utilisant des macro-modèles pour une caractérisation complète du système.

La recherche a porté sur l'établissement des fondements théoriques pour les extractions des macro-modèles et les méthodes semi-automatiques ou automatiques d'optimisation qui ont été intégrés dans un outil de conception (SIMECT), présenté comme une boîte à outils Matlab relativement automatique.

La validation de la méthodologie et des outils développés a été réalisée sur deux architectures de modulateurs Sigma-Delta temps-continu : une du 2ème ordre et une autre du 6ème ordre, basées sur un processus technologique de 350nm fournit par AMS. En tant qu'architectures passe-bande, elles sont conçues à l'aide de micro-résonateurs mécaniques de type résonateur à ondes de Lamb (LWR).

Ensuite, lorsqu'on utilise les réponses mesurées pour des résonateurs réels, fournies par le CEA-LETI, on a exploré des conditions de conception plus réalistes. Cela a permis une caractérisation concrète de la stabilité et des performances attendues des structures Sigma-Delta qui en ont résulté.

Organisation du mémoire

Dans le chapitre II, on étudiera les principes de fonctionnement et les contraintes de conception pour une classe de circuits hautes performances, les modulateurs sigma-delta à temps-continu. Issu des exigences fondamentales pour la conception de ces applications complexes, nous avons exploré les procédées de conception existants et ensuite avons proposé une méthodologie de conception analogique originale.

Dans le chapitre III, on présentera le processus d'extraction automatique des modèles comportementaux ou macro-modèles pour les amplificateurs, les résonateurs et les filtres. L'extraction et le calcul d'une multitude de figures de mérite assure que les modèles incluent les caractéristiques de bas niveau et peut être directement régénéré au cours du processus de la conception. Un examen préliminaire des langages de modélisation actuels et des environnements de simulation pour la macro-modélisation analogique a permis de choisir les outils de mise en œuvre de ces modèles.

Dans le chapitre IV, on présentera deux stratégies d'optimisation pour les cellules analogiques : une première méthode semi-automatique basée sur le pseudo-gradient et une

méthode automatique probabiliste. L'approche automatique repose sur une méthode bayésienne, où l'espace des évaluations est créé par le moyen d'un modèle de krigeage, et la sélection est effectuée en utilisant le critère *Expected Improvement* (EI) soumis à des contraintes. On compare les deux méthodes du point de vue de la cohérence et on présente une conception-optimisation d'un amplificateur à transimpédance.

Dans le chapitre V, la méthodologie de conception a été utilisée pour la conception d'un modulateur Sigma-Delta du 2ème ordre et du 6ème ordre. Dans un première approche, les éléments analogiques actifs des filtres du Sigma-Delta ont été optimisés et macro-modélisés. Ensuite, la modélisation de résonateurs micromécaniques LWR a été réalisé sur la base des paradigmes de conception de MEMS. Enfin, on a modélises les composants basses sur des signaux mixtes des modulateurs (convertisseur analogique-numérique – CAN ou convertisseur numérique-analogique - CNA) et on a intégré tous les composants au niveau système. On a vérifiés les résultats obtenus par rapport à des simulations au niveau transistor, et on a également exploré des conditions de conception plus réalistes fondées sur des mesures des filtres LWR.

Au dernier chapitre, *Conclusions et perspectives*, on réalise l'examen global du travail accompli ainsi que l'on présente des perspectives pour d'amélioration et de diversification de cette direction de recherche.

Méthodologie de conception pour des circuits analogiques

L'évolution des systèmes électroniques d'aujourd'hui vers des procédés CMOS nanométriques combinée avec le développement croissant des systèmes entièrement intégrés de type système-sur-une-puce (SoC) et système-en-paquet (SiP) comprenant des circuits mixtes analogiques et numériques impose l'existence de méthodes efficaces de conception analogique à mettre en parallèle avec les méthodes de conception numériques hautes-performances.

On préfère, en conception analogique, de manière classique, à partir d'un cahier des charges au niveau du système, ce qui entraîne la définition de fonctions, puis des opérateurs au niveau du circuit, et enfin des schémas conduisant à des implémentations au niveau des transistors (les paradigmes de conception descendantes). Cette approche permettrait de gérer ensemble les parties numériques et les parties analogiques et d'assurer l'optimisation des performances directement au niveau du système [12].

Malheureusement, avec les technologies submicroniques les modèles de transistors deviennent de plus en plus complexes. Les modèles classiques Ebers-Moll utilisés pour les transistors bipolaires et les modèles Shichman-Hodges utilisés pour les MOS qui étaient encore adaptés pour des tailles supérieures à 10um deviennent complètement inutilisables avec les dernières technologies. Des modèles mathématiques efficaces pour les transistors modernes exigent des dizaines de paramètres et les équations ne peuvent pas être traitées manuellement.

En outre, ces modèles sont de plus en plus directement intégrés aux simulateurs de sorte que leur utilisation est une étape obligatoire dans la conception de fonctions analogiques. En raison du temps de lancement du logiciel, chaque simulation dure quelques secondes avant de commencer effectivement. Ce temps est multiplié par les différentes analyses nécessaires afin de caractériser le circuit. Dans le cas d'un amplificateur analogique, de nombreux caractéristiques doivent être caractérisées comme le gain, les fonctions de transfert DC et AC, les impédances d'entrée et de sortie (réelle ou complexe), les non-linéarités. Toutes ces caractérisations nécessitent des simulations qui peuvent devenir très longues si elles ne sont pas automatisées. Les solutions de simulation existantes n'offrent pas de possibilités d'automatisation très flexibles.

Dans ce contexte, les techniques de conception impliquant seulement des simulations au niveau des transistors pour des fonctions analogiques complexes (comme les modulateurs sigma-delta en temps continu [14]) sont prohibitifs.

Par exemple, la simulation transitoire d'un modulateur complet avec 4096 échantillons de sortie (permettant d'évaluer son rapport signal sur bruit et la résolution en bits), peut exiger jusqu'à une semaine, en fonction de la complexité de la conception et de la vitesse de la machine.

Par ailleurs, le problème principal qui affecte les approches "purements" descendantes est que les résultats au niveau du système, même lorsqu'ils sont optimisés, se dégradent rapidement au niveau transistor suite à la dispersion technologique [15] et aux imperfections inhérentes aux processus analogiques.

Les possibilités d'optimisation automatiques pour un système entier sont très limitées en raison de ces temps énormes de calcul, requis par chaque étape d'optimisation, de sorte qu'un nombre minimal de scénarios peuvent être étudiés. Même les optimisations manuelles sont limitées aux composants simples ou blocs sous-système et sont très consommatrices en temps.

Pour ces raisons, la modélisation analogique de haut niveau devrait être prise en considération afin de réduire la conception globale du système et de l'effort de vérification. Mais le faible nombre de méthodes systématiques et le manque d'outils efficaces pour automatiser le processus d'extraction de macro-modèles [7] limite cette approche.

Pour être efficaces, les macro-modèles doivent inclure des caractéristiques de bas niveau, extraites au niveau transistor, ou même au niveau « layout », mais doivent aussi assurer une amélioration considérable de la vitesse de simulation.

A partir de toutes ces considérations, les différents paradigmes de conception existants sont étudiés, puis une méthodologie de conception analogique originale est proposée.

Nous avons commencé les travaux par une vue d'ensemble de la modulation Sigma-Delta et en particulier les modulateurs à temps continu, parce que, bien que la méthodologie proposée s’applique bien à leur conception.

Un aperçu des principes généraux de la conversion analogique-numérique, quantification, échantillonnage et mise en forme du bruit de quantification aboutit aux fondements de la conversion Sigma-Delta. Ensuite, les techniques de modulation Sigma-Delta, aussi bien en temps discret qu'en temps continu ont été discutées.

Afin de réduire l'effort de conception, la modélisation efficace de ces systèmes par des modèles de haut niveau doit être considérée. Nous prenons en compte les caractéristiques de l'architecture en boucle fermée et les exigences technologiques qui doivent être strictement observées sur les modèles respectifs et ils devraient assurer une amélioration considérable de la vitesse par rapport à l'implémentation technologique, au niveau des simulations.

Issu de ces exigences fondamentales pour la conception d'applications complexes haute performance Sigma-Delta, nous avons exploré une méthodologie de conception originale analogique.

Cette méthodologie de conception implique l'optimisation automatique ou semi-automatique de schémas et l'extraction des éléments analogiques au niveau transistor sous forme de macro-modèles pour Matlab-Simulink, VHDL-AMS et VerilogA.

Ces modèles de haut niveau intègrent les caractéristiques des circuits et des limites technologiques, assurant ainsi des figures de mérite réalistes.

Les macro-modèles rapidement simulables peuvent être utilisés pour mettre en œuvre et optimiser un système complet dans l'environnement orienté-objet SIMULINK ou avec des langages de description analogiques et à signaux mixtes VHDL-AMS / Verilog-A.

Modélisation comportementale analogique

La première étape de la méthodologie de conception analogique présentée dans la partie précédente, consiste en l'extraction automatique de modèles comportementaux ou macro-modèles pour des amplificateurs (transimpédances, transconductances, miroirs de courant, etc.), des résonateurs et des filtres.

On a menée une première étude sur les pratiques actuelles concernant la modélisation et les stratégies existantes afin de proposer un mécanisme de synthèse des modèles pour cette approche dans le cas de grandes fonctions analogiques à temps continu.

Ensuite, un examen approfondi des langages de modélisation actuels et les environnements de simulation pour la macro-modélisation analogique et a signaux mixtes, a permis de sélectionner des candidats potentiels pour la mise en œuvre des techniques d'extraction des modèles. Dans cette partie, l'environnement Mathworks MATLAB-Simlink ainsi que les langages VHDL-AMS et Verilog-A ont été retenues.

On a étudié pour chaque langage et environnement de simulation, un procédé pour l'extraction des paramètres de macro-modèles en partant des analyses au niveau transistors. On a identifié les simulations au niveau transistors de type DC, AC, paramétrique, transitoires et

balance harmonique comme suffisantes pour fournir toutes sortes de paramètres nécessaires à la modélisation des fonctions analogiques linéaires (pour lesquelles les effets non-linéaires doivent être négligeables par rapport aux contributions linéaires).

On a présenté les principales méthodes et algorithmes pour l'extraction de modèles en *p* et de modèles RLC pour les fonctions de transfert et les impédances de manière synthétique, puis l'extraction des effets non linéaires en DC et finalement l'extraction des faibles effets non-linéaires en hautes fréquences.

Toutes ces caractéristiques extraites sont intégrées dans quatre modèles de base qui permettent de modéliser les entrées ou les sorties en courant ou en tension de chaque bloc. En partant des quatre modèles de base, on a proposé des extensions pour les modes différentiel, afin de pouvoir simuler chaque combinaison unipolaire/différentiel en entrée ou sortie. De cette façon, on a construit seize modèles linéaires (« templates ») sous Simulink, bien que sous VHDL-AMS et Verilog-A. La synthèse se fait directement en partant de la topologie du circuit.

Tous les algorithmes de synthèse et les procédures ont été regroupés dans un outil de conception, SIMECT qui a été implémenté comme une boîte à outils MATLAB, testé et largement utilisé dans la conception et l'optimisation de filtres actifs pour les modulateurs Sigma-Delta.

Nous présentons l'interface entre MATLAB et les outils de CAO Cadence IC, utilisée pour le lancement des simulations analogiques directement à partir de MATLAB et la récupération des résultats de simulation à l'intérieur de l'outil. La forme de représentation des résultats de SIMECT est adaptée à la conception analogique, avec des résultats de simulations DC et paramétriques, ainsi que AC et temporelles.

Enfin, un premier exemple de conception est proposé. Pour illustrer les possibilités de la méthode de conception et de l'outil, une application complexe est proposée : un circuit de type convoyeur de courant de deuxième génération couplé avec un micro-résonateur mécanique à plusieurs résonances. On a utilisé une technologie de conception CMOS 65 nm en tant que processus cible.

L’exemple est proposé dans le seul but de retenir la démarche de modélisation de base, qui sera détaillée et appliquée pour la synthèse de modèles dans le cadre de la conception d’un modulateur Sigma-Delta.

Optimisation Bayésienne des circuits à base de « krigeage »

Le processus d'optimisation, connu également sous son nom historique de "programmation mathématique" représente un ensemble de méthodes mathématiques et les paradigmes utilisés pour la résolution de problèmes quantitatifs dans de nombreuses disciplines, comprenant l'ingénierie, la physique, la biologie, l'économie et les affaires.

Le problème classique d'optimisation consiste à maximiser ou à minimiser une fonction (appelée fonction objectif) par le choix systématique de valeurs dans un ensemble d'entrée autorisé et calculer le résultat de la fonction.

La règle générale de l'optimisation est d'essayer de trouver les «meilleures valeurs» de cette fonction objectif, étant donné un domaine défini, où les variables peuvent être soumises à des contraintes linéaires, non linéaires, ou entières [102].

Le choix d'une stratégie d'optimisation appropriée pour une discipline spécifique dépend du type de problème.

Cela est dû au fait que l'algorithme d'optimisation ne peut pas résoudre tous les types de problèmes et en général les performances de n'importe quelle paire d'algorithmes appliqués à tous les problèmes possibles sont identiques [103].

Une première discussion concerne donc les stratégies d'optimisation dans différentes disciplines et pour différents types de domaine d'entrée, lesquelles permettent l'identification de la classe des problèmes à laquelle l'optimisation des réseaux électriques appartient.

Dans ce contexte, le but de notre travail a été d'établir une autre étape fondamentale de la méthodologie de conception analogique présentée antérieurement : une méthode d'optimisation robuste convenant au dimensionnement automatique des circuits analogiques linéaires pouvant être intégrée avec les méthodes d'extraction de macro-modèles présentés auparavant. Ce processus est aussi connu dans l'industrie sur le nom du dimensionnement automatique au niveau transistor des circuits.

Dans un premier temps, nous définissons le problème de base d'optimisation d'une cellule analogique, puis nous choisissions une stratégie d'optimisation appropriée, qui sera affinée en fonction des besoins de notre méthodologie générale en vue de la conception et l'optimisation.

Le problème d'optimisation pour une cellule analogique est formulé en termes d'objectifs généraux et des contraintes d'optimisation.

La notion de contrainte forte est introduite afin de décrire les faibles effets non-linéaires liés aux fonctions d'amplification (non-linéarités, distorsions). Ces types de contraintes sont utilisés en conjonction avec des contraintes normales (impédances, offsets, gains, etc.) et un objectif d’optimisation (consommation, gains, facteur de qualité, etc.).

Pour ce type d'optimisation, les approches à base de simulations sont obligatoires, éventuellement avec un nombre minimum de points et des améliorations sur la vitesse de simulation.

Puisque la fonction objectif et les contraintes sont coûteuses à évaluer dans le cas de grandes cellules analogiques, on propose une approche probabiliste bayésienne basée sur des fonctions de substitution ou des méta-modèles afin de gagner en temps d'évaluation.

Les fonctions de substitution représentent en pratique une procédure intelligente pour le remplissage de l'espace d'entrée, où les simulations réelles sont effectuées dans un nombre limité de points de l'espace et pour tous les autres points échantillonnés, une prédiction probabiliste est utilisée.

Dans notre cas, on a utilisé la méta-modélisation à base de « krigeage » comme compromis entre la vitesse de convergence et la précision sur la position de l'optimum.

La corrélation spatiale pour l'interpolation statistique appelée « krigeage » est mise en œuvre dans les modèles de substitution. On a donc étudié le prédicteur de « krigeage ».

Les fonctions de régression qui sont utilisées pour composer le modèle de « krigeage » dans les cas : constante, linéaire et quadratique sont ensuite présentées. Les applications pratiques ont montré que fonctions ayant ces ordres sont suffisantes dans notre problème d'optimisation pour des cellules analogiques.

En outre, on présente les fonctions de corrélation pour des processus stochastiques dans les six cas utilisés dans notre approche : exponentiel, le exponentiel au sens général, gaussien, linéaire, sphérique et cubique. Ces fonctions sont utilisées pour trouver le degré de corrélation entre les différents points du domaine d'entrée.

Enfin, on étudie la méthode d'optimisation bayésienne des cellules analogiques dans leur intégralité.

Dans un problème d'optimisation bayésienne, on utilise des variables aléatoires et des processus aléatoires à la place des valeurs déterministes.

Dans ce cas, les variables aléatoires apparaissent dans la formulation du problème d'optimisation lui-même, lequel comporte des fonctions objectives aléatoires ou des contraintes aléatoires [126].

Les méthodes d'optimisation bayésienne généralisent les méthodes déterministes pour les problèmes déterministes. Même si l'ensemble des données se compose d'une mesure précise - déterministe, certaines méthodes introduisent du hasard dans le processus de recherche pour accélérer le progrès [127]. Un tel caractère aléatoire injecté peut également rendre la méthode moins sensible aux erreurs de modélisation. En outre, le caractère aléatoire peut permettre au procédé d'éviter un minimum local et potentiellement d'approcher un optimum global.

En effet, ce principe de randomisation est connu pour être un moyen simple et efficace pour obtenir des algorithmes avec de bonnes performances à peu près uniformément sur les ensembles de données, pour toutes sortes de problèmes.

Nous considérons ici le problème d'optimisation d'une cellule analogique, qui peut être assimilée à une fonction réelle continue, coûteuse à l'évaluation et, par conséquent, ne peut être évaluée qu'un nombre limité de fois.

On choisit l'approche bayésienne de ce problème afin de quantifier l'incertitude dans les régions de l'espace d'entrée qui ne peuvent être explorés initialement, en raison des évaluations limitées.

La méthode bayésienne consiste à combiner les résultats des évaluations et de l'information a priori sur la fonction cible afin de sélectionner efficacement de nouveaux points d'évaluation, jusqu'au moment où le budget pour les évaluations n'est pas épuisé.

L'algorithme appelé *Efficient Global Optimization* (EGO), proposé par Jones, Schonlau et Welch ([128]), est l'un des algorithmes les plus populaires d'optimisation bayésienne. Il est basé sur un critère de sélection appelé *Expected Improvement* (EI), qui suppose un processus gaussien de la fonction cible.

Dans l'algorithme EGO, les paramètres de la covariance du processus gaussien sont estimés à partir des résultats de l'évaluation par la méthode de la ressemblance maximale, et ces paramètres sont ensuite transférés à nouveau au critère de sélection EI [129].

Cependant, il est bien connu que cette stratégie de transfert peut conduire à des résultats très décevants lorsque les évaluations ne portent pas suffisamment d'informations sur la fonction objectif, afin d'estimer les paramètres d'une manière satisfaisante.

Pour éviter ces situations, on considère le critère *Expected Improvement* d'une manière plus élaborée : dans le cas des fonctions analogiques, indépendamment de la fonction objectif, il y a une série de contraintes qui doivent être respectés. Dans ce cas, l'EI est soumis à des contraintes et il s'appelle *Constrained Expected Improvement*.

En outre, les contraintes ont des poids différents dans la sélection des nouveaux échantillons, par exemple les contraintes fortes doivent être respectées obligatoirement sur chaque nouvel échantillon sélectionné, alors que les contraintes normales ne devraient être respectées que sur les résultats finaux. Dans ce cas, une légère modification de l'algorithme EGO est introduite, afin de traiter efficacement les deux classes de contraintes : dans le cas des contraintes fortes, on choisit des points potentiels seulement dans les régions de l'espace d'évaluation avec une forte probabilité de respecter ces contraintes.

On présente donc l'algorithme d'optimisation bayésienne utilisant le critère *Expected Improvement* : dans le cas sans contraintes, le cas avec contraintes et les modifications pour le cas contraintes fortes.

On a également mis en place, à titre de comparaison, dans la phase exploratoire, une méthode d'optimisation semi-automatique basée sur une méthode de descente du gradient.

Les méthodes d'optimisation automatique ou semi-automatique pour les cellules analogiques implémentées au niveau des transistors sont appliques et comparées sur l'exemple de conception d'un amplificateur a transimpédance et ensuite vont être utilisées pour la conception complexe des modulateurs Sigma-Delta.

Conception et optimisation au niveau système des Modulateurs Sigma-Delta

On présente les applications de la méthodologie de conception étudiée principalement par les architectures de modulateurs sigma-delta à temps continu à base de micro-résonateurs mécaniques intégrés.

Ceux-ci présentent une fréquence du travail nettement plus grande que les modulateurs à temps discret ainsi qu'une consommation plus faible.

Les modulateurs sigma-delta temps continu sont des applications aux signaux mixtes contenant des boucles de rétroaction qui doivent être optimisés de manière récursive en fonction des contraintes au niveau système et des limites technologiques afin d'obtenir des architectures stables et de haute performance [138].

Dans ce contexte, il est obligatoire d'avoir des représentations de haut niveau qui peuvent être simulées avec rapidité et précision et qui permettent la modification simple de l'architecture lorsque les critères de performance et de stabilité ne sont pas remplis.

La macro-modélisation ou la modélisation comportementale doit être utilisée, mais il est nécessaire d'utiliser une méthodologie unifiée pour la simulation et l'optimisation de tous les composants multi-domaines dans le but d'obtenir des figures de mérite réalistes.

Dans cette partie, les concepts présentés antérieurement (nouvelle méthodologie de conception pour circuits analogiques, modélisation comportementale analogique et optimisation Bayésienne des circuits à base de « krigeage ») sont appliqués à la conception et à l'optimisation d'applications Sigma-Delta d'ordre élevé.

En ce qui concerne la démarche de conception proposée, les étapes de la conception consistent premièrement dans une description du système au niveau fonctionnel et au niveau des opérateurs.

Ensuite, le niveau de circuits non-optimisés et l'implémentation des schématiques en technologie sont effectuées.

Dans le cas des deux architectures ΣΔ étudiés, cela a déjà été réalisé dans deux travaux précédents réalisés dans notre département [28], [39].

Une application Sigma-Delta du $2^{ème}$ ordre avec des résonateurs idéaux (circuit et lay-out) a été étudiée dans [28] afin de valider la solution et ensuite une architecture optimisée manuellement du $6^{ème}$ ordre a été présenté dans [39].

Notre contribution a été de proposer une méthodologie complète et les outils correspondants pour l'optimisation automatique des cellules analogiques et l'extraction de leurs macro-modèles, puis de reconsidérer la conception au niveau système en utilisant les modèles extraits et en tenant compte de toutes les imperfections extraites.

Comme nous l'avons vu, afin de modéliser l'ensemble d'un modulateur Sigma-Delta, en plus des fonctions analogiques d'amplification, des composants supplémentaires (les convertisseurs analogique-numérique (CAN), les convertisseurs numérique-analogique (CNA) et les micro-résonateurs mécaniques utilisés dans les filtres passe-bande actifs des modulateur) doivent être pris en compte en utilisant le même modèle de conception.

Les CAN et CNA sont des éléments mixtes, conçues en tenant compte des contraintes imposées par le filtre actif et les exigences d'architecture en général.

Pour garder la cohérence avec l'implémentation des autres composants analogiques, ils sont modélisés dans un environnement générique tel Simulink ou en utilisant un langage dédié à modéliser les composants de signal mixte comme VHDL-AMS.

Ceci est un avantage comme les modèles des fonctions d'amplification et les modèles de filtres sont également synthétisés sous Simulink ou VHDL-AMS/VerilogA en utilisant l'outil SIMECT proposé.

En raison de leur nature de signal mixte et de la structure spécifique, du nombre de bits et des autres figures de mérite, les CAN et CNA ne peuvent pas être modélisés automatiquement.

Ainsi, nous proposons des modèles Simulink et VHDL-AMS conçus à la main pour les convertisseurs utilisés dans les modulateurs Sigma-Delta, basées sur les spécifications utilisées dans [28] et [39].

Des composants micro-mécaniques, tels que les capteurs et actionneurs peuvent être intégrés aujourd'hui avec des composants électroniques sur une même puce afin d'étendre les fonctionnalités des systèmes et de réaliser des fonctions qui n'étaient pas disponibles auparavant.

Dans les architectures Sigma-Delta étudiées, on a utilisé des micro-résonateurs mécaniques afin de réaliser des fonctions de résonance à haute performance.

Ces systèmes micro-mécaniques ne peuvent pas être modélisés directement dans des environnements génériques comme Simulink. Dans ce cas, on a utilisé leur modèle fonctionnel ou électrique équivalent.

Une autre option repose sur un logiciel de simulation dédié à la simulation des MEMS comme CoventorWare [136] de Coventor.

Ils donnent les résultats les plus précis dans les simulations des résonateurs, mais ont aussi des inconvénients : les licences des outils dédiés sont coûteuses, l'environnement de

simulation est fermé et ne peut pas être facilement interfacé avec d'autres solutions de modélisation.

L'utilisation de langages pour signaux mixtes (VHDL-AMS) a été une solution pratique en vue de modéliser des résonateurs pour les filtres Sigma-Delta en raison de leur capacité à intégrer de multiples domaines physiques.

La démarche de conception suivie est structurée pour la modélisation des blocs composant des modulateurs et s'est effectuée de la façon suivante :

- Dans un premier temps, les amplificateurs linéaires des filtres Sigma-Delta actifs sont optimisés et un macro-modèle de type VHDL-AMS/Verilog-A ou SIMULINK a été automatiquement extrait pour chaque composant. Nous présentons dans ce rapport trois exemples d'optimisation et d'extraction, correspondant aux trois amplificateurs utilisés dans le filtre : amplificateur à transconductance, amplificateur à transimpédance et convertisseur courant-courant ;
- Ensuite, les stratégies de modélisation des composants multi-physiques - les résonateurs micro-mécaniques dans notre cas – font l'objet d'une présentation : l'approche structurelle, l'approche fonctionnelle et l'approche comportementale. Dans chaque cas, les avantages et les inconvénients de la méthode sont mises en évidence. Cela étant accompli, les filtres passe-bande entiers sont modélisés et on analyse leur réponse dans plusieurs cas : la fonction idéale, la fonction au niveau transistor et au niveau macro-modèles ;
- Une autre étape consiste à modéliser des blocs de nature mixte des modulateurs - les CANs et CNAs - à partir des spécifications initiales et les caractéristiques observées à partir de simulations au niveau transistor ;
- Une fois la modélisation des blocs est accomplie, le niveau système est de nouveau mis en œuvre avec des macro-modèles pour chaque modulateur. On présente les étages des filtres et les résultats sont comparés pour le niveau des étages et le niveau système. La cohérence avec les résultats des simulations au niveau transistor est aussi vérifiée. Dans un premier cas, nous utilisons des fonctions de résonance idéale pour les dispositifs micro-mécaniques;
- Enfin, on utilise les fonctions de résonance réels pour les dispositifs micro-mécaniques dans les architectures de modulateurs, y compris éventuellement des multiples résonances parasites extraites des caractéristiques mesurées.

Le tableau suivant présente d'une manière synthétique les résultats obtenus pour les différents scénarios d'implémentation des deux architectures de modulateurs :

Conception modulateur	Scénario de simulation	Résolution théorique
Passe-bande ΣΔ du 2ème ordre	Niveau transistor, idéal LWR	11.57 bits
	Macro-modèle Simulink, idéal LWR	11.44 bits
	Macro-modèle VHDL-AMS, idéal LWR	11.68 bits
	Macro-modèle Simulink, real LWR	10.82 bits
	Macro-modèle Simulink, real + harmoniques LWR	9.21 bits
Passe-bande ΣΔ du 6ème ordre	Niveau transistor, idéal LWR	15.5 bits [39]
	Macro-modèle Simulink, idéal LWR	14.78 bits
	Macro-modèle VHDL-AMS, idéal LWR	14.35 bits
	Macro-modèle Simulink, real LWR	12.20 bits
	Macro-modèle Simulink, real + harmoniques LWR	Instable

Les résultats sont cohérents sous les mêmes conditions du design et se dégradent lorsque des figures de mérite réalistes pour les résonateurs sont prises en compte.

On a principalement utilisé le langage VHDL-AMS ou l'environnement SIMULINK dans la modélisation de blocs et les implémentations de haut niveau.

On a partiellement utilisé les modèles Verilog-A dans les situations où une amélioration de la vitesse de simulation a été observé ou lorsque des problèmes de convergence sont apparus en raison des paramètres internes des simulateurs.

Pour ce travail, on a aussi utilisé les modèles Verilog-A pour minimiser les problèmes d'interface dans l'environnement Cadence AMS.

En outre, on a exploité les avantages offerts par la méthodologie de conception multi-niveaux discutée auparavant. On a effectué des simulations pour lesquelles une partie de la conception était au niveau des transistors et pour d'autres blocs au niveau comportemental ou fonctionnel, permettant de vérifier le fonctionnement correct d'une partie de conception, tout en conservant des temps de simulation raisonnables. Aussi, on a mis mise en place des simulations pour lesquelles on a utilisé des modèles multi-langues et multi-physiques, ce qui a permis une grande flexibilité dans la conception.

L'utilisation de cette méthodologie assure une amélioration considérable de la vitesse de simulation (d'un facteur entre 10 à 30) et des résultats cohérents. Des résultats comparables sont attendus lors de l'utilisation uniquement des macro-modèles de type Verilog-A.

Le cadre de travail proposé, l'application et des exemples de conception supplémentaires sont disponibles sur le site web de MathWorks à [148].

Conclusions

Ce travail de recherche a constitué en la mise en place d'une démarche de conception, de fondements théoriques, d'algorithmes spécifiques et d'outils de conception correspondants qui peuvent être utilisés dans la conception dynamique des fonctions analogique à temps continu, dans le cadre d'une complexité croissante des systèmes électroniques de nos jours, à la fois structurellement et fonctionnellement.

A partir des paradigmes existants pour la conception descendante et ascendante, on a analysé les outils de CAO actuels et des exigences relatives à la conception de circuits hautes performances ont été formulées.

En étudiant le cas particulier des modulateurs Sigma-Delta temps-continu passe-bande, une méthodologie de conception analogique efficace a été présentée.

Cette méthodologie est une approche descendante combinée à une chaîne de conception ascendante, basée sur l'optimisation automatique des cellules analogiques au niveau transistor en utilisant une méthode bayésienne modifiée et la synthèse de macro-modèles analogiques du type Simulink ou HDL analogiques et à signaux mixtes (par exemple, VHDL-AMS, Verilog-A).

Pour cette démarche de conception, une première étape de la recherche a consisté à établir une technique d'extraction des macro-modèles en faisant intervenir un ensemble complet d'analyses analogiques (DC, AC, transitoire, paramétrique, balance harmonique) effectuées sur les schémas analogiques implémentées dans un processus technologique spécifique.

On a montré comment les macro-modèles de fonctions analogiques peuvent être directement synthétisés à partir d'une multitude de facteurs de mérite extraits dans le processus de simulation : les fonctions de transfert et les impédances extraites de mesures AC, les décalages et les non-linéarités extraites de simulations paramétriques DC et les non-linéarités et distorsions obtenus à partir des scénarios de balance harmonique.

On a étudié différents types d'architectures de macro-modèles de façon à permettre l'extraction de tout type de fonction d'amplification, différentielle ou unipolaire, éventuellement avec des multi-sorties.

Tant que les modèles sont obtenus à partir de simulations au niveau transistor, ils apportent de précieuses caractéristiques de bas niveau à l'échelle du système et peuvent être directement régénérée dans n'importe quelle étape du processus de conception.

Mais il est important que les paramètres d'extraction soient correctement imposés et les simulations spécifiques soient exécutées dans les domaines d'intérêt (fréquence, DC, paramétrique) et sur des cellules déjà optimisés.

Pour optimiser les cellules, on a exploré une autre direction de recherche : l'optimisation automatique des circuits analogiques implémentés au niveau des transistors.

Dans un premier temps, nous nous sommes concentrés sur la recherche et la mise en œuvre d'une méthode simple et intuitive, qui peut être rigoureusement contrôlée et peut apporter des résultats d'optimisation, basé principalement sur l'expérience du concepteur et sur l'automatisation des simulations.

Ainsi, un procédé d'optimisation semi-automatique se basant sur un algorithme de descente de gradient a été exécuté. Cette méthode peut être utilisée pour les explorations initiales des performances des circuits analogiques et pour affiner les résultats de l'optimisation manuelle, mais il consomme beaucoup de temps de simulation, étant entièrement basée sur des simulations au niveau des transistors.

Mais son intérêt principal est lié au fait de pouvoir être utilisé pour approbation et validation de la cohérence avec n'importe quelle méthode d'optimisation automatique.

Ultérieurement, en parallèle avec un stage, une technique d'optimisation automatique pour les cellules analogiques a été développée. Cette technique utilise une méthode bayésienne, où l'espace d'évaluation est créé par le moyen d'un modèle de krigeage, et la sélection est effectuée en utilisant le critère Expected Improvement (EI) soumis à des contraintes.

Notre approche initiale était de considérer les objectifs d'optimisation avec deux types de contraintes: les contraintes fortes (non-linéarités, distorsions) et les contraintes normales (décalages, impédances, gains, largeurs de bande), qui sont adéquates en caractérisation analogique, assurant de solutions optimales robustes sur un nombre limité d'évaluations.

Des applications exécutées en parallèle entre les deux méthodes ont prouvé que les deux techniques sont cohérentes. La méthode d'optimisation automatique est très précieuse et présente de meilleures performances temporelles, lorsque ses paramètres sont réglés correctement.

Pourtant, dans l'implémentation actuelle de la méthode d'optimisation bayésien basée sur le « krigeage », il existe une limitation technique: à cause de la représentation des variables matrices sur la double précision réelle par Mathworks MATLAB, optimiser les circuits avec plus de 15-20 variables paramétriques avec une granularité normale des points n'est pas une tâche commode. Ce point sera discuté et une solution sera proposée comme une perspective de ce travail.

En parallèle avec le développement du travail théorique, un outil de conception a été développée (SIMECT), qui a été intégré comme une boîte à outils MATLAB, y compris tous l'extraction des macro-modèles et techniques d'optimisation automatiques.

La validation de la méthodologie et des outils développés a été réalisée sur deux architectures de modulateur Sigma-Delta à temps continu : un de $2^{ème}$ ordre et un de $6^{ème}$ ordre,

basée sur des processus technologiques de 350nm de Austriamicrosystems et de micro-résonateurs mécaniques de type résonateur à ondes de Lamb (LWR).

Plusieurs types de simulations et des expériences ont été menées en parallèle entre les implémentations au niveau transistor et ceux de haut niveau, en utilisant les macro-modèles extraits. Sur la base des résultats obtenus, la méthodologie de conception a été validé pour cohérence avec les résultats au niveau transistor et il a été montré qu'il est possible d'utiliser les implémentations macro-modélisation pour des explorations supplémentaires avec un niveau de confiance élevé (erreurs relatives maximales normalisées au nombre de points entre [10^{-8}; 10^{-6}] pour les modèles d'ordre deux-six, dans le cas des fonctions petit signaux et de densités spectrales de puissance).

Une fois la méthodologie a été validée, la conception des modulateurs d'ordre deux et d'ordre six a été explorée dans des conditions plus réalistes, en utilisant des macro-modèles et les réponses mesurées pour les résonateurs à ondes de Lamb réelles, fournies par le CEA-LETI.

Cela a permis une caractérisation fiable de la stabilité et d'atteindre les performances attendues des structures Sigma-Delta les plus réputées.

On a montré que le modulateur Sigma-Delta de $2^{ème}$ ordre reste stable et atteint une résolution maximale de 11 bits quand une fonction de résonance unique des LWR réels est employée, alors qu'il peut atteindre seulement 9,2 bits et reste stable seulement dans certaines configurations quand les fonctions réelles à plusieurs résonances sont utilisées.

Pour l'architecture Sigma-Delta de $6^{ème}$ ordre, on a constaté qu'une résolution maximale de 12,20 bits peut être réalisée, en présence d'une fonction de résonance unique des LWR réels. Mais de petites variations sur les gains des amplificateurs ont montré que le modulateur fonctionne à sa marge de stabilité et c'est possible que la mise en œuvre en technologie se traduit par une architecture non-stable. En présence de multifonctions de résonance pour les LWR réels, l'architecture de modulateur d'ordre six s'est révélée non-stable.

Des perspectives à ce travail qui peuvent être d'autres améliorations envisagées pour ces architectures sont présentées.

Une direction de recherche finale au cours de cette thèse de doctorat portait sur une étude préliminaire d'une nouvelle architecture Sigma-Delta passe-bande du deuxième ordre, en bande radiofréquence accordable autour de la fréquence centrale 1,25 GHz.

Basée sur des résonateurs réglables de titanate de baryum-strontium (BST), cette application fait partie du projet européen ARTEMOS, en cours de réalisation [13].

Pour l'architecture préliminaire, une technologie de référence STMicroelectronics 65 nm de basse consommation à usage général (Low Power General Purpose) en processus CMOS a été utilisé.

Perspectives

Les objectifs de recherche présentés dans ce rapport ouvrent des perspectives à court terme et à moyen / long terme.

Une première perspective à court terme concerne l'intégration des effets non-linéaires de haute-fréquence dans la caractérisation des cellules analogiques, la mise en œuvre de l'extraction des coefficients de non-linéaire de haute-fréquence et le placement de ces coefficients dans le procédé de synthèse des macro-modèles.

L'étude théorique a déjà été réalisée et le cadre de travail pour cette opération est déjà mis en place, il reste à trouver les analyses RF des outils de CAO actuels qui pourraient fournir les coefficients spécifiques de non-linéarité et de distorsion.

Une autre direction de recherche concerne les méthodes d'optimisation. La méthode d'optimisation automatique qui a été examinée n'atteint pour l'instant q'un seul objectif sous multiples contraintes.

Les situations d'optimisation pratiques ont montré qu'il est intéressant d'avoir une extension de cette approche vers de multiple objectifs et de multiples contraintes. Cela peut être fait en utilisant le critère de Pareto au moyen d'une pondération appropriée des multiples fonctions objectives de façon à n'atteindre qu'un seul objectif général.

En ce qui concerne la méthode d'optimisation bayésienne avec du krigeage, afin de surmonter la limitation technique en raison de la double précision imposée par Mathworks MATLAB, la mise en œuvre du calcul des méta-modèles de krigeage et des fonctions pour le critère *Expected Improvement* peut être envisagé sous C / C++.

Dans ce cas, l'usage des variables réelles simple précision apporterait une précision de calcul suffisante et les « routines » compilées assureraient une amélioration considérable de la vitesse des calculs.

Une perspective à moyen terme concernant les modélisations pour les architectures Sigma-Delta consiste dans la modélisation structurelle des résonateurs micromécaniques. Ce type de modélisation n'a pas été possible parce que les caractéristiques matérielles et géométriques de la technologie des MEMS n'étaient pas disponibles pendant la période d'essai.

Si le CEA-LETI fournit ces éléments dans les phases de conception suivantes de micro-résonateurs mécaniques, une perspective intéressante de ce travail est de mettre en œuvre les modèles structuraux des résonateurs et d'évaluer les résultats au niveau du système en utilisant la modélisation comportementale, fonctionnelle et structurelle.

De cette manière, l'approche structurelle explorée dans ce travail sera également validée avec les deux autres. Actuellement CEA-LETI et SUPELEC travaillent sur l'intégration des résonateurs réels dans les architectures Sigma-Delta.

A moyen ou à long terme, la perspective d'une implémentation en silicium et le prototype d'au moins un des architectures Sigma-Delta étudiés serait un critère de validation suffisant de la méthodologie de conception. Mais cela peut être fait seulement après que les micro-résonateurs mécaniques sont en phase de prototypage et au moins que la technologie d'intégration utilisée est compatible avec le dépôt de matériaux piézo-électriques afin de mettre en place un circuit monolithique.

Toutes les étapes pré-silicium vont nécessairement être réalisées avec au moins une validation de type « layout », incluant les LWRs réels avec les composants analogiques et à signaux mixtes.

La réalisation des « démonstrateurs » dans le projet ARTEMOS pourrait ouvrir une nouvelle perspective dans cette direction.

I

1. Introduction

1.1. Motivations

The development of nowadays electronic systems follows two main directions, e.g. a roadmap towards an increasing complexity, another towards an increasing functional variability.

The first axis, based on the famous *Moore's law*, consists in scaling to smaller and smaller process geometries, so today we are expecting to see more than 12 billion transistors per chip when using standard 20nm technology processes and bellow [1]. Already, the transistor count has achieved 7.1 billion on commercially available Graphics Processing Units (GPUs) [2] and 2.6 billion on general purpose microprocessors [3].

The second direction tries to integrate on very demanding performance solutions, but which should remain low-cost, an ever increasing number of diverse and complicated functions: the application-specific ICs (ASICs) market (current transistor count >200 million/chip), the application-specific standard parts (ASSPs) (>100 million/chip) tend to be replaced by the completely integrated systems-on-chip (SoCs) and systems-in-package (SiPs) containing mixed analog, digital and radio-frequency (RF) circuitry along with sensors and actuators, designed based on the novel paradigms of reusability. Practically, 70-80% of the today ICs are systems-on-chip [4], ranging from targeted applications (e.g. single-chip TV or single-chip camera [5]) to all the new microprocessors and integrated telecommunications systems.

Such a complexity of these systems achieves a degree never seen before in any artificial system (mechanical, electrical, thermal, architectural, etc.) and can only be compared with biological systems (e.g. the human brain remains for the moment more complex, with 80-100 billions of neurons and a very high degree of inter-connectivity between cells [6]).

A dramatic problem regarding the conception of the novel electronic systems resides in an ever-increasing gap between the actual design methodologies and the circuit scaling plus functional demands [7]. And with the advent of more and more mixed-signal applications,

including analog parts inter-related and controlled from the digital and digital signal processing (DSP) blocks, the problem becomes even more complex.

In fact, 90% of all SoC applications contain analog circuitry, and the analog content of these SoCs averages a relatively constant 20% of the circuit area. For example, a multicore SoC can incorporate more than 30 complex, independent analog systems [8] (multi-domain clocking PLLs, advanced thermal and power management, unit-level trimmers of analog components, high speed input-output interfaces, low jitter clock generators). Figure 1.1 depicts the placement of the multi-domain components in a recent GPU architecture [2].

This analog is implemented in CMOS or BiCMOS processes and is relatively complicated, with hundreds and often thousands of digital control signals [9]. Despite the trend to replace the analog circuitry with digital computations (e.g., digital signal processing in place of analog filtering), there are some typical functions that will always remain analog, the ones needed at the interface between the electronic system and the "real" world [10] (e.g. on the input: low-noise amplifiers, variable-gain amplifiers, filters, oscillators, etc. to treat the signals coming from sensors, microphones, antennas; on the output: drivers, buffers, filters to drive outside loads; analog-to-digital converters and digital-to-analog converters and last but not least, the largest analog circuits today are the high-performance digital circuits - state-of-the-art microprocessors, which are largely custom sized like analog circuits, to push speed or power limits).

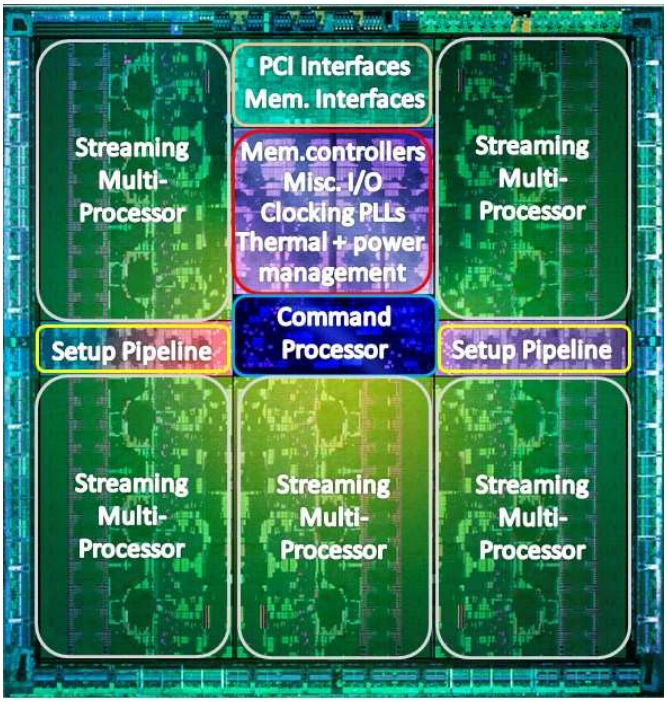


Figure 1.1 SoC components for a GPU example [2]

Without a methodical and well-established design and verification methodology, the complexity of these circuits is resulting in increasing numbers of functional errors, potentially leading to catastrophic failures and multiple design re-spins until obtaining a functional system.

Unfortunately, when compared to the digital design methodologies – already automated, efficient and industry mature – the analog methodologies did not evolve much from the 30-years old SPICE simulation techniques and the specific layout editing environments with their accompanying tools [11] (e.g., some limited optimization capabilities around the simulator, or layout verification tools). Thus, the design cycle for analog and mixed-signal ICs remains long and error-prone and the analog circuits design is often the bottleneck in mixed-signal systems, both in design time and effort as well as test cost, and they are often responsible for design errors and expensive reruns [7].

In this context, there is a strong need of effective analog design methodologies which can be correlated with the existing high-performance digital ones and potentially with other multi-domain methodologies (e.g. conception methodologies for micro-electro-mechanical-systems – MEMS – already integrated with CMOS) in order to obtain a generally-structured design paradigm for these highly complex systems.

Several analog design methodologies employing top-down approaches (for inter-connection with the digital ones) and behavioral modeling techniques based on hardware description languages (HDLs) have been proposed, still the largest problem remains the small number of systematic methods to create good analog behavior which can be exploited at system level and the lack of efficient tools to automate these methodologies and the models extraction process [12].

In conclusion, the establishment of state-of-the-art analog design methodologies and the corresponding electronic design automation (EDA) tools remains a very prolific research field, trying to answer to the increasing needs of the semiconductors industry and finally of the technology markets in general.

1.2. Objectives and Completed Work

Different classes of analog functions can be identified: the analog radio-frequency (RF) functions which are treaded starting from the *S*-parameters formalism, the low-frequency analog functions which can be non-linear functions (e.g. switches, comparators, rectifiers, etc.) and linear functions (e.g. different categories of amplifiers, linear filters, etc.).

In this context, our work is in the class of low-frequency linear analog functions, treating mainly amplification functions, resonators and linear filters for which the maximum linearity is demanded in their design process.

The first objective of this thesis is to establish an effective analog design methodology, to detail its methods, algorithms and to implement its corresponding EDA tools which can be used for the complex conception of linear continuous-time (CT) functions.

A second objective is to validate the proposed methodology on existing high-performance circuits structures (e.g. continuous-time Sigma-Delta modulators), assuring that the performance figures for a complete system can be rapidly investigated, but with comparable accuracy to the low-level (e.g. transistor-level) evaluations.

Finally, the conception paradigms which were developed are used in the exploration of novel architectures, to study their feasibility, the expected performances and the eventual limitations when employing specific technological solutions.

A novel design methodology is presented. This methodology starts from existing analog schematics for which the components (transistors, passive components, voltage/current sources, etc.) should be sized and optimized. This methodology is based on the automatic optimization of the transistor-level analog cells using a modified Bayesian Kriging method and the synthesis of robust analog macro-models in analog and mixed-signal HDLs (e.g. VHDL-AMS, Verilog-A) or simulation environments (e.g. Mathworks Matlab-Simulink). The methodology is easily interfaced with the existing digital, mixed-signal and multi-domain conception paradigms, employing the resulted macro-models for a complete system characterization.

The research focused on establishing the theoretical fundaments for the macro-models extractions and the semi-automatic or automatic probabilistic optimization methods which were integrated in a conception tool (*SIMECT*), presented as a completely automatic Matlab toolbox.

The validation of the developed methodology and tools was conducted on a 2^{nd} order and on a 6^{th} order CT Sigma-Delta modulator architecture, based on a 350nm *AMS* technology process and using micro-mechanical resonators of type Lamb Wave Resonator (LWR).

Then, more realistic design conditions were explored, when employing the measured responses for the real LWRs provided by CEA-LETI. This allowed a reliable characterization of the stability and expected performances of the resulted Sigma-Delta structures.

During this PhD thesis, a preliminary study for a novel 2^{nd} order Sigma-Delta architecture with tunable RF-capabilities, based on barium strontium titanate (BST) adjustable resonators was conducted. This work was supported in the framework of the ARTEMOS European project [13], currently in progress. For the studied architecture, a reference STMicroelectronics 65 nm Low-Power General-Purpose (LPGP) CMOS technology process was used.

1.3. Organization of the Statement

In **Chapter II**, the principles of operation and the design constraints for a class of high-performance circuits, the CT Sigma-Delta modulators, are studied. Deriving the fundamental

requirements for the conception of such complex applications, we explored the existing conception flows and then proposed a novel analog design methodology.

In **Chapter III**, the automatic extraction process of robust analog behavioral models or macro-models for amplifiers, resonators and filters is presented. The extraction and calculus of a multitude of figures of merit assures that the models include the low-level characteristics and can be directly regenerated during the dynamic design process. An initial comprehensive review of the current modeling languages and simulation environments for the analog and mixed-signal macro-modeling allowed choosing the implementation tools for these models.

In **Chapter IV**, two optimization strategies for analog cells are presented: an initial semi-automatic method based on pseudo-gradient calculus and an automatic probabilistic method. The automatic approach relies on a Bayesian method, where the evaluation space is created by the means of a Kriging surrogate model, and the selection is effectuated by using the expected improvement (EI) criterion subject to constraints. The two methods are compared for coherence and a trans-impedance amplifier optimization application is presented.

In **Chapter V**, the design methodology for optimization, macro-models extraction and high-level modeling of complex CT functions was employed for the design of a 2^{nd} order Sigma-Delta modulator and a 6^{th} order modulator architecture. As a first approach, the analog elements of the active Sigma-Delta filters were optimized and macro-modeled. Then, the modeling of the micro-mechanical resonators of type LWR was conducted based on MEMS conception paradigms. Finally, the mixed-signal nature components of the modulators (ADCs and DACs) were modeled and all the components were integrated at system level. The results obtained were approved for correctness when compared with transistor-level simulations, and more realistic design conditions based on LWR measures were also explored.

The last chapter, **Conclusions and perspectives**, makes an overall review on the completed work as well as presenting the perspectives for the improvement and continuation on this research direction.

II

2. Scope of Problem: Analog Design Methodology

2.1. Introduction

The evolution of today electronic systems towards nanometer-scale CMOS processes along with the increasing development of fully integrated systems of type system-on-a-chip (SoC) and system-in-a-package (SiP) including mixed analog and digital circuitry imposes the existence of efficient analog design methods which can balance the high-performance digital implementation and verification techniques, already very well developed and industry mature.

Classically, top-down design paradigms are preferred, starting from a system-level specification, resulting in operators-level functions plus constraints and finally schematics leading to transistor-level implementations. This approach would permit to handle together the digital and the analog parts and to ensure performance optimization directly at system level [12].

Unfortunately, with the sub-micron technologies the transistors models become more and more complex. The classical Ebers-Moll BJT and Shichman-Hodges MOS models which were still adapted for sizes higher than 10 μm become completely unusable with the latest technologies. The effective mathematical models of modern transistors require tens of parameters and equations and cannot be manually handled. Besides this, more and more models are directly integrated to simulators so that their use is an obligatory step in the design of analog functions. Due to the launching time of the software, each simulation needs a few seconds before effectively starting. All this is multiplied by the different analysis required to characterize the circuit. In the case of an analog amplifier, many topics have to be characterized such as the gain, the DC and AC transfer function, the input and output impedance (real or complex), the non-linearities. All these characterizations require simulations which can become

very long if they are not automated. The existing EDA simulation solutions do not have very flexible automation possibilities in this sense.

In this context, the design techniques implying only transistor-level Spice-like simulations of large analog functions such as continuous-time (CT) Sigma-Delta modulators [14] are prohibitive. For example, the transient simulation of a full modulator with 4096 output samples (allowing to evaluate its SNR and bit-resolution), may require up to one week, depending on the complexity of the design and the machine speed.

Furthermore, the main problem affecting the "pure" top-down approaches is that the system level results, even when optimized, are rapidly degrading at transistor level due to technological dispersion [15] and inherent process analog imperfections.

Automatic optimization possibilities for a whole system are very limited due to these huge computation times required by each optimization step, so a minimum number of scenarios can be investigated. Even manual optimization is limited to the simple component blocks and very time-consuming.

On the other hand, analog high-level modeling should be considered in order to reduce the overall system conception and verification effort. Here the largest problem remains the small number of systematic methods to create good analog behavior which can be exploited at system level and the lack of efficient tools to automate the macro-model extraction process [7].

In order to be effective, the macro-models should include low-level characteristics, extracted from transistor stage or layout stage, but they should assure a considerable simulation speed improvement when compared to the technology implementation.

Starting from all these considerations, the different existing design paradigms are investigated and then a novel analog design methodology is proposed.

The chapter will start by an overview of the Sigma-Delta modulation and particularly the continuous-time (CT) modulators, because, although the methodology is general and intended for a large palette of systems, its target applications consist in the CT Sigma-Delta modulators.

2.2. High-performance Continuous-Time (CT) Sigma-Delta Modulators

Discovered in the 1960s [16], and starting from the 1980s [17] [18], the Sigma-Delta conversion imposed itself in communication applications (both audio-frequency and video-frequency) because it allows a good compromise between the accuracy obtained and the costs involved (the circuit area and the integration cost).

As the integration technology evolved and also the theory on Sigma-Delta modulation was further developed, it is possible to consider new applications that operate at frequencies well above the audio range. This major improvement is primarily due to the transposition of

classical discrete time low-pass architectures to continuous time band-pass architectures which can work on considerable higher frequencies.

This section introduces the basic concepts concerning Sigma-Delta modulators functionality, such as the principle of over-sampling, noise shaping quantization and stability conditions, but also the mathematical tools for the design of continuous-time modulators.

The CT architectures and the principal challenges for the conception of such modulators are studied in order to derive some principles for an efficient design methodology in a general perspective of high-performance circuits. This is suitable as long as the CT Sigma-Delta architectures are mixed-signal and potentially multi-domain applications, they include a feedback loop, and their constraints for stability and performance are very strict.

2.2.1. Analog-to-Digital Conversion Principle

As suggested by the name, the analog-to-digital conversion is a process intended to convert an analog (continuous-time) signal ($x(t)$) into a digital (numerical single-level or multi-level) signal ($y(n)$) as shown in Figure 2.1. This operation is done in two steps, implying a sampler (to discretize the time) and a quantizer block (to discretize the continuous values).

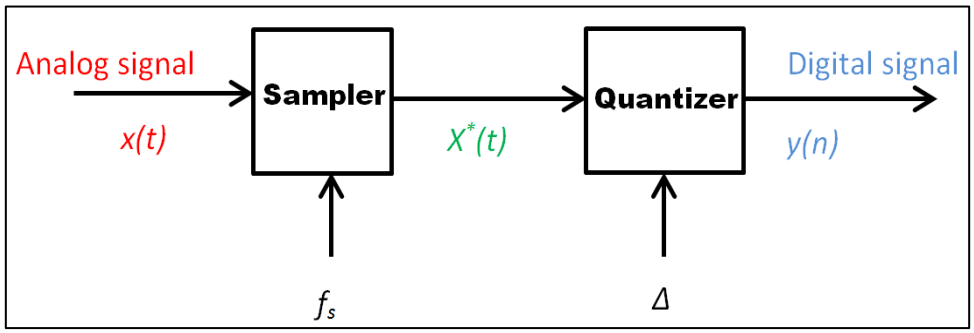


Figure 2.1 Analog-to-digital conversion general principle

Because of the sampler, the time index n of a digital signal ($y(n)$) is an integer number. The sampler output is a discrete-time signal represented by $x^*(t)$ where t equals nT_s and $T_s = {}^1/_{f_s}$ is the sampling period with f_s being the sampling frequency. The relation between $x(t)$ and $x^*(t)$ can be expressed as:

$$x^*(t) = \sum_{n=0}^{\infty} x(t)\delta(t - nT_s) \tag{2.1}$$

Where

$$\delta(t) = \begin{cases} 1, t = 0 \\ 0, t \neq 0 \end{cases} \tag{2.2}$$

is the Dirac delta function.

An analog-to-digital converter transforms first the continuous-time input signal ($x(t)$) into a discrete time signal ($x^*(t)$). Afterward, each sample is approximated by a quantizer to transform $x^*(t)$ into a digital signal ($y(n)$) containing a sequence of finite precision or quantized samples. The quantizer block can be either a simple comparator (in which case a 1-bit quantization is achieved) or it can be a more complex circuit with the output quantified in multi-bit [18].

2.2.2. Quantization Principle and Quantization Error

The quantizer block contains a finite number of levels or discrete values. It will round the input signal ($x^*(t)$), which can take essentially any continuous value to the nearest discrete level, as illustrated in Figure 2.2.

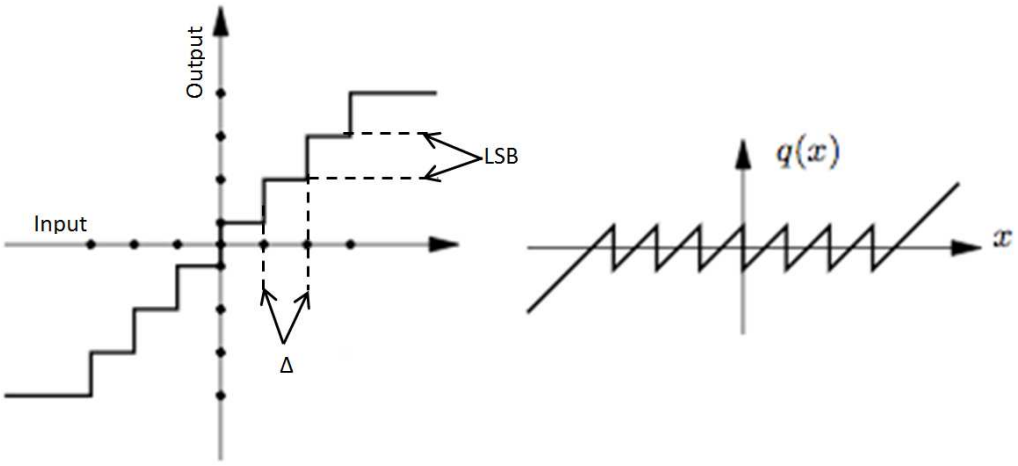


Figure 2.2 Quantization general principle

Because the quantizer essentially makes approximations, this process introduces an error signal that depends on how the signal is being approximated. This error, called the quantization error ($e(n)$), is of same order of magnitude as one Least-Significant-Bit (LSB) [19].

The quantization error is generally "small" when compared with a full amplitude input signal range but it may become relatively large comparing with the useful signal when the input signal has smaller levels, thus constituting a problem.

The relation between the quantization error, the output digital signal and the sampled signal is given by:

$$e(n) = y(n) - x^*(t) \tag{2.3}$$

Although the quantizer is a non-linear system and $e(n)$ is an unknown function, various approximate linear models of the quantizer are proposed to derive $e(n)$ and to be able to apply the classical methods of linear analysis to systems containing quantization nonlinear functions.

These models are all based on the asymptotic results of Bennet [20] or the exact results of Widrow and Sripad on the same subject [21].

Further, we will consider the approximated results of Bennet because they are easy to apply as general principle and do not lose their accuracy if we consider the simpler, ideal situations.

Bennet describes the conditions under which quantization error can be modeled by white noise. Assuming a B-bits quantizer, the number of available levels to quantize $x^*(t)$ is equal to 2^B. Thus, the interval between successive levels (Δ) is given by:

$$\Delta = \frac{1}{2^B - 1} \tag{2.4}$$

In the work [20], it is shown that $e(n)$ is a random quantity in each quantization step with equal probability, if the following conditions are met:

- The input signal does not exceed the dynamic zone of the quantizer
- The quantizer number of bits (B) is sufficiently large.
- The amplitude of the input signal is large compared with the LSB.

Then the variance of $e(n)$, which will be the quantization noise power ($\sigma_e{}^2$), can be calculated as follows [22]:

$$\sigma_e^2 = \frac{1}{2}\int_{-\Delta/2}^{\Delta/2} e^2 de = \frac{\Delta^2}{12} = \frac{2^{-2B}}{3} \tag{2.5}$$

As a result, the quantizer can be replaced by its simple linear model approximation, as presented in Figure 2.3.

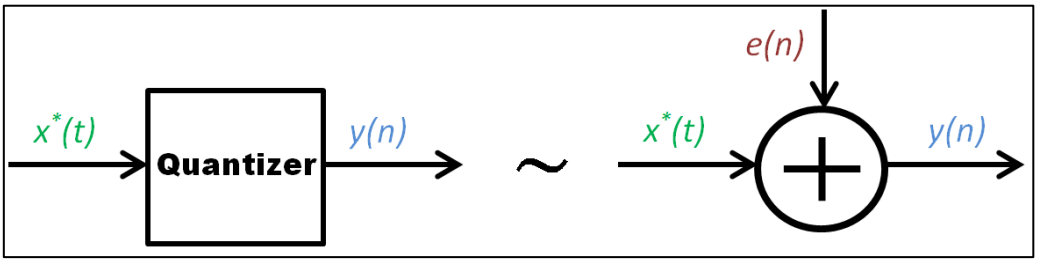


Figure 2.3 Quantizer linear model approximation

In analog-to-digital conversion since the quantization error is modeled as a white noise, the noise power is spread over the entire frequency range equally; the level of the noise power spectral density can be expressed as follows:

$$N(f) = \frac{\Delta^2}{12 f_s} = \frac{2^{-2B}}{3 f_s} \tag{2.6}$$

The noise level is a function of the quantizer number of bits and the sampling frequency. It becomes smaller when f_s or B gets larger.

Figure 2.4 depicts the spectrum of the quantization noise in the signal frequency band $[-f_b, f_b]$.

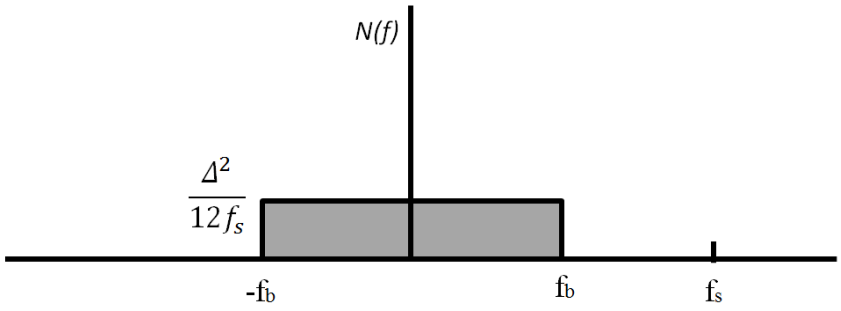


Figure 2.4 Noise spectrum of the quantization error

Considering a sinusoidal signal of amplitude equal to the full scale of the converter, sampled with respect to the Nyquist theorem ($f_s \geq 2f_{sinus}$), the signal to noise maximum ratio (SNR_{MAX}) can be expressed in this case as:

$$SNR_{max}(dB) = 10\,log\left(\frac{P_s}{P_n}\right) \cong 6.02B + 1.76 \qquad (2.7)$$

Thus, starting from SNR_{MAX} we can deduce the converter`s resolution and vice versa. The relation indicated above shows for example that an extra bit of resolution on the output requires +6dB for SNR .

The linear model of the quantizer is indispensable for the analysis of Sigma-Delta modulators [23]. The Noise Transfer Function (NTF) and the Signal Transfer Function (STF) of the Sigma-Delta modulators are found through this model. Although the model of Bennet is valid in a high-order Sigma-Delta modulator loop regardless of the quantizer number of bits, for low-order modulators this model is reliable when the quantizer number of bits is sufficiently large. The quantization noise of a low-order (second or fourth order) modulator containing a low-order quantizer (B = 1 or 2), does not have an equal probability for each quantization step.

It should be noted that in a Sigma-Delta modulator loop, the quantizer number of bits is generally small for practical considerations including chip area and power consumption.

The linear model may be modified to improve the approximation of the 1-bit quantizers as it is shown in Figure 2.5. The quantizer is modeled by a gain stage (ρ) associated with an additive white noise $e(n)$ [24].

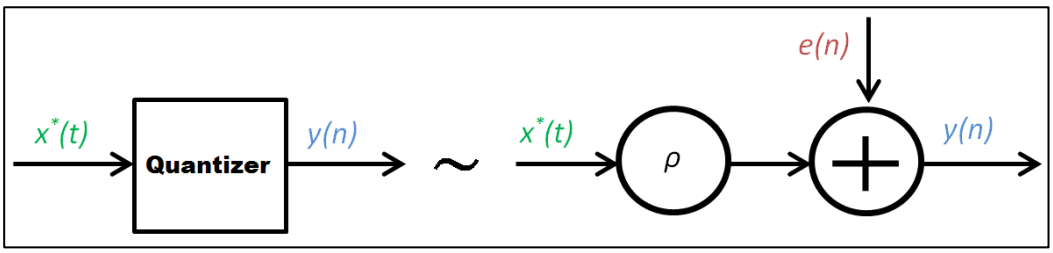


Figure 2.5 Linear model approximation for a 1-bit quantizer

Although ρ is almost equal to unity for high-order quantizer, for low-order ones its value depends on the modulator structure and on the amplitude of the input signal. Therefore, it is a variable number which is difficult to de estimated. In order to relax the analysis of the modulator function of ρ, one may use a dither signal which adds to the system an uncorrelated noise to minimize the correlation between the input signal and the quantization error. The dither must be small enough to result in a white noise [25].

2.2.3. Oversampling and Noise Shaping

Regarding the sampling frequency, the analog-to-digital converters may be categorized into Nyquist-rate ones and oversampling ones. The quantization process is different from one to another. Nyquist-rate analog-to-digital converters sample the input signal with the minimum sampling rate while oversampling analog-to-digital converters sample the input signal with a frequency significantly larger than the Nyquist-rate [26]. Afterward, a digital decimation filter is employed to reduce the signal rate to the Nyquist-rate [27].

Regardless of the quantization process, oversampling eases the design of the anti-aliasing filter. The anti-aliasing filter of Nyquist-rate analog-to-digital converters requires a flat response with no phase distortion over the frequency band of interest. Moreover, in order to prevent signal distortion because of aliasing, all signals above f_b (the maximum frequency of the signal's spectrum) must be attenuated, for example, by at least 96-dB to achieve 16-bits of dynamic resolution. But if considering the oversampling techniques, the anti-aliasing filter has significantly lower demands.

When sampling a signal with a very high frequency, well beyond the Nyquist frequency, the quantization noise spectrum is spread over a wider frequencies range, thus improving the signal-to-noise ratio in the band of the signal. It should be noted that for this improvement a digital decimation filter is employed, afterwards the analog-to-digital converter.

Therefore this high-speed sampling (oversampling) increases the accuracy by lowering the noise power. This mechanism is shown in Figure 2.6. In the first case, the signal is sampled at the Nyquist frequency ($f_{sN} = 2f_b$). As shown earlier, the quantization noise power spectral density has a uniform distribution in the interval $[-f_b; f_b]$. If the same signal is sampled with a frequency K times higher than the Nyquist frequency ($f_{sE} = 2Kf_b$), the noise power spectral density will be divided by K and will be spread uniformly in the interval $[-Kf_b; Kf_b]$, resulting in a constant total noise power.

Another advantage of this technique is that the constraints on the anti-aliasing filter placed at the input of the converter are less severe.

In the case of the oversampled converters, K is called the over sampling ratio (OSR) and is defined by:

$$OSR = \frac{f_{sE}}{f_{sN}} = \frac{f_{sE}}{2f_b} \tag{2.8}$$

The noise power will then be a constant function of frequency of the following form:

$$N_K(f) = \frac{\Delta^2}{12Kf_s} = \frac{2^{-2B}}{3f_s} \tag{2.9}$$

As a result, the SNR is improved by a *K*-dependent term:

$$SNR(dB) = 10\,log\left(\frac{P_s}{P_n}\right) \cong 6.02B + 1.76 + 10\,log[K] \tag{2.10}$$

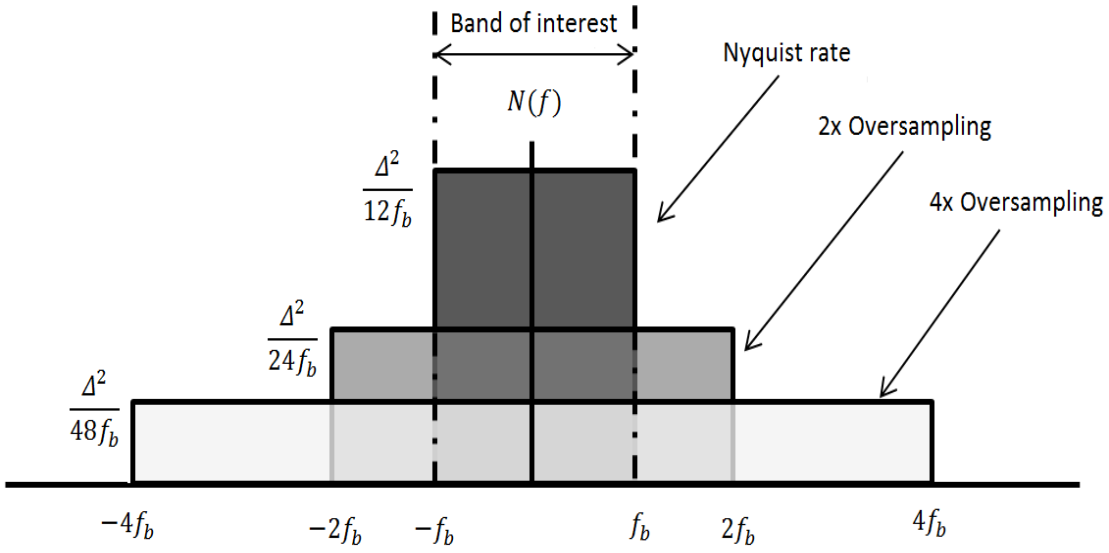


Figure 2.6 In-band noise for the Nyquist-rate ADCs and the oversampling ADCs

For instance, when using an over-sampling ratio equal to 4, the quantization noise in the band of interest is reduced by 6 dB, and therefore a resolution gain of 1 bit is obtained. If we want for example a resolution improvement of 10 bits, it results in sampling the signal with a frequency equal to $10^6 f_{sN}$ where f_{sN} is the Nyquist sampling frequency, which is not technologically possible if the bandwidth of the input signal exceeds few kHz. Usually, depending on the application, the OSR is lower than 500 [28].

Thus, for obtaining a greater resolution without lowering the bandwidth, a feedback filtering is employed, permitting to spread the quantization noise outside the useful band, resulting in a reduction of its energy in the signal's bandwidth. This is called noise shaping and it is widely used in novel analog-to-digital converters.

The oversampling and the noise shaping techniques are the two fundaments of the Sigma-Delta modulation functionality [23]. Figure 2.7 illustrates the noise-shaping proprieties of a first-order and a second-order low-pass Sigma-Delta modulators.

When the low-pass or band-pass filter order increases the system quantization noise shaping proprieties are improved [26]. Still, it is not possible to increase the order indefinitely because the system tends to become unstable.

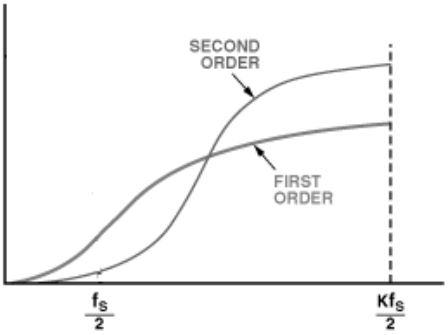


Figure 2.7 Quantization noise for a first and second order Sigma-Delta converters

2.2.4. From Delta Modulation to Sigma-Delta Conversion

The noise shaping concept presented before is a basic concept for Sigma-Delta modulation. The evolution towards Sigma-Delta architectures started with the Delta modulation, where a feedback loop was introduced for the first time to reject the in-band noise.

The Delta modulation is based on quantizing the change in the signal from sample to sample rather than the absolute value of the signal at each sample. The output error term $(x(t) - \hat{x}(t))$, in each sample, is quantized and used to make the next error term. This principle is described in Figure 2.8. An example for slow-varying input signal is presented in Figure 2.9.

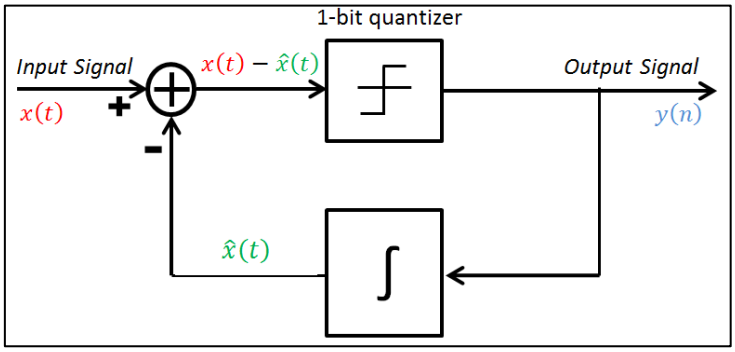


Figure 2.8 Delta modulation principle

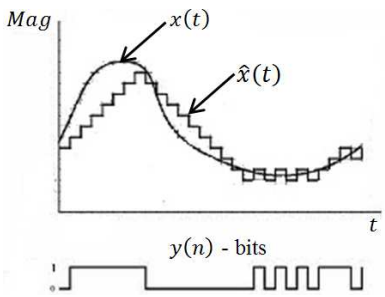


Figure 2.9 Delta modulation example – input, estimated signal and output

The error signal is smoothed by a low-pass filter. As a result, for a high frequency input signal or a high speed rise of the input signal, the modulator is overloaded. Thus, the performance of Delta modulation is limited in terms of speed.

Later on, to overcome this inconvenient, the Sigma-Delta solutions were proposed, with the integrator placed before the quantizer, which can be in this case a multi-bit converter, not only mono-bit as in Delta modulators. The Sigma-Delta modulation principle is illustrated in Figure 2.10.

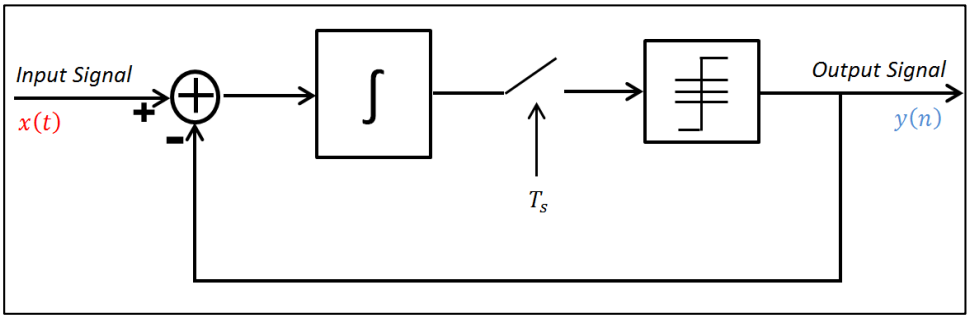


Figure 2.10 Sigma-Delta modulation principle

The name Sigma-Delta modulator comes from putting the integrator (Σ) in front of the delta (Δ) modulator.

Historically, the discrete-time (DT) Sigma-Delta modulators were first developed, where a discrete integration filter was used, and then the continuous-time (CT) modulators were employed, using a continuous-time low-pass filter. Also, the low-pass architectures appeared first, and then high-performance band-pass modulators were developed.

The modulator order corresponds to the number of employed integrators in a low-pass context and two-times the number of employed resonators in a band-pass context.

We will further detail the structure of a simple low-pass discrete-time modulator in order to derive some important functions and characteristics to be used for continuous-time architectures design.

In the ideal discrete-time case, for a first-order low-pass modulator the integration filter can be considered as an ideal integrator with the known transfer function:

$$H(z) = \frac{z^{-1}}{1 - z^{-1}} \tag{2.11}$$

Then, the linearized model of the first-order modulator can be employed (presented in Figure 2.11).

The quantizer block is shaped as an adder for an error signal that is considered constant (as discussed in Subsection 2.2.2).

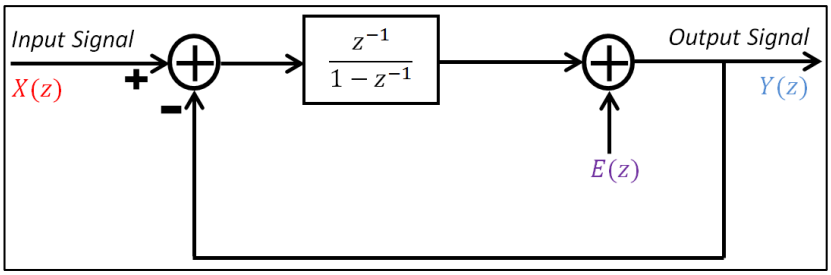


Figure 2.11 First-order low-pass Sigma-Delta modulator linearized *z*-function

The input-output transfer is described by the relation between *Y(z)*, *X(z)*, *H(z)* and *E(z)*:

$$Y(z) = \big(X(z) - Y(z)\big)H(z) + E(z) \tag{2.12}$$

In this manner, it is possible to calculate the two important functions which describe a Sigma-Delta modulator functionality: the signal transfer function (STF) and the noise transfer function (NTF):

$$\begin{cases} STF = \dfrac{Y(z)}{X(z)} = \dfrac{H(z)}{H(z)+1} \\ NTF = \dfrac{Y(z)}{E(z)} = \dfrac{1}{H(z)+1} \end{cases} \tag{2.13}$$

According to the type of filter employed (low-pass or band-pass), the signal transfer function passes the input signal in a low-frequency or intermediate frequency band of interest, as the noise transfer function attenuates the quantization noise in the same band.

It can be shown [18] that for a modulator of order less than 3, the STF is a delay function of the form z^{-L} and the NTF is a high-pass function of the form $\frac{(1+z^{-2})^{L}}{D}$ for the low-pass integrator and $\frac{(1+z^{-2})^{L/2}}{D}$ for the band-pass integrator, with D the function denominator. For the first order low-pass filter considered, the transfer functions STF and NTF are:

$$\begin{cases} STF = \dfrac{Y(z)}{X(z)} = z^{-1} \\ NTF = \dfrac{Y(z)}{E(z)} = 1 - z^{-1} \end{cases} \tag{2.14}$$

Since the signal transfer function is a delay function, the input signal is left unchanged as long as its frequency content does not exceed the filter cut-off frequency. On the other side, the noise transfer function is a high-pass filter and rejects the noise into a higher frequency band. The noise spectrum of a first-order and second-order low-pass Sigma-Delta modulators were compared in Figure 2.7.

A Sigma-Delta modulator generates an output which can be averaged over several input samples periods to produce a very precise result. The averaging is performed by decimation

filters following the modulator. Thus, the discrete-time modulator encodes a higher-resolution discrete signal, into a small number of oversampled bits.

Next, we will explore how continuous-time Sigma-Delta modulators were developed in order to convert analog signals directly to digital signals.

2.2.5. From Discrete-time Sigma-Delta to Continuous-time Sigma-Delta

A continuous-time Sigma-Delta structure is composed of an analog filter of type low-pass or band-pass, an analog-to-digital converter (the equivalent of the quantizer) and a digital-to-analog converter on the feedback loop. The digital-to-analog converter is essential in this case to convert the digital output back into the analog domain for the feedback connection.

The loop filter is used to minimize the average difference between the input signal and its quantized value, and so, the output signal will follow the input signal. A second function of the filter is to ensure the gain and determine the bandwidth of the rejected noise. As the order of the modulator increase, less noise is important in the useful bandwidth, but when the number of integrators or resonators is higher than two, a stability problem may occur, which may be corrected by using highly-parallel architectures [29].

There exist various methods and toolboxes to find the exact function for the global filter of a discrete time modulator corresponding to a series of design requirements [30], [31], [32].

Indeed, the design of discrete time modulators has progressed and this is why the design of continuous time modulators is based on the discrete time equivalents.

Practically, the synthesis of continuous time modulators is performed starting from their discrete time counterparts [33], [34]. However, the problem is not a simple transform from *z*-domain of *F(z)* - the DT filter function - to *s*-domain (*G(s)* - the CT filter function), but the discrete time and the equivalent continuous time modulators must produce the same output for the same input at each sampling period. The Figure 2.12 depicts such equivalence between a discrete-time and a continuous-time architecture.

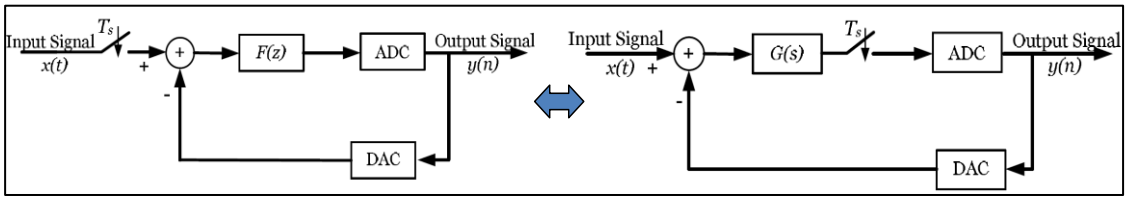


Figure 2.12 Discrete-time Sigma-Delta modulator and its continuous-time equivalent

As a result, the functionality of the analog-to-digital converter and the digital-to-analog converter should be taken into account in the transformation in order to ensure the correct equivalence between the two functions.

The parameters to be considered include the internal delays and the non-linearity of the analog-to-digital converter (ADC) and the digital-to-analog converter (DAC) as well as the rise

time and the form of the ADC and DAC outputs [35] [36]. A comprehensive approach regarding the CT Sigma-Delta modulators synthesis is presented in [37].

The delays introduced by both the digital-to-analog converter and the analog-to-digital converter and the shape of the digital-to-analog converter output signal are taken into account [38].

Standard tools available in symbolic calculation software, such as Laplace and *z*-transforms are employed. The basis of the synthesis of a CT modulator starting from an existing DT function is the following formalism:

$$F(z) = (1 - z^{-1})Z_T\{L^{-1}[G(s)B(s)]\} + D(z) \tag{2.15}$$

Where *D(z)* has the following form:

$$D(z) = \sum_k a_k z^{-k} \tag{2.16}$$

Here $G(s)$ represents the continuous-time filter transfer funtion, $F(z)$ the discrete-time equivalent, L^{-1} denotes the inverse Laplace transform, Z_T is the *z*-transform at the sampling period (T_S) and $B(s)$ denotes the delay and the non-ideality parts of the digital-to-analog converter and the analog-to-digital converter functionalities [28]. The term $D(z)$ is introduced as a result of the loop delay.

The delay is the necessary time period to convert the signal from analog to digital, to process it in the digital domain and convert it back from digital to analog. The treatment and the optimization of the loop delay are important aspects to be taken into account when designing Sigma-Delta architecture [39]. Extra-loop delay (>*Ts*) CT Sigma-Delta converters were recently demonstrated [37], with very good performances, but the necessary delay should be rigorously kept on the transistor-level structure, otherwise the bit-resolution is rapidly degrading. This is an important aspect which will be taken into account on the analog and mixed-signal design methodology.

The main features and drawbacks of CT Sigma-Delta modulators compared with their DT equivalents can be summarized as:

- *The sampling frequency*: in switched capacitor circuits, the sampling frequency is limited by the operational amplifier bandwidth and by the errors produced in the Sample-and-Hold circuit. These errors are due to the switches non-linearity, in charge transfer, at charge injection during clock pulses (Clock Feed through) and clock jitter. This has the effect of increasing the noise in the useful band and therefore limits the sampling frequency for discrete time modulators. While for continuous time modulators, this errors are rejected outside the interest band by the noise transfer function NTF, because the sampling is done inside the loop, before the quantizer. Thus

CT architectures are able to work at frequencies much higher than DT counterparts, possible directly in RF.

- *The power consumption*: the reduced constraints on the active parts of the modulator, especially on the operational amplifier, give a possible reduction of the consumption and also a reduction concerning the supply voltages for the modulators circuits. This is very important if the filter is designed by passive elements – R, L and C. On the other hand, the sampler introduced in the modulator's loop greatly reduces the thermal noise, and fold the signals outside the useful band, and so we can eliminate the anti-aliasing filter. Normally, for the same specifications, a CT modulator needs less current that it's discrete time equivalent.
- *Low supply voltage*: the trend of reduction the supply voltage for circuits, especially in CMOS technology has limited the performances of switched capacitor circuits (reduction in the output dynamics).
- *Design constraints*: the CT modulators design is not an obvious process due to the fact that the inverse transform in 2.15 is usually not satisfied and only approximated, mainly for high-order modulators. A series of constraints are imposed for the low-pass or band-pass analog filter (gains, offsets, bandwidths, delay introduced in the loop, etc.) which have a great impact on the performance and stability of the resulted modulator.

The first type of CT Sigma-Delta applications were the low-pass ones [40], potentially passive and employing low-pass analog filters with fewer constraints, but then high-performance band-pass applications [41], potentially with direct RF-conversion capabilities [42] were developed, in which a multitude of figures of merit should be optimized and intrinsic analog/mixed-signal design paradigms should be employed.

Considering the real mixed-nature of the CT Sigma-Delta modulators, we will explore next the classical approach for the conception of such systems and then we will propose a novel design methodology with the aim to integrate the analog design process in a multi-disciplinary mixed-signal methodology.

2.3. Conception Flows and Models for Mixed-signal Systems

The conception of the current high-performance mixed-signal systems relies mainly on top-down design paradigms in order to benefit from the very advanced digital design and verification techniques [10]. The bottom-up techniques are advantageous only when the low-level components exist and their usage is mandatory and the designer has to configure the system-level starting from these cells.

Unfortunately, although the analog circuits typically occupy only a small fraction of the total area of a mixed-signal IC, their design is often the bottleneck in mixed-signal systems,

both in design time and effort as well as test cost, and they are often responsible for design errors and expensive reruns.

This is mainly due to the fact that analog design in general is less systematic and more heuristic and knowledge-intensive in nature than the digital design, and that it has not yet been possible for analog designers to establish a higher level of abstraction that provides all the device-level and process-level details from the higher level design. This is why also the analog design methodologies are not very systematic and generally applicable.

The top-down methods start from a general and simple model (containing minimum details on the physical implementation) to finish by specifying every detail of the studied system. Such a generic top-down approach is presented in Figure 2.13 [10].

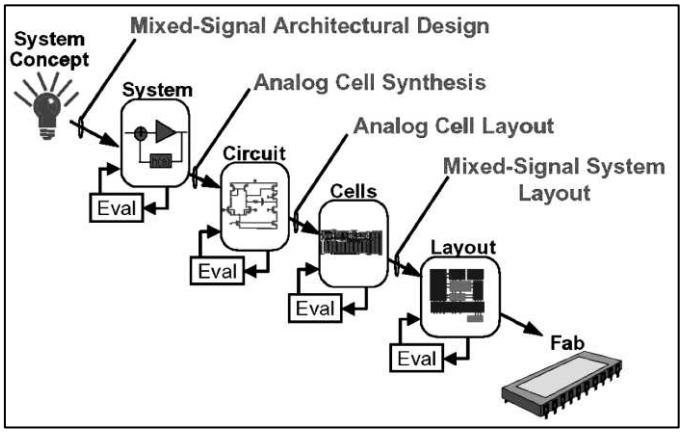


Figure 2.13 Generic top-down mixed-signal design methodology

The main advantages which can be derived for the top-down methodologies are summarized as follows [43], [44]:

- the possibility to perform system architectural exploration and a better overall system optimization (e.g. finding an architecture that consumes less power) at a high level before starting detailed circuit implementations;
- the elimination of problems that often cause overall design iterations, like the anticipation of problems related to interfacing different blocks;
- the possibility to simulate systems with some blocks modeled at a high level in combination with other blocks already implemented with more circuit detail, to fast explore critical parts of the design in an early stage of the flow;
- the possibility to do early test development in parallel to the actual block design;
- the delay of the choice for technical solutions (choice of technology and low-level architecture).

Thus, the modification of certain choices does not affect the early design stages. In addition, this approach guarantees the reusability of the models and therefore easily migration to newer technologies as opposed to models designed according to the bottom-up techniques, which are difficult to reuse [45].

However, the design of the basic blocks in both the analog and the digital domains (e.g. basic amplifiers, logic gates, etc.), can difficultly be achieved by using top-down techniques. There, each sub-block, amplification stage, voltage/current source is designed and then put together to form a basic cell.

One interesting problem is that the levels of abstraction in analog design are not as well defined as in the digital one. Indeed, the analog designer in his refinement approach from one level to another must take into account both the constraints of speed and accuracy and the existing simulation data [46].

By analogy with the digital design flows, it is possible to define the analog flows function of three classification criteria: the function, the structure, and the behavior. The functional models (defined by the functional components) and the behavioral ones (how the system performs the tasks) describe the operation of the system without taking into account the physical implementation (structural model or geometric).

Figure 2.14 shows in a first case these domains by three independent axes in the Digital design situation. The different levels of abstraction are presented as concentric circles intersecting all the axes (with the highest level domain on the outside and the most detailed to the center). Then, Figure 2.14 shows a second "Y" flow inspired by the digital flow. This flow, initially proposed by Gajski [47] has been adapted to the analog design flow [48].

These "Y" conception flows can be used by the designers to model the systems independently of their hardware architecture on their higher levels. Furthermore, as the abstraction level increases, the two flows can be used simultaneously to derive mixed-signal design methodologies.

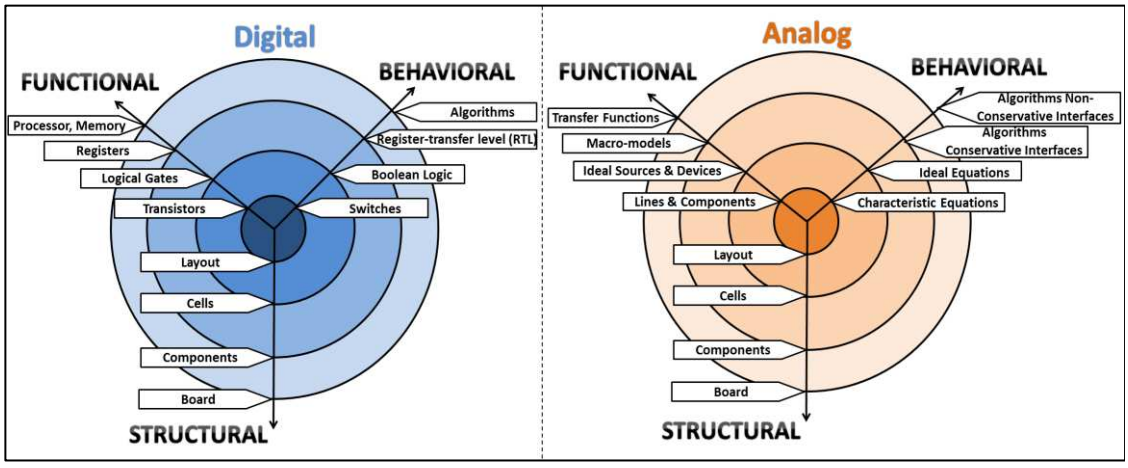


Figure 2.14 Digital and analog design Y conception flots

By studying the conception flows, we can derive that a functional or behavioral modeling is a *sine qua non* step in any advanced design methodology.

The advantages of mixed-signal modeling are widely discussed in the literature, but it is not easy task to formulate some general rules in order to gain maximum benefits from

modeling. The behavioral modeling or functional modeling have great potential for faster and better mixed-signal system verification, but using abstracted models without properly understanding their accuracy and limitations diminishes their effectiveness. Adopting behavioral modeling can be costly in terms of time and resources for developing them, thus a good synthesis method would be a valuable tool in the overall methodology.

The most general design techniques use algorithmic models or analog behavioral models at the top of the design process, along with RTL, to establish system-level behavior. The designers verify the system using these models, validating the mixed-signal architecture. Then, functional blocks are translated into transistor level designs. Here, designers can perform system regressions at any time by using transistor-level implementation for some blocks while retaining other blocks at higher abstraction levels, making the simulations faster. Once the transistor level blocks are verified, layouts are created and post-layout simulations are done. Analog behavioral models can be recalibrated against the transistor-level implementations using transistor-level, layout or post-layout data.

The most important types of models used for analog behavioral modeling can be classified in structured models (with template) and template-free models. Table 2-1 summarizes the most important techniques used to synthetize the analog models.

<table>
<tr><th>Structured models</th><th></th><th>Template-free models</th></tr>
<tr><td>1.polynomials and polynomial functions</td><td rowspan="8"></td><td rowspan="3">1. symbolic model generation as part of the optimization process via different approaches (e.g. [49])</td></tr>
<tr><td>2.rational functions</td></tr>
<tr><td>3.posynomials performance models [50]
(multi-variable functions used in Geometric Programming [51])</td></tr>
<tr><td>4.neural networks with no hidden layer or with one/multiple hidden layers [52], [53]</td><td rowspan="5">2. equation-based description – white box method: characteristic equations, ideal equations, etc. (e.g. [54])</td></tr>
<tr><td>5. models extracted using data mining techniques [55]</td></tr>
<tr><td>6. kernel forms, kernel methods [56]</td></tr>
<tr><td>7. support vector machines [57], [58]</td></tr>
</table>

Table 2-1 Model types for analog behavioral modeling

Starting from these general aspects regarding mixed-signal conception flows and the requirements for complex analog and mixed-signal systems (e.g. Sigma-Delta modulators) we will next propose a novel conception methodology and will detail its techniques.

2.4. Novel Analog Design Methodology

As discussed, the top-down design methodologies are used for complex analog and mixed systems (SoCs and SiPs) in order to handle together the digital and the analog parts. Unfortunately, for the analog design, the system level results, even when optimized, are rapidly degrading at transistor level due to technological dispersion, analog imperfections and mismatches between different cell components.

For example, according to the previous simulation results of Sigma-Delta modulators in transistor-level and layout-level [28] [39], the analog imperfections must be exactly taken into account in the design of high performance modulators.

The designed components should provide a proper performance to ensure the modulators stability and performances while there is no criterion to verify the suitability of a component expecting the performances of the modulator.

In a previous work of our department [28], an optimization method was developed for tuning the modifiable parameters of CT Sigma-Delta modulators (e.g. feed-forward coefficients of the active filter, loop delay, delay provided by ADC and DAC, etc.) accounting analog imperfections.

Although this method was tested with simple models of analog components, system-level models of real components extracted from transistor-level simulations were judged necessary to ensure the reliability of the designed modulators.

Then, the proposed method would get as input the realistic system-level models and would give as output the optimized modifiable parameters of CT Sigma-Delta structures. The main advantage would consist in the possibility to design a reliable modulator with minimum constraints on electronic design, by taking into account the possibilities offered by a specific technology or process.

This approach can be applied as a more general framework, for the entire class of high-performance CT functions and was the starting point for developing a novel analog design methodology for the weakly non-linear CT functions.

In order to overcome the inconveniences of ‘pure’ top-down design methodologies and to avoid the limitations of bottom-up approaches, nowadays the design flows are replaced by design chains or loops where the top-down approaches are combined with bottom-up ones, where the system structure and components can be adjusted according to the low-level simulations and figures of merit.

In this context, we propose a refined analog design methodology for weakly non-linear functions, which can be easily interfaced at its higher level (system-level) with the nowadays advanced digital and multi-domain design methodologies for a native mixed-signal multi-domain approach. Figure 2.15 presents the philosophy and the main components of this

methodology. The top-down path represents the classical top-down approach, while all the bottom-up loops are intended to adjust and enhance the system and circuits structure in a minimum number of iterations.

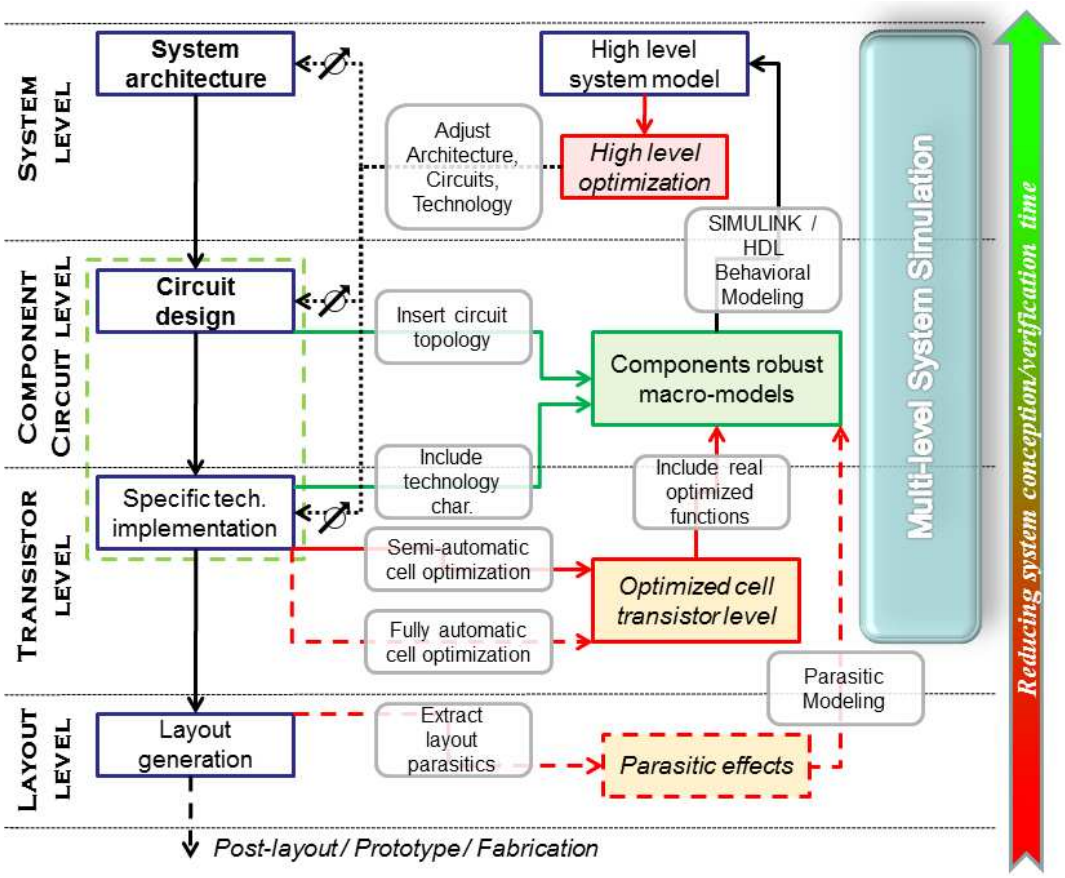


Figure 2.15 Analog Design Methodology based on automatic optimization methods and macromodels extractions

The working steps of the analog design methodology can be summarized as follows:

- Starting from a theoretically validated system architecture, the circuit design and the transistor level implementation on a specific technology are performed;
- Then a semi-automatic or fully-automatic optimization process is performed on each transistor-level cell using a MATLAB-CADENCE application framework;
- Using the optimal performance solution (in terms of desired gains, supply, impedances, nonlinear behavior, etc.) a robust component macro-model is extracted. It includes circuit topology characteristics and real analog functions;
- Using the behavioral macro-models, high-level system modeling can be performed;
- Doing system level optimizations will result potentially in architecture, circuits and transistor-level adjustments;
- The layout can be also generated and the parasitic effects incorporated in the macro-models;
- All these elements will result in Multi-level mixed system simulations.

The performance optimization at transistor-level is becoming increasingly used for designing high performance analog designs, for migrating analog and custom digital designs to new process technology nodes, and for creating high-speed, low-power digital cell libraries.

In our case, two transistor-level cell optimization methods were studied:

- a semi-automatic simpler method based on a *gradient descent* algorithm with *multiple starting points;*
- a fully-automatic optimization method based on a *Bayesian* approach, where the evaluation space is created by the means of a *Kriging surrogate model*, and the selection is effectuated by using the *Expected Improvement* (EI) criterion subject to constraints.

The two methods will be further detailed and exhaustively presented in the following chapters.

Regarding the macro-modeling, an effective modeling technique is implemented. The fitting approach is combined with a constructive approach, where the unipolar and common-mode/differential behaviors are included.

The three levels of models abstraction which were used are presented as example in Figure 2.16. Including these effects and the imperfections at system level (e.g. in the High-level Model 2), it is possible to directly optimize the system performance by taking into account the real parasitic behavior and the actual topology of the analog components.

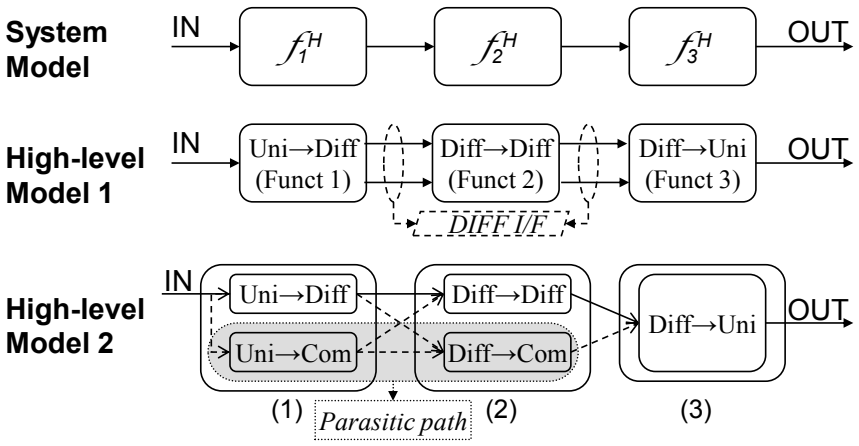


Figure 2.16 Abstraction levels for macro-models

Using the proposed design methodology, it is possible to synthesize robust high-level *analog behavioral models* in environments like Mathworks – MATLAB/Simulink or *hardware description language* (HDL) models of type VHDL-AMS and Verilog-A.

The following chapters will present into details the models extraction strategies and the techniques employed to efficiently use them for systems design and verification.

2.5. Chapter Conclusion

We started this chapter by exploring the nowadays general paradigms and solutions regarding the analog design in the context of the increasing number of mixed-signal applications.

We saw that the design, simulation and optimization of complex continuous-time (CT) circuits designed in nanoscale technologies like Sigma-Delta modulators require large computation times when using only design methodologies based on transistor-level analog simulators like CADENCE Spectre or PSpice.

Effective high-level system modeling is explored in order to reduce the conception effort. We take into account that the closed-loop architecture characteristics and technology requirements should be strictly observed on the respective models and they should assure considerable speed improvement when compared to transistor-level simulations.

An overview of the general principles of analog-to-digital conversion, quantization, oversampling and noise shaping permitted to identify them as fundaments of the Sigma-Delta conversion. Then the Sigma-Delta modulation techniques, starting with the DT ones and particularly the continuous-time (CT) modulators were discussed.

Deriving the fundamental requirements for the design of complex high-performance applications from the study of the Sigma-Delta systems, we explored a novel analog design methodology.

We presented then the design methodology implying the optimization and extraction of transistor-level schematics for analog elements into robust macro-models for MATLAB-SIMULINK, VHDL-AMS and VerilogA.

These high-level models incorporate accurate circuit characteristics and technological limitations, thus assuring realistic figures of merit.

The resulted fast-simulating macro-models can be used to implement and optimize a whole system in the SIMULINK object-oriented environment or the code-based analog and mixed-signal VHDL/Verilog extensions.

Using the HDL analog behavioral models, multi-level mixed simulations assuring a robust design process can be performed, potentially integrating this novel design methodology with digital and multi-domain methodologies for a native mixed-signal multi-physics design paradigm.

As we will further detail in the following chapters, an application framework based on the interface between MATLAB and CADENCE software tools corresponding to this design methodology is also proposed.

III

3. Automatic Analog Behavioral Modeling

3.1. Introduction

Nowadays electronic systems used in consumer devices and communications systems contain mixed analog and digital circuitry along with RF parts and even micro-mechanical and optical parts (oscillators, sensors, actuators), which are integrated in technologies at the nanometer scale.

The digital functions, which are dominant in the designs, are based on micro-processors and micro-controllers, memories and Digital Signal Processing (DSP) blocks, including an internal micro-program which can be potentially rewrite through firmware updates.

The most used analog functions based on amplification and filtering are located on the system inputs assuring the analog-to-digital conversion (ADC) and respectively outputs, for the digital-to-analog conversion (DAC) thus interfacing the internal digital functions with the "analog" outside world.

Given the heterogeneous nature of these complex systems, the existence of a unified methodology for the design and verification of all multi-domain components is imposed in order to achieve first-silicon success.

As discussed in Chapter 2, top-down combined with bottom-up design chains using macro-models are preferred, aiming to ensure the integration of the low-level effects at higher levels; however, these efficient analog design methodologies should still be refined for the automatic generation of the high-level models in the conception and optimization processes [7].

Furthermore, transistor-level Spice-like simulations are already prohibitive in the case of large functions and will not be practicable for future designs - increasingly complex - even when using high-performance simulators (e.g. FastSPICE from Cadence Design Systems).

The capability to freely mix and switch between different levels of abstraction (e.g. digital RTL and gates, transistors and analog and mixed-signal (AMS) functional or behavioral language descriptions) will be the solution to design and verify those systems [59].

With the migration to smaller geometries, two important directions should be noted: the number of transistors per chip increases and also the number of different functionalities integrated per chip. This results principally in an increased interaction between the analog and digital portions for an A/D mixed-signal system.

While a system containing only simple unidirectional A/D functionality can be designed and verified using a divided A/D simulation flow, in the cases of bidirectional interaction or feedback, this type of flow is no longer sufficient. Functional design failures and mixed-signal connectivity problems cannot be prevented in such designs without using mixed-signal simulation methodologies and accurate mixed models [12].

In this context, the models of the analog blocks should be natively inter-connectable with the digital parts (e.g. simulators which supports analog, digital and mixed-signal technology are already widely used), they should provide good analog behavior but also speed-up the simulations and potentially the methods to extract or synthesize these models should be automated.

3.2. Analog Behavioral Modeling for Large CT Functions

In the classic top-down design of continuous-time (CT) functions, the process is started on a high abstraction level using some type of macro-models.

Dependent on the starting point in terms of required abstraction, the analog blocks are described as behavioral or functional models in mixed-signal multi-domain HDLs, such as VHDL-AMS, SystemC-AMS, VerilogAMS, or even higher level system description languages (e.g. MATLAB®/Simulink®, SystemVerilog) allowing for fast chip-level simulations and improvement in the system architecture in early design stages [46].

A good example of large mixed-signal functions that are of interest for us are the continuous-time (CT) Sigma-Delta architectures [14], which were discussed in Chapter 2. Recently they adhered to the fast-growing world of nanoscale IC technologies. The use of well-adapted technologies along with new processes like integrating CMOS circuits with micromechanical devices (e.g. LWRs - Lamb Wave Resonators) offered the possibility to obtain very good bit resolutions for high levels of integration and low power consumption [60] [61] [62].

The conception effort for this kind of systems is very costly even when using high level models because, among others, the macro-models of the analog blocks are manually written, they should be iteratively updated or even re-wrote function of the architecture changes and also because automatic optimization methods for an entire system are not practicable [7]. This

situation raises the need of finding a way to automate the synthesis of high-level models, which can simulate fast, with comparative accuracy to the transistor-level simulators [39].

Several macro-modeling techniques have been proposed [63] [56] [64], still the largest problem remains the small number of systematic methods to create good analog behavior which can be exploited at system level and the lack of efficient tools to automate the models extraction process [12].

Specifically for Sigma-Delta converters some directions were explored. One consists in the extraction of generic semiconductor devices models and migration towards behavioral languages like Verilog-AMS, VHDL-AMS or independent platforms [65] [66] [67]. Still this procedure is not mature enough and the large technology diversity limits its efficiency. As an alternative, top-down design techniques using macro-models combined with optimization processes prior to transistor-level simulations were investigated [68] [69]. This approach is well-suited for initial design steps, but cannot guarantee the optimization of the transistor-level structure.

As a consequence, the use of macro-models should necessarily observe the following directions:

- Because the systems performance specifications are dependent on the technology employed and rapidly degrade for systems sensitive to dispersion (e.g. closed-loop Sigma Delta architectures, PLLs), the macro-models to be used should incorporate accurate transistor-level technology characteristics, analog imperfections and mismatches while enabling faster simulation;
- The models should be easily re-adjustable, function of changes at the system or transistor level [7];
- Efficient tools to automate the macro-model extraction process and the tuning of analog cells function of the changes at system level should be considered.

To address these issues, we presented in Chapter 2 a novel design methodology where the design flows are replaced by design chains or loops, where the top-down approaches are combined with bottom-up ones and the system structure and components can be adjusted according to low-level simulations.

In our case, the macro-models of linear CT functions like trans-conductance amplifiers (Gm), trans-impedance amplifiers (TIA), current conveyers (CCII), current mirrors, passive resonators, etc. are automatically extracted by performing a complete set of analyses (DC, AC, Transient, Parametric, Harmonic Balance) on the analog transistor-level implementation on a specific process for each function. Prior to extraction, an online optimization process of the analog cell can be conducted based on a Kriging method, assuring fast convergence of the optimization solution because analog functions are expensive to evaluate over large input space. The optimization feature is presented in Chapter 4.

The entire model extraction and optimization process is automated and integrated in a dedicated MATLAB®-CADENCE® toolbox, named SIMECT [70], from transistor-level simulations and the extraction of circuit figures of merit to semi-automatic and automatic optimization and macro-model extraction;

In the following sections we will detail the languages and environments used for modeling, the analyses, the algorithms which were developed and the architectures used for the analog behavioral models used in this design methodology and the corresponding design environment.

3.3. Choice of the Languages/Environments for Mixed-signal Modeling

As a first approach, an overview of the principal simulation environments and languages for the analog and mixed-signal behavioral modeling is presented. This will permit us to emphasize their strong and weak aspects and to select potential candidates in which the analog macro-models will be synthesized.

3.3.1. Mathworks MATLAB®/Simulink® Environment

The MATLAB®/Simulink® environment is well known by the specialists in electronics, automatics and telecom for its calculus power by using matrices and for the modeling flexibility provided by the different proprietary toolboxes (e.g. Signal Processing Toolbox, Filter Design Toolbox, Simpower™, etc. provided by Mathworks) and non-proprietary toolboxes (e.g. PLECS® [71] provided by Plexim, SLPS® [72] provided by Cadence Design Systems Inc.). This calculus power allowing to solve numerical problems quickly and the high ability to interface MATLAB®/Simulink® with others platforms and tools is essential in our modeling approach.

In our case, an initial interface between MATLAB® and Cadence Spectre RF circuit simulator was studied in order to be able to perform all kind of analog analyses directly from MATLAB® functions and scripts. This was possible by using the Cadence OCEAN [73] scripting language. Another task of the interface was to read the simulations results from Cadence generated files, which was done by using the Cadence MMSIM compiled MATLAB® functions. The interface description will be detailed in the following sections. For now, we will retain the possibility to interface MATLAB® with these tools.

MATLAB® and particularly Simulink® were studied in order to perform another functionality in the design cycle: the extraction of Simulink-type macro-models and the simulation of an entire mixed-signal system by using these models.

Simulink® is used widely in the dynamic problems from automatic control and signal processing fields. Its usage was extended recently to electronic systems, particularly for power electronics by using the Mathworks proprietary SimPowerSystems™ toolbox.

Simulink® is a good candidate for the synthesis of behavioral macro-models and the simulation of analog and mixed-signal systems mainly because:

- In order to create a model it is possible to use and interface a very heterogeneous suite of blocks: from ideal/real signal sources, mathematical functions to transfer functions and user-defined routines;
- The blocks can be automatically placed and parameterized by using pre-defined MATLAB® functions;
- The transfer functions can be simulated directly by using their s-domain form;
- It allows the usage of embedded functions (embedded C or MATLAB® functions simulated by Simulink®) – in the discrete domain, cycle-accurate or in the continuous-time domain;
- The embedded functions give access also to the MATLAB® pre-defined powerful functions and libraries;
- Electronic components which are not technology-specific can be directly co-simulated by using some toolboxes like SimPowerSystems™, PLECS® or even SLPS® for Spice interfacing; however, this is not sufficient in the case of technology-specific devices.

The existence of solid fixed-step and variable-step solvers, highly customizable and the possibility to control strictly the simulations were retained as good aspects for a simulation environment of the mixed-signal models. Still, there are some drawbacks and limits which should be noted as MATLAB®/Simulink® are generic simulation tools, and not targeted for mixed-signal electronic systems as we will further see for AMS languages for example.

Due to convergence problems which were noted when simulating large mixed-signal systems, we will examine the numerical methods used by the solvers of MATLAB®/Simulink® in order to better understand the extent at which this environment is reliable for mixed-signal modeling.

MATLAB® and Simulink® use the two mentioned types of solvers. For the fixed-step solvers, the simulation step can be parameterized by the user to a fixed value, but convergence is not assured. For variable-step solvers, the simulation engine chooses iteratively the step size until the solution is convergent, with the exception of step size inferior to a minimum step size (also parameterized), when the problem is declared non-convergent.

MATLAB® being an interpreted language, it can be notably accelerated when using compiled code. For the MATLAB® embedded functions in Simulink® this functionality is provided by the proprietary toolbox Real-Time-Workshop (RTW). It is interesting to accelerate simulations in the perspective of analog macro-modeling; the main drawback is that RTW

works only with fixed-step solvers, which are not applicable in the case of continuous-time functions simulations.

Mathworks provides a series of variable-step solvers for the treatment of problems which can be modeled as systems of differential equations.

One important note is that these solvers are different between MATLAB® and Simulink® [74], so the same simulation will give slightly different results when performed in Simulink® than MATLAB® and finally, it can be non-convergent in one environment and convergent in the other.

One of the causes of non-convergence of the solvers is the "stiffness" of some problems: such a problem is identified by Simulink® if rapid variations are detected in the system, but the final solution is changing slowly than these variations, when compared to the simulation interval [75]. In this case, the *ode45* variable-step solvers (used by default in Simulink®) will not converge. The *ode15s* and *ode23s* (which use the modified Rosenbrock order 1 and 2 formulae) are then recommended.

Theses variable-step solvers optimize the simulation step function of the speed variation in the model states. The solver passes at the next simulation stage (marks one step) only if the model generates an output. The management of events is synonymous with the management of Zero crossing (ZC): the moments when singularities appear in the simulation. In MATLAB® one should write its own functions for singularities treatment, while in Simulink® it is sufficient to select one automatic option or to use the possibility to write some procedures of ZC treatment as embedded functions. Effectively, the ZC block is integrated in the algorithm solver and particularly in the management of the step size which should be modified [48].

Another problem, specific for the electronic modeling in MATLAB®/Simulink® is related to the fact that these environments are not well adapted to conservative systems, thus it is not possible to define bidirectional physical quantities (e.g. voltages and currents) as in the case of AMS modeling languages. These quantities can still be separated and treated as inputs/outputs of the blocks as we will further discuss.

Related to the non-conservative behavior, another drawback is the aspect of algebraic loops which appear frequently in large systems. MATLAB® uses its specific algorithms to solve these algebraic loops, with a tradeoff in simulation speed, sometimes very significant. If it fails, it is necessary to introduce in the model an element which will break the loop. This operation uses a *Memory* block (from the library *Continuous*) and sometimes changes the dynamics of the system, affecting the results [76]. In this case, a good knowledge of the Simulink® behavior is required.

Moreover, using a dedicated environment like Simulink® affects the portability of the models, meaning that the synthesized models can only be simulated inside this environment, itself dependent on MATLAB®.

Given all these aspects, we can conclude that it is possible to use MATLAB®/Simulink® for analog and mixed-signal macro-modeling, but we should adapt the modeling techniques to the specificity and limitations of this generic environment.

As we will further see, a better approach is to use some dedicated analog and mixed-signal (AMS) languages for the models implementation.

3.3.2. Mixed-signal Modeling Languages

For a long time, the CAD (Computer Aided Design) and verification of mixed-signal systems was practically split into two directions: one focusing only on the digital portions and the other on the analog domain of the design. One common practice was to design and verify the digital blocks represented in Hardware description languages (HDLs) like VHDL or Verilog and to use digital pseudo models for the analog blocks in order to estimate the entire system behavior and correct connectivity issues. For more complex systems like M(O)EMS, the optical, mechanical and other non-electronic parts were also separately designed using FEM (Finite Element Modeling) environments and all the interactions between these heterogeneous components were hard to predict.

Such a modeling procedure must often be repeated from the beginning for each new project. And for systems where analog-to-digital or digital-to-analog feedbacks are present, or other type of feedback connections between electrical and non-electrical parts exist, the connectivity issues cannot be evaluated [59].

To overcome these problems, a common modeling framework for all the heterogeneous domains was established. This framework is represented by the modeling languages for multi-physics and mixed-signals systems. One of the major requirements these languages are trying to answer is to provide easy-to-simulate models of complex systems comprising all the parts from different fields of physics and whatever the nature of signals (discrete or continuous) [77]. In fact, the flexibility offered by these languages allows the creation of reusable models at different levels of abstraction with a minimum modeling effort (and therefore a lower cost): it allows, for example, non-technical users to write and maintain a library of abstract models or to create some templates for simple models which can be automatically generated starting from simulation characteristics.

On the other hand, the multi-physics and mixed-signals modeling languages provide an opportunity for designers to focus on the important parts of the design using a global approach in which only the critical components will be represented at physical level (the rest of the circuit will be maintained at a higher level of abstraction).

Next we will present the principal positive aspects and the main drawbacks of the most important mixed-signals modeling languages which were studied for automatic analog behavioral modeling: VHDL-AMS, Verilog-A(MS) and SystemC-AMS.

3.3.2.1. VHDL-AMS as a Mixed-signal Modeling Language

VHDL-AMS is an IEEE standard (IEEE 1076.1999) inherited from VHDL (1999). The great strength of this language resides in its fully mixed nature, allowing modeling of continuous time functions (analog systems domain), discrete logic (digital circuitry) or by mixing the two domains. At this flexibility, one can add the ability offered to designers to describe their models at different levels of abstraction. Indeed, VHDL-AMS provides mechanisms to manage both behavioral and structural abstractions.

At the behavioral level, it is the function performed by the system which is analytically modeled and not the physical phenomena behind it. However, at the structural level, the system is divided into sub-parts which can themselves be modeled using different abstractions (concept inherited from the digital design field in VHDL) [48].

For modeling the analog functions, VHDL-AMS proposes some specific concepts like the terminals (keyword *terminal*) to represent the analog nodes for which the laws of energy conservation must be applied. These terminals are specific for the physics field studied and are defined in some associated packages: for example the package *electrical_systems* defines the electrical nature [78].

Another specific concept is the representation of the unknown analog quantities as analog signals (keyword *quantity*) which can be either a standard type (integer, real, etc.) or a subtype of a type previously defined (e.g. *voltage* or *current* of the *electrical* nature). These quantities can be either free (for intermediate calculation) or related to physical terminals [79]. It is possible also to define for a terminal an intensive quantity – *through* – and an extensive one – *across*. In the case of electrical systems, *through* is a current and *across* is a voltage. The analog quantities are defined in continuous time and not discrete, unlike their counterparts in VHDL.

These are already some very strong aspects which oriented us towards the choice of VHDL-AMS as a modeling language for the synthesis of our macro-models. Apart from the general aspects, a number of specific benefits for automatic modeling were found very important in our case:

- Like in VHDL, the models are reusable (high modularity, possibility to create libraries and pre-compiled units);
- The models are multidisciplinary by using the different packages (electrical, mechanical, thermal) – in our case, it is possible to integrate the mixed-signal electronic models with micro-mechanical (e.g. Lamb Wave Resonators), electrical or optical parts models.

- It provides sequential instructions, like in classical programming but also simultaneous statements by using the *procedural* constructions and continuous instructions, targeting the analog domain modeling [80];
- The interaction between the analog and digital interfaces is allowed as well as that between continuous signals and discrete signals leading to natural mixed-signal descriptions;
- It provides continuous time and discrete time powerful tools: access to Laplace transforms (*'LTF*), Z formalism (*'ZTF*), derivative and integral (*'DOT* and *'INTEG*) which can be potentially cascaded [81], modeling propagation by using the *'DELAYED* operator and realistic conversions of the digital signals in the analog domain thanks to the *'RAMP* operator;
- It offers implicitly the possibility to solve algebraic and Differential Algebraic Equations (DAE); the Partial Differential Equations (PDE) are not implicitly supported, but some solutions [82] were proposed in order to extend the use of VHDL-AMS for solving PDEs;
- The language defines a semantics for energy conservation (in the declaration of quantities, the direction of the *across* quantity is defined – in the case of electrical systems this represents the sense of currents);
- The initialization by the user in case of discontinuity is possible by using the *break* statements;
- In terms of data transmission, VHDL-AMS supports both conservative and non-conservative laws in order to represent the data flow of a system by signals.

Still, VHDL-AMS has a few limitations and drawbacks which should be taken into account when developing the analog modeling framework:

- Different tools do not fully support the standard. For example, for the two tools we used: Cadence® Virtuoso® AMS Designer does not allow using arrays of values in the *'LTF*, *'ZTF*, *'DOT* and *'INTEG* attributes, while SMASH™ of Dolphin Integration allows this operation. Otherwise, the latter tool does not support other primitives defined by the standard as the simultaneous *procedural* statements;
- The portability of VHDL-AMS models from an environment to another is not assured. For example, the IEEE standard does not define how the tolerances must be taken into account. Therefore, the simulators use their own configuration for the tolerances, which is normally set in the environment of the tool. This configuration is not standardized for all simulators therefore it can cause sensitive differences in simulations;

- The solvability condition is very strict: at each time step, the number of equations should be equal to the number of unknowns. In practical terms, this condition results in the fact that the number of simultaneous equations must be equal to the number of *through* quantities plus the number of "free" quantities and the number of interface quantities in out mode. This criterion is more difficult to observe when the number of quantities is large.

Given all the presented aspects, we can conclude that VHDL-AMS is a very good candidate for analog and mixed-signal macro-modeling when taking into account the tools targeted for models simulations.

3.3.2.2. The Verilog-AMS and Verilog-A Languages

Verilog-AMS was created under the supervision of Accellera (EDA Standards Organization), inherited as the analog and mixed-signal extension of VHDL (1999). Unlike VHDL-AMS, Verilog-AMS is not an IEEE standard.

Verilog-AMS presents a large number of similarities with VHDL-AMS, we will note here only the reasons which were particularly interesting in our modeling approach:

- The sequencing of equations is a transparent task in VHDL-AMS, e.g. by using the simultaneous statements provided by the language. However, in Veriolg-AMS we are forced to use sequential equations in analog blocks;
- VHDL-AMS allows the use of intermediate variables named "free quantities" (in contrast to the *branch* quantities, bound to nodes). This feature is very useful when modeling complex circuits with complicated mathematical descriptions. In contrast, the variables in Verilog-AMS must be all associated with nodes. Therefore, the designer in Verilog-AMS is forced to add a fictive node for each intermediate variable which significantly increases the complexity of the circuit. The latter feature indicates the language circuit-oriented approach, but induces a loss in terms of degrees of freedom when compared to VHDL-AMS.

Verilog-A contains only the analog definitions of Verilog-AMS. It shares the same modeling characteristics, benefits of modularity and mixed-signal orientations, but also the limitations, applied to the analog domain.

A practical figure which oriented our choice for analog behavioral modeling towards Verilog-A rather than Verilog-AMS was related to the fact that the Cadence® Virtuoso® AMS environment translates all the analog interface between Spectre®/SPICE netlists and other AMS models in Verilog-A [83].

In our case, when co-simulating behavioral models with Spectre®/SPICE netlisted blocks, there are only two languages which should be complied and simulated when using Verilog-A and minimum three languages (Verilog-A is default for the interfaces) when using

VHDL-AMS or Verilog-AMS. A speed improvement and fewer convergence problems were observed for the Verilog-A approach in the Cadence® environment.

3.3.2.3. The SystemC-AMS Language

SystemC is a set of open-source libraries of classes written in C++ allowing the system level modeling of SoCs containing digital circuitry and embedded software. It is intended to do modeling and functional verification of the different block components and interfaces [84].

SystemC-AMS was developed by the SystemC AMS Working group (AMSWG) as an extension of SystemC, introducing system level design and modeling of embedded Analog/Mixed-Signal systems [85].

The current version of SystemC-AMS is optimized for applications in the signal processing domain. In theory, this language allows the modeling of conservative and non-conservative systems with dynamic and static non-linear equations. In practice, only few applications were proposed [86] and the SystemC-AMS standard is not yet supported by the majority of design tools.

In our case, SystemC-AMS, though promising, was not selected for automatic analog behavioral modeling mainly because the tools we used did not support the SystemC-AMS standard. Cadence® Virtuoso® AMS Designer does support the SystemC standard, but not the AMS extension, while Dolphin Integration SMASH™ supports only partially the SystemC-AMS. A perspective of this thesis will be the extension of the analog macro-modeling framework to SystemC-AMS.

3.4. Extraction of the Analog Macro-models Parameters

3.4.1. Performed Analyses for Macro-models Parameters Extraction

This design methodology and the resulted conception tools are essentially dedicated to the optimization and macro-model extraction of CT functions such as amplifiers and filters. The input and output quantities can be either currents or voltages. The input/output interfaces can be unipolar or differential.

We start from the general paradigm that each CT amplification function can be interpreted as a real dependent source of type:

- Voltage-controlled-voltage-source (VCVS);
- Voltage-controlled-current-source (VCCS);
- Current-controlled-voltage-source (CCVS);
- Current-controlled-current-source (CCCS);

This will generate 4 types of macro-models specifying the direct transfer function. Now, considering on both the input and output of the circuit the two quantities (voltage and current), a reverse transfer function, input and output impedances can be characterized. A sense convention was established for the model ports: on each side, the controlling quantity will be considered as an input port, while the controlled one as an output. This leads to a generic model structure as presented in Figure 3.1.

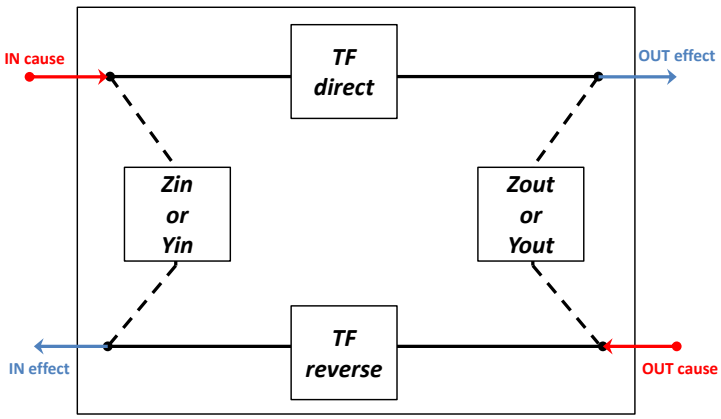


Figure 3.1 Generic macro-model structure

To create this kind of model, we will need information about the DC linear and nonlinear figures of merit, the AC linear and nonlinear characteristics, and the transient behavior, potentially on the input and output of the circuit. So the methodology was developed in order to allow performing the four types of analyses: DC analysis, DC parametric analysis, AC analysis and transient analysis. Radio Frequency (RF) Harmonic Balance (HB) possibilities were also studied.

In order to present the figures of merit, we will consider a unipolar amplifier example with the input as a current and the output as a voltage, but all other cases can be treated as we will further detail.

A. Input DC analysis

By performing a DC analysis by varying the input current we can deduce the gain, the resistive part of the input impedance, the DC transfer function and the input DC non-linearities; intermediary voltages can also be verified. By varying the output current, we can get the resistive part of the output impedance, the reverse DC gain and output DC non-linearities.

B. Output DC parametric analysis

A DC parametric analysis by varying the input and output currents can also be performed. It is possible to extract the output voltage as a function of the input and output currents and get a model of the non-linearities of the transfer function.

Then, the non-linear coefficients are extracted resulting in, e.g. a second order model,

$$V_{out} = gI_{in} + Z_{out}I_{out} + c_{20}{I_{in}}^2 + c_{11}I_{in}I_{out} + c_{02}{I_{out}}^2 \quad (3.1)$$

The extraction algorithm for higher order models is presented in the following sections. In SIMECT, the nonlinearity coefficientsc_{i0}, c_{0j}, c_{kk} are assimilated as NLX, NLY and IMOD.

C. AC analysis

The AC analysis will be used to get the input and output complex impedance and gains function of the frequency – normalized or real. It is then possible to deduce an s-model of input and output impedances and transfer function. The order of the model can be either fixed or automatically calculated by choosing the minimum order for which the error between the model and simulations in a specified frequency range is under a threshold. RLC models of the input / output impedance can also be computed. In the following sections we will detail the extraction algorithm. Other figures of merit at a defined *f0* frequency can be verified along with the $f_{cut\text{-}off}$.

D. Transient analysis

With a transient analysis, it is possible to verify, for example, the stability of the element.

3.4.2. Extraction of the *s*-models for Transfer Functions and Impedances

Referring to the generic model structure in Figure 3.1, the direct transfer function (TF direct), the reverse transfer function (TF reverse), the input impedance or admittance (Z/Yin) and the output impedance or admittance (Z/Yout) are stable s-functions of a maximum specified order, extracted from AC analyses results using the damped Gauss-Newton method for iterative search [87].

Considering H_n the n-vector of circuit responses at discrete frequencies w_n and wt_n a weighting vector with values in [0; 1], the B and A polynomials of the s-function will be extracted from the frequency response as:

$$\begin{cases} B(s) = \overline{H_n} \\ A(s) = 1 \end{cases}; When \langle\langle H_n \rangle\rangle = 0; \tag{3.2}$$

B and A for $\langle\langle H_n \rangle\rangle \neq 0$ are found by minimizing the criteria:

$$c_{ord,b,a} = \min_{ord} \left(\min_{b,a} \sum_{k=1}^{n} wt[k] \left| H[k] - \frac{B(w[k])}{A(w[k])} \right|^2 \right) \tag{3.3}$$

with b, a - the vectors of polynomial coefficients in B, A and ord the model order. An initial linear estimate is used.

The model structure presented before is the most simple - unipolar, but many designs, including Sigma-Delta architectures use differential amplifiers and filters. A more detailed approach is used for the differential designs, where the common-mode/differential and normal input-output characteristics are studied.

For this purpose, the differential input and output possibilities were added. Each of the 4 model types will define 4 sub-types corresponding to the combinations unipolar/differential on input and output. The differential structures are natural extensions of the basic models for which the differential and common-mode s-functions were converted into normal, path-associated functions. With the 16 macro-models structures one can extract all unipolar/differential circuits but also multiple-inputs-multiple-outputs (MIMOs) [88].

For differential macro-model extraction, the proposed methodology provides common-mode and differential AC analyses on the circuit input and separate AC analyses for the outputs. This assures the minimum number of components per model (e.g. only 2 reverse transfer functions and separate output impedances), without degrading the differential behavior. The transistor-level differential/common-mode responses will be used to compose the 4 normal input-output responses (3.4).

The resulted vectors are converted in the actual s-functions (3.5), using the equations 3.2 and 3.3 which were noted as the Φ formalism:

$$\begin{aligned} TFD_n^1[i] &= (TF_{comm}^{out1}[i] + TF_{diff}^{out1}[i])/2 \\ TFD_n^2[i] &= (TF_{comm}^{out2}[i] + TF_{diff}^{out2}[i])/2 \\ TFD_n^3[i] &= (TF_{comm}^{out1}[i] - TF_{diff}^{out1}[i])/2 \\ TFD_n^4[i] &= (TF_{comm}^{out2}[i] - TF_{diff}^{out2}[i])/2 \end{aligned} , i = 1 \ldots n \tag{3.4}$$

$$TFD^j(s) = \Phi\left(TFD_n^j\right) , j = 1..4 \tag{3.5}$$

Figure 3.2 presents an example of extracted s-functions (maximum order=2, frequency-de-normalized characteristics) for a differential-input differential-output current amplifier. Function decomposition on each path was applied. The design methodology provides through the SIMECT tool directly this figure of merit once the requested analyses ran, in order to verify the model correctness before synthesis.

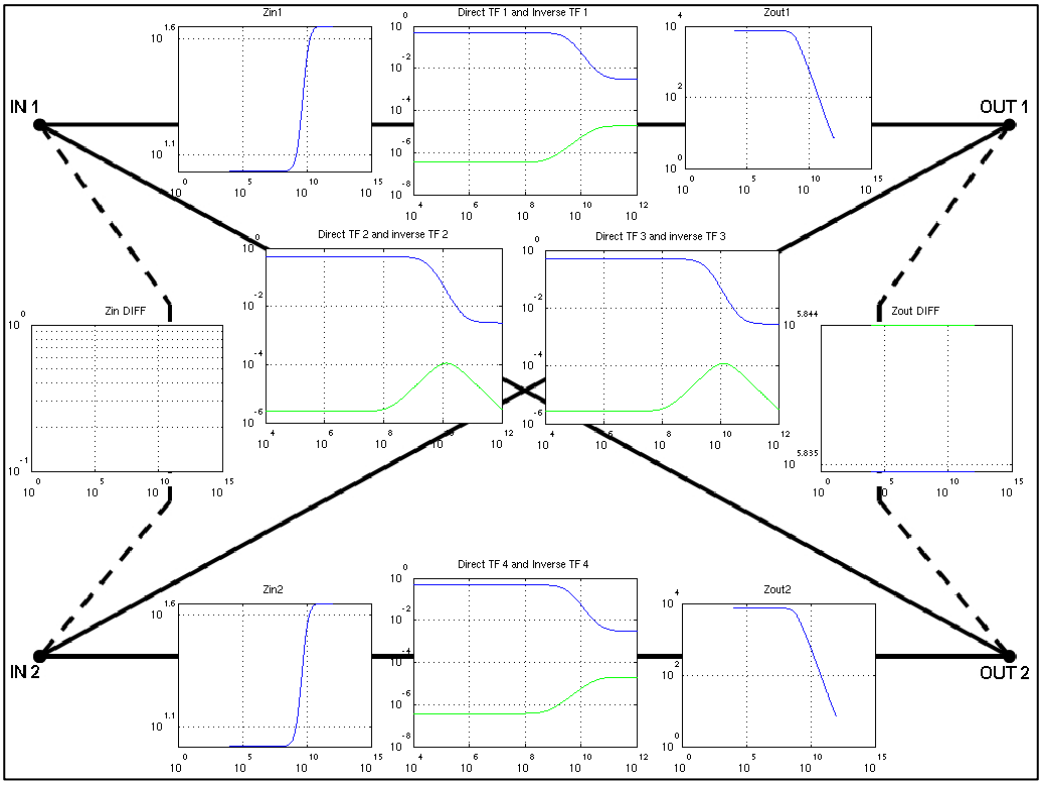


Figure 3.2 Differential macro-model topology example presented as s-functions

The extraction algorithm accuracy depends on the transistor-level function complexity in terms of poles and zeros versus the maximum s-model order selected. For the Sigma-Delta modulator components which were studied in our work, we obtained a maximum total relative error normalized to the number of points in the interval [10^{-7}; 10^{-6}] when 2^{nd} order models were used for extraction and [10^{-8}; 10^{-7}] for 6^{th} order models (for the studied differential current amplifier, 6.56e-7 for 2^{nd} order and 2.98e-8 for 6^{th} order).

3.4.3. Extraction of the RLC Models for Transfer Functions and Impedances

For the input impedance/admittance (Z/Yin) and the output ones (Z/Yout), it is possible to approximate these functions (measured as functions of de-normalized frequency) with RLC models. These RLC models will not be used in the behavioral models synthesis but can be used to quickly verify the impedance adaptation or physical correctness of the designs.

Two types of models were studied which cover practically all the linear cases studied: Rs-RLC and parallel RC groups [89].

The extraction of the RLC model consists in a least square fitting of the measured data starting from the theoretical solution of the RLC resonator.

Figure 3.3 exemplifies an Rs-RLC impedance model along with the transistor level simulation data and the extracted model.

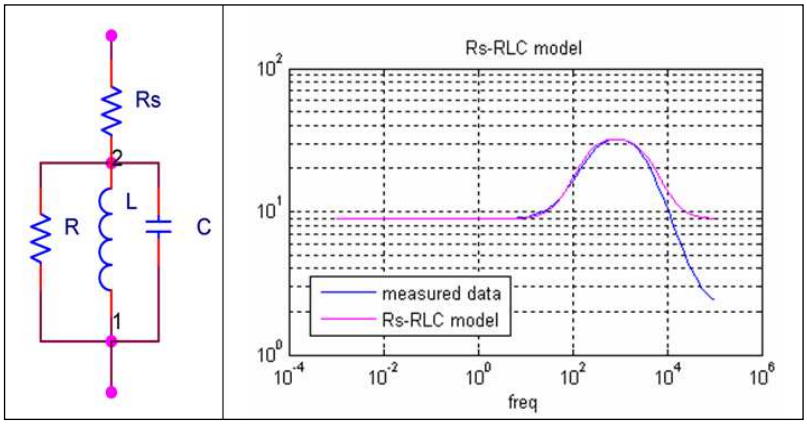


Figure 3.3 Rs-RLC impedance model, transistor-level simulation and extracted data

In this case, the pure resistive part *R+Rs* can be directly extracted as the real part of the measured impedance in DC:

$$R + Rs = Re\left(Z_{meas}(j\omega)\right)\Big|_{\omega=0} \tag{3.6}$$

The resistance *R* is obtained as:

$$R = max\left(Re\left(Z_p(j\omega)\right)\right)\Big|_{\omega\to\omega max} \tag{3.7}$$

Where the parallel group impedance *Zp* is:

$$Z_p(j\omega) = Z_{meas}(j\omega) - Rs \tag{3.8}$$

The values for *L* and *C* are obtained starting from the parallel group impedance and the resonance frequency:

$$L(\omega) = \frac{\left(\frac{\omega}{\omega_c}\right)^2 - 1}{Im\left(\frac{1}{Zp(j\omega)}\right)\cdot\omega};\ \ \omega \to \omega_c \tag{3.9}$$

$$C(\omega) = \frac{1}{L(\omega)\cdot\omega_c^2};\ \omega \to \omega_c \tag{3.10}$$

The proposed algorithm extracts directly the *Rs* and *R* from the measured data and varies the frequency around the resonance point in order to compute *L* and *C* values. This approach permits the optimal fitting of the transistor level measured impedance.

Figure 3.4 presents an example of a parallel RC case. The model components can be computed as follow: in DC, the resistive effect permits the evaluation of *R*:

$$R = Re(Z_{meas}(j\omega))|_{\omega=0} \tag{3.11}$$

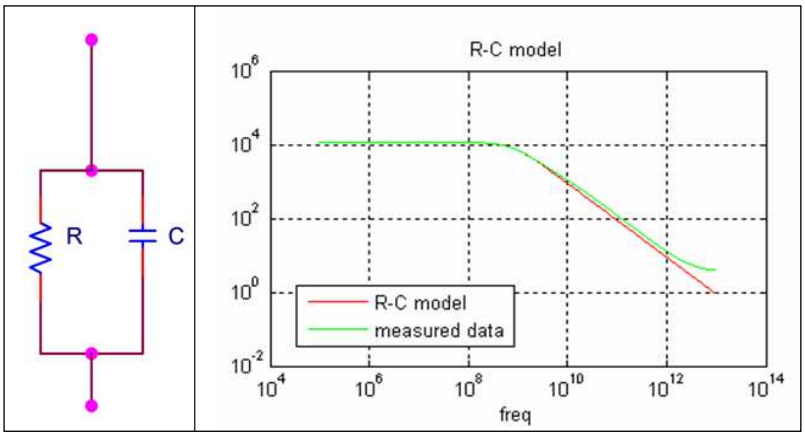


Figure 3.4 RC parallel impedance model, transistor-level simulation and extracted data

When considering the impedance values, the capacitance value *C* can be expressed as:

$$C(\omega) = \frac{Im\left(\frac{1}{Z_{meas}(j\omega)} - \frac{1}{R}\right)}{\omega} \tag{3.12}$$

When considering the central frequency, we have:

$$C(\omega) = \frac{1}{Re(Z_{meas}(j\omega))\cdot\omega};\ \omega \to \omega_c \tag{3.13}$$

The algorithm extracts directly the *R* value from measured data and approximates *C* values in both ways in order to optimally fit the measured impedance.

3.4.4. Extraction of the DC Nonlinear Effects

Reconsidering the unipolar CCVS model type, the general equations for the output ports including linear and nonlinear DC behavior can be expressed by a generalization of 3.1 as:

$$V_{out} = \underbrace{V_{out0} + g^d I_{in} + Z_{out} I_{out}}_{linear} + \underbrace{\sum_{k=2}^{K} c_{k0} I_{in}^k + \sum_{m=2}^{M} c_{0m} I_{out}^m + \sum_{n,p=1}^{N} c_{np} I_{in}^n I_{out}^p}_{nonlinear} \tag{3.14}$$

$$V_{in} = \underbrace{V_{in0} + g^r I_{out} + Z_{in} I_{in}}_{linear} + \underbrace{\sum_{k=2}^{K} c_{k0}^r I_{out}^k + \sum_{m=2}^{M} c_{0m}^r I_{in}^m + \sum_{n,p=1}^{N} c_{np}^r I_{out}^n I_{in}^p}_{nonlinear} \tag{3.15}$$

Where V_{out0} and I_{in0} are the corresponding DC offsets measured on the circuit input and output effects, g^d and g^r are the DC direct and the reverse gains (corresponding in AC to the direct and reverse transfer functions), while $Z_{in/out}$ /$Y_{in/out}$ will be the impedances/admittances corresponding to the quantities which are implied on the circuit input and output. The gains and

impedances/admittances are computed using both the simple DC extractions and the parametric DC interpolation, for coherence.

The $c_{ij}^{(r)}$ factors are the DC nonlinearity coefficients, extracted by an LMS approximation of the response area obtained from DC (on the input) parametric (on the output) analyses. The nonlinearity orders are considered as follows: *K* is the maximum nonlinearity order for the direct path, *M* is the maximum nonlinearity order for the reverse path and *N* the maximum intermodulation order between the input and the output causes.

The algorithm for surface least-square approximation of the DC parametric response and offsets/gains/nonlinearity extraction is presented for a 3rd order [89].

Considering x_s the *n*-size vector containing all the DC input variable values (values of I_{in} at each analysis step) and y_s the *m*-size vector containing all the DC parameter values (values of I_{out} at each analysis step), we have the two mean values and simple variances:

$$\begin{cases} x_0 = \overline{x_s} \\ y_0 = \overline{y_s} \end{cases} \tag{3.16}$$

$$\begin{cases} \Delta x = max(x_s) - x_0 \\ \Delta y = max(y_s) - y_0 \end{cases} \tag{3.17}$$

Following, the order 2, order 4 and order 6 means can be computed as:

$$\begin{cases} Mx_2 = \overline{(x_s - x_0)^{\langle 2 \rangle}} \\ My_2 = \overline{(y_s - y_0)^{\langle 2 \rangle}} \end{cases} \tag{3.18}$$

$$\begin{cases} Mx_4 = \overline{(x_s - x_0)^{\langle 4 \rangle}} \\ My_4 = \overline{(y_s - y_0)^{\langle 4 \rangle}} \end{cases} \tag{3.19}$$

$$\begin{cases} Mx_6 = \overline{(x_s - x_0)^{\langle 6 \rangle}} \\ My_6 = \overline{(y_s - y_0)^{\langle 6 \rangle}} \end{cases} \tag{3.20}$$

Where $(a)^{\langle b \rangle}$ is the element wise *b* power calculus for elements of vector *a*;
The inter-correlated mean will be:

$$Mx_2\,y_2 = \overline{\overline{(y_s - y_0)^{\langle 2 \rangle} \times (x_s - x_0)^{\langle 2 \rangle}}} \tag{3.21}$$

The response surface will be assimilated with the *m*-by-*n* z matrix containing all the response values (corresponding to all the measured values of V_{out} in our case). Its mean value will be:

$$Mm = \overline{\overline{z}} \tag{3.22}$$

Next we have the partial derivative coefficients:

$$Mxm = \overline{\overline{[[1]_{ys} \times (x_s - x_0)] \otimes z}} \tag{3.23}$$

Where $[1]_{ys}$ is the identity matrix of size *m* x *m, m* being the size of the y_s vector and $\otimes$ denotes the element-wise matrix multiplication;

$$Mym = \overline{\overline{[(y_s - y_0) \times [1]_{xs}] \otimes z}} \tag{3.24}$$

Where $[1]_{xs}$ is the identity matrix of size *n* x *n*, *n* being the size of the x_s vector;

$$Mxym = \overline{\overline{[(y_s - y_0) \times (x_s - x_0)] \otimes z}} \tag{3.25}$$

The 2nd order partial derivative coefficients:

$$Mx_2m = \overline{\overline{\left[[1]_{ys} \times (x_s - x_0)^{\langle 2 \rangle}\right] \otimes z}} \tag{3.26}$$

$$My_2m = \overline{\overline{[(y_s - y_0)^{\langle 2 \rangle} \times [1]_{xs}] \otimes z}} \tag{3.27}$$

The 3rd order partial derivative coefficients:

$$Mx_3m = \overline{\overline{\left[[1]_{ys} \times (x_s - x_0)^{\langle 3 \rangle}\right] \otimes z}} \tag{3.28}$$

$$My_3m = \overline{\overline{[(y_s - y_0)^{\langle 3 \rangle} \times [1]_{xs}] \otimes z}} \tag{3.29}$$

Following these vectors can be calculated:

$$r^1 = [Mx_2m \quad Mm \quad My_2m] \times \begin{bmatrix} Mx_2 & 1 & My_2 \\ Mx_4 & Mx_2 & Mx_2y_2 \\ Mx_2y_2 & My_2 & My_4 \end{bmatrix}^{-1} \tag{3.30}$$

$$r^2 = [Mxm \quad Mx_3m] \times \begin{bmatrix} Mx_2 & Mx_4 \\ Mx_4 & Mx_6 \end{bmatrix}^{-1} \tag{3.31}$$

$$r^3 = [Mym \quad My_3m] \times \begin{bmatrix} My_2 & My_4 \\ My_4 & My_6 \end{bmatrix}^{-1} \tag{3.32}$$

From which result directly the following seven de-normalized coefficients:

$$c_0 = r_1^1 \tag{3.33}$$

$$cx_1 = r_1^2; cx_2 = r_2^1; cx_3 = r_2^2 \tag{3.34}$$

$$cy_1 = r_1^3; cy_2 = r_3^1; cy_3 = r_2^3 \tag{3.35}$$

And an inter-modulation coefficient:

$$cxy = \frac{Mxym}{Mx_2 \cdot My_2} \tag{3.36}$$

The real response surface can be fitted by a nonlinear surface which includes all the linear and nonlinear contributions until 3rd order:

$$p = c_0 + cx_1 \cdot \left([1]_{ys} \times (x_s - x_0)\right) + cy_1 \cdot ((y_s - y_0) \times [1]_{xs}) + cx_2 \cdot \left([1]_{ys} \times (x_s - x_0)^{\langle 2 \rangle}\right) + cy_2 \cdot \left((y_s - y_0)^{\langle 2 \rangle} \times [1]_{xs}\right) + cx_3 \cdot \left([1]_{ys} \times (x_s - x_0)^{\langle 3 \rangle}\right) + cy_3 \cdot \left((y_s - y_0)^{\langle 3 \rangle} \times [1]_{xs}\right) + cxy \cdot \left((y_s - y_0) \times (x_s - x_0)\right) \quad (3.37)$$

To evaluate different orders of nonlinearity the real response surface is fitted also by its linear approximation (the ideal response surface obtained by a linear interpolation in the median coordinates). In this case, the following coefficients are computed:

$$c_0^s = Mm \quad (3.38)$$

$$cx_1^s = \frac{Mxm}{Mx_2} \quad (3.39)$$

$$cy_1^s = \frac{Mym}{My_2} \quad (3.40)$$

The linear interpolation will be computed as follows:

$$p_{lin} = c_0^s + cx_1^s \cdot \left([1]_{ys} \times (x_s - x_0)\right) + cy_1^s \cdot ((y_s - y_0) \times [1]_{xs}) \quad (3.41)$$

Subtracting the nonlinear 3[rd] order approximation from the real response we have:

$err = z - p$ which represents the residual nonlinearity of order > 3;

Subtracting the linear approximation from the real response we have:

$err_{lin} = z - p_{lin}$ which represents the total response nonlinearity;

A graphical representation of *err* and err_{lin} is a very useful description of the total and residual design nonlinearity (the residual effect after the extraction of the above contributions), giving the possibility to evaluate and optimize all the aspects concerning nonlinear behavior. Figure 3.5 presents the two nonlinearities for the parametric DC response of a real amplifier. The characteristic is function of the input and output DC variation.

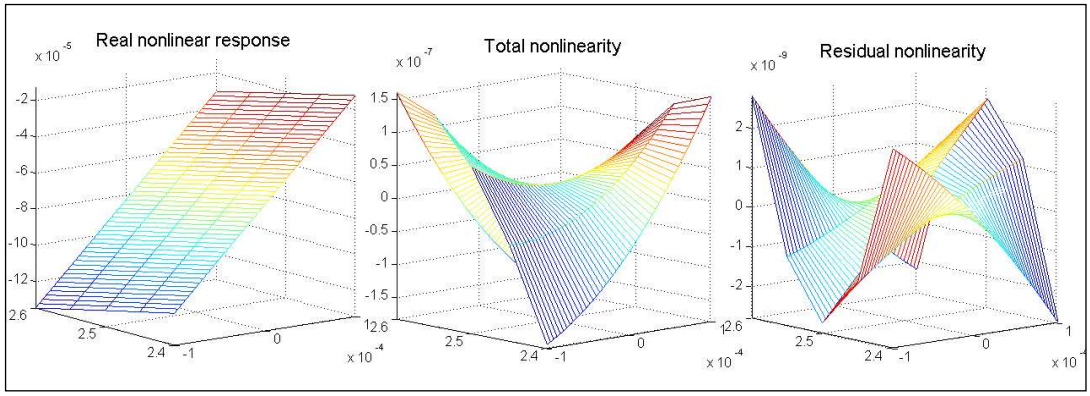


Figure 3.5 Initial response, total and residual nonlinearity after 3rd order substraction

Finally, for a common reference and non-dimensional value of the nonlinearity coefficients, they are be normalized with $gn \cdot \Delta x$ (gn=gain extracted from normal DC analysis).

The six nonlinearity coefficients will be:

$$cx_1^n = \frac{cx_1 \cdot \Delta x}{gn \cdot \Delta x} \; ; cy_1^n = \frac{cy_1 \cdot \Delta y}{gn \cdot \Delta x} \tag{3.42}$$

$$cx_2^n = \frac{cx_2 \cdot (\Delta x)^2}{gn \cdot \Delta x} \; ; cy_2^n = \frac{cy_2 \cdot (\Delta y)^2}{gn \cdot \Delta x} \tag{3.43}$$

$$cx_3^n = \frac{cx_3 \cdot (\Delta x)^3}{gn \cdot \Delta x} \; ; cy_3^n = \frac{cy_3 \cdot (\Delta y)^3}{gn \cdot \Delta x} \tag{3.44}$$

The normalized inter-modulation coefficient will be:

$$cxy^n = \frac{cxy \cdot \Delta x \cdot \Delta y}{gn \cdot \Delta x} \tag{3.45}$$

A total nonlinearity coefficient is computed in order to easily evaluate the nonlinear behavior of a studied design:

$$c_t = \sqrt{\overline{\overline{\overline{\overline{err \otimes err_{lin}}}}}} \tag{3.46}$$

The response offset (V_{out0}) is modeled by the c_0 coefficient extracted previously, which will not be normalized as it is dimensional (in *V* or *A*, depending on response type).

The proposed approach is applied to the direct path, but reverse nonlinear coefficients can also be computed by considering the output cause and the input effect.

In the case of linear and weakly nonlinear CT functions, the DC nonlinear effects can be limited to the most important contributions: nonlinearity coefficients up to order 3 and order 1 intermodulation coefficient. By performing this limitation, we obtained normalized residual nonlinearities in the interval [10^{-10}; 10^{-8}] for the real Sigma-Delta components which were studied. For the vast majority of functions, this approach will generate sufficiently accurate models while maintaining a low model complexity, in order not to affect simulation speed.

3.4.5. Extraction of the HF Weak Nonlinear Effects

The nonlinear effects in the frequency domain can be characterized by using a specific RF Harmonic Balance approach.

Harmonic balance is very accurate and very efficient if the circuit is near linear and the voltage and current waveforms are near sinusoidal. In fact, assuming the component models are correct, harmonic balance becomes exact in the limit where the circuit is weakly non-linear and the stimulus are sinusoidal [90].

In our case, because the models should simulate fast, the nonlinear contributions should be synthetized in a small number of coefficients, capturing the most important effects. A model order reduction technique will be of main interest [91].

Writing the time domain modified nodal analysis (MNA) formulation [92] [93] of the unipolar CCVS circuit for example we have:

$$g_{dir}x(t) + c_{dir}\dot{x}(t) + f_{dir}^{nlin}\big(x(t)\big) = r_{dir}u(t) + b_{dir}(t) \tag{3.47}$$

$$i(t) = r_{dir}^{T}x(t) \tag{3.48}$$

Where g_{dir}, c_{dir} are matrices that contain the direct contributions of memory-less and memory elements of the circuit, $x(t)$ is a vector which contains node voltages, independent voltage sources currents and initial capacitor charges, while f_{dir}^{nlin} is a vector used to model the nonlinear behavior at transistor level; r_{dir} is a selector table which correlates the voltages and currents of the input/output ports, *u(t)* and *i(t)* are the currents and voltages of the ports while b_{dir} contains the independent voltage and current sources values.

Now considering the first direct path as the main section of the circuit and all the other paths as subsections (e.g. the supplementary paths for differential designs and multi-output ones), the Harmonic Balance equations corresponding to 3.47 and 3.48 can be expressed in the frequency domain [94] as :

$$\begin{bmatrix} G_{dir} & D_1R_1^T & \cdots & D_nR_n^T \\ R_1D_1^T & G_1 & 0 & 0 \\ \vdots & 0 & \ddots & 0 \\ R_nD_n^T & 0 & 0 & G_n \end{bmatrix} \times \begin{bmatrix} X_{dir} \\ X_1 \\ \vdots \\ X_n \end{bmatrix} + \begin{bmatrix} C_{dir} & 0 & \cdots & 0 \\ 0 & C_1 & 0 & 0 \\ \vdots & 0 & \ddots & 0 \\ 0 & 0 & 0 & C_n \end{bmatrix} \times \begin{bmatrix} X_{dir} \\ X_1 \\ \vdots \\ X_n \end{bmatrix} + \begin{bmatrix} F_{dir}(X_{dir}) \\ F_1(X_1) \\ \vdots \\ F_n(X_n) \end{bmatrix} = \begin{bmatrix} B_{dir} \\ B_1 \\ \vdots \\ B_n \end{bmatrix} \tag{3.49}$$

The respective *dir* Fourier coefficients are associated with the main section while the *n* other equations characterize the subsections. D_i are selector tables that map the port voltages and currents of the subsections to the node space of the circuit.

The interesting indications which can be used for macro-modeling the nonlinearities in the frequency domain are the Fourier coefficients of the nonlinear effects contained in the matrices F_{dir}, $F_1 \ldots F_n$. These coefficients will be computed and used as follows:

- The overall analysis domain of frequencies will be partitioned in *k* intervals: $[f_0, f_1], [f_1, f_2], \cdots, [f_{k-1}, f_k]$;
- For each interval, single-tone single-source Harmonic Balance analyses will be performed on the input and output of the circuit for the central frequency of the interval $f_c^i = (f_i - f_{i-1})/2$. These will produce the F_{dir} coefficients and potentially $F_1 \ldots F_n$ when subsections are present which will characterize the input and output nonlinear effects;
- Then single-tone multi-source Harmonic Balance analyses (with one source on the input and the other on the output) will be performed for the same f_c^i. These will produce the F_{dir} coefficients and potentially $F_1 \ldots F_n$ which will characterize the intermodulation effects between the direct and reverse path of the circuit;
- Additionally, multi-tone single-source and multi-tone multi-source analyses can be performed between the frequencies f_{i-1}, f_i in order to capture the normal or crossed frequency intermodulation products.

The nonlinear effects characterization in frequency domain can be limited with satisfactory results to 5 nonlinear coefficients + 2 frequency intermodulation products coefficients for each interval resulting for example in the following matrix for the direct path of the circuit:

$$\begin{bmatrix} \phi_{20}^{nl,1} & \phi_{02}^{nl,1} & \phi_{30}^{nl,1} & \phi_{03}^{nl,1} & \phi_{11}^{nl,1} & \phi_{20}^{imod,1} & \phi_{02}^{imod,1} \\ \phi_{20}^{nl,2} & \phi_{02}^{nl,2} & \phi_{30}^{nl,2} & \phi_{03}^{nl,2} & \phi_{11}^{nl,2} & \phi_{20}^{imod,2} & \phi_{02}^{imod,2} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \phi_{20}^{nl,k} & \phi_{02}^{nl,k} & \phi_{30}^{nl,k} & \phi_{03}^{nl,k} & \phi_{11}^{nl,k} & \phi_{20}^{imod,k} & \phi_{02}^{imod,k} \end{bmatrix} \begin{matrix} \rightarrow [f_0, f_1] \\ \rightarrow [f_1, f_2] \\ \vdots \\ \rightarrow [f_{k-1}, f_k] \end{matrix} \quad (3.50)$$

The coefficients $\phi_{20}^{nl,i}$ and $\phi_{30}^{nl,i}$ are the second and the third harmonic of the input quantity contributing to the output, $\phi_{02}^{nl,i}$ and $\phi_{03}^{nl,i}$ are the second and the third harmonic of the complementary output quantity contributing to the output, $\phi_{11}^{nl,i}$ is the fundamental of the intermodulation between the two paths while $\phi_{20}^{\mathrm{imod},i}, \phi_{02}^{\mathrm{imod},i}$ are the contributions of the input and output respectively at the intermodulation frequency $f_{i\mathrm{mod}}^{i} = (f_i + f_{i-1})$.

We studied the effect of the proposed limitation of the number of coefficients to the most important contributions.

The simulation tools which were used for this purpose are: the Harmonic Balance engine from Agilent ADS® and the RF analyses (principally the Periodic Steady State – PSS and Periodic AC – PAC) provided by SpectreRF® from Cadence.

In the case of the real Sigma-Delta components which were studied we obtained a decreasing factor in the interval [40; 60] dBc between the fundamental and the second harmonic and [50; 80] dBc between the fundamental and the third harmonic.

The others harmonics are further decreased, while the intermodulation effects will be even smaller.

This confirms the validity of nonlinear coefficients number limitation and assures that the macro-models which will be synthetized using this approach will be robust.

As an example, in Figure 3.6 the effects on the direct path of a real current-controlled-current-source are presented, when performing a single tone 5th order Harmonic Balance analysis under Agilent ADS®.

We applied the excitation on the input with a power source of maximum 10 dBm and extracted the nonlinearity on the output with an impedance Z_{load} = 50Ω. The fundamental frequency is *100MHz*.

We can observe that the magnitudes of the second and third harmonics have already decreased with 47dBc and 67dBc when compared to the fundamental.

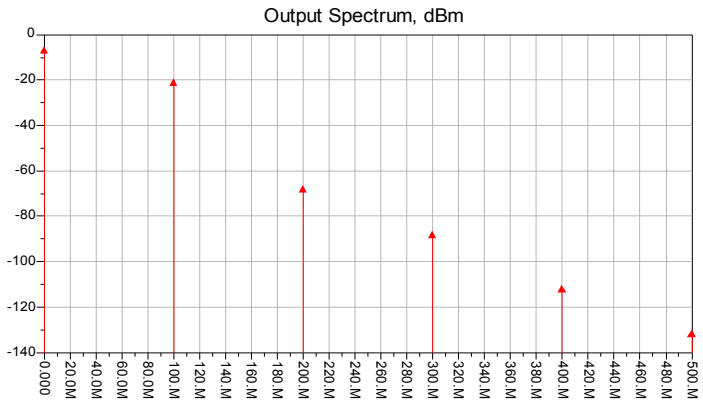


Figure 3.6 Output spectrum, 5th order Harmonic Balance (Agilent ADS) – current-to-current converter

The SpectreRF® simulation option from Cadence adds some important capabilities to the Spectre circuit simulator that are particularly useful to analog and RF designers [95]. In order to extract the nonlinear frequency effects using SpectreRF®, a periodic large signal analysis and a periodic small signal analysis should be used. The first step is to compute the periodic steady-state response of the circuit, by using for example the Periodic Steady-State (PSS) analysis. Next, a small signal analysis as Periodic AC (PAC) is used in order to observe the nonlinear effects on a predefined number of harmonics [96].

SpectreRF® provides also another class of RF analyses, known as the shooting methods. This feature is interesting when trying to exactly determine the distortion of low distortion amplifiers and filters. Still, it is a complex task to extract efficiently a small number of nonlinearity coefficients from the shooting analysis.

For a comprehensive presentation of the RF analyses and a comparison of the different nonlinearity extractions methods in RF the work [90] should be referred.

3.5. Macro-models Synthesis

3.5.1. Linear Models Synthesis

Once all the figures of merit are extracted or computed for a specific circuit, the design methodology permits to automatically synthetize Simulink® macro-models and VHDL-AMS or Verilog-A behavioral models.

The linear model synthesis for Simulink® resides on 16 pre-coded templates corresponding to the 16 macro-models structures (see Section 3.4.1), allowing the extraction of all combinations of unipolar/differential circuits with voltages or currents on the input and output.

A unipolar Simulink® macro-model example is presented in Figure 3.7. It is the model for the CCVS circuit which was previously discussed.

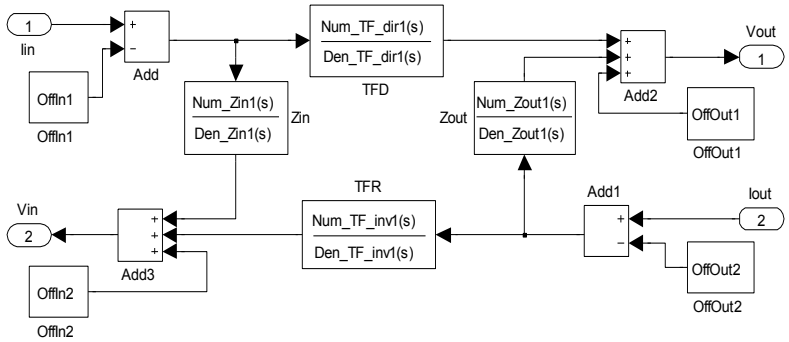


Figure 3.7 Simulink® linear macro-model of type CCVS

The poles and zeros of the *s*-functions and all offsets are injected into the templates as MATLAB® variables along with the simulation engine settings.

For the differential structures, the synthesis concepts are the same, but multiple paths are added between the input and the output ports. The conception examples provided in Section 3.7 and Chapter 5 will clarify the above considerations.

The VHDL-AMS behavioral architectures [80] are synthetized by using an inline approach were each component is selected function of the circuit topology.

The following entity components, the data transfers and principal functions are used:

- Input and output entity ports with the established sense convention are used for all the input/output quantities, e.g.:

```
PORT    (TERMINAL Diff_In_PORT_1, Diff_In_PORT_2, Diff_Out_PORT1, Diff_Out_PORT2:
ELECTRICAL);
```

- The *s*-functions numerators and denominators are exported as vectors of real coefficients, e.g.:

```
CONSTANT Num_TF_dir1 : REAL_VECTOR := (-2.964843e-03, 1.815268e+09, -
2.478997e+20);
CONSTANT Den_TF_dir1 : REAL_VECTOR := (1.000000e+00, 4.6192672e+10,
5.015811e+20);
```

- Other values (e.g. extracted DC offsets, user-defined constants, fixed components values, etc.) can be declared as real constants, e.g.:

```
CONSTANT Out_1_OffOut11 : REAL := 1.926695e-04;
CONSTANT R_Comp : REAL := 2.500000e+03;
```

- Intermediary electrical values are used by declaring their electrical nature as quantities, e.g.:

```
QUANTITY deltaIin1, deltaIin2: current;
QUANTITY deltaVout1, deltaVout2: voltage;
```

- Voltages and currents are associated to the ports using the specific concepts *across* and *through*, e.g.:

```
QUANTITY Vin1 ACROSS Iin1 THROUGH Diff_In_PORT_1 TO ground;
QUANTITY Vout1 ACROSS Iout1 THROUGH Diff_Out_PORT1 TO ground;
```

- A linear model behavior will use the Laplace formalism to find output signals starting from inputs, e.g.:

```
Iout1 == deltaIin1'LTF(Num_TF_dir1, Den_TF_dir1) + deltaIin2'LTF(Num_TF_dir3,
Den_TF_dir3) + deltaVout1'LTF(Den_Zout1, Num_Zout1) + OffOut11;

Vin2 == deltaIin2'LTF(Num_Zin3, Den_Zin3) + deltaIin1'LTF(Num_Zin_diff1,
Den_Zin_diff1) + deltaVout1'LTF(Num_TF_inv1, Den_TF_inv1) +
deltaVout2'LTF(Num_TF_inv4, Den_TF_inv4) + OffIn22;
```

If the VHDL-AMS behavioral models are intended to be simulated with the Cadence AMS® simulator, due to the limitations of this simulator (see Subsection 3.3.2), the vectors containing the numerators and denominators of the transfer functions are divided in the composing elements, which will be declared as *REAL* constants and used in the Laplace functions in the explicated form.

For the VerilogA models an inline approach is also implemented in SIMECT, where the following elements [83] are used for model generation:

- The circuit inputs and outputs are declared as electrical terminals, e.g.:

```
input iinp;
electrical iinp;
output iout0p;
electrical iout0p;
```

- The values of the poles/zeros and offsets are declared as real parameters, e.g.:

```
parameter real Num_TF_dir11  = -2.564606096184e-01 ;
parameter real Num_TF_dir12  = 1.524397274239e-12 ;
parameter real Den_TF_dir11  = 1.000000000000e+00 ;
parameter real Den_TF_dir12  = 9.882579238525e-11 ;
parameter real OffIn12 = 2.156916e+00;
```

- Specific voltages and currents should be mapped to the input and output ports of the circuit, e.g.:

```
Iin1 = I(iinp, gnd);
V(iinp, gnd) <+ Vin1 ;
Vout1 = V(iout0p, gnd);
I(iout0p, gnd) <+ Iout1;
```

- The behavior of the circuit is formulated using the Laplace formalism for transfer functions as in the case of the VHDL-AMS models, e.g.:

```
Iout1 = laplace_nd (deltaIin1, {Num_TF_dir11, Num_TF_dir12}, {Den_TF_dir11,
Den_TF_dir12}) + laplace_nd (deltaIin2, {Num_TF_dir31, Num_TF_dir32},
{Den_TF_dir31, Den_TF_dir32}) + laplace_nd (deltaVout1, {Den_Zout11, Den_Zout12},
{Num_Zout11, Num_Zout12}) + OffOut11;

Vin1 = laplace_nd (deltaIin1, {Num_Zin11, Num_Zin12}, {Den_Zin11, Den_Zin12}) +
laplace_nd (deltaIin2, {Num_Zin_diff11, Num_Zin_diff12}, {Den_Zin_diff11,
Den_Zin_diff12}) +laplace_nd (deltaVout1, {Num_TF_inv11, Num_TF_inv12},
{Den_TF_inv11, Den_TF_inv12}) + laplace_nd(deltaVout2, {Num_TF_inv21,
Num_TF_inv22}, {Den_TF_inv21, Den_TF_inv22})  + OffIn12;
```

Because the simulation speed is a key design factor when using high-level models to evaluate system level performances, some methods for speed optimization used for the extraction of HDL macro-models were implemented:

1. AC characteristics normalization – "working in baseband"

It is possible to specify in our methodology an *f0 normalization frequency* (e.g. the central resonance frequency for resonators or the central in-band frequency for band-pass filters) with which the measured AC characteristics will be normalized, resulting in the extraction of normalized models - with smaller singularities values; This approach is useful for early design explorations, when a large number of simulations are performed and the circuit can be studied independently of the real working frequency.

2. Decomposition of the transfer functions

For the VHDL-AMS and VerilogA models, it is possible to perform a decomposition of higher order transfer functions into a product series of order 0, 1 or maximum 2 transfer functions with real coefficients which can be rapidly evaluated. This technique is interesting as long as the *'LTF*, *'ZTF*, *'DOT* and *'INTEG* attributes can be cascaded.

A general transfer function of type 3.51 can be totally decomposed into a non-zero series of real and complex transfer functions of type 3.52.

$$H(s) = \frac{B(s)}{A(s)} = \frac{b(1)\cdot s^{nb-1} + b(2)\cdot s^{nb-2} + \cdots + b(nb)}{a(1)\cdot s^{na-1} + a(2)\cdot s^{na-2} + \cdots + a(na)} \tag{3.51}$$

$$H(s) = g\cdot\prod_i (s - z_i)\cdot\prod_j \frac{1}{(s - p_j)}\cdot\prod_k \frac{(s - z_k)}{(s - p_k)}\cdot\prod_n \frac{b_n^2\cdot s^2 + b_n^1\cdot s + b_n^0}{a_n^2\cdot s^2 + a_n^1\cdot s + a_n^0} \tag{3.52}$$

3. Elimination of the out-of-band singularities

When performing AC analyses for *s*-models extraction, the designer selects a minimum and a maximum AC frequency, defining a frequency band of interest. It is possible then to specify another option for HDL models in order to eliminate the out-of-band poles and zeros resulted from *s*-models extraction. Though interesting because it allows model order reduction, this option was used carefully, as long as it can change the dynamic behavior of the macro-models when compared to the transistor-level implementations.

4. Reverse function exclusion

The reverse transfer function implemented in the HDL models (as specified in Subsection 3.4.1) can be included or excluded from the model extraction, providing a simplification and a speed optimization of the subsequent analyses. In our case, the reverse function was excluded in simulations where highly anti-symmetric designs were studied, as long as the reverse effects are particularly small when compared to the direct ones. The reverse function can also be excluded when the output-input feedback is not interesting in system level simulations.

3.5.2. Non-linear Models Synthesis Considerations

The non-linear model synthesis resides on the extracted non-linear figures of merit presented in Subsections 3.4.4 and 3.4.5.

Indeed, to include the non-linear behavior, the linear macro-models presented before are completed with a suite of components which integrate the non-linear coefficients extracted from DC parametric analyses for the DC non-linear behavior or Harmonic Balance analyses for RF non-linear behavior.

The Simulink implementation of the DC nonlinear macro-models resides on the substitution of the normal adders composing the effects in the linear macro-models (e.g. the adders Add2 and Add3 in Figure 3.7) with nonlinear adders having the structure presented in Figure 3.8. The polynomial forms *P1* and *P2* include the extracted linear and non-linear simple coefficients, up to 3rd order. The intermodulation coefficient is included by performing a product between the two signal paths. These coefficients will be normalized with the respective gains or impedances/admittances on the input and output.

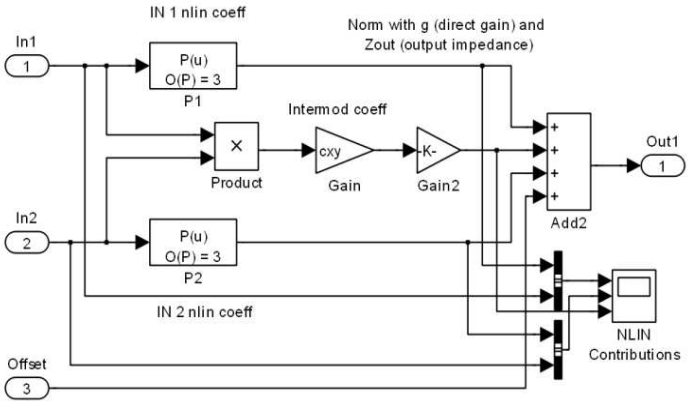


Figure 3.8 Nonlinear adder implementation in Simulink®

For the VHDL-AMS and Verilog-A, equations of type 3.14 and 3.15, limited to 3rd order contributions can be used for non-linear behavioral models synthesis.

In order to implement the RF non-linear effects extracted from Harmonic Balance analyses, a selector matrix of type 3.50 can be used, where the coefficients values in the

nonlinear adders will be dynamically updated with values corresponding to the frequency sub-band where the model is simulated. These matrices of coefficients along with the frequency intervals can be stored as matrix variables of type *REAL* directly into the Simulink® generated blocks or the VHDL-AMS/VerilogA behavioral architectures of the models.

At this time, the nonlinear models can be manually synthetized based on the figures of merit extracted by SIMECT and the above considerations. The implementation of the automatic synthesis algorithms for non-linear models extraction in SIMECT is a perspective of this work.

3.5.3. Multi-output Models Extensions

In addition to the differential nature of many amplifiers and filters, multi-output blocks are used in the design of mixed-signal ICs; therefore an additional option was considered in the design methodology in order to extract multi-output macro-models when needed.

The multi-output extension in SIMULINK is based on the decomposition of the circuit in subsections where the input is paired with each output. Then, for each pair a simple macro-model as in Figure 3.7 is extracted and finally all the contributions are recomposed in a single structure. Figure 3.9 presents the components generated for a real differential multi-output current mirror used in the first stage of the studied Sigma-Delta filter.

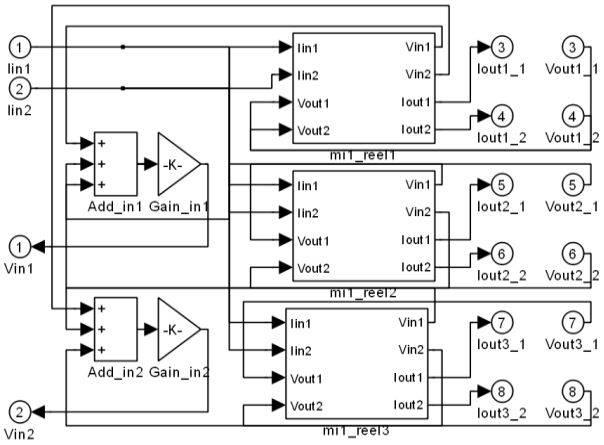


Figure 3.9 Structure of the SIMULINK macro-model for a multi-output amplifier

These multi-output models are generated as hierarchical structures in the design tool, SIMECT. The automatic placement and routing of the components and input/output ports was implemented allowing to synthetize models with an unlimited number of outputs which could be easily read and modified in the design process.

The VHDL-AMS and VerilogA models for the multi-output designs are directly obtained by a generalization of equations of type 3.14 and 3.15 where all the outputs contributions are considered. For each *quantity* and *constant* in the model, a prefix of type *Out_<number>* is added for each new output of the circuit.

3.6. Interface MATLAB® - Cadence IC Design Tools

For the task of macro-models parameters extraction and automatic synthesis of analog behavioral models, we developed an integrated design framework as an interface between MATLAB® and Cadence IC design tools. This framework is available as a MATLAB® toolbox, under the name SIMECT [70], protected by a GNU public license.

In our case, MATLAB® is used as a master tool where the user interface is implemented, which calls and accesses the analog simulator and all the other tools, and then reads the analyses results.

The design framework allows performing all kind of analog analyses directly from MATLAB®, to read the simulations results from Cadence generated files, to extract and calculate the figures of merit, the macro-models parameters and finally to synthesize Simulink®, VHDL-AMS and Verilog-A analog behavioral models.

The interface description will be summarized in the following sections. For an exhaustive presentation of the application, please consult the SIMECT user guide from [70].

3.6.1. Starting Analog Simulations from MATLAB

Analog simulations can easily be started in a batch mode using the Cadence Open Command Environment for Analysis (OCEAN) tool [73]. OCEAN is a text-based process that can be run from a UNIX shell. It can be used with any simulator integrated into the Virtuoso Analog Design Environment. In our case, the Cadence integrated Spectre simulator was used, but Spice, UltraSim or AMS are also available.

OCEAN works in an existing netlist directory. So the first step is to create the schematic of the circuit which has to be analyzed and modeled. All the sizes of components should be parameterized with specific names (e.g. w_i, l_i for MOS transistor sizes, R_i, L_i, C_i for passive components, and also user-defined variables) which will be used as MATLAB variables in the SIMECT framework. When the schematic is finished, the next step is to start the Analog Design Environment and create the simulation netlist. Once this step is completed, the CADENCE environment can be closed and SIMECT started.

The next operations are transparent for the designer as they are completely automated. We will present them and the architecture of the framework, insisting on the transactions between the tools in order to better understand the extraction algorithms presented afterwards.

The OCEAN tool is called from MATLAB using a batch file such as following:

```
unix 'ocean < sim.bat > ocean.log' ;
```

The batch file contains the names of the OCEAN scripts to be loaded and executed:

```
load( "init.ocn" )
load( "desvar.ocn" )
```

```
load( "an.ocn" )
```

These scripts will be created by SIMECT from MATLAB just before calling OCEAN. In order to develop the MATLAB functions which generate OCEAN scripts, we studied the OCEAN scripting language syntax [73], Cadence proprietary.

The first file contains the simulator name, the name of the design netlist, the results directory and the path to the components models.

The second file contains the schematic variables values such as:

```
desVar( "w1" 65u )
```

The third file contains the analyses to be performed. For example, a DC analysis for the circuit design variable *iin* varying from -0.001 to 0.001 will be performed using:

```
analysis('dc ?dev ""  ?param "iin"  ?start "-0.001"  ?stop "0.001"  )
```

MATLAB passes the command to OCEAN, which invokes the analog simulator, starting all the required transistor-level simulations.

The simulation runs automatically and writes the results of each analysis in a separate directory.

3.6.2. Getting Simulation Results in MATLAB

Once all simulations have run, the results can be imported into MATLAB®. SIMECT will perform automatically this action at the end of the simulations, when OCEAN re-passes the command to MATLAB®.

The MATLAB® functions required to read the simulations results are available within the Virtuoso Multi-Mode Simulation MMSIM Spectre/RF tools [97]. They are available as MEX compiled routines. The data can be imported by using the function *cds_srr* [98]. In order to read the results, the following syntax is used:

```
sig = cds_srr('res_dir', 'd_set_name', 'sig_name')
```

This function reads the simulation results for the signal *sig_name* from the simulation repository *res_dir* using the dataset name *d_set_name*. Usually, the dataset is compatible with the performed analysis type: *ac-ac*, *dc-dc*, *tran-tran*.

The function can be used also in order to find all the signals which were saved from a specific analysis with the following syntax:

```
[Sig_array] = cds_srr('res_dir', 'd_set_name')
```

Where the parameters are the same as above.

The overall interfacing between MATLAB® and Cadence IC design tools and the simulation process is shown in Figure 3.10. See also Annex A of this work for an example of OCEAN command files.

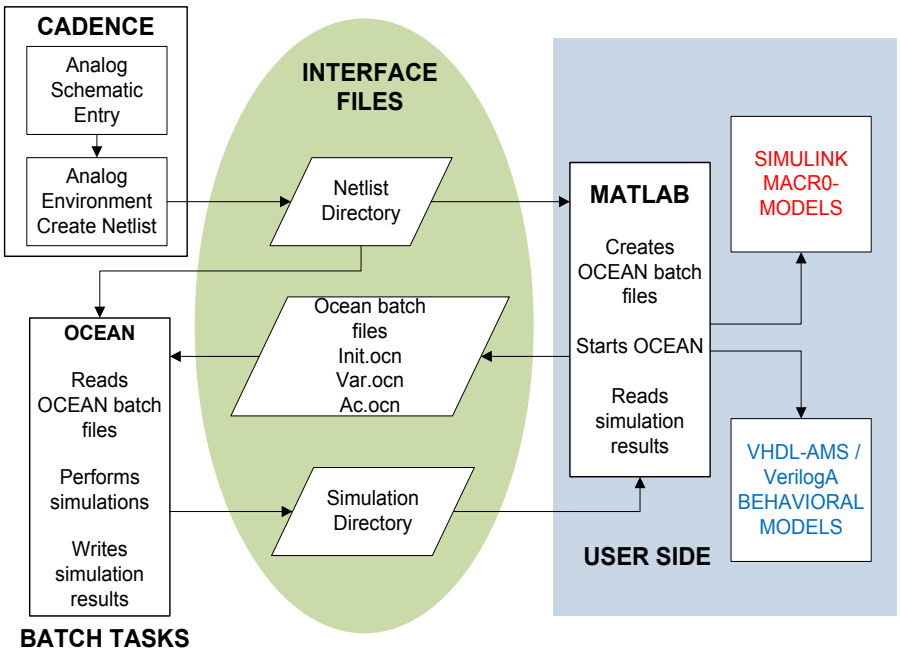


Figure 3.10 Macro-models extraction application architecture

3.6.3. Simulation Results Representation in MATLAB

All the simulations results are synthetically grouped in two figures, one for the DC (input) and Parametric DC (output) results and one for the AC and transient results.

Figure 3.11 presents as an example the results obtained from the DC analysis on input and Parametric DC analysis on output for a simple trans-impedance (TIA) unipolar amplifier. This amplifier was also presented in [99].

The following figures of merit extracted from DC and parametric analyses can be represented:

- The amplifier direct gain extracted from simple DC and parametric DC;
- The output impedance extracted from parametric DC on output;
- The input impedance extracted from simple DC on input;
- The inverse (output-input) gain extracted from parametric DC;
- All the input and output offsets, in our case the extracted ones are the voltage offsets when the currents are 0;
- All the non-linearity coefficients (for the output – on the top and for the input – on the bottom of the Figure 3.11): NLX (order 2), NLX (order 3), NLY (order 2), NLY (order 3), IMOD (intermodulation input-output), NLTOT - the maximum total non-linearity error for the whole input or output range;
- A 2D plot of the output voltage function of the input current for output current=0;
- A 2D plot of the input voltage function of the input current for output current=0;
- A 3D plot of the output DC voltage function of the input current and output current;

- A 3D plot of the input DC voltage function of the input current and output current;
- A 2D plot of the supply current (circuit consumption) function of the input current;
- A 2D where intermediary values (voltages, currents) can be verified function of the varied DC quantity on input;

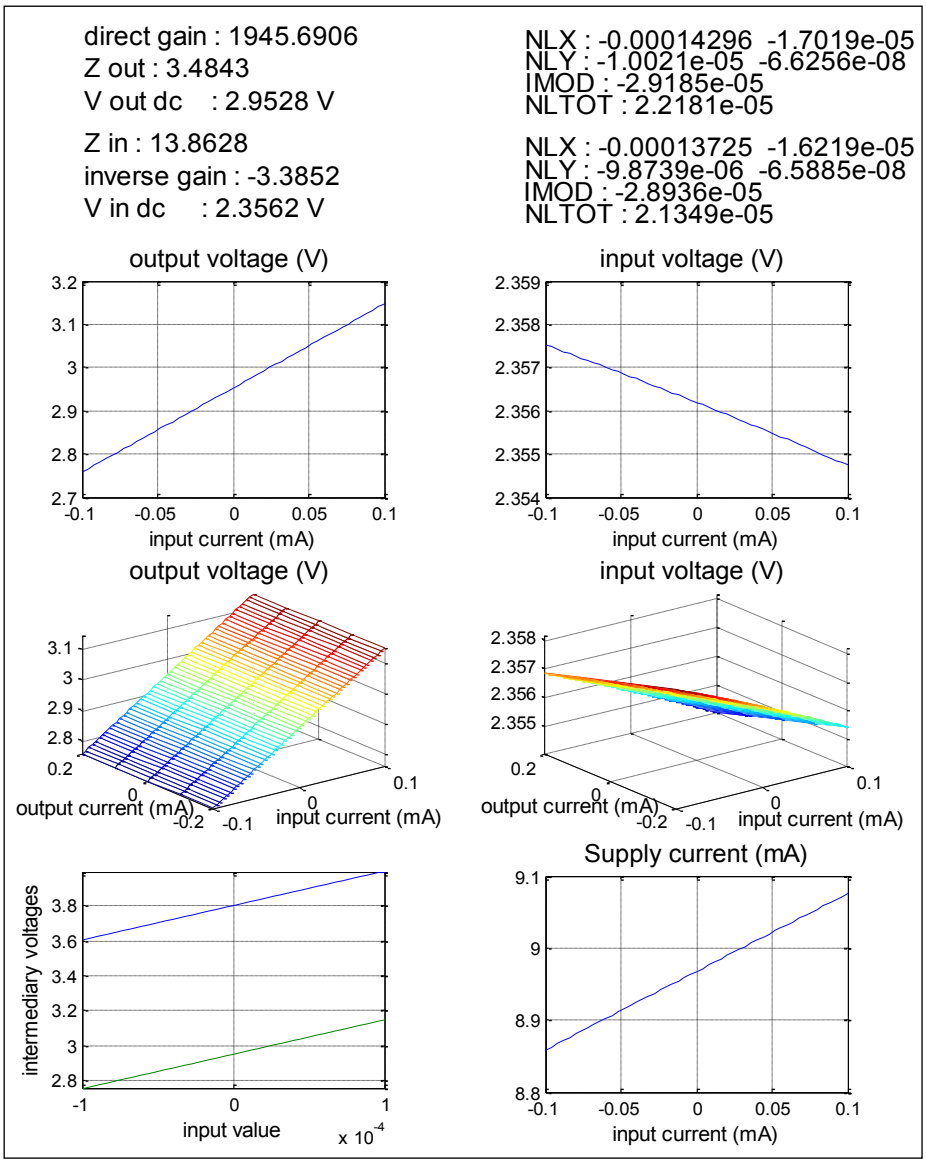


Figure 3.11 Input DC and Output DC analyses results example presented by SIMECT

Figure 3.12 details as an example the AC and transient results for the same TIA unipolar amplifier.

The following figures of merit extracted from AC and transient analyses are given:

- The resistive part of the input and output impedances at the frequency *f0*;
- The delay introduce by the circuit at the *f0* frequency (Δt);
- The transfer-function cutoff frequency (*FC Trans*);
- Input and output impedances equivalent RLC model parameters;
- AC plots of the measured input impedance, output impedance and transfer function;

- AC plots of the extracted s-models and/or RLC models for the three characteristics;
- The transient response (*out*) along with the generated transient input signal;

It was already discussed that the frequency characteristics can be normalized or de-normalized function of *f0*. In this example, the characteristics are normalized.

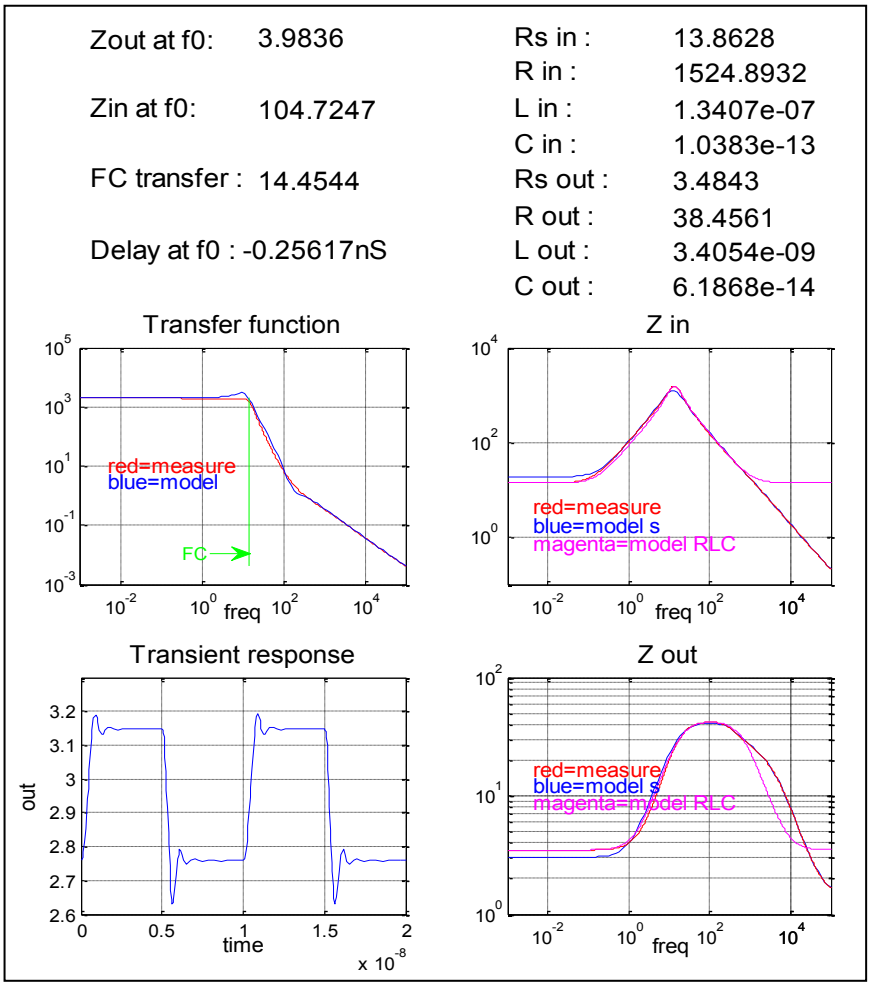


Figure 3.12 AC and Transient analyses results example presented by SIMECT

The results representation which was presented is generated in the case of simple, unipolar circuits. In the case of multi-output and/or differential designs, more complex figures of merit and indications are presented. This will be further discussed in Chapter 5 of this work.

It should be noted that in all cases, the denomination “measure” refers to the results of transistor-level simulations, which are to be distinguished from the extracted functions in the *s*-domain, referred as “model”.

3.7. Conception Example: Second Generation Current Conveyer (CCII)

Here we study the interfacing of a second generation current conveyer with an acoustic resonator (of type Lamb Wave Resonator - LWR [60]) and the possibilities to model the ensemble of two components by using a single high-level model. We show the full capability of the modeling approach which can be used even for multiple blocks synthesis and is not limited to a single circuit or cell.

The architecture of the current conveyer which was studied was published by B.Calvo et all. in [100] and is formally depicted in Figure 3.13. In our case, it was implemented using a STMicroelectronics 65 nm Low Power General Purpose CMOS technology process.

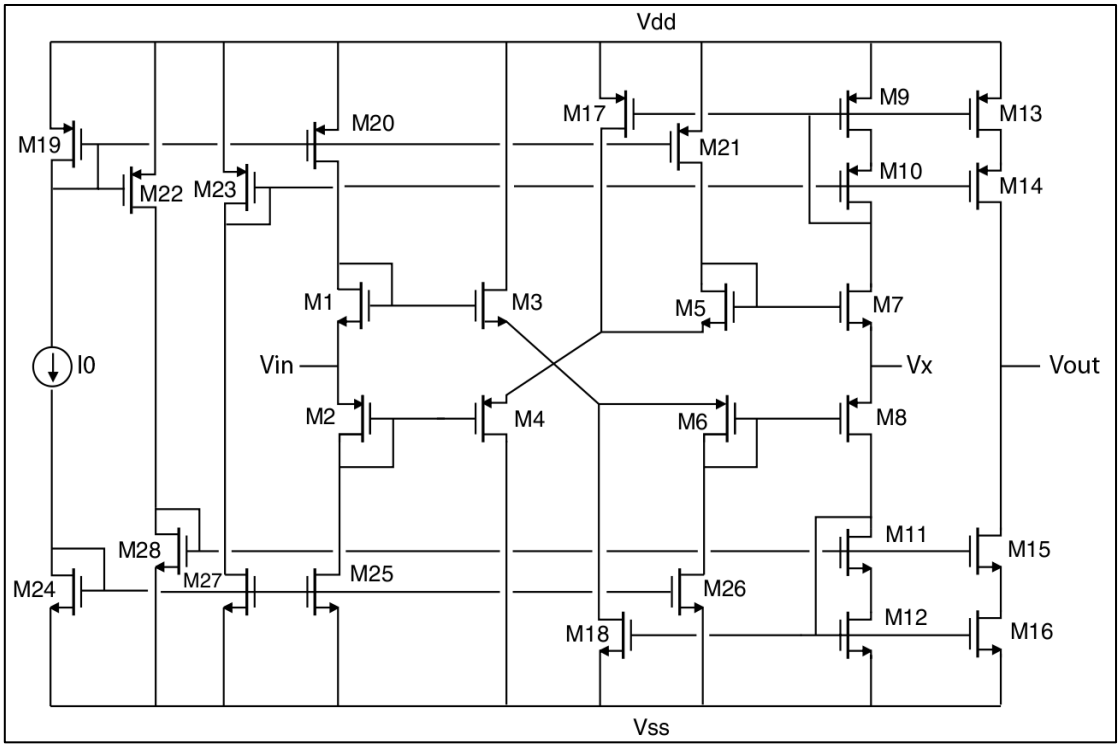


Figure 3.13 Second generation current conveyer

In our case, the current conveyer is used to copy the resulted resonance current from an acoustic resonator with the fundamental resonance centered at *fc*=2.5 GHz. This resonator requires a voltage as stimulus and responds with a current.

As long as this conveyor should be capable of correctly copying the output current of the resonator, the following design specifications are required for its sizing:

- The direct gain between *Vin* and *Vout* should be as close as possible to 1mA/V in the frequency band [0, 2.5+] GHz for a reference resistor connected to *Vx*, *R=100Ohm*. The phase would also be linear in this region;
- The input and output impedances (*Zin* and *Zout*) should be relatively high for the correct operation of the input voltage sink and output current source;

- The input current offset and output current offset (considered as circuit responses) should be close to 0;
- The non-linear effects (nonlinearities, intermodulation, etc.) should also be close to 0 (the most important being the output nonlinearity).

Starting from these design specifications, the conveyer was sized. Initially, the original components sizes were studied, as proposed by B.Calvo et all. in [101]. Then, the transistors sizes were re-optimized for a correct functionality in the STMicroelectronics 65 nm CMOS technology process. Each stage was considered first with ideal current sources and finally the real current mirrors were added.

The Table 3-1 presents the sizes obtained for all the transistors of the conveyer.

Transistors / Transistor pair	W [µm]	L [µm]
M1, M5	5.1	0.06
M2, M6	8.4	0.06
M3, M7	10.8	0.06
M4, M8	16.8	0.06
M9, M10, M13, M14	13.8	0.06
M11, M12, M15, M16	6.6	0.06
M17	8.4	0.06
M18	4.2	0.06
M19, M20, M21	15.6	0.12
M22	6	0.06
M23	0.54	0.06
M24, M25, M26	6.6	0.12
M27	1.38	0.06
M28	3.6	0.12

Table 3-1 Second generation current conveyer – transistors sizes

The resonator is implemented at transistor-level under Cadence Virtuoso Integrated Circuits Front to Back (ICFB) by using an ideal multi-resonance paradigm (Figure 3.14) based on the Butterworth-Van Dyke electrical model for micro-mechanical resonators [28]. In Chapter 5 we will further detail the modeling approaches for the micro-mechanical systems integrated with electronic systems.

The resonator is connected to the V_x input-output port of the conveyer, as depicted in Figure 3.15.

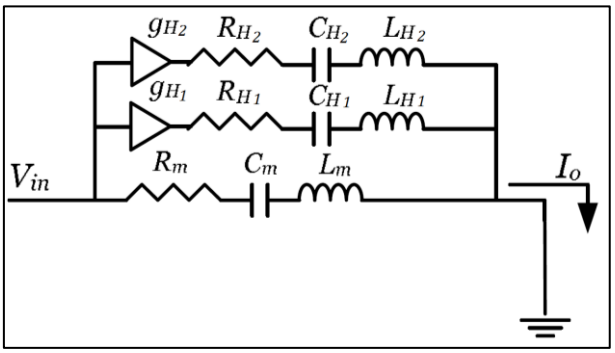


Figure 3.14 LWR acoustic resonator multi-resonant model

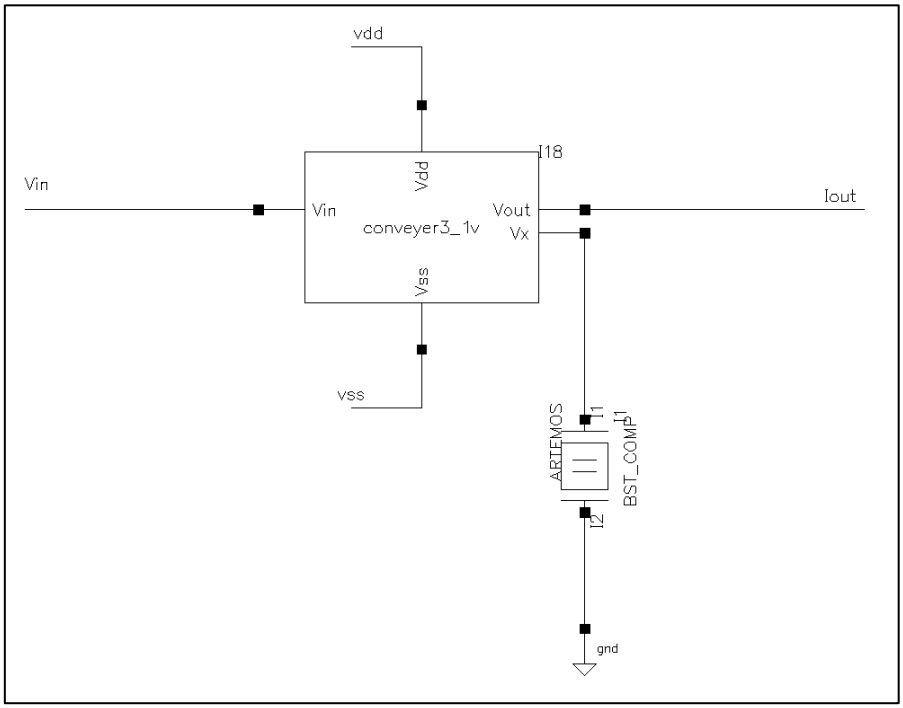


Figure 3.15 Current conveyer with acoustic resonator

For a correct DC analysis of the structure composed of the current conveyer and the acoustic resonator, we will replace in DC the resonator by its equivalent resistance (R_{ech}) at the principal resonance frequency (*fc*=2.5 GHz). This operation is analogous to the baseband translation of the band-pass spectrum for the acoustic resonator.

In this context, we will perform a simple DC analysis where the input voltage, *Vin*, varies in [-0.5; 0.5] V and afterwards a parametric analysis over the previous defined DC, with the output voltage, *Vout* varying in [-0.5; 0.5] V.

Then, for AC and transient analyses, the actual resonant model in Figure 3.14 is employed. The AC analyses are performed in the frequency domain [10MHz; 1THz] from which we will extract *s*-models of maximum orders 7 for all the frequency functions. A transient analysis over 2 periods of *fc*=2.5GHz is performed by using a sine input signal.

Because the design is practically unipolar on the input and on the output, when running the simulation the results are presented in two figures, one for the DC and Parametric analyses results (Figure 3.16) and one for the AC and transient analyses results (Figure 3.17).

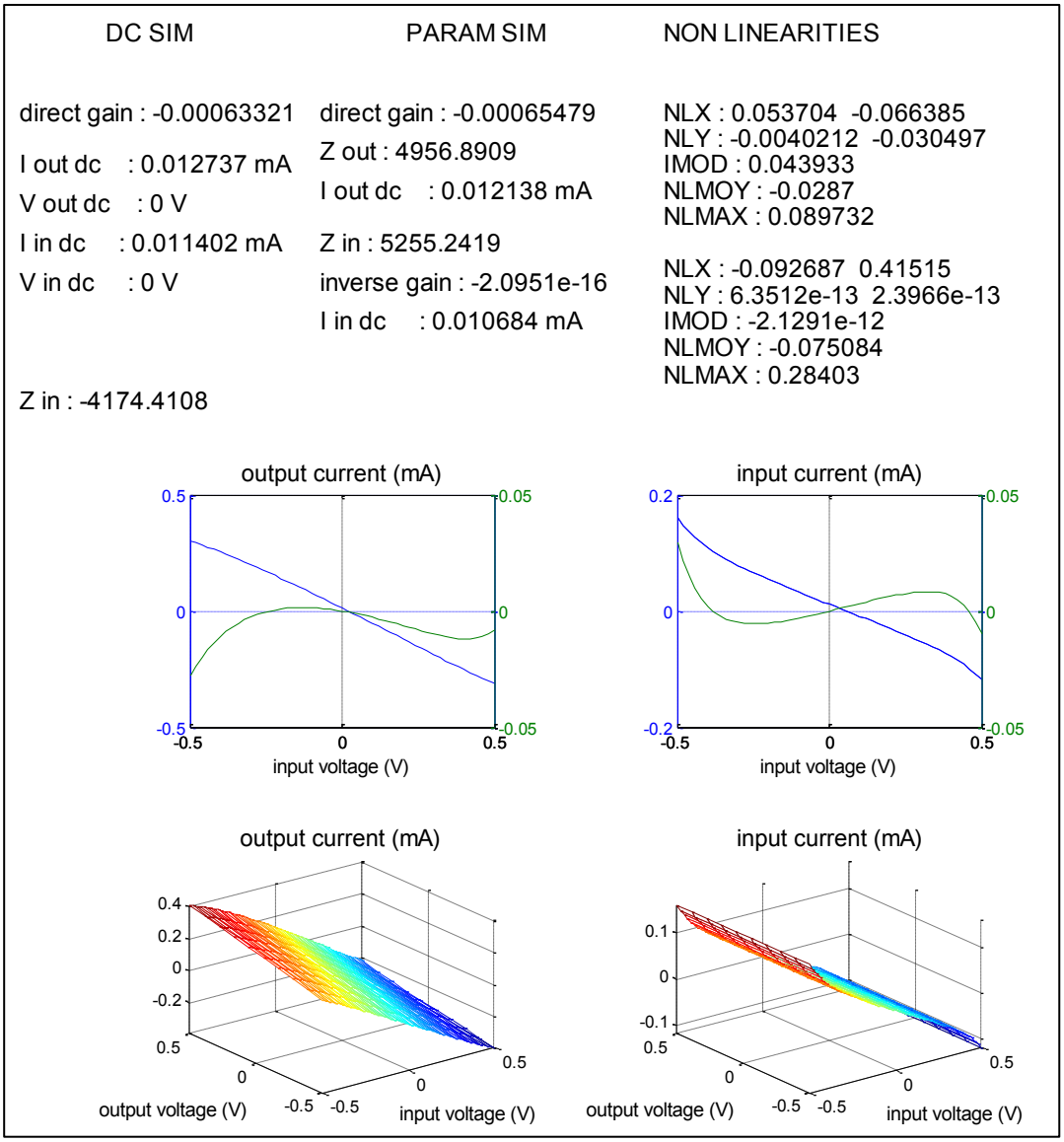


Figure 3.16 Current conveyer + R_{ech}: DC in and DC out responses and parameters

The DC gain of this conveyer (g_d=-0.63 mS) reflects the voltage-to-current conversion from the *Vin* input to the *Vout* output in the presence of the equivalent resistance (R_{ech}) on the *Vx* port. A good output linearity is assured, while the minus sign on the gains reflects the signal phase inversion from input to output. The extracted DC nonlinear effects are highly sensitive in this case, due to the small DC gains of the conveyer.

The input and output DC impedances (purely resistive in DC) are high enough for the correct conveyer operation (Z_{in}, Z_{out} ~ 5kΩ) and accurately extracted from Parametric DC analyses, while the simple DC extraction gives a rough approximation of the input impedance. The minus sign on the DC extraction suggests a phase inversion between the input voltage and

the resulted input current, while the Parametric DC extraction retains only the modulus of the impedance.

The input and output current offsets are also computed, resulting in the characterization of the offset imperfections for the current conveyer.

Apart from the DC characteristics, the conveyer should be able to correctly copy the high-frequency resonance of the LWR and potentially all the other parasitic resonances. This is inspected starting from the AC results.

As we can see in Figure 3.17, the extracted model for the transfer function is a good approximation of the real filter response, though the model order is considerably high (order 7).

The resonance effect at *fc*=2.5 GHz is present and the principal imperfections around the central frequency are correctly taken into account.

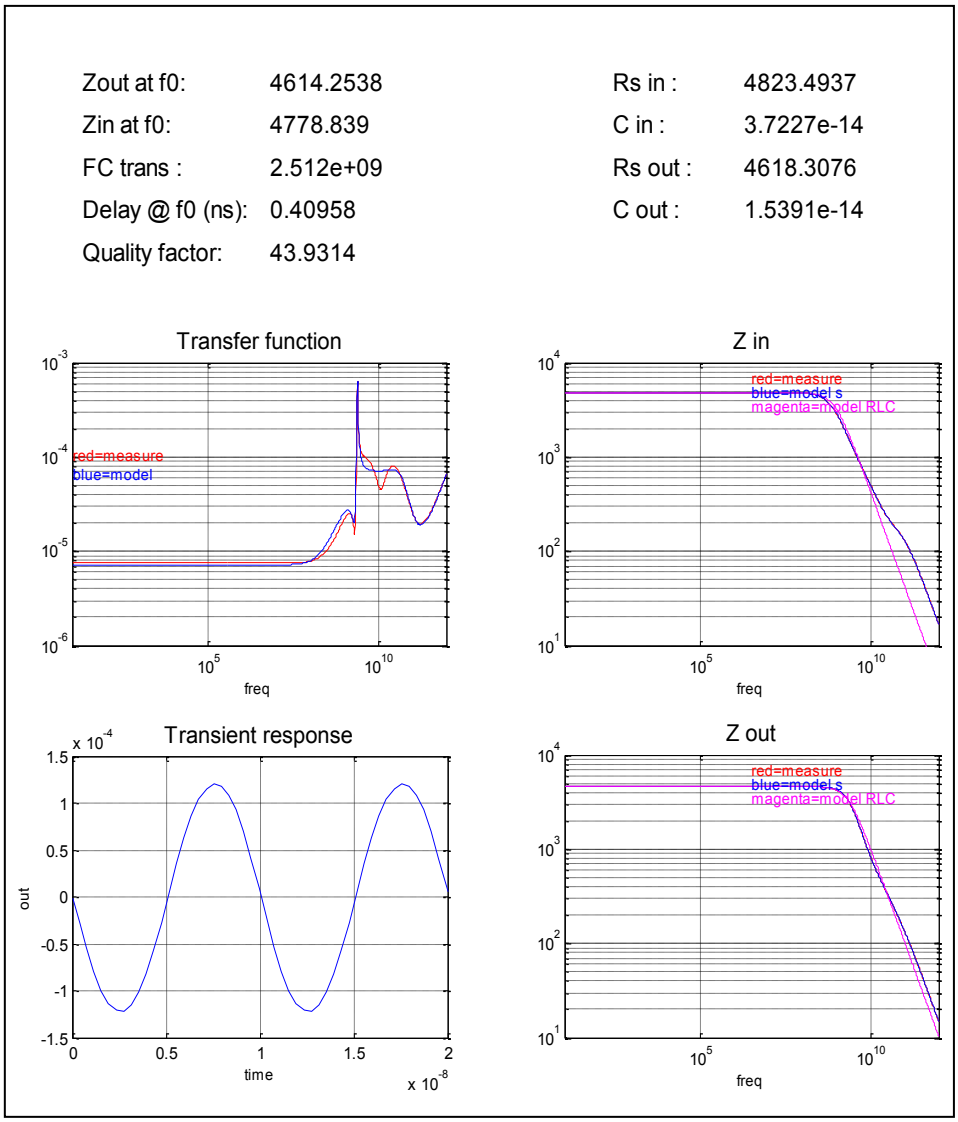


Figure 3.17 Current conveyer + LWR: AC and transient responses and parameters

The model topology as *s*-functions is inspected in Figure 3.18. This includes the *s*-models for the input impedance, the output impedance, the direct transfer function and the reverse transfer function.

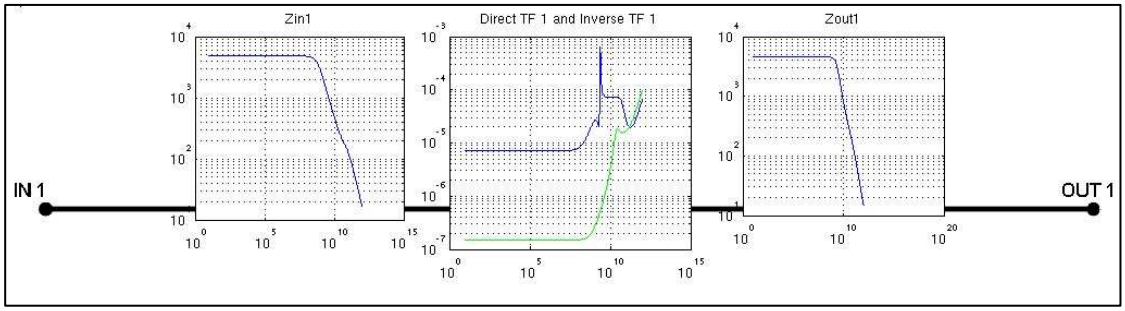


Figure 3.18 Current conveyer + LWR: Macro-model topology as *s*-functions

Once all the figures of merit and interesting characteristics are extracted or computed from the transistor-level simulations, the SIMULINK and HDL models of the complete structure composed from the current conveyer and LWR are extracted. The SIMULINK model will have the unipolar structure illustrated in Figure 3.19.

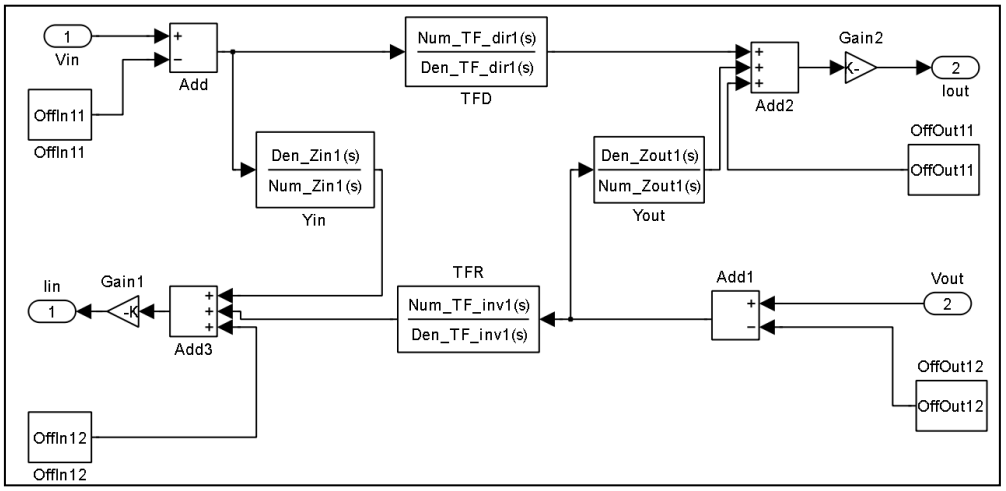


Figure 3.19 Current conveyer + LWR: Simulink macro-model

The HDL behavioral models for the second generation current conveyer (CCII) coupled with the acoustic resonator can be extracted using the proposed methodology.

A maximum 7th order VHDL-AMS linear model used for simulations with Dolphin Integration SMASH, a VHDL-AMS model for the Cadence AMS simulator and a Verilog-A model used with Cadence AMS simulator are presented in Annex C of this work.

They are automatically synthetized starting from the previous extracted and computed figures of merit.

The Chapter 5 of this work will propose more conception examples, where these modeling techniques are used along with the optimization methods presented in Chapter 4.

3.8. Chapter Conclusion

This chapter presents the first fundamental step of the novel analog design methodology presented in Chapter 2, e.g. the automatic extraction process of robust analog behavioral models or macro-models for amplifiers, resonators and filters.

The chapter started by a discussion on the current practices regarding the analog design modeling for large continuous-time functions and the existing strategies for this approach.

Then, a comprehensive review of the current modeling languages and simulation environments for the analog and mixed-signal macro-modeling permitted to select potential candidates for the implementation of the model extraction techniques. In this part, Mathworks MATLAB-Simlink environment and the VHDL-AMS and Verilog-A languages were retained. The main benefits and drawbacks for each selected option were retained to be used further in the model generation process.

Furthermore, a method for the extraction of the analog macro-models parameters from transistor-level analyses was studied. The DC, AC, parametric, transient and Harmonic Balance transistor-level simulations were identified as adequate to provide all kinds of parameters needed for macro-modeling the weakly non-linear analog functions.

The principal methods and algorithms for the extraction of *s*-models and RLC models for transfer functions and impedances, the extraction of the DC nonlinear effects and the extraction of the high-frequency weak nonlinear effects were synthetically presented.

Based on the extracted figures of merit, the model synthesis automatic procedure is presented: the case of linear models synthesis, the non-linear one and finally the multi-output extensions.

All these algorithms, procedures and results materialized in a conception tool, SIMECT which was implemented as a MATLAB toolbox, tested and extensively used in the conception and optimization of Sigma-Delta Modulators active filters.

Some elements are presented regarding the interface between MATLAB and Cadence IC tools used for starting analog simulations directly from MATLAB and to get the simulation results inside the tool. Naturally, the results representation form is also discussed briefly.

Finally, a conception example is proposed. To illustrate the capabilities of the design methodology and tool, a complex application is proposed: a second-generation current conveyer circuit coupled with a multi-resonance micro-mechanical resonator. A mature 65nm CMOS technology is used as target process.

This example is proposed with the sole purpose to derive a basic workflow for models synthesis, which will be detailed and applied in the complex design framework (Chapter 5).

IV

4. Bayesian Kriging Optimization of Analog Cells

4.1. Introduction

The process of optimization, also known on its historic name of “mathematical programming” represents a collection of mathematical methods and paradigms used for solving quantitative problems in many disciplines, including engineering, physics, biology, economics, and business.

The classical optimization problem consists of maximizing or minimizing a function (called objective function) by systematically choosing values from within an allowed input set and computing the result of the function. Generally, optimization tries to find the "best available" values of this objective function given a defined domain, where the variables can be subject to linear, nonlinear, or integer constraints [102].

Choosing an appropriate optimization strategy for a specific discipline is dependent of the problem type. This is due to the fact that an optimization algorithm cannot solve all types of problems and on average the performance of any pair of algorithms applied to all possible problems is identical [103].

The aim of our work is to establish a robust optimization method for automatic sizing of linear analog circuits which can be integrated with the macro-model extraction tools presented in Chapter 3.

In this context, several general approaches to optimization were identified [104] [105], which can be classified function of the input domain type:

- For continuous-type domains:
    a. Unconstrained optimization (the input variables have no associated constraints), which includes the following major algorithms:
        i. Global optimization

ii. Non-differentiable optimization
iii. Optimization based on non-linear equations
iv. Optimization based on non-linear least-squares

b. Constrained optimization (in this case some constraints are applied to the variables), represented by:
i. Linear programming
ii. Network programming
iii. Nonlinear constrained optimization
iv. Stochastic continuous programming

- For discrete-type domains:
c. Integer programming
d. Stochastic discrete programming

In a first step we will define the basic optimization problem of an analog cell, and then we will chose an appropriate optimization strategy which will be further refined according to the requirements of our general methodology for the design and optimization as presented in Chapter 2.

4.2. The Basic Optimization Problem of an Analog Cell

The optimization of electrical networks represents a classical problem. Many optimization tools were proposed and generally, with each generation of analog simulators, a new class of optimization methods was also included [106]. Still, automatic sizing of analog circuits continues to be a research goal for the EDA industry due to the heterogeneous designs and very specific circuit functions [107], [108], [109].

Three general directions were described in order to cast circuit sizing as an optimization problem [105], [110]:

5. *Knowledge-based approaches* – the first attempts to capture the knowledge of an analog designer and translate it into a set of rules which could be used to automatically size a circuit for a given set of performance specifications. The main drawback - their reusability: they are circuit-class specific and technology specific;
6. *Equation-based approaches* – in this case the circuit equations are automatically formulated by using some assumptions on the active devices (transistors, diodes) and used afterwards to optimize the requested figures of merit. Black-box optimization algorithms or convex optimization based on geometric programming are used. Their main advantage resides in the possibility to find an optimum solution with moderate resources, while the drawbacks are related to not-accurate circuit equations when compared to analog circuit simulators. In this case the technology limitations and imperfections are not considered.

7. *Simulation-based approaches* – reside on the solutions produced by an analog circuit simulator (e.g. SPICE, Cadence Spectre®) in order to optimize the requested figures of merit. Black-box optimization algorithms are used in this case. Compared to the previous directions, the simulation-based approaches capture the most realistic approximations of the circuits fabricated on silicon, but their timing performances are poorer, as analog simulations are time-consuming.

Our work is in the class of continuous-time (CT) linear functions including operational amplifiers (e.g. trans-impedance, trans-admittance, current mirrors, current conveyers) and analog passive and active filters. The resulted cells will be used in larger functions sensitive to analog technology imperfections and limitations (e.g. CT Sigma-Delta modulators) [28] so the most realistic models of devices should be used. In this case, simulation-based approaches are mandatory, potentially with improvements on the timing characteristics.

Each of the analog cells can be seen as a union of active devices (e.g. transistors, diodes) and passive devices (e.g. resistors, capacitors, inductors) with intrinsic or extrinsic variables. As intrinsic, we can define: transistor sizes, passive components values, values of static currents and voltages; as extrinsic – user defined variables which model a circuit behavior (e.g. common-mode vs. differential-mode gain).

In Figure 4.1 a generic analog cell is composed with n devices parameterized with their respective $x_i, y_i, \ldots, z_i$ design variables (intrinsic or extrinsic). In order to properly formulate the basic optimization problem, the white-box representation of the circuit and its black-box conversion are considered. The goals are to optimize one objective (f objective function) and potentially to respect a number of constraints (g_m constraints on the circuit input, h_n constraints on the circuit output, k_p general input-output constraints).

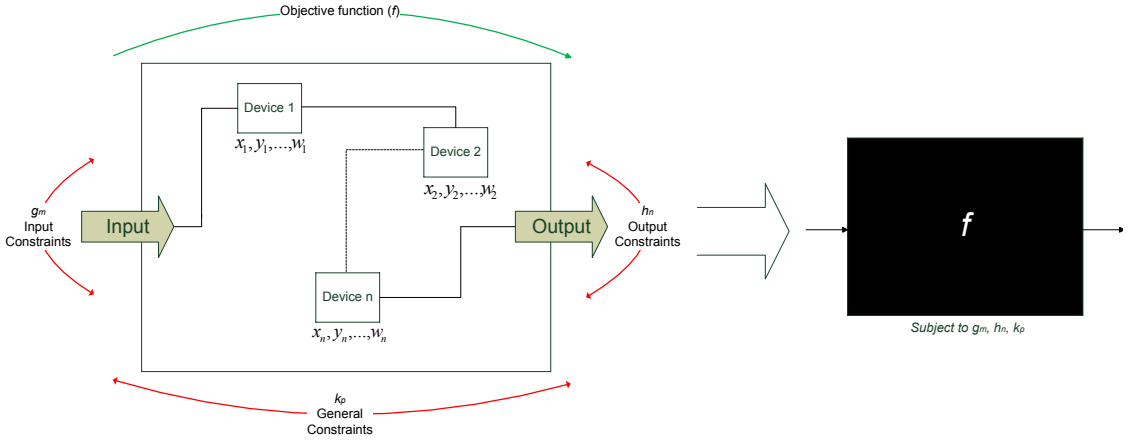


Figure 4.1 White-box and black-box representation of a generic analog cell

When constraints are present, the problem is said to be a *constrained optimization problem,* otherwise we have to solve an *unconstrained problem*. In the case of linear analog functions, we define the following classification for each constraint:

- *Strong constraint* – a constraint (e.g. extracted nonlinearity or distortion coefficients) which should be satisfied for all the *evaluation points* selected by the optimization algorithm (numerical values for all the sets of type $(x_i, y_i, \dots, z_i)$, producing a specific value for the objective and constraints);
- *Normal constraint* – a constraint (e.g. input/output impedance, offsets, quality factors) which is not mandatory to be satisfied for all the intermediary *evaluation points* but should be satisfied for the final result.

Considering the totality of design variables of the circuit, it is possible to define a point $\boldsymbol{X_k}$ of the multi-dimensional input domain as a realization of all the sets of type $(x_i, y_i, \dots, z_i)$:

$$\mathbf{X_k} = \bigcup_{\mathrm{i}} (\mathrm{x_i}, \mathrm{y_i}, \dots, \mathrm{w_i}), \quad \mathrm{i} = \overline{1\text{:}\,\mathrm{n}} \tag{4.1}$$

In the *unconstrained* case, a performance criterion P will be derived in terms of all the design variables:

$$P = f(\boldsymbol{X_k}) \tag{4.2}$$

In the *constrained* case, P will be also subject to:

$$\begin{aligned} a_m \le g_m(\boldsymbol{X_k}) \le b_m, \ \ (a_m, b_m) \in R^2 \\ \mathrm{a_n} \le \mathrm{h_n}(\mathbf{X_k}) \le \mathrm{b_n}, \ \ (\mathrm{a_n}, \mathrm{b_n}) \in \mathrm{R}^2 \\ a_p \le k_p(\boldsymbol{X_k}) \le b_p, \ \ (a_p, b_p) \in R^2 \end{aligned} \tag{4.3}$$

In a practical approach choosing the right performance criterion can be itself a difficult task. The most basic optimization problem is to adjust the design variables of the circuit in such a way as to minimize the quantity P. This is the *unconstrained problem* stated as:

$$\min_{\mathbf{X}} \mathrm{P} = \mathrm{f}(\mathbf{X_k}), \text{where } \mathbf{X_k}^{\mathrm{T}} = [(\mathrm{x_1 y_1} \dots \mathrm{w_1}), (\mathrm{x_2 y_2} \dots \mathrm{w_2}), \dots, (\mathrm{x_n y_n} \dots \mathrm{w_n})] \in \mathbb{R}^{\mathrm{n}} \tag{4.4}$$

The general problem can be focused on minimization without loss of generality, because the maximum of P can be readily obtained by finding the minimum of the negative of P and then changing the sign, since:

$$\max[\mathrm{f}(\mathbf{X_k})] = -\min[-\mathrm{f}(\mathbf{X_k})] \tag{4.5}$$

Still, for the vast majority of analog circuits, it is necessary to solve a *constrained optimization problem* because the optimization of only one performance criterion (e.g. power consumption) will rapidly degrade all others figures of interest (e.g. gains, impedances, bandwidths) [110].

4.3. Usage of Probabilistic Meta-models for Bayesian Optimization

4.3.1. Choice of the Probabilistic Meta-model

As indicated ahead a simulation-based optimization approach will be used as the analog functions are sensitive to technology imperfections and limitations. Analog simulations are highly time-consuming so an improvement on the timing characteristics should be considered.

Because the objective function and constraints are expensive to evaluate, we propose a Bayesian approach based on a probabilistic *meta-model* to spare a significant amount of evaluation time.

The general optimization strategy which is to be developed is based on the following generic directions, which will be detailed in the following sections:

- in a first stage, a limited number of transistor-level evaluations should be performed, resulting in exact circuit responses, and a method to extract the objectives and constraints for optimization from these simulations should be studied;
- then, a *meta-model* should be used to approximate the circuit response in the rest of the input space based on the exact responses of transistor-level evaluations;
- finally, a selection criterion is employed to interpret the information contained by the *meta-model* and to choose new candidate points, providing the convergence towards an optimum.

The probabilistic *meta-models* [111] or *surrogate functions* [112] can be defined as approximations of simulation input-output response functions. They have the major advantage of high computational efficiency when compared to the original response function. When using a *meta-model*, the function is evaluated in a limited number of points of the input space and for all others a probabilistic prediction is computed. This can lead for example to an efficient search of the optimum, given a restrictive evaluation budget.

In [113] the following *meta-models* types and their proprieties are identified:

- linear and quadratic polynomial – can be used only for local approximations in the case of stochastic responses;
- higher order polynomial – used for global approximation but not recommended (stochastic responses);
- nonlinear regression – used for phenomenon-specific global approximation in the case of stochastic responses;
- radial basis function – used for global approximation in the case of deterministic experiment responses;
- spatial correlation (Kriging) – can be used for global approximation in the case of deterministic or stochastic experiment responses;

- neural networks – also used for global approximation in the case of deterministic or stochastic responses;
- self-organizing maps (SOMs) which are useful for visualizing low-dimensional views of high-dimensional data [114];
- splines – also used for global approximation in the case of deterministic or stochastic responses;

In our case, of analog cells optimization problem, a global approximation is intended. Certain solutions based on neural networks [115], stochastic nonlinear regression [116] or genetic algorithms [117] can be used. But recent advances in Kriging metamodel-based optimization extend the existing modeling frameworks and suggested that this approach is the most adequate in the case of optimization problems with constraints, when a trade-off between speed and optimum position finding exists [118], [119]. Furthermore, the special class of constraints called *strong constraints* can be used in conjunction with the Kriging meta-models. Moreover, compared with the other methods, the Kriging computes the best linear unbiased estimator.

Kriging modeling combined with Latin Hypercube sampling (LHS) was already used in [120] to build surrogate models of analog circuit performances, indicating the need of less sample points and providing two times higher accuracy than classic quadratic response surface models. Kriging modeling of circuit performance can be utilized to estimate parametric yield. Besides, it may facilitate the global optimization of parametric yield or circuit performance.

As will be further discussed, a Bayesian approach based on a Kriging probabilistic surrogate model is recommended also to spare a significant amount of evaluation time. Regarding this criterion, the heuristic methods such as genetic algorithms [117], Tabu search, simulated annealing can be used but are known to generally require more evaluations [121].

In the next sub-sections, the Kriging modeling solution is further presented.

4.3.2. Kriging Surrogate Models

Kriging or spatial correlation represents a general method of statistical interpolation based on least squares estimation algorithms. Originally, Kriging was a group of geostatistical techniques used to interpolate the value of a random field at an unobserved location from observations of its value at nearby locations [122].

We will further present the techniques for the design of Kriging meta-models. The functions in the DACE toolbox [123] with the specific modifications for strong constraints are used for algorithm implementation.

We consider a set of m design sites (points of the input space) $x = [X_1, X_2, \dots, X_m]^{\mathrm{T}}$ with $X_i \in \mathbb{R}^d$ and responses $y = [y_1, y_2, \dots, y_m]^{\mathrm{T}}$ with $y_i \in \mathbb{R}^q$. The response y will be a circuit

response extracted from simulations which can signify an objective function (of type *P* performance criterion) or a constraint (strong or normal) from the optimization point of view.

In a medium practical problem for a circuit implementation, the dimensionality of these different spaces can be as follows: d – the total number of design variables in the circuit can vary between 1 and a few tens; m – the number of input design sites and their associate responses can vary between tens/hundreds to a few millions; q – the total number of optimization goals can vary between 1 (single-objective without constraints) to ten-twenty goals (objectives and constraints).

The data is assumed to satisfy the normalization conditions:

$$\mu[x_{:,j}] = 0, V[x_{:,j}, x_{:,j}] = 1, \qquad j = 1, 2, \dots, d$$
$$\mu[y_{:,j}] = 0, V[y_{:,j}, y_{:,j}] = 1, \qquad j = 1, 2, \dots, q \tag{4.6}$$

Where $x_{:,j}$ is the vector given by the j^{th} column in matrix x, and $\mu[\cdot]$ and $V[\cdot,\cdot]$ denote respectively the mean and the covariance. The first step in the model construction is to normalize the given x, y so that these conditions are met [123].

Following [124] we adopt a model $\hat{y}$ that expresses the deterministic response $y(x) \in \mathbb{R}^q$, for the d dimensional input $x \in \mathcal{D} \subseteq \mathbb{R}^d$, as a realization of a regression model $\mathcal{F}$ and a random function (stochastic process) z,

$$\hat{y}_l(x) = \mathcal{F}(\beta_{:,l}, x) + z_l(x), \quad l = 1, 2, \dots, q. \tag{4.7}$$

We use a regression model which is a linear combination of p chosen functions $f_j : \mathbb{R}^n \to \mathbb{R}$,

$$\mathcal{F}(\beta_{:,l}, x) = \beta_{1,l} f_1(x) + \beta_{2,l} f_2(x) + \cdots + \beta_{p,l} f_p(x) \equiv f(x)^T \beta_{:,l} \tag{4.8}$$

The coefficients $\{\beta_{k,l}\}$ are regression parameters.

The random process z is assumed to have mean zero and covariance between $z(w)$ and $z(x)$ equal to:

$$\mathbb{E}[z_l(w) z_l(x)] = \sigma_l^2 \mathcal{R}(\theta, w, x) \quad l = 1, 2, \dots, q \tag{4.9}$$

Where σ_l^2 is the process variance for the l^{th} component of the response and $\mathcal{R}(\theta, w, x)$ is the correlation models with parameters θ. An interpretation of this model is that deviations from the regression model, though the response is deterministic, may resemble a sample path of a (suitably chosen) stochastic process z.

It is important to note that the true value of the response can be written as

$$y(x) = \mathcal{F}(\beta_{:,l}, x) + \alpha(\beta_{:,l}, x) \tag{4.10}$$

Where α is the approximation error. The assumption is that by proper choice of β this error behaves like "white noise" in the region of interest, i.e. $x \in \mathcal{D}$. The proper choice of the regression functions will be detailed in Subsection 4.3.4 and it will be shown that the choice is application dependent.

4.3.3. The Kriging Predictor

An efficient optimization algorithm should be able not only to use a meta-model for speed improvement but also to combine evaluation results and prior information about the function in order to select new input points efficiently (e.g. in order to converge towards the optimum on the output), as long as the budget for evaluations is not exhausted. In this case, an output predictor for the choice of new evaluation points should be used. We will consider now the Kriging predictor for the output y.

For the set x of design sites we have the expanded $m \times p$ design matrix F with $F_{ij} = f_j(X_i)$,

$$\mathrm{F} = [\mathrm{f}(\mathrm{X}_1), \mathrm{f}(\mathrm{X}_2), \dots, \mathrm{f}(\mathrm{X}_\mathrm{m})]^\mathrm{T} \tag{4.11}$$

With $f(x)$ defined above.

Further we consider R as the matrix $\mathcal{R}$ of stochastic-process correlations between z points at design sites,

$$\mathrm{R} = \mathcal{R}\big(\theta, \mathrm{X}_\mathrm{i}, \mathrm{X}_\mathrm{j}\big), \qquad \mathrm{i,j} = 1, 2, \dots, \mathrm{m}. \tag{4.12}$$

When selecting a new input point X_{m+1} we have the vector of correlations between z points at design sites and X_{m+1}:

$$\mathrm{r(x)} = [\mathcal{R}(\theta, \mathrm{X}_1, \mathrm{X}_{\mathrm{m+1}}), \mathcal{R}(\theta, \mathrm{X}_2, \mathrm{X}_{\mathrm{m+1}}), \dots, \mathcal{R}(\theta, \mathrm{X}_\mathrm{m}, \mathrm{X}_{\mathrm{m+1}})]^\mathrm{T} \tag{4.13}$$

We will assume that $q = 1$ in Equation 4.7 implying that $\beta = \beta_{:,1}$ and $Y = y_{:,1}$, and consider the linear predictor

$$\hat{\mathrm{y}}(X_{m+1}) = \mathrm{c}^\mathrm{T}\mathrm{Y}, \tag{4.14}$$

with $c = c(X_{m+1}) \subseteq \mathbb{R}^m$.

The error in modeling is

$$\begin{aligned} \hat{y}(X_{m+1}) - y(X_{m+1}) &= c^T Y - y(X_{m+1}) \\ &= (F\beta + Z) - (f(X_{m+1})^T\beta + z) \\ &= c^T Z - z + (F^T c - f(X_{m+1}))^T \beta \end{aligned} \tag{4.15}$$

Where $Z = [z_1, z_2, \dots, z_m]^\mathrm{T}$ are the errors at the design sites.

To keep the predictor unbiased we demand that $F^\mathrm{T}c - f(X_{m+1}) = 0$, or

$$F^{\mathrm{T}}c = f(X_{m+1}) \tag{4.16}$$

Under this condition the mean squared error (MSE) of the predictor is

$$\varphi(X_{m+1}) = \mathrm{E}[(\hat{y}(X_{m+1}) - y(X_{m+1}))^2] = \sigma^2\left(1 + c^{\mathrm{T}}Rc - 2c^{\mathrm{T}}r\right) \tag{4.17}$$

After computing the Lagrangian function for the problem of minimizing φ with respect to c and subject to the constraint $F^{\mathrm{T}}c = f(X_{m+1})$, and finding its gradient, as explained in [123], from the first order necessary conditions for optimum, we get the following system of equations with its solutions:

$$\begin{bmatrix} R & F \\ F^T & 0 \end{bmatrix}\begin{bmatrix} c \\ \tilde{\lambda} \end{bmatrix} = \begin{bmatrix} r \\ f \end{bmatrix}$$

$$\tilde{\lambda} = \left(F^{\mathrm{T}}R^{-1}F\right)^{-1}\left(F^{\mathrm{T}}R^{-1}r - f\right) \tag{4.18}$$

$$c = R^{-1}\left(r - F\tilde{\lambda}\right)$$

Where we have defined

$$\tilde{\lambda} = -\frac{\lambda}{2\sigma^2}, \lambda \text{ being the Lagrangian multiplier} \tag{4.19}$$

The matrix R and therefore R^{-1} is symmetric, and because we considered $\hat{y}(X_{m+1}) = c^{\mathrm{T}}Y$ we find

$$\hat{\mathrm{y}}(\mathrm{X_{m+1}}) = \left(\mathrm{r} - \mathrm{F}\tilde{\lambda}\right)^{\mathrm{T}}\mathrm{R}^{-1}\mathrm{Y} = \mathrm{r}^{\mathrm{T}}\mathrm{R}^{-1}\mathrm{Y} - \left(\mathrm{F}^{\mathrm{T}}\mathrm{R}^{-1}\mathrm{r} - \mathrm{f}\right)^{\mathrm{T}}(\mathrm{F}^{\mathrm{T}}\mathrm{R}^{-1}\mathrm{F})^{-1}\mathrm{F}^{\mathrm{T}}\mathrm{R}^{-1}\mathrm{Y} \tag{4.20}$$

It can be shown [110] that for the regression problem

$$F\beta \cong Y \tag{4.21}$$

The generalized least squared solution (with respect to R) is

$$\beta^* = (F^{\mathrm{T}}R^{-1}F)^{-1}F^{\mathrm{T}}R^{-1}Y, \tag{4.22}$$

And in this case the predictor can be expressed as:

$$\begin{aligned} \hat{y}(X_{m+1}) &= r^T R^{-1}Y - (F^T R^{-1}r - f)^T\beta^* \\ &= f(X_{m+1})^{\mathrm{T}}\beta^* + r(X_{m+1})^T\gamma^* \end{aligned} \tag{4.23}$$

For multiple responses ($q > 1$) this relation hold for each column in Y, so that the last equation holds with $\beta^* \in \mathbb{R}^{p\times q}$ and $\gamma^* \in \mathbb{R}^{m\times q}$ computed via the residuals, $R\gamma^* = Y - F\beta^*$.

Note that for a fixed set of designed data the matrices β^* and γ^* are fixed. For every new X_{m+i} we just have to compute the vectors $f(X_{m+i}) \in \mathbb{R}^p$ and $r(X_{m+i}) \in \mathbb{R}^m$ and add two simple products. Getting an estimate of the error involves a larger computational work.

Again we first let $q = 1$ and we get the following expression for the MSE of the predictor,

$$\varphi(\mathrm{X_{m+i}}) = \sigma^2\left(1 + \mathrm{c^T(Rc - 2r)}\right)$$
$$= \sigma^2\left(1 + \mathrm{u^T(F^TR^{-1}F)^{-1}u - r^TR^{-1}r}\right) \quad (4.24)$$

Where $u = F^{\mathrm{T}}R^{-1}r - f$ and σ^2 is the process variance. This expression generalizes immediately to the multiple response case: for the l^{th} response function we replace σ by σ_l, the process variance for l^{th} response function.

It can be seen that if we consider X_{m+i}, the i^{th} design site selected by the Kriging predictor, then $r(X_{m+i}) = R_{:,i}$, the i^{th} column of R, and $R^{-1}r(X_{m+i}) = e_i$, the i^{th} column of the unit matrix, $e_i = I_{:,i}$. Using these relations we find that $\hat{y}(X_{m+i}) = y_i$. This shows that the Kriging predictor interpolates the design data. Further, we get $u = F^{\mathrm{T}}e_i - f(s_i) = 0$ and the associated MSE will be $\varphi(X_{m+i}) = 0$, since $R_{ii} = 1$.

As indicated by the name MSE (mean squared error) we expect that $\varphi(X_{m+1}) > 0$, but it may happen that $r^{\mathrm{T}}R^{-1}r > 1 + u^{\mathrm{T}}(F^{\mathrm{T}}R^{-1}F)^{-1}u$, in which case $\varphi(X_{m+1}) < 0$. This point needs further investigation, but as a first explanation we offer the following: the equation $\varphi(X_{m+1}) = \mathrm{E}[(\hat{y}(X_{m+1}) - y(X_{m+1}))^2]$ is based on the assumption that the difference between the regression model and the true value is "white noise", and if there is significant approximation error, $y(X_{m+1}) = \mathcal{F}(\beta_{:,l}, X_{m+1}) + \alpha(\beta_{:,l}, X_{m+1})$, then this assumption and its implications do not hold.

Next we can evaluate the gradient of $\hat{y}$ as

$$\hat{\mathrm{y}}' = \left[\frac{\partial\hat{\mathrm{y}}}{\partial\mathrm{X_1}}, \frac{\partial\hat{\mathrm{y}}}{\partial\mathrm{X_2}}, \ldots, \frac{\partial\hat{\mathrm{y}}}{\partial\mathrm{X_d}}\right]^{\mathrm{T}} \quad (4.25)$$

which can be further expressed as

$$\hat{y}'(x) = J_f(X_{m+1})^{\mathrm{T}}\beta^* + J_r(X_{m+1})^T\gamma^* \quad (4.26)$$

where J_f and J_r is the Jacobian of f and R, respectively,

$$(J_f(X_{m+1}))_{ij} = \frac{\partial f_i}{\partial X_j}(X_{m+1}), \qquad (J_r(X_{m+1}))_{ij} = \frac{\partial \mathcal{R}}{\partial X_j}(\theta, X_i, X_{m+1}). \quad (4.27)$$

From Equation 4.17 it follows that the gradient of the MSE can be expressed as

$$\varphi'(X_{m+1}) = 2\sigma^2((F^TR^{-1}F)^{-1}u' - R^{-1}r')$$
$$= 2\sigma^2\left(\llbracket(F)\rrbracket^{\mathrm{T}}R^{-1}F\right)^{-1}\llbracket(F)\rrbracket^{\mathrm{T}}R^{-1}r' - J_f\beta^*\big) - R^{-1}r'\Big) \quad (4.28)$$

This is the procedure of predictor implementation in practical computations, and although complex calculus is employed, further simplifications might be done for modeling analog functions as we will further present.

4.3.4. Regression Functions

The *f* functions composing the regression model $\mathcal{F}$ can be represented by polynomials of orders 0, 1 or 2 for simple practical implementation.

More specific, with x_i denoting the j^{th} component of x we have the following cases:

8. **Constant**, $p = 1$: $f_1(x) = 1.$
9. **Linear**, $p = n + 1$: $f_1(x) = 1,\ f_2(x) = x_1,\ \dots,\ f_{n+1}(x) = x_n$
10. **Quadratic**, $p = \frac{1}{2}(n+1)(n+2)$: $f_1(x) = 1$

$$f_2(x) = x_1,\ \dots,\ f_{n+1}(x) = x_n$$
$$f_{n+2}(x) = {x_1}^2,\ \dots,\ f_{2n+1}(x) = x_1 x_n$$
$$f_{2n+2}(x) = {x_2}^2,\ \dots,\ f_{2n+1}(x) = x_2 x_n$$
$$\dots \quad \dots$$
$$f_{\ p}(x) = {x_n}^2$$

The corresponding Jacobians are (index $n \times q$ denotes the size of the matrix and O is the matrix of all zeros):

11. **Constant**: $J_f = [O_{n\times 1}]$
12. **Linear**: $J_f = [O_{n\times 1} \quad I_{n\times n}]$
13. **Quadratic**: $J_f = [O_{n\times 1} \quad I_{n\times n} \quad H]$

Where we illustrate $H \in \mathbb{R}^{n\times(p-n-1)}$ by

$$n = 2: \quad H = \begin{bmatrix} 2x_1 & x_2 & 0 \\ 0 & x_1 & 2x_2 \end{bmatrix} \tag{4.29}$$

4.3.5. Correlation Functions Models

The $\mathcal{R}$ correlation functions for stochastic-processes can be products of stationary, one-dimensional correlations [124] of type:

$$\mathcal{R}(\theta, w, x) = \prod_{j=1}^{n} \mathcal{R}_j\big(\theta, w_j - x_j\big), \tag{4.30}$$

More specific, the DACE toolbox [123] allows the implementation of the following 6 types of correlation functions: EXP-Exponential, EXPG-General Exponential, GAUSS-Gaussian, LIN-Linear, SPHERICAL and CUBIC.

The correlation functions along with their expression formulae and calculus conditions are presented in Table 4-1.

Name	$\mathcal{R}_j(\theta, d_j)$
EXP	$\exp(-\theta_j \lvert d_j \rvert)$
EXPG	$\exp\left(-\theta_j \lvert d_j \rvert^{\theta_{n+1}}\right), \quad 0 < \theta_{n+1} \leq 2$
GAUSS	$\exp(-\theta_j {d_j}^2)$
LIN	$\max\{0, 1 - \theta_j \lvert d_j \rvert\}$
SPHERICAL	$1 - 1.5\xi_j + 0.5{\xi_j}^3, \xi_j = \min\{1, \theta_j \lvert d_j \rvert\}$
CUBIC	$1 - 3{\xi_j}^2 + 2{\xi_j}^3, \quad \xi_j = \min\{1, \theta_j \lvert d_j \rvert\}$

Table 4-1 Correlation functions available in DACE

For two points that are close in the input domain, their associated errors should be also close, so we can assume that their associated errors are correlated. The correlation is high when the two points are close and low when the points are far apart. So correlation between errors is related to the distance between the corresponding points, distance that can be computed in many ways, the most simple being the geometric distance between the two points.

Some of the choices are illustrated in Figure 4.2 below. Note that in all cases the correlation decreases with $|d_j|$ and a larger value for θ_j leads to a faster decrease. Due to the normalization of the data we are interested in cases where $|d_j| \leq 2$, as illustrated.

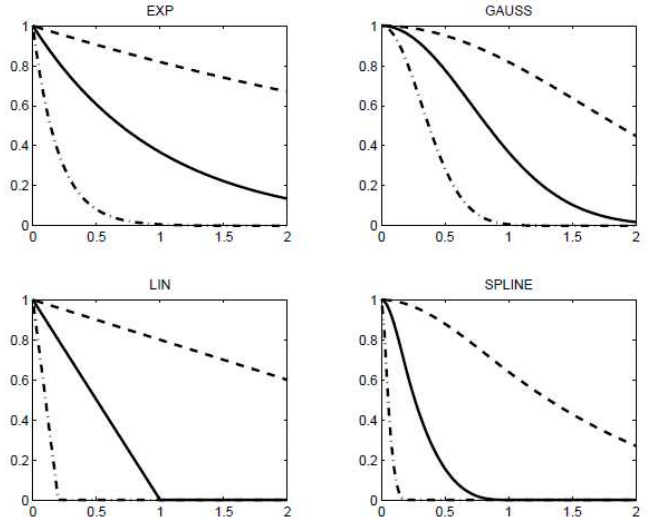


Figure 4.2 Correlation functions for $0<d_j \leq 2$ (θ_j=0.2: Dashed-full; θ_j=1.5: Dash-dotted)

The correlation functions in Table 4-1 can be separated into two groups, one containing functions that have a parabolic behavior near the origin (GAUSS, CUBIC), and the other containing functions with a linear behavior near the origin (EXP, LIN and SPHERICAL). The General Exponential EXPG can have both shapes, depending on the last parameter: $\theta_{n+1} = 2$ and $\theta_{n+1} = 1$ gives the Gaussian and the exponential functions, respectively.

The choice of correlation function is motivated by the underlying phenomenon of the physical process we want to model. In the vast majority of real problems, Gaussian or cubic

functions prove to be very reliable, but different correlation patterns can be chosen depending on the problem, as indicated in [125].

4.4. Bayesian Optimization Using the Expected Improvement Criterion

4.4.1. Bayesian Optimization Applied to Analog Cells

In a Bayesian optimization problem, random variables and random processes are used instead of deterministic values.

In this case the random variables appear in the formulation of the optimization problem itself, which involve random objective functions or random constraints [126].

The Bayesian optimization methods generalize deterministic methods for deterministic problems. Even when the data set consist of precise measurement, some methods introduce randomness into the search-process to accelerate progress [127]. Such randomness can also make the method less sensitive to modeling errors. Further, the injected randomness can enable the method to escape a local minimum and potentially to approach a global optimum.

Indeed, this randomization principle is known to be a simple and effective way to obtain algorithms with almost certain good performance uniformly across data sets, for many sorts of problems.

We consider here the problem of optimizing an analog cell, which can be assimilated with a real-valued continuous function, that is expensive to evaluate and, consequently, can only be evaluated a limited number of times.

The Bayesian approach to this problem is chosen in order to quantify the uncertainty in the input space regions which cannot be initially explored, due to limited evaluations.

The Bayesian method consists in combining evaluation results and prior information about the target function in order to efficiently select new evaluation points, as long as the budget for evaluations is not exhausted.

The algorithm called efficient global optimization (EGO), proposed by Jones, Schonlau and Welch ([128]), is one of the most popular Bayesian optimization algorithms. It is based on a sampling criterion called the expected improvement (EI), which assumes a prior Gaussian process about the target function.

In the EGO algorithm, the parameters of the covariance of the Gaussian process are estimated from the evaluation results by maximum likelihood, and these parameters are then re-transferred in the EI sampling criterion [129].

However, it is well-known that this plug-in strategy can lead to very disappointing results when the evaluation results do not carry enough information about the function in order to estimate the parameters in a satisfactory manner.

To avoid these situations, the expected improvement criterion is considered in a more careful way: in the case of analog functions, apart one objective, there are a series of constraints which should be respected. In this case, the expected improvement will be derived subject to constraints.

One of our contributions in the expected improvement criterion application resides in the introduction of the constraints with different weights in the selection of new samples, e.g. the *strong constraints* should be necessarily respected on each new selected sample, while the *normal constraints* should only be respected on the final results. In this case, a slight modification of the EGO algorithm is introduced, as will be further discussed, to treat efficiently the two classes of constraints.

4.4.2. Unconstrained Expected Improvement

As presented in Section 4.2 a simulation-based optimization approach is used for the analog functions. The simplest scenario of the simulation-based optimization resides on a sequential strategy [130] where points are chosen and the function is evaluated in one point at a time. The next point should be selected 'rationally' in order to converge towards the global optimum of the function.

In this section we will present a criterion for determining the next sampling location, based on the Kriging predictor presented in Subsection 4.3.3. It generalizes the criterion proposed by Mockus et all. [131], which can be interpreted as the expected improvement (EI) in the maximum y value found so far when the next function evaluation is made.

For the objective function f: X →R, where X⊂ $\boldsymbol{R}^d$ we want to find the minimum, for example. The main idea of an EI-based algorithm [128] is a Bayesian one: *f* is viewed as a sample path of a random process ξ defined on $\boldsymbol{R}^d$. We combine evaluation results and prior information about *f* to select, according to the EI criterion, new evaluation points efficiently, as long as the budget for evaluations is not exhausted.

After *n* evaluations, the available information corresponds to the previous evaluation points $(X_1, X_2,\ldots, X_n)$ and to the associated evaluation results (ξ(X_1), ξ(X_2),…, ξ(X_n)). Let F_n denote the set (X_1, ξ(X_1), X_2, ξ(X_2),…, X_n, ξ(X_n)) and m_n the current minimum, min(ξ(X_1), ξ(X_2),…, ξ(X_n)). Given a budget of N, the purpose of this algorithm is obviously to minimize the value m_N. The next evaluation point X_{n+1} is the maximizer of the EI criterion ρ_n:

$$\rho_n(x) = E_n((m_n - \xi(X_{n+1}))_+ | X_{n+1} = x) \tag{4.31}$$

Where $E_n$is the conditional expectation, given F_n and $x \in \boldsymbol{X}$.

ξ is assumed to be a Gaussian process. $\xi(x)$, conditionally with respect to F_n, is Gaussian with mean $\hat{\xi}_n(x)$ and variance $s_n(x)$, whose explicit expressions can be found in

[132]. The main point of choosing ξ as a Gaussian process is the existence of an analytical form for the EI criterion expression:

$$\rho_n(x) = s_n(x)\Phi'\left(\frac{m_n - \hat{\xi}_n(x)}{s_n(x)}\right) + \left(m_n - \hat{\xi}_n(x)\right)\Phi\left(\frac{m_n - \hat{\xi}_n(x)}{s_n(x)}\right) \qquad (4.32)$$

Where Φ is the normal cumulative distribution. This expression shows that, given a set of evaluation points and a Gaussian process, the EI sampling criterion can be computed with a moderate amount of resources.

Practically, to use this analytical expression, one needs to define an explicit mean and covariance functions for ξ. Experimentally, a Gaussian covariance function is used and the parameter values are estimated at each iteration by maximum likelihood. This approach corresponds exactly to the EGO algorithm in [128]. It should be noted that both Kriging predictions and maximum likelihood estimations are performed using the functions in the DACE toolbox [123].

4.4.3. Expected Improvement Subject to Constraints

Once we presented the Expected Improvement criterion in the unconstrained case, we want to explore the possibilities to integrate the optimization constraints for analog cells as defined in the Section 4.2. Let us consider p functions $g_1, g_2, \ldots, g_p$, where the new optimization problem is the minimization, for $x \in \boldsymbol{X}$, of $f(x)$ subject to $a_i \leq g_i(x) \leq b_i$, with $(a_i, b_i) \in \boldsymbol{R}^2$, for $1 \leq i \leq p$. For simplicity, we will consider that the $g_i(x)$ constraints can be on the circuit input, on the output or general constraints on the input-output. It should be noted that these constraints are of type *Normal constraint* as defined in the Section 4.2.

The general approach is the same as for the unconstrained EI. At step n, we choose the next evaluation point X_{n+1} as the maximizer of the Expected Improvement. However, if the constraints g_i are expensive to evaluate as well, we can also consider them as Gaussian random processes G_i. Let G_n^i denote the set (X_1, $G_i(X_1)$, X_2, $G_i(X_2)$, …, X_n, $G_i(X_n)$). The only difference is that the expected improvement (EI) criterion is now subjected to the constraints. Let $\bar{\rho}_n$ denote this criterion:

$$\bar{\rho}_n(x) = E_n(I_n(x)\,|X_{n+1} = x) \qquad (4.33)$$

where $I_n(x)$ equals to $((m_n - \xi(X_{n+1}))_+$ if $a_i \leq G_i(x) \leq b_i$ is verified for all i values and equals 0 otherwise.

Without going further into details, it can be shown [132] that if $f, g_1, g_2, \ldots, g_p$ are all statistically independent, this constrained EI criterion can be written as follows:

$$\bar{\rho}_n(x) = \rho_n(x)\prod_{i=1}^{p} P\left(a_i \leq G_i(x) \leq b_i |\, G_n^i\right) \qquad (4.34)$$

This criterion is simply the product of the unconstrained EI with the probability that each constraint is verified. This probability is easy to compute because, for $x \in \boldsymbol{X}$, the random process $G_i(x)|\ G_n^i$ is a Gaussian process, i.e., this constrained EI criterion can also be computed with a limited amount of resources [129].

4.4.4. Modification of the Criterion for the Strong Constraints

In practice, with respect to the application to linear analog functions considered, we notice that a modification of the algorithm should be performed for the constraints of type extracted nonlinearity or distortion coefficients, called *strong constraints*.

Because these functions should be highly linear, the evaluation points which do not respect the imposed linearity constraints cannot be considered acceptable partial solutions, as in the case of *normal constraints*.

This modification of the criterion can improve drastically, the convergence to a global optimum where the constraints are respected.

The regions of the domain satisfying the *strong constraints* are the most promising. In other words, most of the budget of the evaluations should be consumed in these specific parts of the domain. However, an important property of the EI criterion is avoiding convergence to a local minimum because of an exploration/exploitation tradeoff. This means that because of the exploration phase of the algorithm, many evaluation points are chosen in areas where the probability of verifying the nonlinearity constraint is very low.

This problem can be solved by tuning the parameters of the Gaussian process (the mean and the parameters of the covariance function) in a more careful way (maximum likelihood estimation for the parameters is known for not always being relevant [133]).

We used a slightly different approach: for the nonlinearity constraint g_j, a more strict selection is performed to solve the problem of exploration in areas non-interesting for strong constraints. This can be done by a reset of the value of $\bar{\rho}_n(x)$ to 0 when $P\left(a_j \leq G_j(x) \leq b_j|\ G_n^j\right)$ is below a threshold α (usually $\alpha = 0.05$).

The value of the threshold α is a priori selected function of the total number of evaluation points, the number of strong constraints and other practical requirements for the specific cell, which is to be optimized.

4.5. Automatic Optimization of Analog Cells – Algorithm Implementation

This automatic optimization method, using a Bayesian technique, where the evaluation space is created by the means of a Kriging surrogate model, and the selection is effectuated by using the expected improvement (EI) criterion subject to constraints was integrated in the

general design methodology presented in Chapter 2 and implemented in the corresponding design framework SIMECT [134].

We will further detail the implementation of the automatic optimization algorithm by taking into account the theoretical fundaments which were presented.

The optimization algorithm based on the DACE toolbox functions [123] consists of two main stages in which very different operations are performed:

14. *The starting phase*: in this part, a defined number of *starting points* are selected by using an efficient sampling method which assures both the selection of random values and a good distribution of the points over the entire input domain. Also in this phase, the function to optimize is evaluated in all the starting points;
15. *The optimization phase*: also known as the exploration/exploitation phase, in this stage, statistical models for the objective function and constraints are constructed based on the evaluations in the starting phase and new "promising" points are selected iteratively from the input space based on the EI criterion subject to the *normal constraints* and *strong constraints*.

Regarding the starting points selection, a number of sampling methods for multi-dimensional spaces can be used: uniform sampling, Monte Carlo sampling (or random sampling), Latin Hypercube sampling (LHS) and Orthogonal sampling. In our case, the Latin Hypercube sampling method was employed in order to benefit from its good space-filling proprieties with limited amount of calculus resources. The Latin Hypercube designs are particularly useful when a random sample is needed, but that is guaranteed to be relatively uniformly distributed over each dimension [128].

The LHS, first described by McKay et al. [135] and further developed in [136], is a strategy for generating random sample points ensuring that all portions of the vector space are represented. Considering the space $S \subset \boldsymbol{R}^n$ and m points which are sampled in this n-dimensional vector space, the Latin Hypercube sampling strategy is as follows:

16. the interval of each dimension is divided into m non-overlapping intervals having equal probability (when a uniform distribution is considered, the intervals should have equal size);

- then, points in each interval in each dimension are sampled randomly from a uniform distribution;
- finally, the points from each dimension are paired randomly (equal likely combinations).

The Figure 4.3 presents an example of Latin Hypercube sampling in a 2-D space [0; 1] x [0; 1], with 50 sampling points selected.

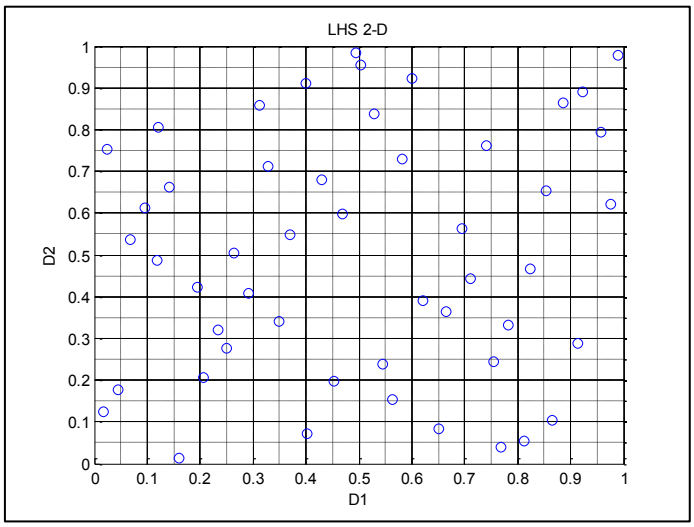


Figure 4.3 Latin Hypercube sampling in a 2-D space example

The total number of *evaluation points* for the Bayesian optimization method consists of the *starting points* and the *optimization points*. The number of *starting points* and the number of *evaluation points* are custom parameters and should be carefully selected function of the specific application. Practical implementations suggested that the quantity of evaluation points should be 5 to 10 times the number of design variables [110]. A comprehensive example study regarding this aspect will be presented in Section 4.7, on the proposed optimization application.

Starting from these considerations, the optimization algorithm is structured as follows:

1. Choice of the starting points (the starting phase)
    a. Latin Hypercube sampling for the design variables of the analog cell used for optimization (transistor size, current source value, component value, user-defined variables, etc.);
    b. Evaluation of the objective function and constraints for the starting points – in the case of analog cells, this is performed by running the transistor-level simulation for each starting point and extraction of the figures of merit which were defined as objective and constraints;
2. Build a statistical model for the objective function and constraints, respectively
    a. Build the regression model as a linear combination of regression functions: constant, linear or quadratic;
    b. Build the correlation model of type: Exponential, Gaussian or Linear;
3. Selection of a new evaluation point
    a. Build the evaluation space (composed with all the remaining points after the starting points selection, or when too many, the LHS is again employed);

b. The Expected Improvement subject to *normal constraints* and *strong constraints* is used as criterion to select the most "promising" point in the evaluation space;

c. As long as the *strong constraints* are mandatory, the algorithm makes choices only in the regions where the *strong constraints* are very likely to be respected (see discussion in Subsection 4.4.4). Figure 4.4 presents an example of the regions of an evaluation space where the EI will make selections, e.g. the points 2, 3, 5 can be selected.

4. Verify the stop condition and if met the iteration ends;
5. Otherwise, the objective function and constraints are computed in the new selected point and the algorithm is re-iterated from Step 2. The statistical models will be reshaped based on the information of the new point.

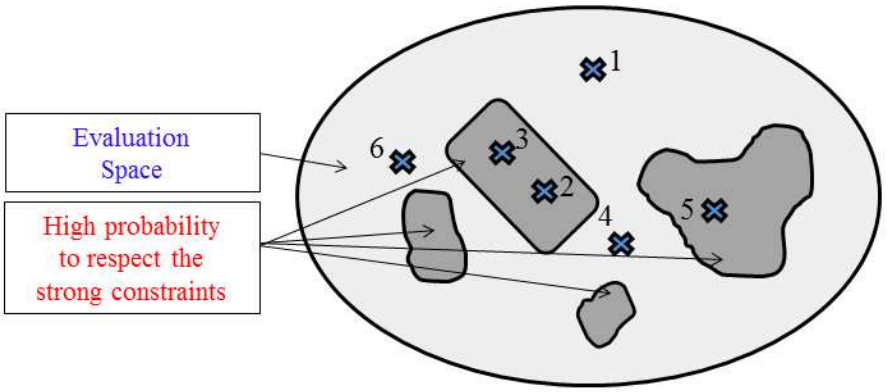


Figure 4.4 EI selection function of the strong constraints - example

Next, we will consider also a simple optimization method of type semi-automatic, which can be used for initial explorations but also for validation of the automatic method results.

4.6. Semi-automatic Optimization Based on Pseudo-gradient Descent

In the first stages of our research, a semi-automatic optimization method based on a steepest descent algorithm (pseudo-gradient descent) using multiple starting points was studied.

This method, although time-consuming and entirely based on transistor-level simulations, was intended for initial explorations of the analog designs performances and for fine-tuning the manual optimization results, when needed. It can be used also for validation and coherence approval with the automatic optimization method.

The semi-automatic optimization method is based on two specialized analyses: macro-parametric performance analysis and multi-criteria sensitivity analysis.

The automated part of the process consists in performing macro-parametric analyses: each design variable of the schematic which is considered (transistor size, current source value,

component value) can be varied and it is possible through transistor-level simulations to extract all the interesting characteristics for optimization. For the bipolar transistors the only parametric design variable is their area. In the case of MOS transistors, optimal parameters would be their length (L) and geometric ratio (W/L).

One performance parametric analysis result is shown as an example in Figure 4.5, but it is possible to obtain macro-parametric analyses for each analyzed parameter.

The total simulation time might take minutes or hours depending on the number of varied parameters, the schematic complexity, and the machine speed, but the simulations are automatic and no human intervention is required.

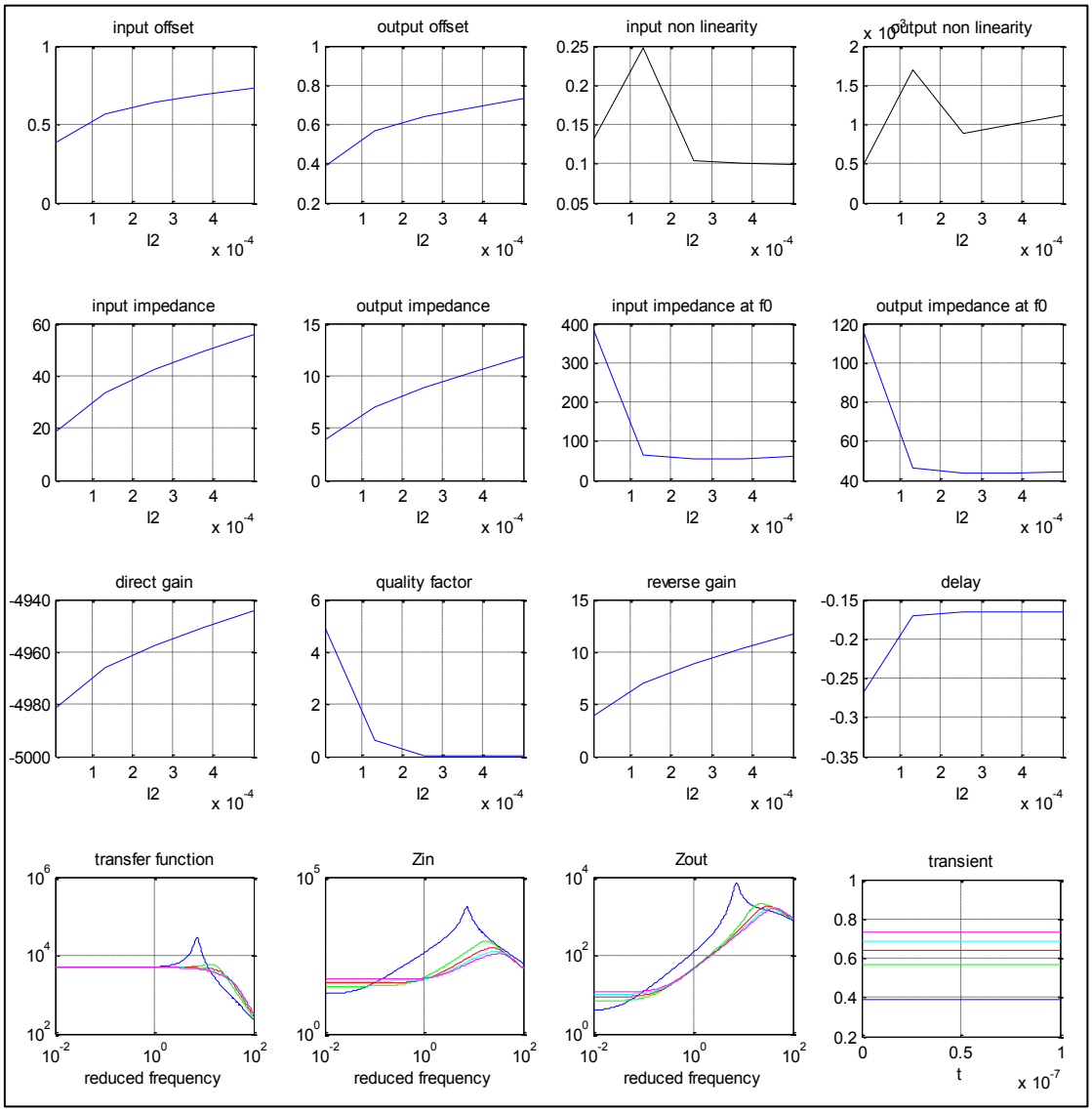


Figure 4.5 Macro-parametric analysis for a TIA amplifier

The parametric analysis presented in Figure 4.5 is part of the semi-automatic optimization process of a trans-impedance amplifier (TIA) presented in the following Section 4.7. In this example, each performance characteristic of the circuit is analyzed function of the polarization current I2. We can deduce for example that I2 has a strong influence on the input

impedance and the output impedance, while the group delay (*de*) at a given frequency are less changed by the I2>20µA variation. The input non-linearity is sufficiently large in this first exploration, so more adjustments on the other design variables will try to reduce it.

Once all the macro-parametric analyses were performed, the designer will deduce which parameter influences each performance characteristics and the components values can normally be found manually after 3 iterations.

Otherwise, it is possible to semi-automate the process using the multi-criteria sensitivity analyses, where the influence of each design variable on a specific criteria are grouped.

A multi-criteria sensitivity analysis is given as an example in Figure 4.6 (also for the TIA proposed example).

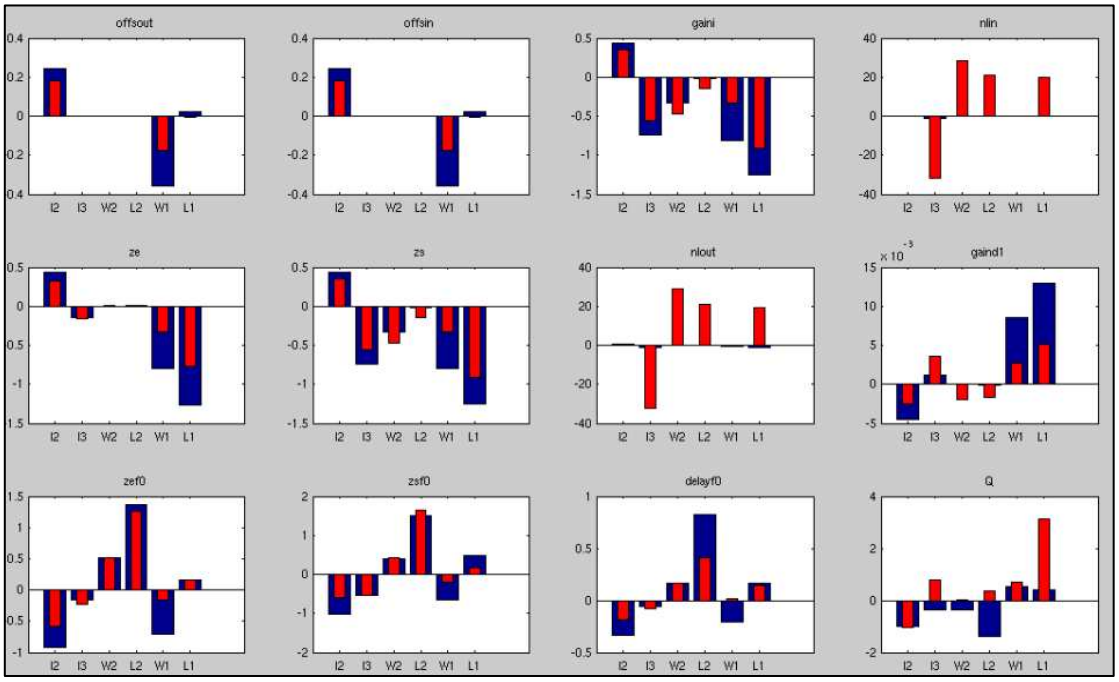


Figure 4.6 Multi-criteria sensitivity analysis for a TIA amplifier

With red we have the short term sensitivity of the criterion function of each variable, defined as the simple derivative of the indication (*I*) in the point of interest (only for differentiable functions):

$$\varepsilon_I = \left.\frac{\partial I(x_k)}{\partial x_k}\right|_{x_k=x_0} \tag{4.35}$$

With blue we computed the long term sensitivity as the slope of the linear variation between the starting and the ending point of the design variable domain:

$$\vartheta_I = \left.\frac{\Delta I(x_k)}{\Delta x_k}\right|_{x_k=\overline{x_s, x_e}} \tag{4.36}$$

We consider $[\mathbf{I}]_{n\times 1}$ the n-dimensional input vector containing all the design variables, $[\mathbf{E}]_{m\times 1}$ the m-dimensional output vector containing all the criteria and $[\mathbf{S}]_{m\times n}$ the sensitivity

matrix (containing the short term or long term sensitivity coefficients) that retains the dependency between inputs and outputs.

For a first order approximation around a specific point, the following can be written:

$$[\boldsymbol{E}]^T = [\boldsymbol{S}] \times [\boldsymbol{I}] \tag{4.37}$$

$$[\boldsymbol{I}]_{p+1} = [\boldsymbol{I}]_p + [\boldsymbol{S}]^{-1} \times \left([\boldsymbol{E}]_d^T - [\boldsymbol{E}]_p^T\right) \tag{4.38}$$

Where $[\mathbf{E}]_d$ is the desired output vector, $[\mathbf{E}]_p$ is the current output vector (for the current point), $[\mathbf{I}]_p$ is the current input point (around which the linearization is done) and $[\mathbf{I}]_{p+1}$ is the input point that is chosen for the next iteration. The $[\mathbf{S}]$ matrix must be reversible, and therefore a square one, so in a general case, an appropriate bordering is considered.

The principle of this optimization method is presented as a generic first order example in Figure 4.7. The objective function is considered as black continuous line, and the area where the constraints are met is contained by the red ellipse. As we can see, if we want to minimize this function, respecting the constraints, the optimum point is located near the point indicated by number 5. The method is based on successive steps that consist in choosing points which respect the criterion (e.g. 4) and then separately the constraints (e.g. 1, 3, 5), until an optimum solution is found.

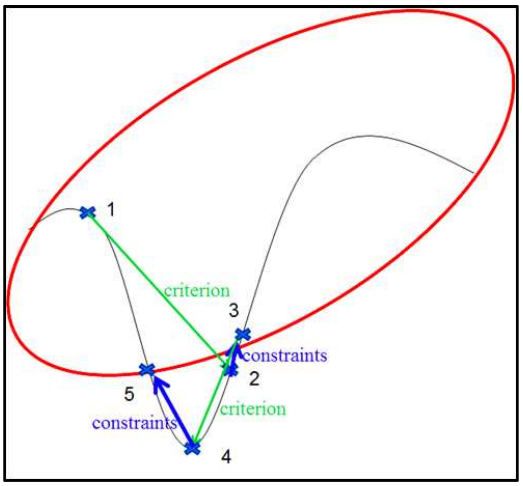


Figure 4.7 Semi-automatic optimization of one criteria and constraints

The method only gives approximate solutions for non-linear functions, and no guarantee exists that a global optimum is achieved. Furthermore, the $[\mathbf{S}]$ matrix is not the gradient, as required for multi-variable problems. Still, for a reasonable number of design variables and a proper exploration of the scenarios through transistor-level simulations, the method can be employed for early explorations. For a large number of variables, the schematics are optimized with ideal current sources and afterwards the real sources are added and optimized also.

4.7. Validation of the Bayesian Kriging Optimization Method

The validation of the optimization methods was realized throughout the entire conception of the components for the Sigma-Delta applications presented in Chapter 5.

Here we discuss, as an example, the optimization of a single amplifier used in a novel CT Sigma-Delta modulator architecture.

The optimization methodologies were applied to the design of a fast trans-impedance amplifier used in a CT Sigma-Delta modulator based on the architecture presented in [137].

A differential trans-impedance amplifier is composed in a differential CT Sigma-Delta filter of two single-ended amplifiers. A schematic of the single-ended structure is shown in Figure 4.8. It is implemented using a STMicroelectronics 65nm Low-Power General Purpose (LPGP) CMOS technology process.

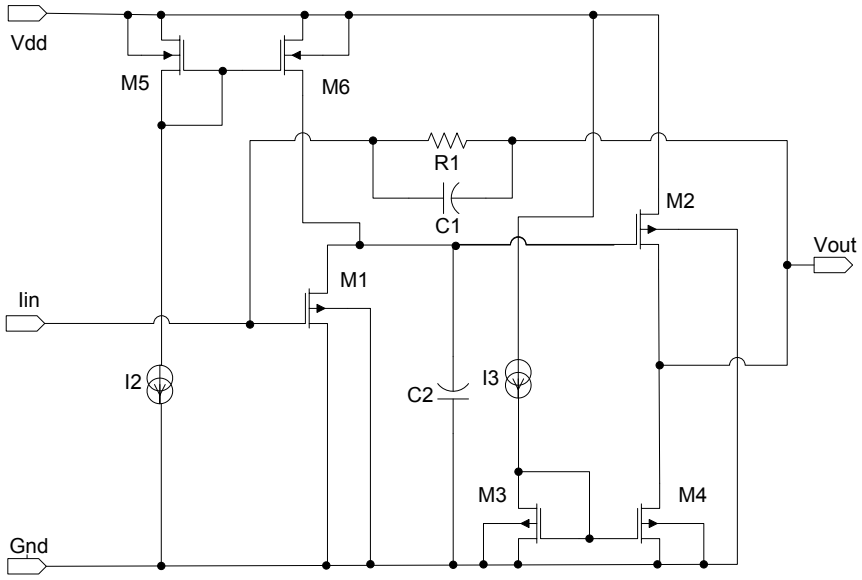


Figure 4.8 Single-ended transimpedance amplifier

Transistors M1 and M2 and resistor R1 act as a current-controlled voltage source. The source's gain is equal to R1. The pairs M3-M4 and M5-M6 and the current generators I2 and I3 are real current sources used to bias the main transistors M1 and M2.

When this amplifier is connected to a high capacitance, it may become unstable, and capacitances C1 and C2 help stabilize the amplifier.

The optimization objectives and constraints for this circuit are summarized in Table 2-1. Assuring the coherence with the presented Bayesian optimization method, but also with the semi-automatic method briefly discussed, one objective, two *normal constraints* and two *strong constraints* are employed.

Parameter	Type	Desired Value(s)	Others
Input impedance (Z_{in})	Objective	Minimum	-
Output impedance (Z_{out})	Constraint	[5 ; 10] [Ω]	Normal
Direct gain (g^d)≈ R_1(5000 Ω)	Constraint	[4900 ; 5100] [Ω]	Normal
Input Nonlinearity (nl^{in})	Constraint	<1%	Strong
Output Nonlinearity (nl^{out})	Constraint	<1%	Strong

Table 4-2 Trans-impedance amplifier optimization goals

The initial current sources values are determined by manual DC considerations, while the initial transistor sizes are sufficiently large to minimize the saturation voltages drain-source.

Then, the automatic optimization process is launched. Table 4-3 lists the variation range of the design variables.

Design Variable	Variation Range	Number of points
I2	[20; 80] µA	25
I3	[20; 160] µA	60
w_i/w_{min}	[1; 60] ; w_i→[0.065; 3.9] µm	60
l_i/l_{min}	[1; 30] ; l_i→[0.065; 1.95] µm	30
w_j/w_{min}	[1; 60] ; w_j→[0.065; 3.9] µm	30
l_j/l_{min}	[1; 30] ; l_j→[0.065; 1.95] µm	15

Table 4-3 Variables variation; w_{min}= l_{min}=65 nm; i=1,2; j=3-6

The starting points used to compute the meta-model will be sampled to create a Latin Hypercube sample, which can be viewed as a generalization in all dimensions of a Latin square sample.

Then, the algorithm will start choosing optimization points according to the EI criterion and constraints. Figure 4.9 presents the evaluation points (40 starting points and 40 optimization points) for the first three design variables.

The number of points is not arbitrarily chosen, but is dependent on the application. A first approach was to choose a number between 5 to 10 times the quantity of design variables (12). But the problem was further studied and a comparative performance table is given at the end of this section, where a comparison between the optimization methods is intended.

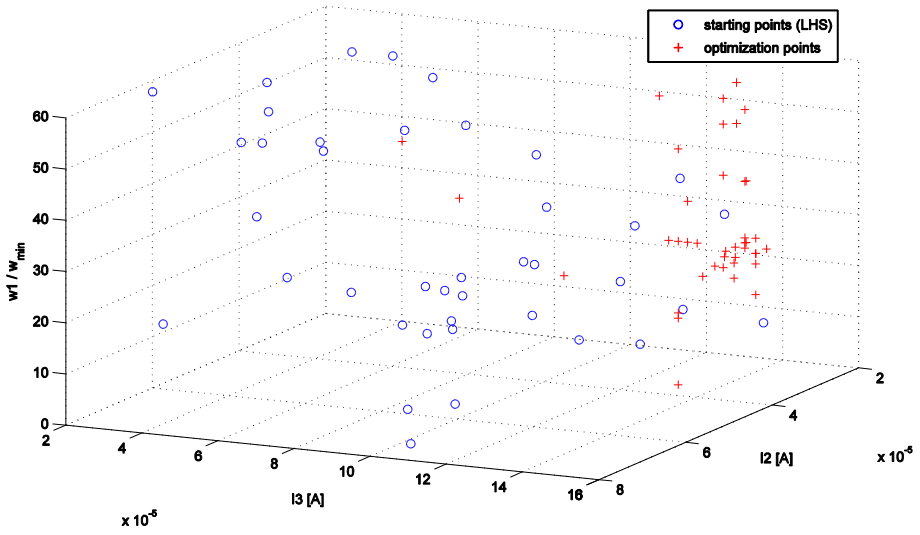


Figure 4.9 Distribution of the starting points and optimization points

An interesting measure of the algorithm's performance is the temporal evolution of the optimization goals. Figure 4.10 depicts the variation of three figures of merit with the advancement of the optimization: the optimization objective (*Zin*), a normal constraint (*Zout*) and a strong constraint (*nlout*).

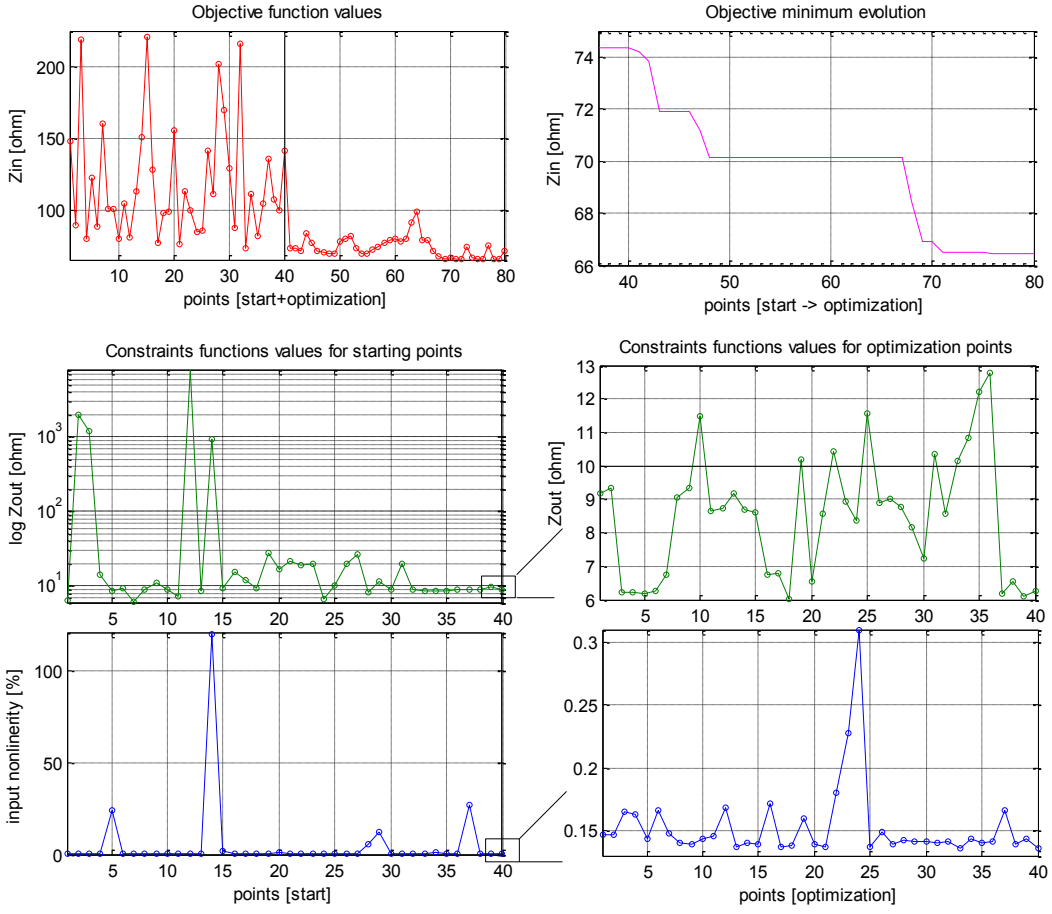


Figure 4.10 Temporal evolution of the optimization goals

For the starting domain, because the points are arbitrarily chosen with respect to objective and constraints, these optimization goals also have arbitrary values.

As soon as the meta-model is designed and optimization points are considered, the goals start to converge towards the desired values (e.g. the Objective minimum evolution reaches points where smaller values are obtained with the optimization epoch advancement). The non-uniform variation of the goals results from the algorithm exploring different regions of the design variables domain, where the probability of finding "better" candidate points is higher.

When using the Bayesian Kriging Optimization method for the optimization of this amplifier, the optimum point is found in the following point of the multi-dimensional space created by the design variables: $I2$=27.5µA, $I3$=145µA, w_1/w_{min}=28, l_1/l_{min}=27, w_2/w_{min}=19, l_2/l_{min}=6, $w_3/w_{min}=w_4/w_{min}$ =24, $l_3/l_{min}=l_4/l_{min}$=10, $w_5/w_{min}=w_6/w_{min}$=4, $l_5/l_{min}= l_6/l_{min}$=2. For this point, the following figures of merit defining the optimization goals were obtained: Z_{in}=66.42Ω, Z_{out}=10Ω, g^d=4926Ω, nl^{in}=0.15%, nl^{out}=0.1%.

The semi-automatic optimization method was also employed for the optimization of the TIA amplifier. In this case, the amplifier was optimized with ideal current sources (transistors M3, M4 and the source I3, transistors M5, M6 and the source I2 were replaced by two ideal current sources) and then the real sources were added and optimized also to match the value obtained for the real source.

By using the semi-automatic method, after 3 macro-parametric iterations (e.g. Figure 4.5) on each configuration (ideal and separate real sources), an optimum point is found at $I2$=30µA, $I3$=125µA, w_1/w_{min}=28, l_1/l_{min}=27, w_2/w_{min}=16, l_2/l_{min}=8, $w_3/w_{min}=w_4/w_{min}$ =26, $l_3/l_{min}=l_4/l_{min}$=12, $w_5/w_{min}=w_6/w_{min}$=3, $l_5/l_{min}= l_6/l_{min}$=2. For this, we obtained Z_{in}=69.5Ω, Z_{out}=9.8Ω, g^d=4940Ω, nl^{in}=0.2%, nl^{out}=0.1%.

In terms of computation time and performance, the two optimization methods are compared in Table 4-4.

The experiment was conducted for a set of 10 cases of optimization on the same amplifier design, using the same optimization goals. Due to the fact that different starting points are chosen for every new scenario by the LHS, the automatic method will not repeat the same calculus in each case, so the computation times and also the optimization results will be slightly different, tough the coherence of the optimization is assured. This is why the computation times are indicated as average.

For the automatic method, different numbers of starting points and optimization points were selected in order to show the impact of this parameter on the convergence of the method and the total time of an optimization cycle.

The indications presented are: the number of optimization steps to convergence – indicating on which position between the optimization points the optimum was found; the total optimization time – the time needed to compute all the calculus for the overall evaluation points

(starting and optimization); the actual time to convergence – the time needed to compute the starting points and to reach convergence, hypothetically without computing all the optimization points calculus.

It is important to derive some figures of merit from these timing characteristics:

6. selecting a higher number of starting points will accelerate convergence due to more a priori information when constructing the meta-model;
7. selecting a higher number of starting points will also minimize the actual time to convergence, thus it is possible to reduce the number of evaluation points in the optimization stage, without the risk of non-convergence;
8. a too high number of starting points will result in a drawback: very fast convergence, which can result in a local minimum and potentially in a too complex meta-model, requiring very huge computation resources;
9. too many optimization points will require a much longer evaluation time (for each new point, the meta-model is reshaped), while too few optimization points can result in cases of non-convergence due to insufficient exploration (e.g. in the case 20 starting and 50 optimization points in Table 4-4).

For the semi-automatic method, the given time is an indication of the actual simulation process, but the approach is not entirely automatic as in the first case, so the designer experience is important in order to reach the optimization goals.

1. Automatic method		Optimization steps to convergence	Total optimization time – average	Actual time to convergence – average
Number of points				
start	optimization			
40	40	33	242 min.	220 min.
20	50	47*	249 min.	239 min.
30	50	40	266 min.	223 min.
40	50	32	291 min.	214 min.
50	50	15	308 min.	168 min.
60	30	7	245 min.	152 min.
2. Semi-automatic method		27* macro-parametric	1-2 days	1-2 days

Table 4-4 Trans-impedance amplifier optimization – methods performances

* the convergence is not assured

The presented aspects indicate that the two optimization methods are coherent and the automatic method finds approximately the same results as the semi-automatic one, but with better temporal performances, when its parameters are correctly tuned.

4.8. Chapter Conclusion

The fourth chapter of this work studies another fundamental step of the analog design methodology detailed in Chapter 2, e.g. the semi-automatic optimization or automatic optimization possibilities for analog cells implemented at transistor-level. This process is known in industry also on the name of automatic transistor-level sizing of circuits.

A first discussion regards the optimization strategies for specific disciplines and different input domain types, identifying the problem set to which the optimization of electrical networks belongs.

Then, the basic optimization problem for an analog cell is formulated in terms of general optimization objectives and constraints. The notion of *strong constraint* is introduced to describe the weak non-linear effects related to amplification functions (non-linearity, distortions). These types of constraints are used in conjunction with *normal constraints* and one *optimization objective* for the optimization problem statement.

For this kind of optimization, simulation-based approaches are mandatory to be used, potentially with a minimum number of points and improvements on the simulations timing.

As long as the objective function and constraints are expensive to evaluate in the case of large analog cells, a Bayesian approach based on probabilistic surrogate functions or meta-models is proposed to spare a significant amount of evaluation time.

The surrogate functions represent practically a “smart” procedure for space-filling, where the real simulations are performed in a limited number of points of the input space and for all the other sampled points, a probabilistic prediction is employed. In our case, the Kriging meta-modeling was used as a trade-off between speed and optimum position finding.

The spatial correlation for statistical interpolation known as Kriging which is implemented in the surrogate models and the Kriging predictor were detailed.

The regression functions which are used to compose the Kriging model are presented in the constant, linear and quadratic cases. Practical implementations have shown that these orders of functions are sufficient in our analog cells optimization problem.

Also the correlation functions for stochastic processes are presented in the six cases used in our approach: Exponential, General Exponential, Gaussian, Linear, Spherical and Cubic. These functions are used to find the degree of correlation between different points of the input domain.

Finally, the Bayesian optimization method employing the Expected Improvement criterion is presented: the unconstrained case, the Expected Improvement subject to constraints and the modifications for the strong constraints case. For comparison and early explorations, a semi-automatic optimization method based on gradient descent algorithm was also implemented.

V

5. System-level Design and Optimization of Sigma-Delta Modulators

5.1. Introduction

The applications studied here are represented by continuous-time (CT) Sigma-Delta Modulator architectures based on micro-mechanical integrated resonators. The functionality of the continuous-time ΣΔ modulators and the main differences between continuous-time and their discrete-time counterparts were discussed in Chapter 2.

The CT Sigma-Delta Modulators are mixed-signal applications containing feedback loops which should be recursively optimized function of the system-level requirements and technology limitations in order to obtain stable and high-performance architectures [138].

In this context, it is mandatory to have high-level representations which can be simulated fast and accurately and which permit easy architecture reshaping when the performance and stability criteria are not met. Macro-modeling or behavioral modeling must be employed, but it is necessary to use a unified methodology for the simulation and optimization of all multi-domain components in order to obtain realistic figures of merit.

In this part, the concepts presented in Chapter 2, Chapter 3 and Chapter 4 are applied to the design and optimization of high order CT Sigma-Delta modulators applications.

With respect to the proposed design methodology, the first design steps consist in the system-level design at functional level and operators' level. Next, the non-optimized circuit level and technology implementations are performed. In the case of the two studied ΣΔ architectures, this was already achieved in two previous works of our department, e.g. [28], [39].

A 2^{nd} order application with ideal resonators (circuit and layout design) was studied in [28] in order to validate the solution and a manually optimized 6^{th} order architecture was presented in [39].

Then our contribution was to propose a complete methodology and the corresponding tools for automatic optimization of the composing analog cells along with their macro-models extraction and to re-implement the system level designs by using the extracted models and taking into account all the extracted low-level effects.

The automatic optimization of cells containing weakly non-linear amplification functions is performed by using a Bayesian Kriging strategy, as discussed in Chapter 4.

Next, their behavioral models are synthetized by using the figures of merit extracted or computed starting from transistor-level simulations. All the amplifiers of the Sigma-Delta filters are modeled using the automatic behavioral modeling technique presented in Chapter 3, thus we will provide here, for each amplifier nature, one example of realization.

Apart from the amplification functions, in order to model a whole Sigma-Delta modulator, additional components should be treated using the same design paradigm: analog-to-digital converters (ADCs), digital-to-analog converters (DACs) and micro-mechanical resonators used in the CT band-pass active filter.

The ADCs and DACs are mixed-signal cells designed by taking into account the constraints imposed by the active filter and the general architecture requirements. As explained in Chapter3, Section 3.3, they can be modeled in a generic environment like Simulink® or by using a dedicated mixed-signal language like VHDL-AMS. This is an advantage as long as the amplification functions behavioral models are also synthetized as Simulink® or VHDL-AMS/VerilogA by using the proposed SIMECT tool.

Due to their mixed-signal nature and the specific structure, the number of bits and others figures of merit, the ADCs and DACs cannot be automatically synthetized. We will propose Simulink® and VHDL-AMS models for the converters used in the Sigma-Delta modulators, based on the specifications used in [28] and [39].

Micro-mechanical parts like sensors and actuators can be integrated nowadays with electronic parts on the same chip in order to extend the functionality of the systems and to realize functions which were unavailable before. In the studied Sigma-Delta architectures, micro-mechanical resonators were employed to realize high-performance resonance functions.

These micro-mechanical systems cannot be modeled directly in generic environments (Simulink®). In this case, their functional or electrical equivalent model is used. Another option relies on dedicated simulation software for MEMS simulation like CoventorWare [139] from Coventor. This would provide the most accurate results for resonators simulations; still the licensing is costly for the dedicated tools, while the simulation environment is closed and cannot be easily interfaces with other modeling solutions.

A practical solution for the Sigma-Delta resonators modeling was the usage of mixed-signal languages (VHDL-AMS) due to their capability to integrate multiple physical fields.

In this chapter, we will start by presenting the general Sigma-Delta Modulator structures at the different levels of abstraction. Then the macro-modeling of the amplification functions will be detailed. This will permit to recompose the filter structure at high-level starting from the transistor-level cells. The ADCs and DACs models will be presented along with the different approaches used to model the micro-mechanical resonators. Multiple types of simulations are used to analyze the resulted systems, in order to validate the methodology. Finally, the stage-level and system-level results are discussed.

5.2. Sigma-Delta Modulators Architectures Employing LWRs

5.2.1. Sigma-Delta Modulators Topologies

Classically, a continuous-time ΣΔ modulator is composed with an analog low-pass or band-pass filter and a feedback loop integrating an ADC and one or more DACs. The applications studied here are band-pass ones, able to convert signals in the domain of hundreds of MHz to GHz, with effective bandwidths of tens of MHz. There are different possibilities to design the sigma-delta modulators filters. Some of them use multiple loops at the DAC level. Others, the feed-forward architectures, use a sum of terms at the ADC input.

In our case, two multi-path feed-forward structures are considered [28], [39]:

- A 2nd order band-pass ΣΔ modulator with the analog filter based on a single micro-mechanical lamb wave resonator (LWR);
- A 6th order band-pass ΣΔ modulator containing 3 LWRs, each resonator having its central frequency shifted by a few percent aiming to cover a larger input bandwidth.

The following characteristics are noted for the two architectures:

- they are 'pure-resonator' architectures [137];
- they work at a central frequency *fc=0.25fs*;
- the loop delay is 1.5 times the sampling period (*Ts*);
- the ADC sampling frequency is *fs=400MHz*;

For these structures working in multi-bit, the performance depends on the DAC quality. In order to have single DAC units in the topology, feed-forward architectures were chosen [39].

The original proposed topology (transfer functions level) for the 2nd order modulator [28] is presented in Figure 5.1.

It consists of the resonant filter transfer function (H_{res}), the ADC, DAC and a supplementary T_3 term, which is a fast connection between the DAC and ADC. The pure-resonator function H_{res} will be:

$$H_{res} = \frac{s}{s^2 + \omega_0^2} \tag{5.1}$$

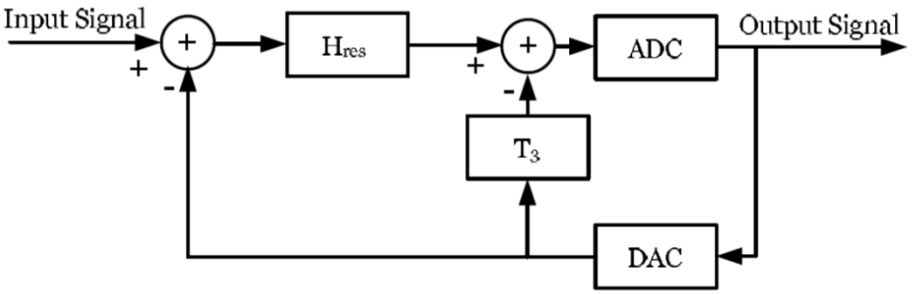


Figure 5.1 Transfer functions level of the 2nd order Sigma-Delta Modulator

Practically, the ideal resonance function cannot be achieved, due to a limited quality factor of the resonators (Q), thus the transfer function of the filter will be of type:

$$H_{res} = \frac{\omega_0 \, s}{s^2 + \frac{\omega_0}{Q} s + \omega_0^2} \tag{5.2}$$

The transfer functions topology of the 6th order modulator [39] can be found in Figure 5.2. It consists of 3 in series resonators and multiple re-loops. The transfer functions of type $\frac{b_k' \omega_k s}{s^2+\omega_k^2}$ are ideal pure-resonators functions, with the coefficients b_k, k_i and g_j used for filter tuning in order to assure the stability and performances of the modulator. This topology includes an in-series pure resonator filter at the input which eliminates the continuous components and the high frequency terms of the DAC. In practice, due to the limited quality factor of the resonators (Q_k), the resonant functions will be of the form 5.1.

The ADC and DAC are included in the structure, while the T_3 term is also a fast DAC connection in this case.

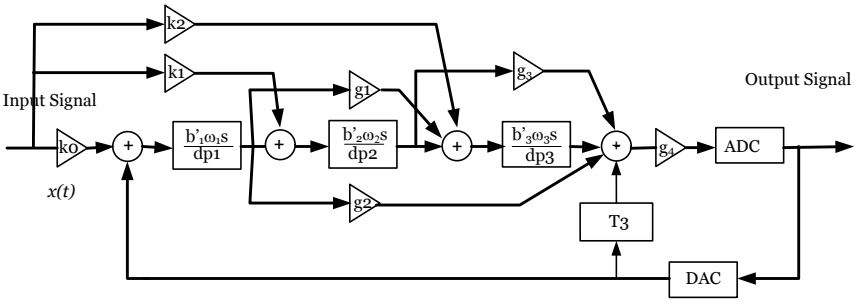


Figure 5.2 Transfer functions level of the 6th order Sigma-Delta Modulator

5.2.2. Lamb Wave Resonators

Micro-electro-mechanical systems (MEMS) such as piezoelectric resonators were recently integrated in manufacturing processes at the nanometer scale [140] [141] permitting to realize high-performance completely integrated applications.

In the case of micro-mechanical resonators, the resonance is obtained via the vibration of a micro-structure such as a vibrating beam, a ring resonator, a suspended micro-bridge or

membrane, etc. Multiple resonator architectures can be identified function of the type of waves established into the composing material volume. The principal categories include the following [142], [143]:

- Thickness mode resonators which can be of type thin film bulk acoustic resonator (FBAR) or solidly mounted resonator (SMR);
- Bulk Acoustic Wave (BAW) resonators – in which volume waves are established;
- Surface Acoustic Wave (SAW) resonators – in which surface waves are established;

In our case, for Sigma-Delta modulators, a specific kind of SAW technology was studied known as Lamb Wave Resonators (LWR) [144].

The LWR structure consists of a piezoelectric layer sandwiched between two thin electrodes. Figure 5.3 presents the LWR concept and a demonstrative realization of a resonator by CEA-Leti.

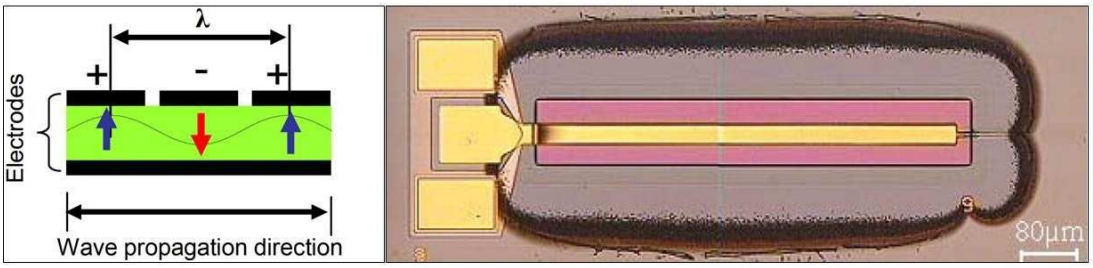


Figure 5.3 Lamb Wave Resonator structure and example realization from [6]

These resonators have some interesting features which make them a good choice for the design of band-pass CT Sigma-Delta modulators: they use as stimulus an input voltage and respond with a current, and also they are able to provide resonances at very high frequencies (up to some GHz [145] and up to 20GHz [146]) with high Q factors (up to 170000 [28]).

Unfortunately there are a number of drawbacks which should be correctly taken into account and treated in the early design stages in order to obtain stable CT Sigma Delta structures:

17. The LWRs present an anti-resonance effect which is very important and close to the principal resonance along with multiple parasitic resonance frequencies and harmonics of the principal resonance;
18. They require a large biasing voltage (e.g. 10 V) in order to reduce the impedance at the resonance frequency, which is distinctly varied (e.g. between 0÷12 V) to adjust the position of the principal resonance. This range of biasing voltages is not permitted in low-consumption integrated circuits.
19. MEMS resonators are generally encapsulated in order to increase their Q-factor, but this process is costly and not easily compatible with integration technologies.

Thus, the real response of a LWR is generally not a pure-resonance, and it includes a predominant parasitic capacitive effect along with multiple harmonics of the principal resonance. The real response can be modeled by using the generalized Butterworth-Van Dyke electrical model for micro-mechanical resonators [28].

The Figure 5.4 presents the generalized electrical model of a LWR cell, including the principal resonance and two harmonics along with a generic frequency response. The group $R_mC_mL_m$ and the capacity C_0 account for the dominant effects of this structure: the principal resonance and the parasitic capacitive effect, while the other series resonant groups are responsible for the harmonic content.

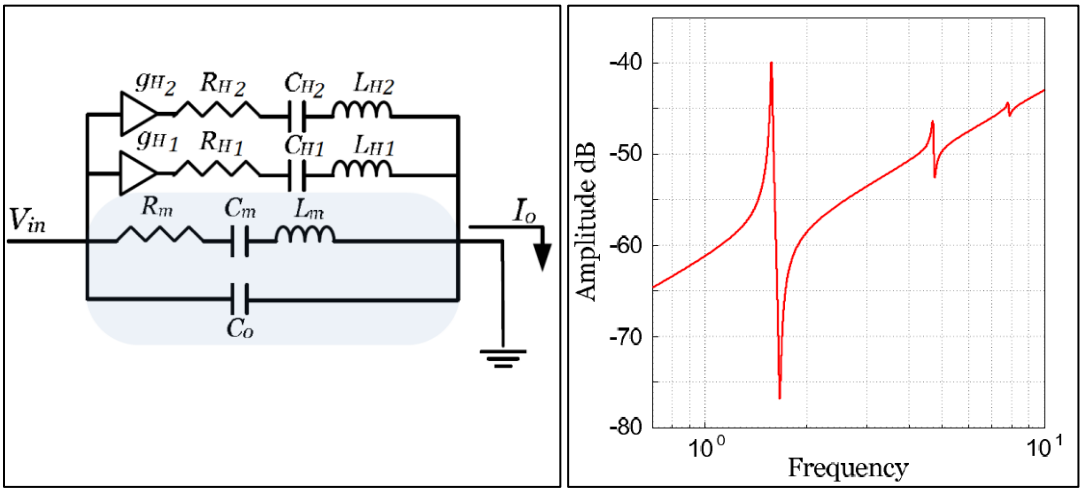


Figure 5.4 Lamb Wave Resonator electrical model and generic response

For the conception of CT band-pass Sigma-Delta modulators, one need pure-resonance functions with Q factors of minimum order 50 [39].

Thus, the parasitic capacitance effect is a drawback and should be compensated. This effect along with the anti-resonance effect can be treated by using a symmetric differential structure for the filter [147]. Practically, a lattice-like structure (L) is used, employing compensation capacitance paths (Zc), as presented in Figure 5.5.

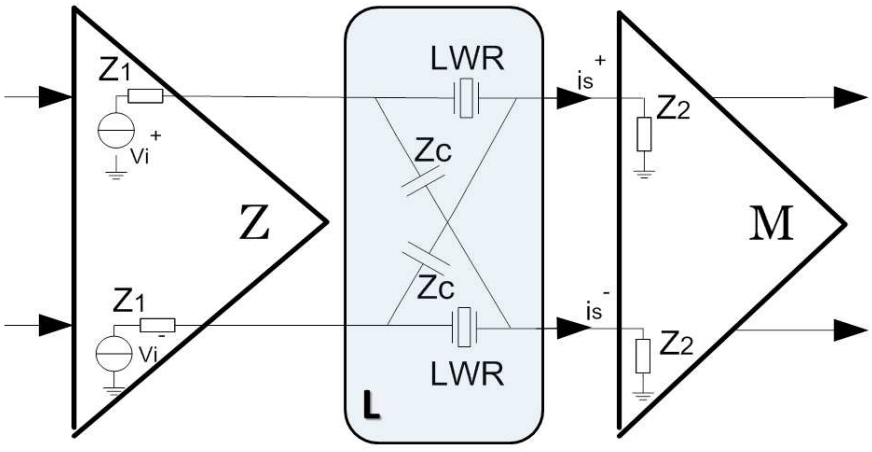


Figure 5.5 Differential filter structure for LWR compensation

The filter structure in the Sigma-Delta modulators will then be composed starting from the basic filtering cell presented in Figure 5.5. This cell, apart for its cancellation proprieties, includes a trans-impedance amplifier (*Z*) to convert an input current to a voltage used to load the resonators and a current-to-current converter (*M*) for impedance adaptation and to employ the specific gains corresponding to the feed-forward coefficients in the Sigma-Delta topologies.

Practically, the basic filtering cell will incorporate a supplementary trans-conductance amplifier (*Gm*) because the studied Sigma-Delta modulators are used for the A/D conversion of voltage levels.

As we will further present, in the design of these amplifiers we will take into account, as optimization criteria, the gains corresponding to the specific coefficients in the Sigma-Delta topologies, the non-linear effects which should be as low as possible, and the specific impedances, based on the circuit functionality in the topology.

One important study concerns the effect of amplifiers non-idealities on the resonant response of LWRs.

In this context, it is possible to derive the effect of the finite input/output impedances values for the amplifiers and that of the compensation impedance. Ideally, if we consider that the LWRs have no parasitic capacitance and harmonics, Z_{LWR} will respect the pure-resonance function with high, finite *Q* factor (5.2). It can be shown that when we compensate the LWRs, their *Q* factor is degraded due to these impedances. In Figure 5.5, considering the differential input voltage $\Delta v = v_i^+ - v_i^-$ and the differential output current $\Delta i = i_s^+ - i_s^-$, the differential impedance of the resonant structure can be expressed as:

$$Z_{diff} = \frac{\Delta v}{\Delta i} = \frac{Z_{LWR} + (Z_1 + Z_2)\left(1 + \frac{Z_{LWR}}{Z_C}\right) + 4\,Z_1 Z_2 \cdot \frac{1}{Z_C}}{1 - \frac{Z_{LWR}}{Z_C}} \tag{5.3}$$

The expression 5.3 shows that a high resistive contribution from the Z_1 and Z_2 amplifiers impedances increases the resistive part of the differential resonant structure, thus its quality factor will be degraded. This will be an important criterion in the conception of the amplifiers, where the optimization will focus on minimizing their resistive contribution at maximum 10-15% of the characteristic resistance of LWRs, thus a *Q>50* can be assured.

Furthermore, even Z_C compensation degrades the resonators, and its effects are increasingly worst in higher frequency. As we will further discuss, in the compensation of LWRs, a more complex structure is used, not a pure capacitance.

As long as the harmonic effects are intrinsic technological effects, the main concern which remains to be treated in the system level design is related to the modeling of parasitic harmonic resonances of LWRs leading to the transition from ideal resonance functions to real measured functions, obtained in the technological process.

5.2.3. Sigma-Delta Modulators with LWRs

The operators' level for the two topologies is implemented starting from the electronic control circuit of the LWRs presented in Figure 5.5, resulting in a differential structure, able to remove the intrinsic anti-resonance effect of the resonators [39].

Since the summing operators are easier to implement using currents than voltages, the structure uses current converters in the nodes where summations are necessary.

As discussed, the main motivation for a structure where voltage-to-current and current-to-voltage converters are implemented resides on the fact that the lamb wave resonators use as stimulus a voltage and respond in current.

Figure 5.6 presents the operators' level for the 2nd order band-pass ΣΔ modulator. Its filter consists of one unipolar to differential voltage-to-current converter (*Gm*), two current-to-voltage converters (*Z* is a fully differential trans-impedance and *Z'* – a unipolar trans-impedance amplifier), one differential to unipolar current-to-current converter (*M*) and the Lamb Wave differential cell.

A 3-bits flash ADC was used for its high-speed conversion, having a direct output (the modulator *Output Signal*) and an inverted one, used to create the differential flow after the DAC connections.

The DAC is a current-type converter, providing a maximum differential output current – full scale current I_{fs}. It uses a parallel architecture composed of 7 DC current sources, where each cell delivers $I_{fs}/7$. The DAC1 represents the slow DAC connection, while DAC2 is the fast one.

The need to differential structures for the ADC and DAC resides on the interfacing requirements for the feed-back path.

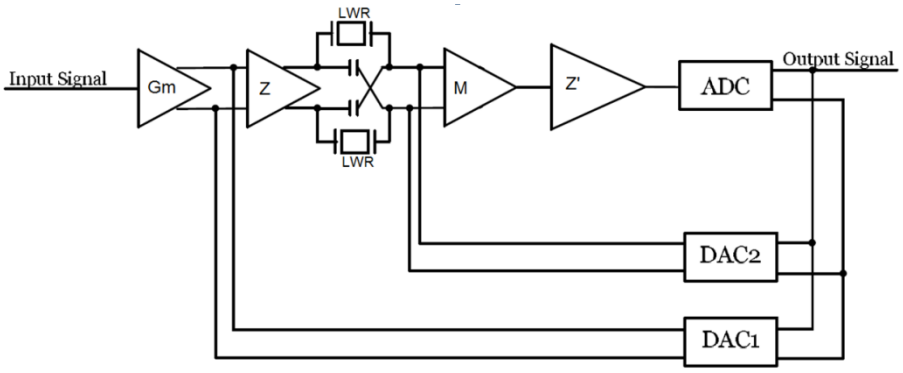


Figure 5.6 Operators level of the 2nd order Sigma-Delta Modulator

The operators' level of the 6th order band-pass ΣΔ modulator is also implemented starting from the differential filter cell [39].

Figure 5.7 presents the architecture, based on the same principles like in the 2nd order case. Its filter is composed with three trans-conductance amplifiers (*Gm1-Gm3*), four trans-impedance amplifiers (*Z1-Z4*), and three multi-output current-to-current converters (*M1-M3*).

This structure includes multiple feed-forward connections, aiming to provide multiple degrees of freedom for modulator tuning [28].

As in the 2nd order case, the ADC is a 3-bits flash structure and the DAC is a differential current-type converter. In Figure 5.7, the DAC1 provides the full loop delay, while DAC2 functionality consist in a fast conversion re-loop.

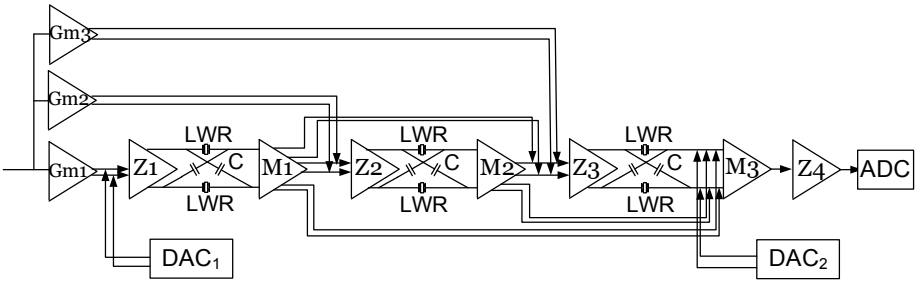


Figure 5.7 Operators level of the 6th order Sigma-Delta Modulator

The 2nd order and the 6th order modulators were implemented at transistor-level using an *Austriamicrosystems* BiCMOS 0.35 µm low power technology working at 5V power supply. Since CT Sigma-Delta modulators are mixed-signal circuits where analog and digital circuits are implemented on a monolithic chip, a hybrid technology was chosen.

All the simulation results for the first realizations at transistor level of the 2nd order and the 6th order modulators and the layout level for the 2nd order modulator are presented in the previous works [28] and [39].

5.3. Modeling Strategies for the Sigma-Delta Modulators Components

5.3.1. General Approach for the Design and Optimization of ΣΔ Modulators

The 2nd order and the 6th order architectures which were presented were non-optimal in terms of performance at block level or they were manually optimized.

The process of manual optimization of blocks is very time-consuming and cannot assure a system level optimal solution. Multiple re-loops from system-level to transistor-level and vice-versa should be performed in order to have a final optimal solution [138].

This is why the conception methodology discussed in Chapter 2 and the corresponding design tool (SIMECT [148]) – based on the algorithms presented in Chapter 3 and Chapter 4 were applied to the design and optimization of the two Sigma-Delta structures.

Once the transistor-level and potentially the layout-level were implemented, our task was to optimize all the composing circuits, to extract realistic high-level models for these blocks and to re-implement the system level structures. Then the overall converters performances were evaluated when taking into account the analog imperfections at system level and compared with the transistor level results for coherence.

Since the initial design of the Sigma-Delta structures was conducted by using ideal RLC models of the LWRs, a further research direction was to study the effects of real resonators functions and parasitic resonance effects on the global modulators figures of merit.

The following steps are considered for the modeling of the composing blocks:

- In a first stage, the weakly non-linear amplifiers of the Sigma-Delta active filter are optimized and a macro-model of type VHDL-AMS/Verilog-A or SIMULINK is automatically extracted for each component. Here we present three examples of optimization and extraction, corresponding to the three classes used in the filter: trans-conductance amplifier, trans-impedance amplifier and current-to-current application.
- Next, the strategies for modeling the multi-physics components – the micro-mechanical resonators in our case - are presented: the structural approach, the functional approach and a behavioral one. In each case, the benefits and drawbacks of the method are highlighted.
- Once this is accomplished, the basic band-pass filter cell is modeled and its response is inspected in multiple cases: the ideal function, the function at transistor level and at macro-models level.
- A further step consists in the modeling of the mixed-nature blocks of the modulators - the ADCs and DACs – starting from the initial specifications and the resulting transistor-level characteristics.

- Once the blocks modeling is accomplished, the system-level is re-implemented for each modulator, the filters stages are presented and stage-level/system-level results are compared for coherence with transistor level simulations results. In a first case we use ideal resonance functions for the micro-mechanical devices;
- Finally, the real resonance functions for the micro-mechanical devices, eventually including multiple parasitic resonances extracted from measured characteristics are employed in the modulators architectures. The respective results are commented and final conclusions are presented.

The VHDL-AMS language or the SIMULINK environment were mainly used for blocks modeling and high-level implementations [149]. Verilog-A models were partially used in situations where a simulation speed improvement was observed or when convergence problems appeared due to simulators engine internal parameters.

Additionally, the benefits offered by the multi-level conception methodology in Chapter 2 were exploited: simulations were conducted where a part of the design was at transistor level and other blocks at behavioral or functional level, allowing to verify the correct functionality of a design portion, while maintaining reasonable simulation times; simulations were implemented where multi-language and multi-physics models were employed, thus permitting a great flexibility in the design.

5.3.2. Optimization and Modeling of the Trans-impedance Amplifiers (TIA)

The presented optimization and model extraction techniques are applied to the design of a fast trans-impedance amplifier used in the architecture of the CT Sigma-Delta modulators.

The amplifier *Z* in Figure 5.6 and *Z1, Z2, Z3* in Figure 5.7 are differential TIA, composed of two single-ended amplifiers, while *Z'* and *Z4* are designed starting from the actual single-ended structures. The schematic of the single-ended cell is shown in Figure 5.8.

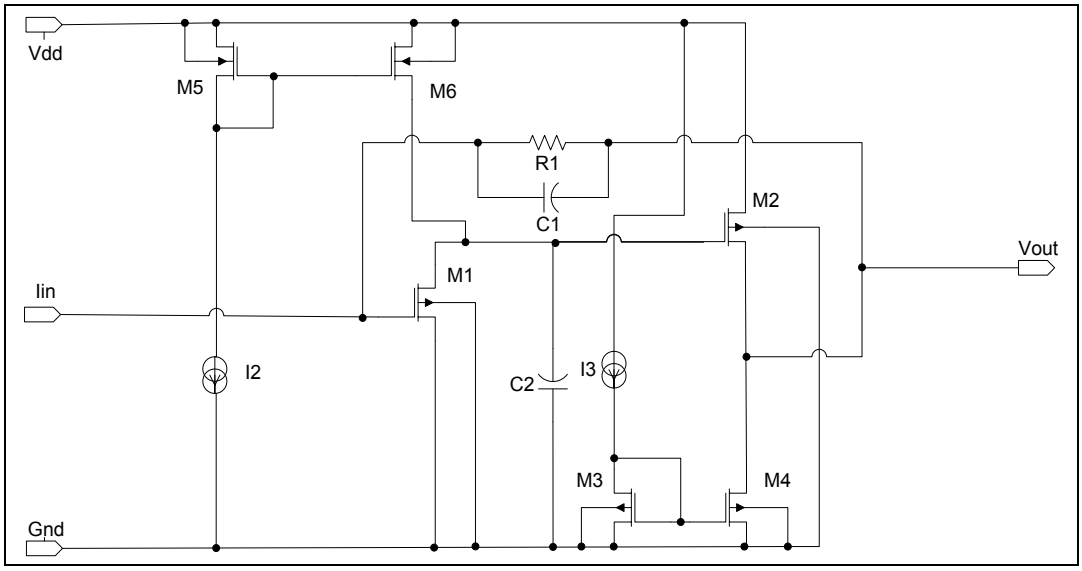


Figure 5.8 Unipolar current to unipolar voltage amplifier

(single-ended transimpedance amplifier)

The functionality of this amplifier is based on the transistors M1 and M2 and resistor R1 which act as a current-controlled voltage source, the source's gain being equal to R1. The pairs M3-M4 and M5-M6 and the current generators I2 and I3 are real current sources used to bias the main transistors M1 and M2. When the structure is connected to a high capacitance, it may become unstable, and capacitances C1 and C2 help stabilize the amplifier [70].

To simulate and optimize the current-to-voltage amplifiers, we studied first their characteristics and the type of signals they should treat. The output was also considered as the first three amplifiers are connected to the LWR filters and should not affect their response. The first stage amplifiers (*Z* and *Z1*) are the ones who suffer the most constraints. They receive as direct input the signal from the *Gm* amplifiers and also as feedback the output of the slow DACs, which are theoretically square signals, but with steep fronts. This leads to output current peaks, since the TIAs charge directly the parasite capacitances of the LWR filters and the compensation capacitances. The next two amplifiers on the 6^{th} order architecture (*Z2* and *Z3*) imply fewer constraints in the sense that input fronts are less stiff, thus the output current characteristics due to capacitors are less important. The output stage amplifiers (*Z'* and *Z4*) are

only charged by the input capacitances of the ADC and does not output signal to LWR filters. As a result, they do not need compensation capacitances.

The criteria retained for the optimization of the TIA amplifiers contain the following figures of merit:

- the gains of the amplifiers, required to form the coefficients of the band-pass filter in the Sigma-Delta topologies;
- the resistive part of the output impedances (for *Z*, *Z1*, *Z2* and *Z3*) which should not exceed 10-15% of the resistive part of LWRs impedances, Z_{out}^{LWR} which is 100Ω (in the band of interest);
- the input impedances should also be as low as possible, because the TIA configurations are current sinks;
- the nonlinearities on the output and input, as low as possible, with a maximum 1% nonlinearity on the output, for a correct linear transfer of the signal, without distortions.

The automatic optimization strategy and detailed optimization steps for a TIA amplifier were presented in Chapter 4, Section 4.7. We will retain here only the optimization goals and the results for the objective function and constraints in this case (Table 2-1).

Parameter	Type	Desired Value(s)	Value(s) after optimization
Input impedance (Z_{in})	Objective	Minimum	11.9 [Ω]
Output impedance (Z_{out})	Constraint (Normal)	Minimum: [1 ; 10] [Ω]	3.35 [Ω]
Direct gain (g^d)≈ R_1(1950 Ω)	Constraint (Normal)	[1930 ; 1969] [Ω] (max. 2% gain error)	1947.7 [Ω]
Input Nonlinearity (nl^{in})	Constraint (Strong)	desired <1% (acceptable <5%)	0.026
Output Nonlinearity (nl^{out})	Constraint (Strong)	<1%	$164.5 \cdot 10^{-6}$

Table 5-1 Transimpedance amplifier *TIA 1* optimization goals and results

Next, for macro-modeling this amplifier we will perform a DC analysis with *Iin* varying in [-100; 100] µA, afterwards a parametric analysis over the previous defined DC, with *Iout* varying also in [-100; 100] µA, AC analyses in the frequency domain [1Hz; 1THz] from which we will extract *s*-models of maximum orders 3 for all the frequency functions. A transient analysis over 10 periods of *f0*=100MHz by using a square input signal was also enabled.

Running the simulations, the results are plotted in two figures, one for the DC and Parametric analyses results and one for the AC and transient analyses results. The DC-

Parametric figure and the AC-transient figure in the case of the TIA amplifier can be found in Annex B of this work.

The following figures of merit extracted from DC and parametric analyses can be summarized:

20. The amplifier gain extracted from simple DC and parametric DC: 1947.7 Ω, inverted output signal;
21. The output impedance extracted from parametric DC: 3.35 Ω;
22. The input impedance extracted from simple DC and parametric DC: 11.9 Ω;
23. The reverse gain extracted from parametric DC: 3.26 Ω^{-1};
24. All the input and output offsets, in our case the extracted ones are $V_{out\ DC}$ = 2.56 V and $V_{in\ DC}$ = 2.36 V;
25. All the non-linearity coefficients (for the output – on the top and for the input – on the bottom of the DC figure in Annex B): NLX (order 2), NLX (order 3), NLY (order 2), NLY (order 3), IMOD (intermodulation input-output), NLTOT - the maximum total non-linearity error for the whole input or output range;
26. A 2D plot of the output voltage function of the input current for output current=0;
27. A 2D plot of the input voltage function of the input current for output current=0;
28. A 3D plot of the output DC voltage function of the input current and output current;
29. A 3D plot of the input DC voltage function of the input current and output current;

The following figures of merit extracted from AC and transient analyses are given:

- The resistive part of the input and output impedances at the frequency *f0*=100MHz: $Z_{in}|_{f=f0}$ = 85.24 Ω, $Z_{out}|_{f=f0}$ = 3.78 Ω;
- The delay introduce by the circuit at the *f0* frequency (Δt=0.24ns) and its quality factor (not applicable in this case, because it is not a resonant circuit);
- The transfer-function cutoff frequency (*FC Trans*): f_c=1.32 GHz;
- Input and output impedances equivalent RLC model parameters;
- AC plots of the measured input impedance, output impedance and transfer function;
- AC plots of the extracted s-models and/or RLC models for the three characteristics;
- The transient response (*out*) along with the generated transient input signal;

In the next step, prior to macro-model extraction, the model topology as s-functions is inspected. Figure 5.9 depicts the structure of the macro-model for the TIA circuit.

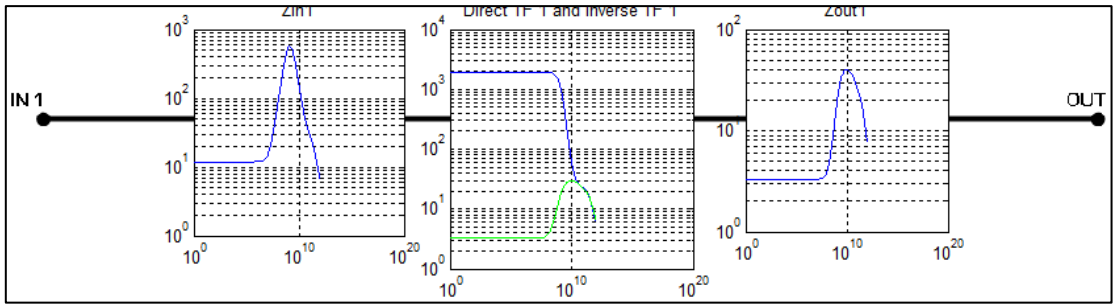


Figure 5.9 TIA Amplifier Macro-model topology as s-functions

The SIMULINK and HDL models of the TIA are extracted by SIMECT. The SIMULINK model will have the structure presented for unipolar amplifiers, as illustrated in Figure 5.10.

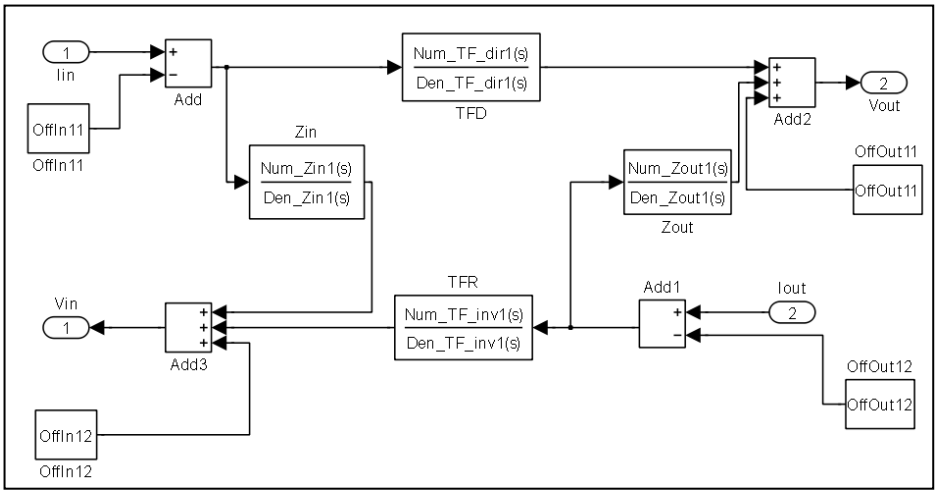


Figure 5.10 TIA Amplifier Simulink macro-model

The 3rd order VHDL-AMS linear models used for simulations with Dolphin Integration SMASH, Cadence AMS simulator and the Verilog-A model used with Cadence AMS are presented in Annex C of this work.

The synthesis of these models relies on the concepts presented in Chapter 3, Section 3.5 and on the aforementioned methods for simulations speed optimization: the transfer functions decomposition, splitting of the vectors containing the poles and zeros into singular elements to be used in the *Laplace Transfer Functions* attributes and out-of-band poles/zeros removal when possible.

5.3.3. Optimization and Modeling of the Trans-conductance Amplifiers (Gm)

The model extraction technique is applied next to the design of a voltage-to-current converter. The schematic of a real amplifier is shown in Figure 5.11. This structure is a unipolar input to differential output converter.

The symmetry between I_{out+} and I_{out-} and their linearity is important as the converter is situated on the modulator input and any resulting error spreads all over the modulator. As a result, the amplifier is based on the BJT-pair Q0 and Q1 and a resistor (R). They are employed because they have a higher cut-off frequency and a larger linearity zone compared with MOS-transistors. However, the swing of the input voltage (V_{in}) must not exceed the linearity zone of the transistors. V_{dc} is a low noisy DC-voltage equal to the DC-offset of V_{in} to ensure the symmetry between I_{out+} and I_{out-} [39].

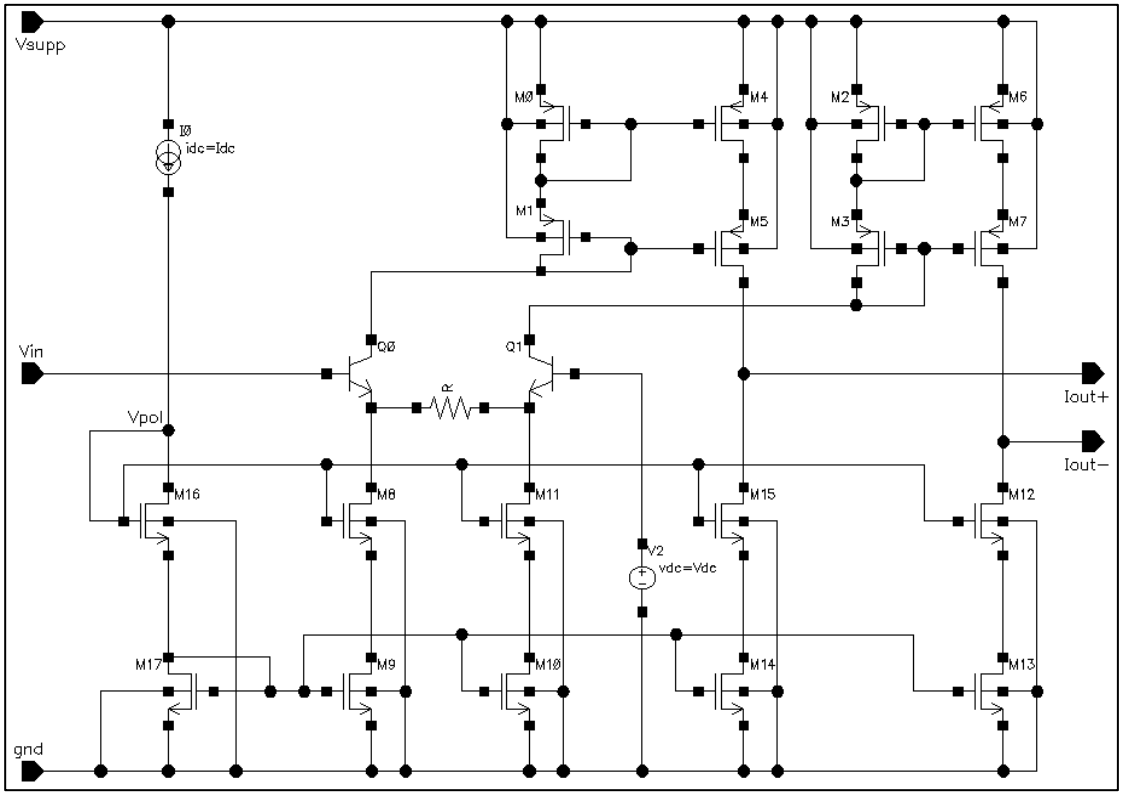


Figure 5.11 Unipolar voltage to differential current amplifier (differential *Gm*)

Once the schematic is implemented and basic DC/AC functionality is tested, the automatic optimization process is performed. In this case, because the *Gm* amplifiers are situated at the input stage of the modulator, they receive the input voltage which should be converted to differential currents for summing in different circuit points. An important optimization goal will be the minimization of the output common-mode contributions and of the output non-linarites of the amplifiers. Another concerned aspect is to assure the desired gains and input/output impedances in DC and at high frequency.

The amplifier gain is controlled by the resistance *R*. For example, if we want a full-scale input voltage of amplitude 1V (+/-0.5V) to provide an output current of +/-100μA on each branch, the desired trans-conductance should be 200μS, implying a resistance of 5Ω. For each *Gm* amplifier, the resistance value was adjusted to the desired gain in a first step, and then the overall structure was optimized in order to achieve the proposed figures of merit.

Table 5-2 presents the optimization goals and the results obtained for the objective function and constraints in the case of the *Gm* amplifier. In this case, one objective function, three normal constraints and two strong constraints are considered.

Parameter	Type	Desired Value(s)	Value(s) after optimization
Common-mode direct gain (g_{MC}^{d})	Objective	Minimum	$1.005 \cdot 10^{-12}$ [Ω^{-1}]
Differential direct gain at f0 ($\left\|TF_{diff}^{out}\right\|_{f=f0}$)	Constraint (Normal)	[140 ; 150] $\cdot 10^{-6}$[Ω^{-1}] (ideal = 148 μS)	$144.7 \cdot 10^{-6}$ [Ω^{-1}]
Output impedance (Z_{out})	Constraint (Normal)	>100 [kΩ] (↑↑↑)	318.8 [kΩ]
Input impedance (Z_{in})	Constraint (Normal)	>100 [kΩ] (↑↑↑)	50.13 [MΩ]
Input Nonlinearity (nl^{in})	Constraint (Strong)	<1%	0.080
Output Nonlinearity (nl^{out})	Constraint (Strong)	<1%	0.0169

Table 5-2 Transconductance amplifier *Gm* optimization goals and results

Once the automatic optimization process is performed, it is possible to extract the macro-model of the amplifier. To this aim, for the trans-conductance amplifier we will perform a DC analysis with V_{in} varying in [-100; 100] mV, afterwards a parametric analysis over the previous defined DC, with V_{out} varying in [-100; 100] mV, AC analyses in the frequency domain [10kHz; 1THz] from which we will extract *s*-models of maximum order equal to 5 for all the frequency functions. A transient analysis over 10 periods of *f0*=100MHz by using a sinusoidal input signal was also enabled.

Running the simulations, the DC results can be plotted in two different manners in the case of differential-output designs: by showing the extracted parameters for each separate output or by showing the differential or common mode figures of merit.

The Annex B of this work contains two figures which illustrate the DC and Parametric figures of merit in the case when separate outputs are considered and two figures containing the

same types of parameters in the common-mode/differential case. An AC-transient figure is also presented in the case of the Gm amplifier.

The figures of merit extracted from DC and parametric analyses for the trans-conductance amplifier when studying the I_{out-} output can be summarized as follows:

30. The amplifier direct gain extracted from simple DC and parametric DC analyses (a small difference can be observed, the parametric DC being more exact): -57.3 $\mu\Omega^{-1}$; In our case an important figure of merit is the amplifier direct gain at f0 which is extracted from AC analyses as we will further see.
31. The output impedance extracted from parametric DC analyses: 318.8 kΩ;
32. The input impedance extracted from simple DC and parametric DC analyses (again a small difference between the values can be observed, due to the parametric DC variation, resulting in an exact value for the parametric analysis): 50.13 MΩ;
33. The reverse gain extracted from parametric DC analysis: $1.7\text{x}10^{-20}$ Ω; Practically this amplifier is highly anti-symmetric in DC, but the reverse gain increases with frequency, potentially reaching values of the same magnitude as the direct gain at $f \gg f_0$.
34. All the input and output offsets and all the non-linearity coefficients are extracted, as we explained before (for the considered output I_{out-} – on the top and for the input – on the bottom of the DC-Parametric figures in Annex B): NLX (order 2), NLX (order 3), NLY (order 2), NLY (order 3), IMOD (intermodulation input-output), NLTOT - the maximum total non-linearity error for the whole input or output range;
35. A 2D plot of the output current function of the input voltage for output voltage=0;
36. A 2D plot of the input current function of the input voltage for output voltage=0;
37. A 3D plot of the output DC current function of the input voltage and output voltage;
38. A 3D plot of the input DC current function of the input voltage and output voltage;

For the I_{out+} output, the extracted figures of merit are symmetric on the output and similar on the input, when compared to the I_{out-} output.

The figures of merit for the differential and common-mode DC and parametric analyses can be summarized as follows:

39. The amplifier differential direct gain (the parameter direct gain in the differential DC-Parametric figure from Annex B) is -28.7 $\mu\Omega^{-1}$ while the common-mode direct gain (the parameter direct gain in the common-mode DC-Parametric figure from Annex B) is of order of 10 $p\Omega^{-1}$, thus assuring a very good DC common-mode rejection;
40. The other macro-model parameters: input and output impedances, input and output offsets, all the non-linearities are extracted or computed from the simple DC and

parametric DC differential and common-mode analyses in the same manner as for the separate outputs;

The figures of merit extracted from AC and transient analyses can be summarized as follows:

- The resistive part of the input and output impedances at the frequency *f0*=100MHz – in this case, two output impedances are presented, because the design is differential: $Z_{in}|_{f=f0}$ = 85.733 kΩ, $Z_{out1}|_{f=f0}$ = 8.680 kΩ, $Z_{out2}|_{f=f0}$ = 9.672 kΩ;
- The delay introduce by the circuit at the *f0* frequency (Δt=2.06ns) and its quality factor (not applicable in this case, because it is not a resonant circuit);
- The transfer-function cutoff frequency (*FC Trans*): f_c=4.36 GHz;
- Input and output impedances equivalent RLC model parameters – again, in this case the models are presented for the two output impedances;
- AC plots of the measured input impedance, output impedance and transfer function;
- AC plots of the extracted s-models and/or RLC models for the three characteristics;
- The transient response (*out*) along with the generated sinusoidal input signal;

It should be noted that for the differential output designs, the transfer function and the output impedance are presented as differential and common-mode characteristics in the two corresponding AC plots.

Once all the parameters needed for model synthesis are computed, prior to macro-model extraction, the model topology as s-functions can be inspected. Figure 5.12 depicts the structure of the macro-model for the *Gm* circuit.

The SIMULINK model is synthetized in order to have an adequate structure for differential output amplifiers, as illustrated in Figure 5.13.

The HDL behavioral models for the *Gm* amplifier are subsequently extracted. A 5th order VHDL-AMS linear model used for simulations with Dolphin Integration SMASH, a VHDL-AMS model for the Cadence AMS simulator and a Verilog-A model used with Cadence AMS are presented in Annex C of this work. They are automatically synthetized by using the model extraction algorithms implemented in the SIMECT tool. In this case, the transfer functions decomposition and the vectors splitting techniques are used.

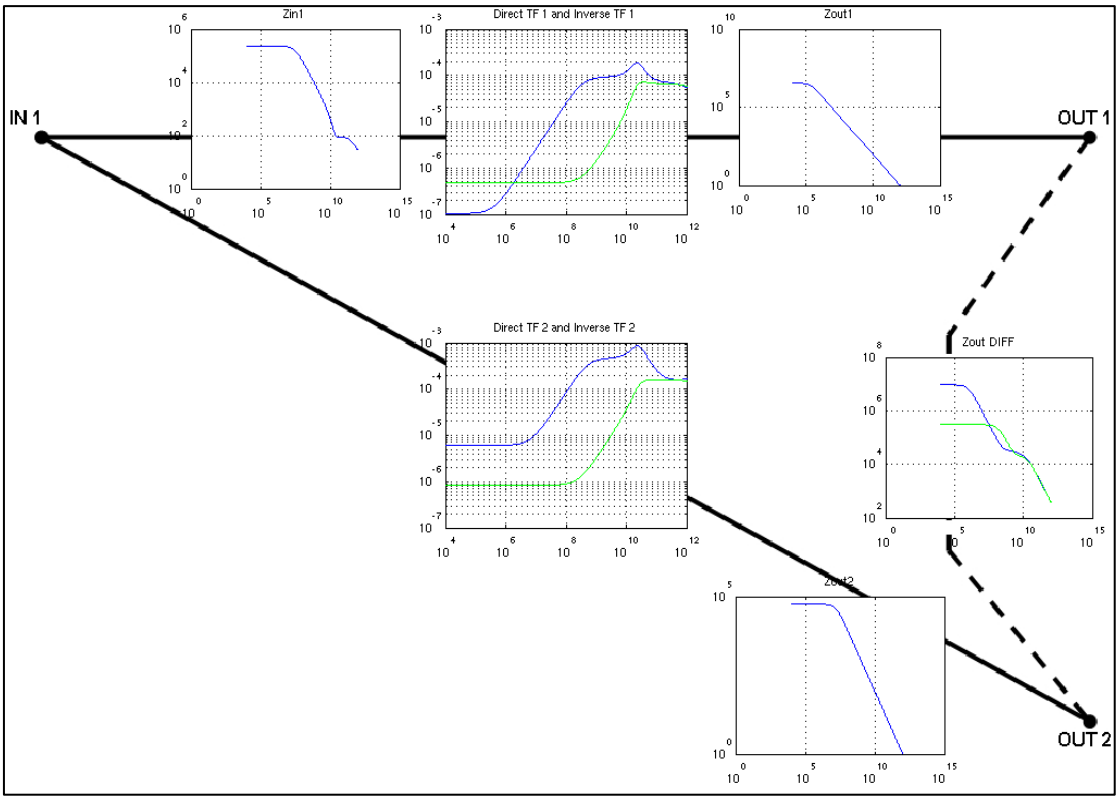


Figure 5.12 Trans-conductance amplifier Macro-model topology as s-functions

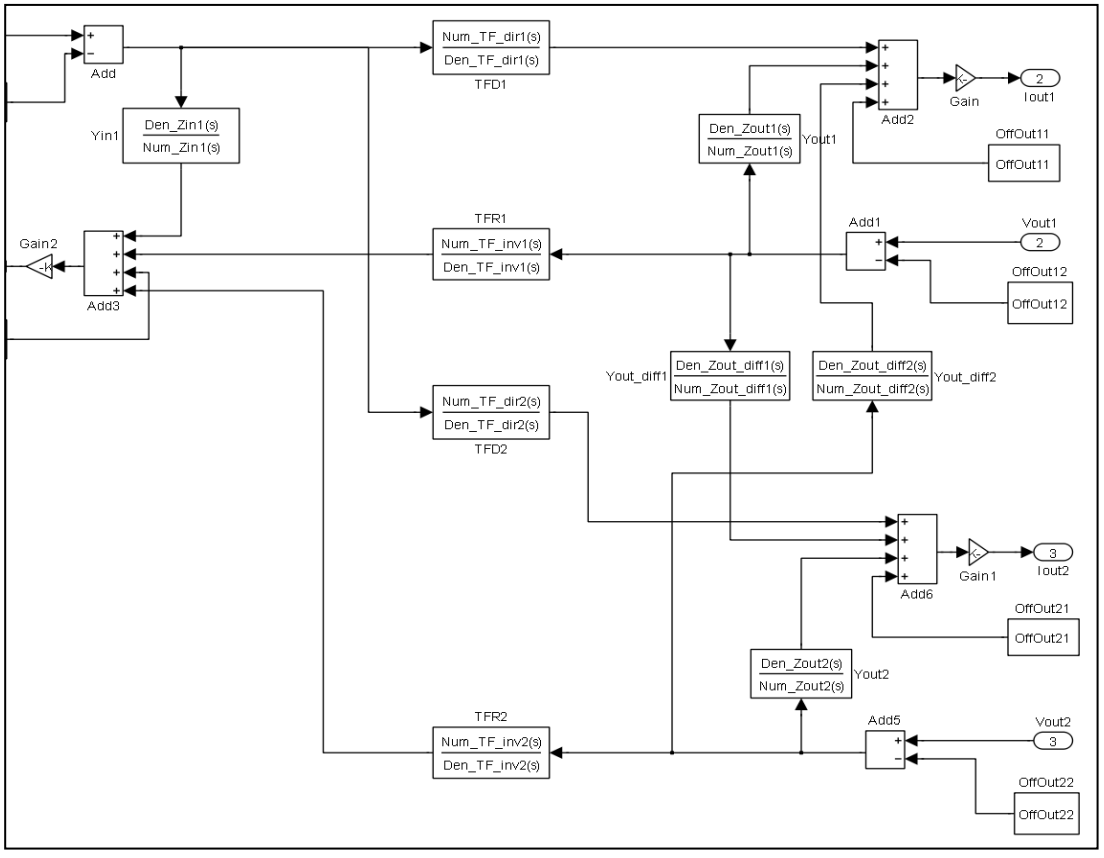


Figure 5.13 Trans-conductance amplifier Simulink macro-model

5.3.4. Optimization and Modeling of the Current Mirrors (M)

The automatic optimization and the model extraction techniques are finally applied to the design of a fully differential current-to-current converter (Figure 5.14), used in multiple stages of the studied Sigma-Delta Modulators with different number of outputs and specific gains. Here we present the single-output case, but an analogous process was performed for the multi-output converters.

Like in the case of the *Gm* amplifiers, all the features of the hybrid bipolar-CMOS 0.35 μm Low Power target technology were used.

The input stage of this amplifier is a BJT common-base arrange (Q0-Q1) able to provide a low-input impedance, because this amplifier collects the output currents of a low-impedance acoustic resonator. Then, subsequent pairs of MOS-transistors (N-mirrors and P-mirrors) are used to form the positive and negative signal paths. For a comprehensive description of this class of amplifiers, refer to [28].

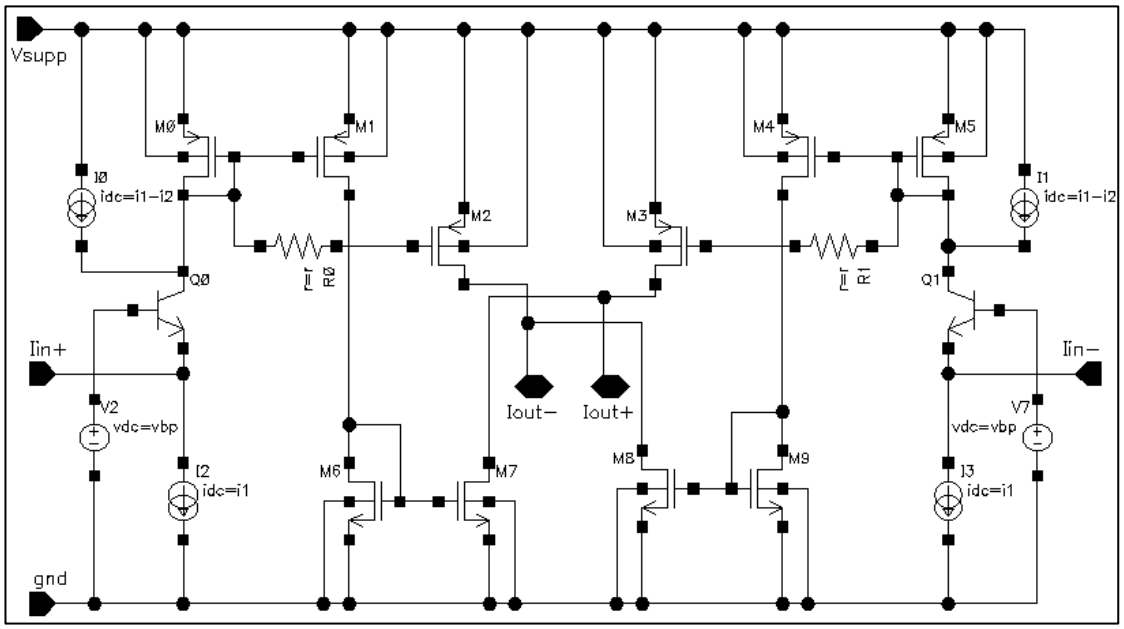


Figure 5.14 Differential current to differential current amplifier (differential mirror)

Like for the previous blocks, the formulation of the optimization objective and constraints for the differential current mirrors takes into account their system level functionality and their location in the general topology.

The current mirrors are loaded by a LWR differential cell and usually drive a trans-impedance amplifier. Taking into account the usual DC value of the output impedance for the LWRs which is $Z_{out}^{LWR} = 100\Omega$, the current mirrors should provide a low input impedance (maximum 10-15% of the Z_{out}^{LWR}), potentially in all the frequency band, thus they will not degrade the resonant effects of the LWRs. As current sources, they should provide relatively high output impedance and accurate linearity on the output and input.

Another optimization aspect concerns the differential and common-mode gains and the symmetry of the outputs. The different values for the differential gains should be assured on each output, in the entire frequency band, while the contributions in common-mode should be as low as possible. An interesting aspect for this block resides in the usage of a semi-parasitic RC filter on the output P-mirrors, able to provide the same delay between the N-paths and P-paths for each output [28]. This filter controls directly the variations of the common-mode gain in high frequency ($f \to 1GHz$). If the R value of the resistance is not correctly set, the common-mode transfer function will increase dramatically at higher frequencies, so a correct optimization of the DC common-mode gain will be of no use. Introducing R in the optimization process will only increase the number of variables, while providing only a bijective relationship with this gain. This is why a post-optimization fine tuning of R is done, to correctly set a quasi-constant value of the common-mode gain in the studied frequency domain.

Table 5-3 summarizes the optimization goals and the results obtained after the automatic optimization for the objective function and constraints in the case of the M amplifier. In this case, one objective function, three normal constraints and two strong constraints are considered.

Parameter	Type	Desired Value(s)	Value(s) after optimization
Common-mode direct gain (g^d_{MC})	Objective	Minimum	0.003÷0.004
Differential direct gain (g^d_{DIFF})	Constraint (Normal)	[0.95 ; 1.05] (ideal = 1.00)	1.02
Input impedance (Z_{in})	Constraint (Normal)	<15 [Ω] (↓↓↓)	12.23 [Ω]
Output impedance (Z_{out})	Constraint (Normal)	>5 [kΩ] (↑↑↑)	7.9÷8.2 [kΩ]
Input Nonlinearity (nl^{in})	Constraint (Strong)	<1%	$2.65 \cdot 10^{-3}$
Output Nonlinearity (nl^{out})	Constraint (Strong)	<1%	$4.36 \cdot 10^{-5}$

Table 5-3 Current mirror M optimization goals and results

For the current mirror macro-models extraction the following analyses were performed: a DC analysis with I_{in} on each input varying in [-2; 2] mA, a differential DC with the differential input current varying in [-1; 1] mA, afterwards a parametric analysis over the previous defined DC analyses, with V_{out} varying in [-1; 1] V, AC analyses in the frequency domain [10kHz;

1THz] from which we extracted *s*-models of maximum order equal to 2 for all the frequency functions. A transient analysis over 10 periods of *f0*=100MHz by using a square input signal was also enabled.

The Annex B of this work contains two figures which present the DC macro-model parameters for each separate output.

It should be noted that in the case of differential-input differential-output designs, some supplementary characteristics are extracted when compared to unipolar-input differential-output designs (the Trans-conductance amplifier example in the precedent Sub-section):

- The *C.M. gain* in the two figures represents the gain obtained for each separate output, when the input is varied in common-mode;
- The *direct gain* in the two figures represents the gain obtained for each separate output, when the input is varied in differential-mode;
- A 3D plot of the output DC current function of the input current and output voltage is included, when the input is varied in common-mode;

The Annex B of this work contains also two figures which illustrate the DC-Parametric characteristics in differential and common-mode for the output. Here again, some supplementary characteristics are extracted, with the following specifications: the *C.M. gain* refers in each case to the input varied in common mode, while the *direct gain* refers to the input varied in differential mode.

The figures of merit extracted from AC and transient analyses for the current mirror can be found in Annex B. Here, the four types of transfer functions are presented: the differential and common-mode output transfer functions when the input is in differential mode and the differential and common-mode output transfer functions when the input is in common mode.

The model topology as s-functions for the fully differential current mirror is presented in Figure 5.15, corresponding to the SIMULINK macro-model structure from Figure 5.16.

It should be noted that the fully differential model for the current mirrors, in the case of a single-output circuit, contains four direct and four reverse transfer functions, along with the impedances for all the inputs and outputs and the differential ones. This leads to a total of sixteen *s*-functions which should be simulated in a single block, so it is important to keep the *s*-functions orders as low as possible, while not to deprecate the precision of the model.

The HDL behavioral models for the current-to-current converter can be extracted using the SIMECT tool. A 2^{nd} order VHDL-AMS linear model used for simulations with Dolphin Integration SMASH, a VHDL-AMS model for the Cadence AMS simulator and a Verilog-A model used with Cadence AMS are presented in Annex C of this report. They also are automatically synthetized starting from the previous extracted and computed figures of merit.

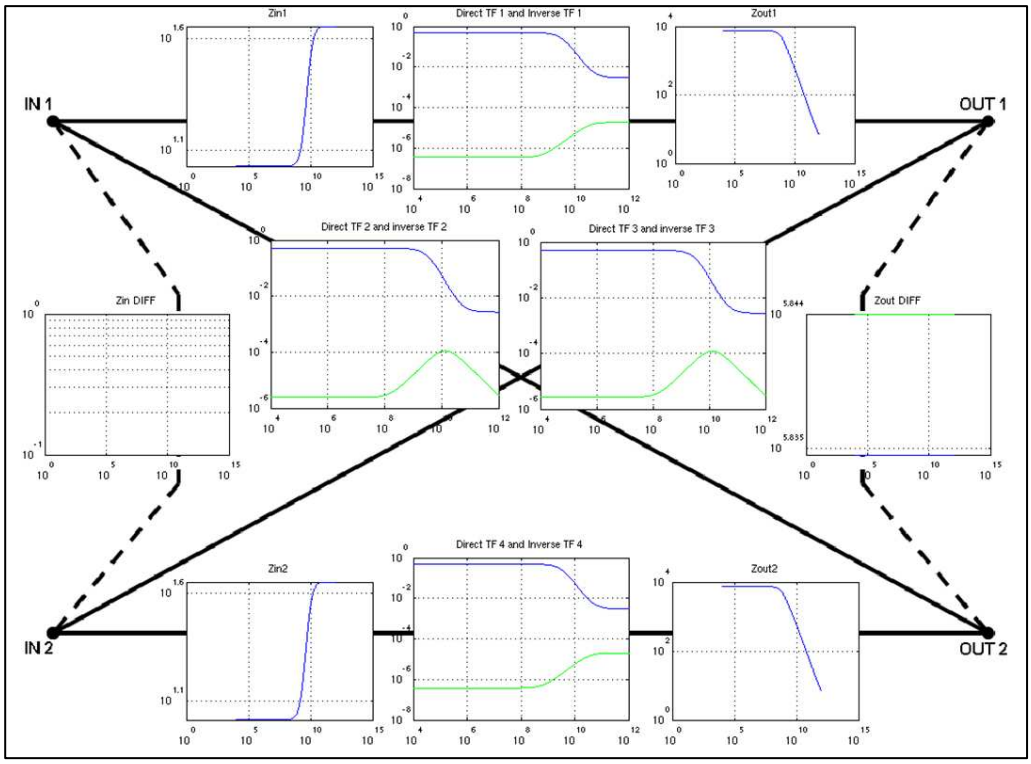


Figure 5.15 Current mirror Macro-model topology as s-functions

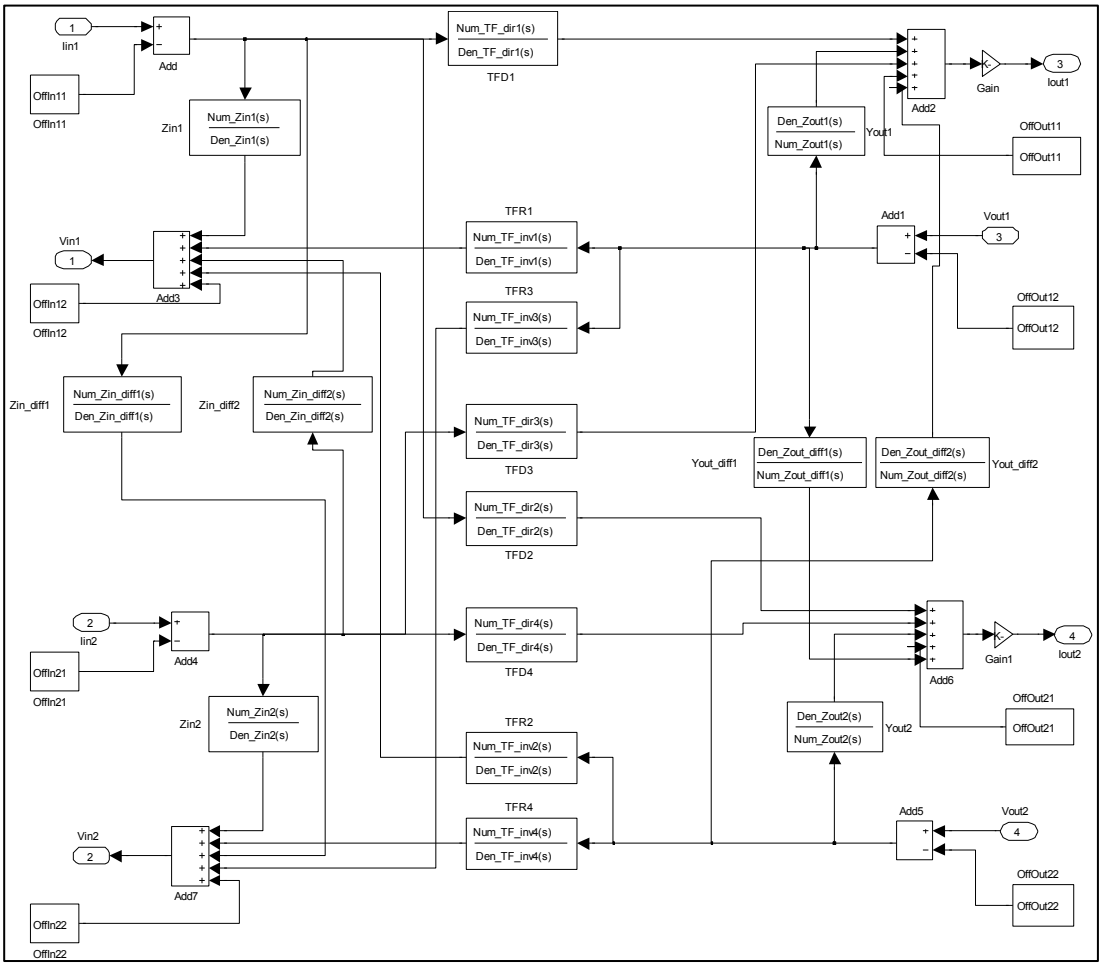


Figure 5.16 Current mirror Simulink macro-model

5.3.5. Micro-mechanical Resonators Modeling

The micro-mechanical resonators include a series of physical and material effects and their responses are dependent on these phenomena so no automatic model extraction technique is available for LWRs, like in the case of the amplifiers. Other works [150] [48] also reported manual modeling of the resonators using the following approaches:

- *structural modeling*, starting from the material characteristics and the geometrical dimensions [143] [151];
- *functional modeling* by taking into account the direct, reverse and potentially the differential transfer functions of the structures [140];
- *behavioral modeling* starting from measures on the output characteristics which try to reproduce the responses without taking into account the internal structure and functions [80].

The most realistic models would be obtained from structural modeling, where the material behavior, conform to the integration technology, is simulated. The functional modeling can provide a similar level of accuracy like the structural one, still it remains very costly in terms of simulation resources. This is why behavioral modeling is preferred in early design stages or when material characteristics are unknown.

5.3.5.1. Structural Modeling of the Micro-mechanical Resonators

The structural modeling of the micro-mechanical resonators of type LWR resides on the description of the electro-mechanical effects established in the resonant structure.

In a general approach, the resonator can be assimilated to an electrical port, the resonant layer and one or more types of coupling, the most important being the piezoelectric one (depicted in Figure 5.17).

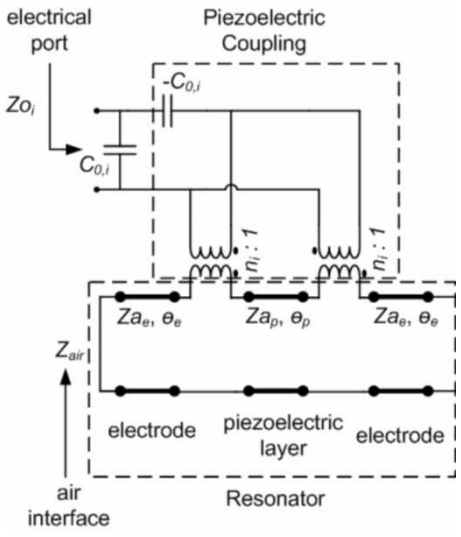


Figure 5.17 Lamb Wave Resonator piezoelectric propagation model of the filter – generic structure

The network elements' values are dependent on the structure materials (in terms of piezoelectric matrix, electric permittivity, Young modules) and the physical dimensions.

In the simplest generic case, the following equations give the elements' values [152]:

$$C_{0,i} = \varepsilon \frac{A_i}{t_{p,i}} \tag{5.4}$$

$$n_i = e \frac{A_i}{t_{p,i}} \tag{5.5}$$

$$\theta_{layer} = \frac{2\pi f_r}{va_{layer}} \cdot t_{layer} \tag{5.6}$$

$$Za_{layer} = \sqrt{c \cdot \rho} \tag{5.7}$$

$$va_{layer} = \sqrt{\frac{c}{\rho}} \tag{5.8}$$

Where ε is the dielectric coefficient for the piezoelectric material, e is the piezoelectric coefficient, c the stiffness coefficient, ρ represents the material mass density, $t_{p,i}$ is the piezoelectric layer's thickness, A_i - the electrode's area and f_r the resonance frequency.

The equations can be implemented in VHDL-AMS, Verilog-A, C-embedded or Matlab code for Simulink, relating the impedance on the electrical port $Z_{o,i}$ to the material effects as presented for example in [153]. Additionally, all the input-output voltages and currents of the structure can be related to the low-level effects [48].

The structural modeling can be done also by using dedicated software like CoventorWare to design the MEMS starting from its internal structure and geometry. An exhaustive study of the two types of structural modeling for BAW, SAW, FBAR and LWR was done in [146].

In our case, structural modeling was not practicable due to the fact that material and geometrical characteristics of the MEMS were not available. If CEA-Leti will provide these elements, an interesting perspective of this work is to implement structural models of the resonators and evaluate the system-level results using the different approaches presented here and the structural one.

5.3.5.2. Functional Modeling of the Micro-mechanical Resonators

The functional modeling of the micro-mechanical resonators resides on the description of the transfer functions for the resonators in order to capture the resonance/anti-resonance effects, the parasitic resonances and the harmonics of the principal resonance frequency.

The resonators which were used are designed and produced by CEA-Leti, Grenoble. In this case it was possible to accurately model the transfer functions starting from the real measured characteristics provided by the CEA-Leti.

Because the resonators use as stimulus the input voltage and respond in current, the interesting function for modeling is the input-to-output admittance, e.g. Y_{12} (Figure 5.18).

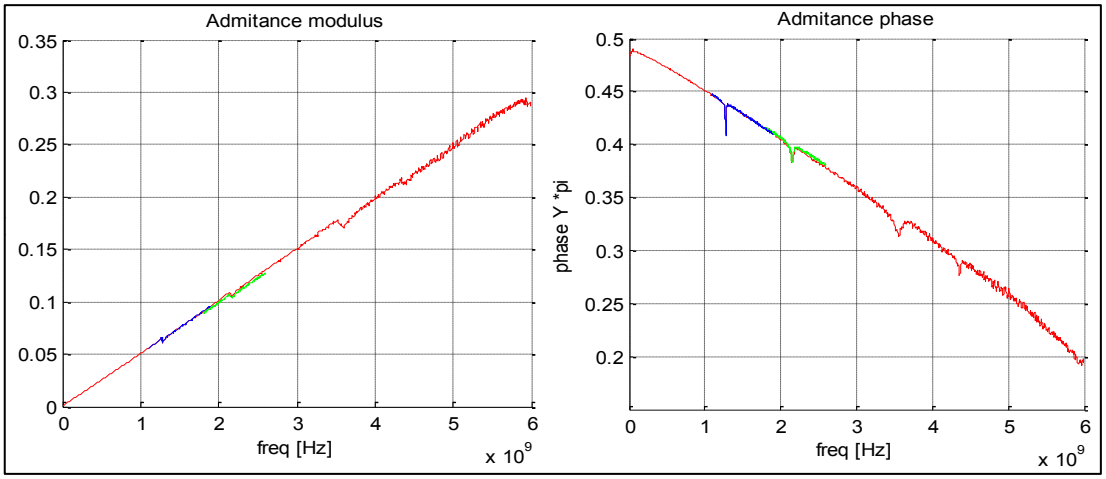


Figure 5.18 Lamb Wave Resonator measured characteristics - CEA-Leti

Multiple resonances can be observed on the admittance modulus/phase along with a pregnant parasitic capacitive effect. The selection of the most important resonant modes can be done by using the S11 parameter for the measures.

Figure 5.19 depicts this parameter for the measured characteristics, resulting in the first two resonant modes as the most effective. Practically, it was determined that the modeling of the first two resonances would be sufficient to predict the impact of the real resonator on the Sigma-Delta global performances, providing more than 95% of the parasitic effects since the cut-off frequencies of the amplifiers are close to the principal resonance.

Still, an exhaustive study, where all the parasitic resonant effects were considered is in progress by SUPELEC and CEA-Leti in the framework of [13].

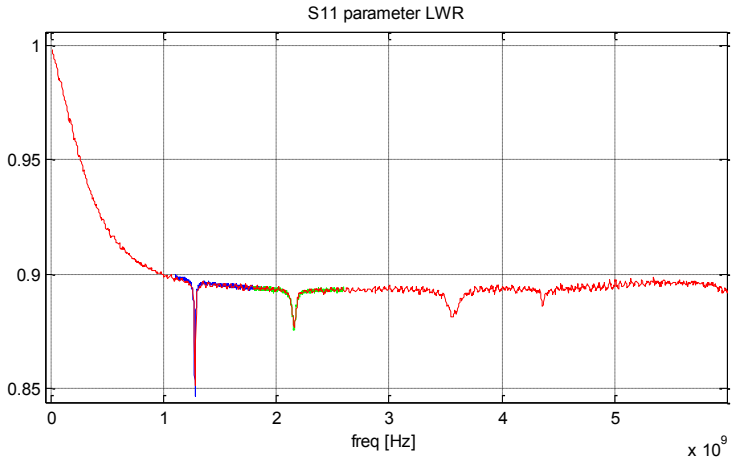


Figure 5.19 Lamb Wave Resonator measured characteristics *S11*-parameter

In a first step, the measured characteristics from CEA-Leti were compensated for capacitive and resistive parasitic effects and anti-resonant effects. Technologically, this compensation is implemented using a lattice form [154] where two series cells containing resonant layers are integrated on the same dice with two parallel cells without the resonant layers (only the substrate and the coupling which produce the parasitic effects). Figure 5.20 shows the principle and the equivalent passive group used for compensation in the studied case.

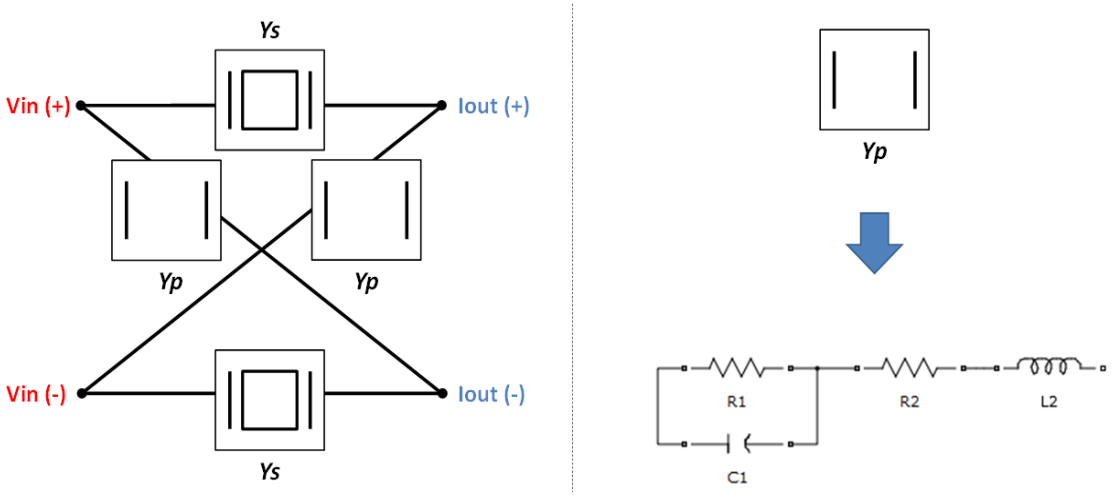


Figure 5.20 Admittance compensation for Lamb Wave Resonators

The compensation admittance and the resulting compensated one can be deduced:

$$Y_p = \frac{1}{sL_2 + R_2 + \frac{1}{\frac{1}{R_1} + sC_1}} \tag{5.9}$$

$$Y_{comp} = Y_{mes} - Y_p\big|_{f=0 \to fmax} \tag{5.10}$$

The compensation was performed using the previous formulae and a direct *s*-function identification under Matlab, then a manual tuning on the passive components was studied and finally a non-convex optimization was performed to find the best fit (Figure 5.21).

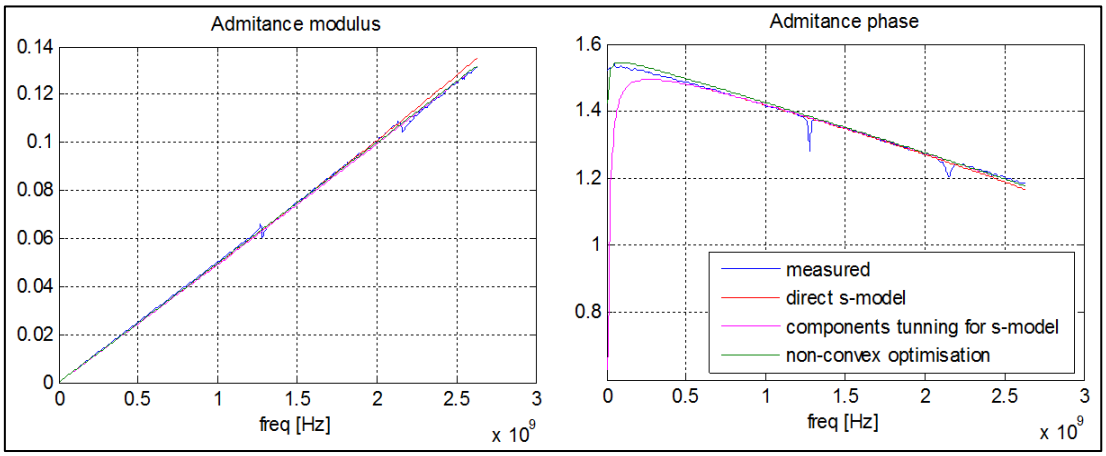


Figure 5.21 Lamb Wave Resonator measured characteristics compensation

The characteristics resulted from compensation for the three cases are presented in Figure 5.22. Starting from the best compensation, an *s*-function is extracted for the first and second resonances respectively. This method can be applied to an undefined number of resonances, the main drawback being the increasing model order.

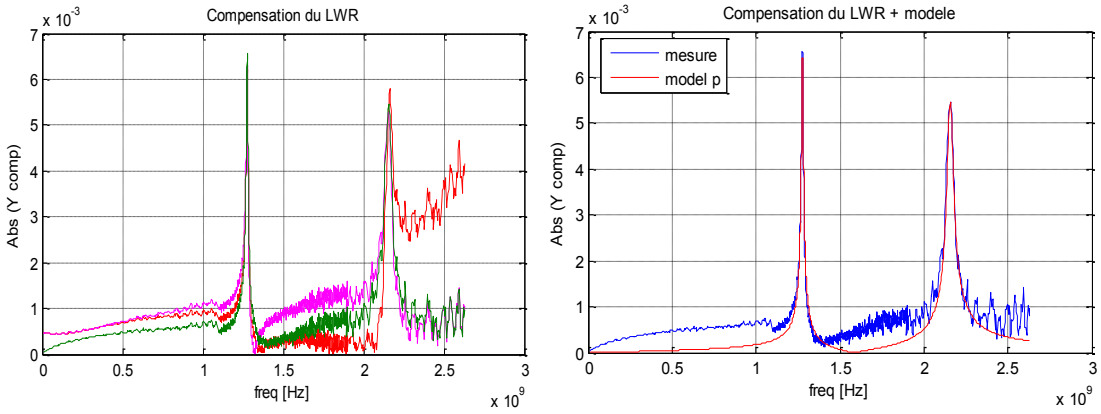


Figure 5.22 Lamb Wave Resonator measured characteristics and s-functions

The transfer function was determined by an identification of the measures with a function of type 5.9:

$$H_c(s) = \frac{g_1 \cdot \frac{R_1}{L_1} \cdot s}{s^2 + \frac{R_1}{L_1} \cdot s + \frac{1}{L_1 C_1}} + \frac{g_2 \cdot \frac{R_2}{L_2} \cdot s}{s^2 + \frac{R_2}{L_2} \cdot s + \frac{1}{L_2 C_2}} \qquad (5.11)$$

Where we can consider two separated resonances in parallel and g_i, R_i, L_i, C_i, are the corresponding elements from the extended Butterworth-Van Dyke electrical model of resonators when harmonics are present [155]. The resulted function is implemented directly in VHDL-AMS or Verilog-A by using the *Laplace Transfer Function* attributes and in Simulink by using the *Transfer Fcn* block set.

5.3.5.3. Behavioral Modeling of the Micro-mechanical Resonators

For behavioral modeling, a simpler formalism is used, where the indicial or impulsion responses obtained from measures are used and not the actual *s*-models.

Three kind of models were developed for this purpose, a Matlab embedded function for Simulink, a C embedded function for Simulink and a VHDL-AMS model, all based on the indicial response. With the three kinds of models, it is possible to interface with the previous blocks in Simulink or VHDL-AMS and potentially accelerate simulations by using the C embedded pre-compiled code.

In the three models, two selectable levels of precision were used:

1. *Low precision*: The response point is computed by using two samples of the indicial response in each summation term;

2. *High precision*: The response point is computed by using four samples of the indicial response in each summation term;

The Matlab embedded function in Simulink for the indicial response is based on the following routine:

```
ech=floor((t-vt(k))/trep);  % integer part of delay
dlt=(t-vt(k))/trep-ech;     % fractional part of delay
d12=(dlt-1).*(dlt-2) ;
dd1=dlt.*(dlt+1) ;
       if prec
             y=sum( (vu(k)-vu(k+1)).*(-repind(1+ech)'.*(dlt).*d12/6 +
repind(2+ech)'.*(dlt+1).*d12/2 -repind(3+ech)'.*dd1.*(dlt-2)/2
+repind(4+ech)'.*(dd1).*(dlt-1)/6  ) ) ;
       else
             y=sum( (vu(k)-vu(k+1)).*(repind(1+ech)'.*(1-dlt)+repind(2+ech)'.*(dlt))) ;
end
```

Where the following variables are considered:

- *t* is the current continuous-time of the system and *vt* is an array containing the current time and a number of previous time samples;
- *u* is the current input sample and *vu* is an array containing the current input sample and a number of previous input samples;
- *repind* is an array containing the indicial response and *trep* is its sampling period;
- *prec* is a Boolean variable selecting the precision of the response computation;
- *ech* will be the integer part of the delay of the current input sample when compared to the indicial response period, while *dlt* will be its fractional part;
- *y* will be the block output response;

The C embedded function in Simulink for the indicial response is built starting from the following routine:

```
out_calc=0.0;

for (i=0; i<kmax; i++) {
       ech=((crt_time-vt[i])/trep);
        dlt=(crt_time-vt[i])/trep-ech;
        d12=(dlt-1.0)*(dlt-2.0);
        dd1=dlt*(dlt+1.0);

        if (prec==1)
        {
             out_calc = out_calc + ((vu[i]-vu[i+1])*(-repind[ech]*(dlt)*d12/6.0 +
repind[1+ech]*(dlt+1.0)*d12/2.0 -repind[2+ech]*dd1*(dlt-
2.0)/2.0+repind[3+ech]*(dd1)*(dlt-1.0)/6.0  ) );
        }
        else
        {
             out_calc = out_calc + ((vu[i]-vu[i+1])*(repind[ech]*(1.0-
dlt)+repind[1+ech]*(dlt)));
        }
    }
    *y = out_calc;
}
```

The same principle applies here, with the difference that the current time is retained in *crt_time* and an intermediary *double* variable (*out_calc*) is used to retain the output value at each step, written then on the output pointer *y*.

Unlike the Matlab and C embedded functions where generic input and output are employed, the VHDL-AMS model for the indicial response uses as input and output actual electrical quantities, providing a coherent electrical behavior:

```
port(terminal in_a, out_a : electrical); --input/output port
generic(Zout : real := 50 ); --output impedance

quantity Vin1 ACROSS Iin1 THROUGH in_a TO ground;
quantity Vout1 ACROSS Iout1 THROUGH out_a TO ground;
```

The VHDL-AMS model for the indicial response is based on the following mixed-signal process:

```
compute_out : process is
…
  begin

       loop
       wait until clk = '1';

       …

       y:=0.0;

        for k in 1 to kmax loop

            ech:=integer(floor((now-vt(k))/trep));
            dlt:=(now-vt(k))/trep-real(ech);
            d12:=(dlt-1.0)*(dlt-2.0);
            dd1:=dlt*(dlt+1.0);

            if prec=1 then
               y:=y+((vu(k)-vu(k+1))*(-repind(1+ech)*(dlt)*d12/6.0 +
repind(2+ech)*(dlt+1.0)*d12/2.0 -repind(3+ech)*dd1*(dlt-
2.0)/2.0+repind(4+ech)*(dd1)*(dlt-1.0)/6.0  ) );
            else
               y:=y+((vu(k)-vu(k+1))*(repind(1+ech)*(1.0-dlt)+repind(2+ech)*(dlt)));
            end if;

       end loop;

       v_aux <= y;

    end loop;

  end process compute_out;

  Iout1 ==  v_aux/Zout;
```

The calculus method remains the same but the following AMS specific concepts are used:

- the infinite mixed-signal process *compute_out* is used, sample by the digital clock *clk*;
- *y* is a real variable, keeping at each step the computed output voltage value, which is assigned to the analog signal *v_aux*;
- the input samples in *vu* are taken from the *Vin1* input voltage (LWRs input is a voltage);
- the output of the block is the *Iout1* current (LWRs response is a current), computed from the intermediary voltage *v_aux* and the static output impedance *Zout*.
- the current continuous-time is obtained in this case by using the VHDL-AMS specific function *now()*;

The Matlab embedded function for Simulink, the C embedded function for Simulink and the VHDL-AMS model using the indicial response can be found in Annex D of this report.

The structural models for LWRs, the functional ones and finally the behavioral models are designed taking into account the requirements to interface them with the amplifiers extracted models and respectively with the ADCs and DACs models further presented. In Simulink, voltages and currents conventions were established (see Chapter 3) and in VHDL-AMS electrical nature constructs are used.

5.3.6. Basic Filter Cell Implementation

Once all the amplifiers and the micro-mechanical resonators of type LWR were modeled, it is possible to implement and inspect the responses of the basic cells composing the band-pass Sigma-Delta filters.

A cell of this type is composed of the LWR lattice compensation structure and the two corresponding amplifiers (a trans-impedance amplifier *TIA* and a current-to-current converter *M*) – see Figure 5.5 - along with a trans-conductance amplifier *Gm,* on the input of the structure.

The two Sigma-Delta architectures discussed in Section 5.2 are composed starting from these basic filtering stages.

Parallel DC, AC and transient simulations were conducted for different implementations of the cells in order to evaluate and validate the modeling approach. The ideal resonator and amplifier functions were first evaluated, then the transistor-level simulations and finally two types of macro-models (SIMULINK, VHDL-AMS), obtaining coherent results. Figure 5.23 illustrates the small-signal results for the different implementations.

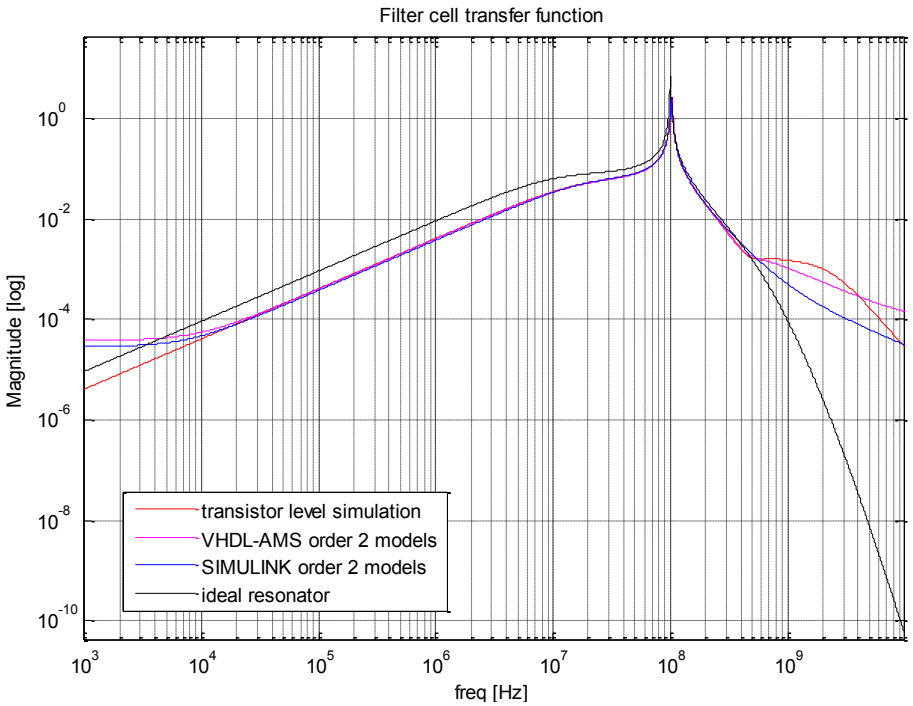


Figure 5.23 Filter cell transfer function – ideal function, transistor-level and macro-models functions

5.3.7. Modeling of the ADCs and DACs

In the original 2nd order and 6th order ΣΔ architectures, a 3-bits flash ADC was used with a direct and an inverted output used to create a pair of differential currents after the DAC.

For the behavioral modeling, a simpler solution was employed: to use a single-output ADC, able to provide the non-inverted output bit stream (this is the own modulator output) and to re-create the differential outputs only at the DAC level. Some specific characteristics of the ADC result from the signals it should be able to convert [28] [39]:

- for this application the output DC offset of the ΣΔ filter is $V_{out\ DC}=2.5V$, the offset value of the last TIA amplifier;
- the converter is a voltage converter with the variation interval for the input voltage being +/- 1V around $V_{out\ DC}$ for the presented case;
- the ADC is a multi-bit converter and should be able to provide 3-bit binary conversion and thermometric code for this application;

The ADC model in Simulink is presented in Figure 5.24. It is based on the sample-and-hold technique (the *Sample* block set) and quantization. The Quantize block uses also a compensation coefficient for the ADC. An initial adder is used to subtract the output DC of the final stage TIA from the input signal.

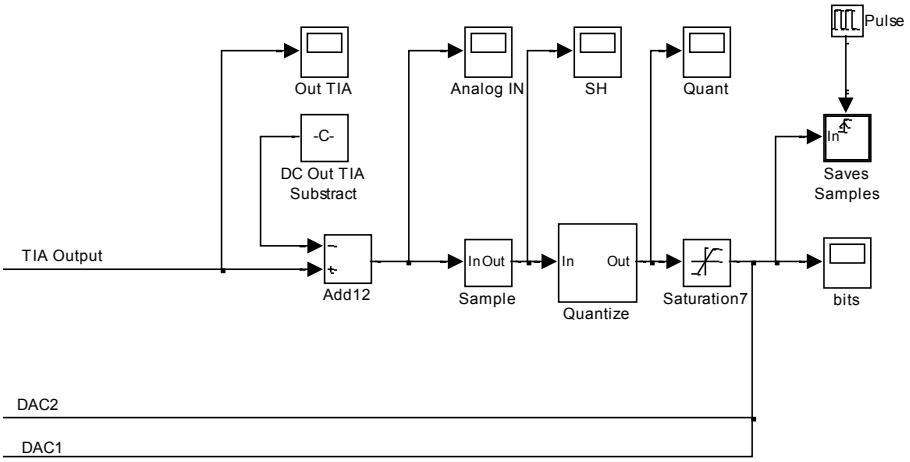


Figure 5.24 Simulink model of the modulator ADC

The VHDL-AMS model for the ADC is a fully mixed-signal model based also on the sample-hold and quantization principles [156] [149]. For this model, the conversion levels, the DC offset value and the number of bits are generic adjustable parameters:

```
generic (MIN : real := -1.0; -- minimum input voltage
         MAX : real := 1.0;  -- maximum input voltage
        DC_OUT_FILT : real := 2.5; -- TIA4 DC output voltage
         NBITS : positive := 3
             );
```

Its ports include: an electrical analog terminal (*IN_ADC*), the clock signal and two digital outputs (*VAL* and *THERM*), for the output bits and thermometric code respectively. Furthermore, quantities of electrical nature are used on the input side of the ADC:

```
port (terminal IN_ADC : electrical;
          signal CLK : in bit;
          signal VAL : out bit_vector (NBITS-1 downto 0) := (others => '0');
          signal THERM : out bit_vector (2**NBITS-2 downto 0) := (others => '0')
          );

quantity Vin ACROSS Iin THROUGH IN_ADC TO ground;
```

The conversion is performed using an AMS sequential process, sampled by the clock signal *CLK*. The comparison between the input value and the different conversion levels is based on recursive subtraction:

```
process
        …

         begin

        wait on CLK;
         delta_v := MAX - MIN;
         input_hold := Vin - DC_OUT_FILT - MIN;

         if (CLK'event and CLK = '1') then
              count:=0;

              --bits and count
              for i in NBITS-1 downto 0 loop
                  delta_v := delta_v / 2.0;
                  if input_hold >= delta_v then
                      VAL(i) <= '1';
                      input_hold := input_hold - delta_v;
                      count := count + 2**i;
                  else
                      VAL(i) <= '0';
                  end if;
              end loop;

              --thermometric
              for i in 1 to 2**NBITS-1 loop
                  if (i<=count) then
                  THERM(i-1) <= '1';
                  else
                  THERM(i-1) <= '0';
                  end if;
              end loop;

          end if;

end process;
```

Delta_v is a real variable which retains the conversion interval to be compared with the input value, *input_hold* contains the value of the input voltage centered on 0 (the DC is previously subtracted), while *count* is an integer used to retain the decimal value in the thermometric conversion process.

A VHDL-AMS temporal response of the ADC model, simulated with Dolphin Integration SMASH, is presented in Figure 5.25. As input, a sine wave containing all the interesting voltage levels was used.

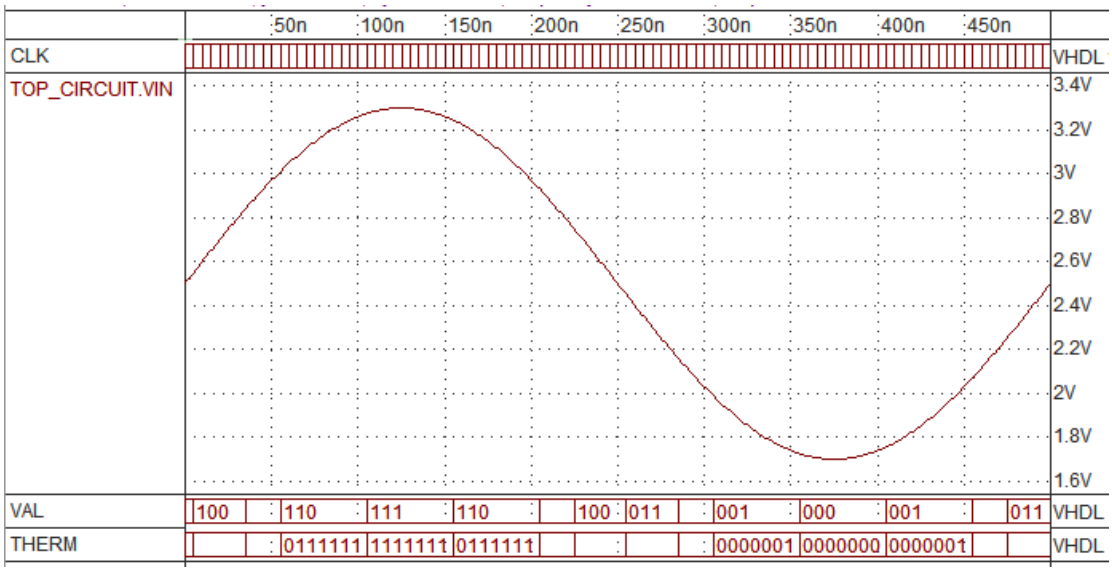


Figure 5.25 VHDL-AMS ADC model temporal response

The DAC models in Simulink and VHDL-AMS takes into account its principal characteristics from transistor-level [39] [28]:

- the DAC is a multi-bit (3-bits for this application) current-type converter;
- it should be able to provide differential current output for feedback summing in different filter points;
- it should be able to provide a fixed propagation delay (>*Ts*) in order to form the loop delay: *1.5 Ts* in this application;
- the output levels are different between the fast DAC connection and the slow one. In our case, the full scale output current for the slow connection is +/- 100µA and +/- 25µA for the fast one;
- a finite output transition time should be provided. In the analog transistor-level case, this value was determined: 4ps;

The Simulink model for the DAC is presented in Figure 5.26. Two *Gain* blocks assure the required current level at the output, the differential structure is employed for each connection and the *Transport Delay* assures the propagation delay.

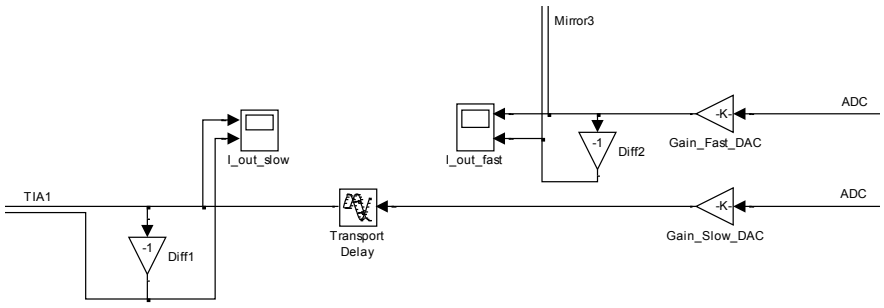


Figure 5.26 Simulink model of the modulator fast and slow DACs

The VHDL-AMS model for the DAC is based on a mixed-signal process containing a digital-to-analog conversion function [157] [158]. The following parameters are generic and adjustable:

```
generic (
  NBITS : positive := 3;-- number of bits
  TT : REAL := 4.0e-12; -- output transition time
  PR_T : time := 1.00ns; -- loop delay fraction = 0.4 Ts
  REF_I_IDEAL : REAL := 230.00e-6; --ideal max reference current
  REF_I_OFFSET : REAL := 100.00e-6 --ideal offset for current

);
```

The input and output ports of the block along with the electrical quantities on the output will be:

```
port (signal CLK : in bit;
      signal DIGITAL_IN : in bit_vector(NBITS-1 downto 0);
      terminal ANALOG_OUT_P : electrical;
      terminal ANALOG_OUT_M : electrical
);

quantity Voutp ACROSS Ioutp THROUGH ANALOG_OUT_P to ground;
quantity Voutm ACROSS Ioutm THROUGH ANALOG_OUT_M to ground;
```

The conversion function returns a real value starting from the input bit stream:

```
function d_to_a(in:bit_vector(NBITS-1 downto 0);max_val:real) return real is

variable s_out: real:=0.0;
variable sum:real;

begin
sum:=0.0;
for i in 0 to NBITS-1 loop
      if input(i)='1' then sum:=sum+max_val/real(2**(NBITS-i));
      else sum:=sum;
      end if;
end loop;
    s_out:=sum;
    return s_out;
end d_to_a;
```

Finally the conversion process and the instructions to form the output differential currents are the following:

```
process is
…
    begin
    wait on CLK;


    if(CLK'event and CLK='1')then
           AOUT<=d_to_a(DIGITAL_IN,REF_I_IDEAL) after PR_T;--conversion with delay
    end if;
end process;


Ioutp==-AOUT'ramp(TT)+REF_I_OFFSET; --current output +
Ioutm==AOUT'ramp(TT)-REF_I_OFFSET; --current output -
```

The ADC model coupled with the DAC model is simulated with Dolphin Integration SMASH for consistency. The simulation result is presented in Figure 5.27 and coherent with the transistor-level simulations. As input, a fast sine wave containing all the interesting voltage levels was used.

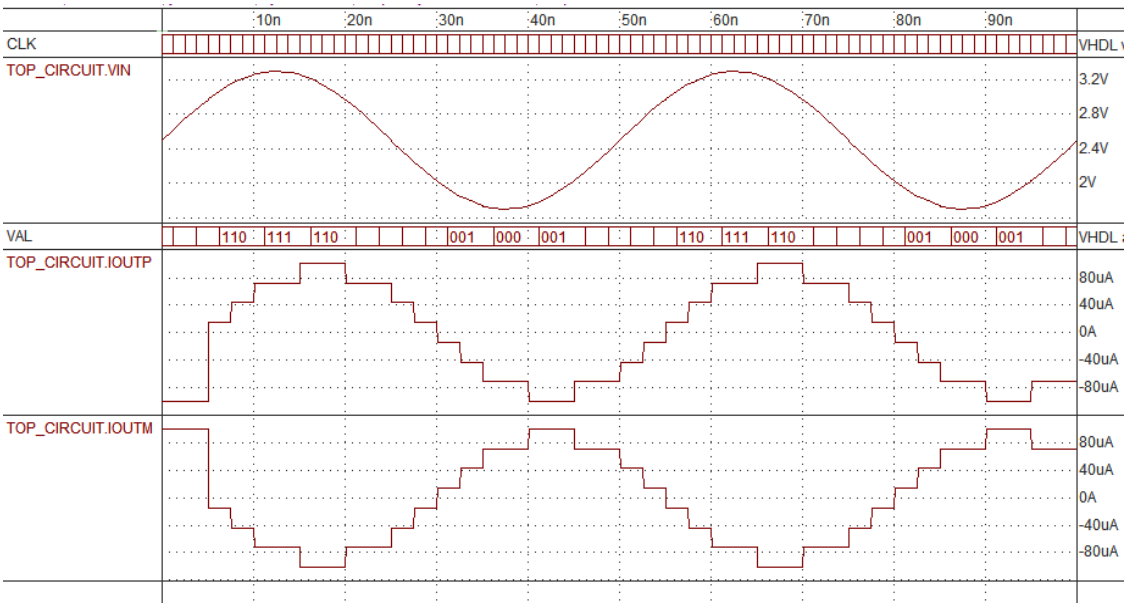


Figure 5.27 VHDL-AMS ADC and DAC models coupled test - temporal response

The complete HDL models for the ADC and DAC can be found in Annex E of this work.

5.4. System-level Design and Verification of Sigma-Delta Modulators

High-level implementations of the 2nd order modulator and 6th order modulator were realized in MATLAB-Simulink and Dolphin Integration SMASH (for VHDL-AMS) starting from the extracted macro-models of the analog components, the proposed LWR modeling strategies and the ADC and DAC models.

The high-level ΣΔ implementations were conducted for different orders of analog behavioral models (2-6). In all the cases, the system-level performance characteristics were evaluated and compared to the transistor-level results to validate the methodology.

In a first design step, the LWRs ideal resonance functions were used for comparison with the transistor-level ideal implementations of LWRs.

Afterwards, the real functions obtained from measures, eventually including the parasitic effects were employed. The aim of this study was to precisely explore the possibilities to realize a functional and high-performing modulator when using real micro-mechanical resonators.

5.4.1. 2nd Order ΣΔ Modulator System-level Design

The second order ΣΔ modulator was implemented in Simulink starting from its general system-level architecture in Figure 5.6 and using the previously designed composing blocks.

Figure 5.28 presents the Simulink implementation which contains the analog filter model, along with the ADC and DAC.

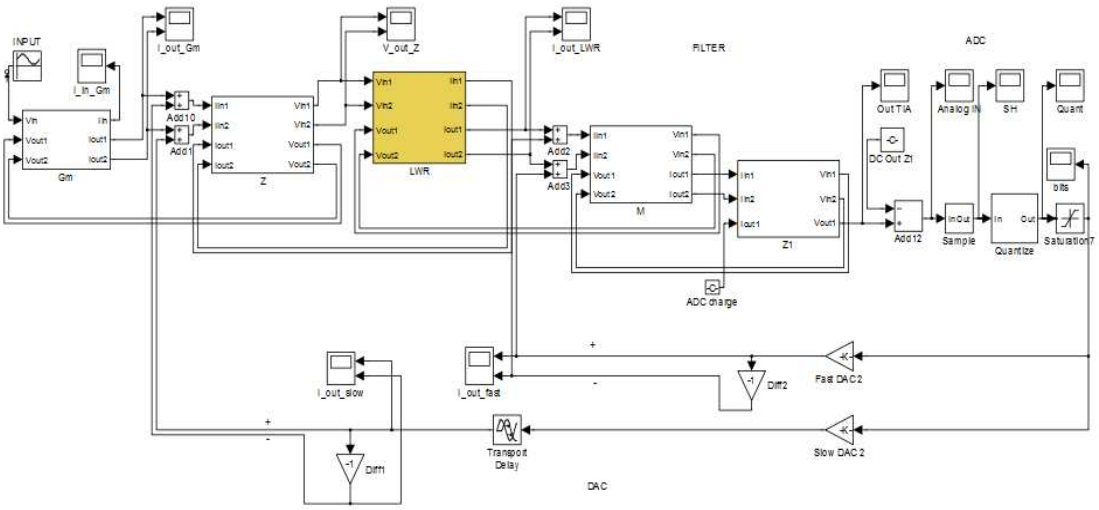


Figure 5.28 2nd order ΣΔ modulator implementation in Simulink

The *Gm* is a unipolar to differential trans-conductance amplifier of gain=800μS, *Z* is a differential inverted trans-impedance structure, having a differential gain of 1750Ω. The *LWR*

circuit is the model for a differential resonance structure with f_r=*100MHz*. The current mirror *M* is a fully differential adapter of differential gain=4 and very low common-mode gain. Finally, *Z1* is a differential-to-unipolar current-to-voltage converter of gain 2500Ω [28].

The non-inverted ADC and DAC fast and slow connections are realized starting from the concepts presented in Sub-section 5.3.7.

The modulator was also implemented in VHDL-AMS using the HDL automatically extracted models of the amplifiers, the functional/behavioral models of LWRs and the HDL models of the ADC and DAC. For this purpose, a top-level test-bench was implemented, containing a top-level *entity* and *architecture* of the modulator, where all the components were instantiated and all the connections made. The top-level test-bench contains also the stimulus sources.

Parallel transient and frequency-domain simulations were conducted on the transistor-level schematic and the macro-model structures in order to validate the conception methodology. In a first stage the basic functionality and stability of the modulator were tested and then power spectral densities were extracted from the transient simulations. This allowed characterizing the modulator performance.

Figure 5.29 contains the small-signal (AC) results compared between the transistor-level simulations, the Simulink macro-models and the VHDL-AMS models for the transfer function of the entire Sigma-Delta filter. A supplementary VHDL-AMS analysis is included for comparison, where the real resonance function of LWRs including parasitic effects was introduced into the filter structure. This supplementary function will be used in the final part of this chapter.

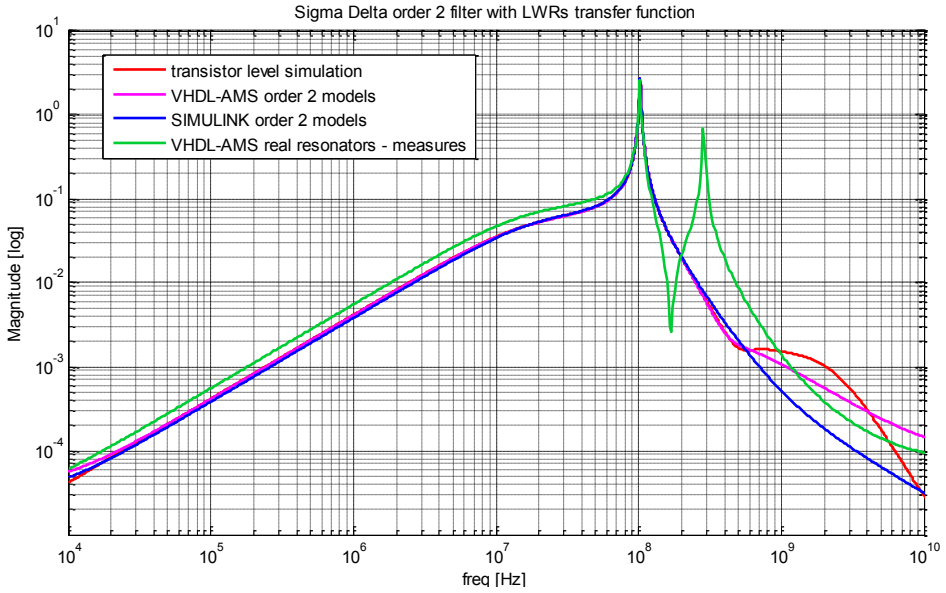


Figure 5.29 2nd order ΣΔ modulator filter response: transistor-level simulation, Simulink and VHDL-AMS

Next, power spectral densities (PSDs) on the modulator output were evaluated. The PSD of the output when normal input signal is applied and the PSD of the output when small input signal is applied are computed with the aim to inspect the modulator noise shaping proprieties.

For all the power spectral density simulations (2^{nd} order modulator and 6^{th} order modulator), the input signal for the evaluations is a 4096-samples modulated sine (in order to cover the converter band), beginning with a ramp (not to saturate the modulator at power-up). In all the cases we will present for the input signal and output signals temporal evolutions only a limited number of samples.

Two types of signals are used for PSD evaluations: a normal amplitude signal (e.g. 0.5V) and a low amplitude one (e.g. ~0V). Figure 5.30 presents the temporal evolution and the spectrum in reduced frequency for the input signal of normal amplitude.

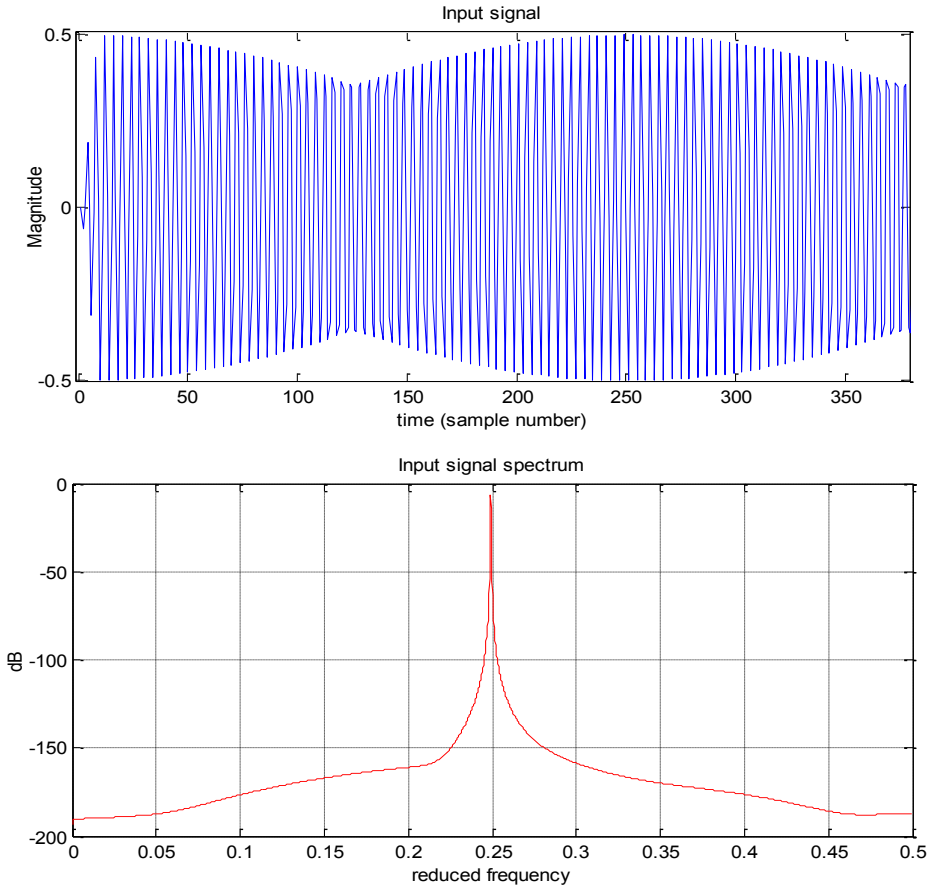


Figure 5.30 Input signal and its spectrum for PSD simulations

The output signal power spectral density when normal-amplitude input signal is applied and the output power spectral density when small-amplitude input signal is applied for the 2^{nd} order modulator in the case of the transistor-level simulations are presented in Figure 5.31.

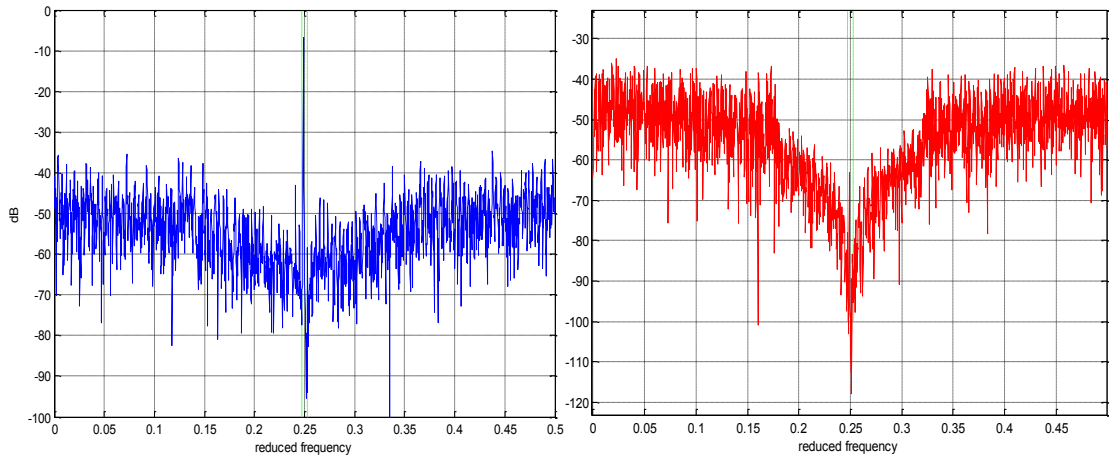


Figure 5.31 2nd order modulator transistor-level: *(a)* PSD of the output with normal amplitude stimulus and *(b)* PSD of the output with small amplitude stimulus

In these conditions, a theoretical resolution of 11.57 bits can be achieved.

The temporal evolution of the output in the case of the transistor-level simulation is shown in Figure 5.32.

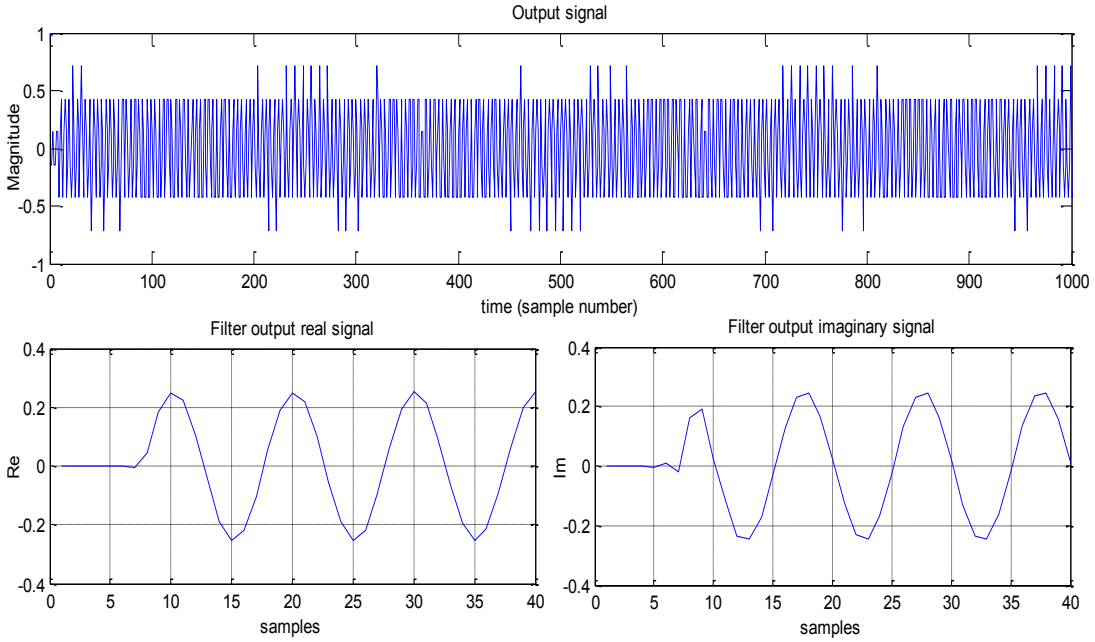


Figure 5.32 Output bits and Filter real and imaginary output signals

The output signal power spectral density when normal-amplitude input stimulus is applied and the output power spectral density when small-amplitude input stimulus is applied for the 2nd order modulator in the case of the SIMULINK simulations are presented in Figure 5.33.

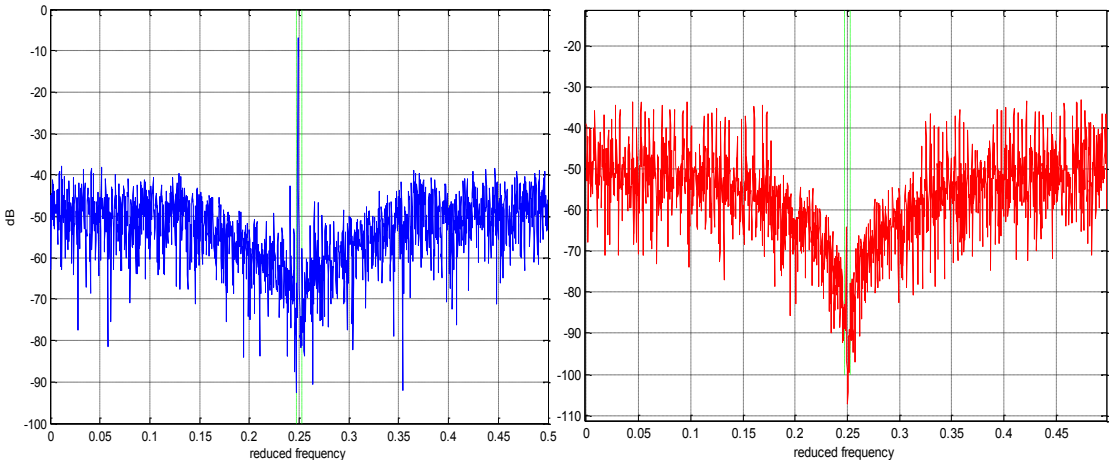


Figure 5.33 2nd order modulator SIMULINK: *(a)* PSD of the output with normal amplitude stimulus and *(b)* PSD of the output with small amplitude stimulus

The two Power Spectral Densities present coherent values with the transistor level ones.

In these conditions, a theoretical resolution of 11.44 bits can be achieved, comparable with the transistor-level one.

The temporal evolution of the output in the case of the SIMULINK macro-models simulation is shown in Figure 5.34.

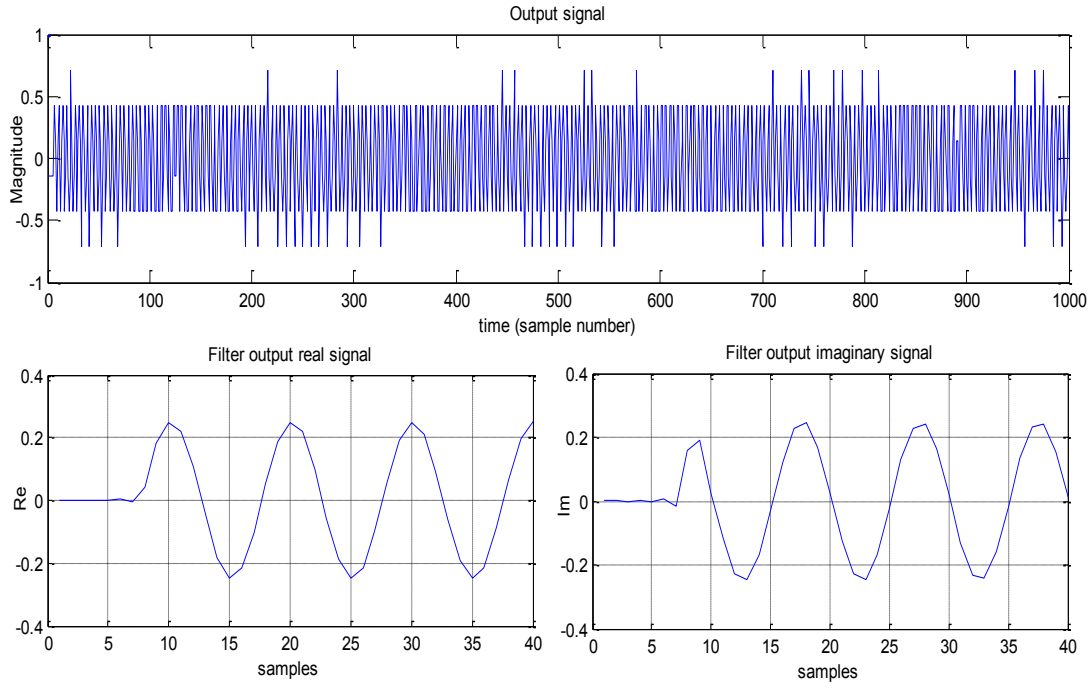


Figure 5.34 Output bits and Filter real and imaginary output signals

The output signal power spectral density when normal-amplitude input signal is applied and the output power spectral density when small-amplitude input signal is applied for the 2nd order modulator in the case of the VHDL-AMS simulations are presented in Figure 5.35.

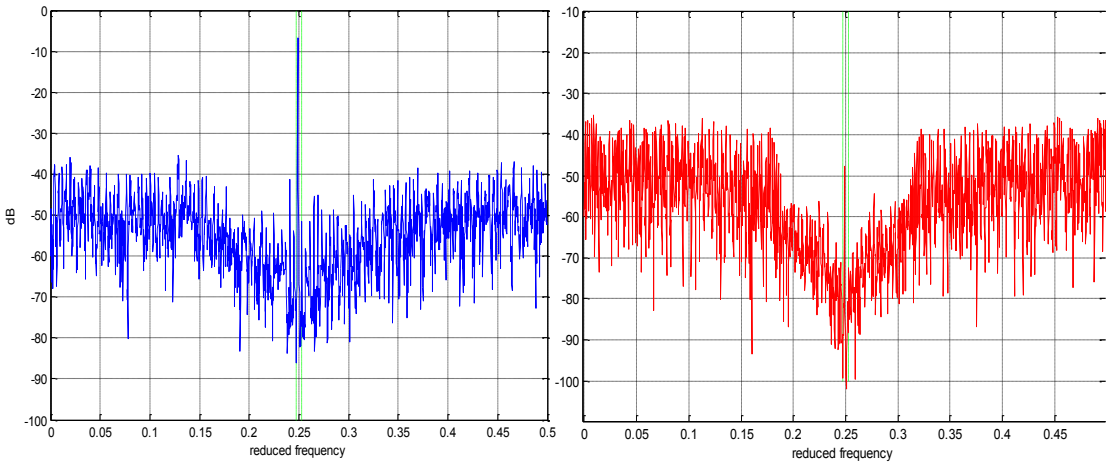


Figure 5.35 2nd order modulator VHDL-AMS: *(a)* PSD of the output with normal amplitude stimulus and *(b)* PSD of the output with small amplitude stimulus

In these conditions, a theoretical resolution of 11.68 bits can be achieved, compatible with the other measures.

The temporal evolution of the output in the case of the VHDL-AMS HDL macro-models simulation is shown in Figure 5.36.

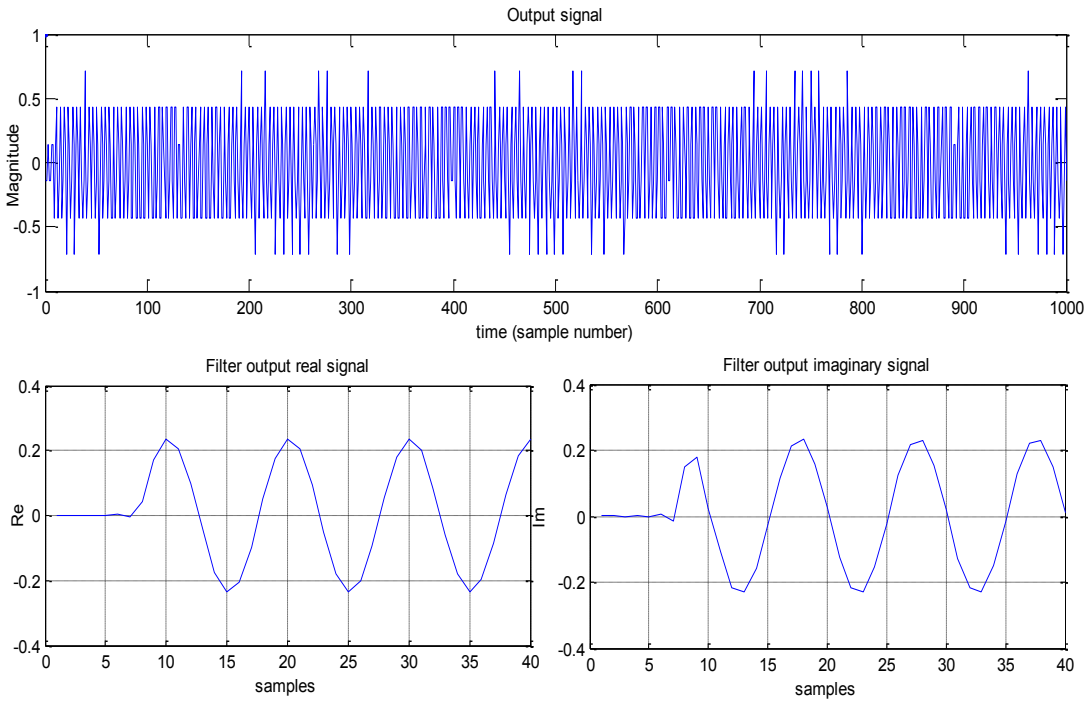


Figure 5.36 Output bits and Filter real and imaginary output signals

The results obtained from temporal simulations and frequency domain ones are coherent and more accurate when the model orders increase.

We present two supplementary evaluations in the transient domain for the VHDL-AMS simulations. The 2nd order modulator power-up is shown in Figure 5.37, for an input signal of type *100MHz* sinusoidal. The 2nd order functionality for an input of type modulated sine is

shown in Figure 5.38. For the two simulations the filter output voltage *Vout_1* is displayed along with the input currents for the *Z* trans-impedance and *M* current mirror (here the feedbacks from DACs are summed).

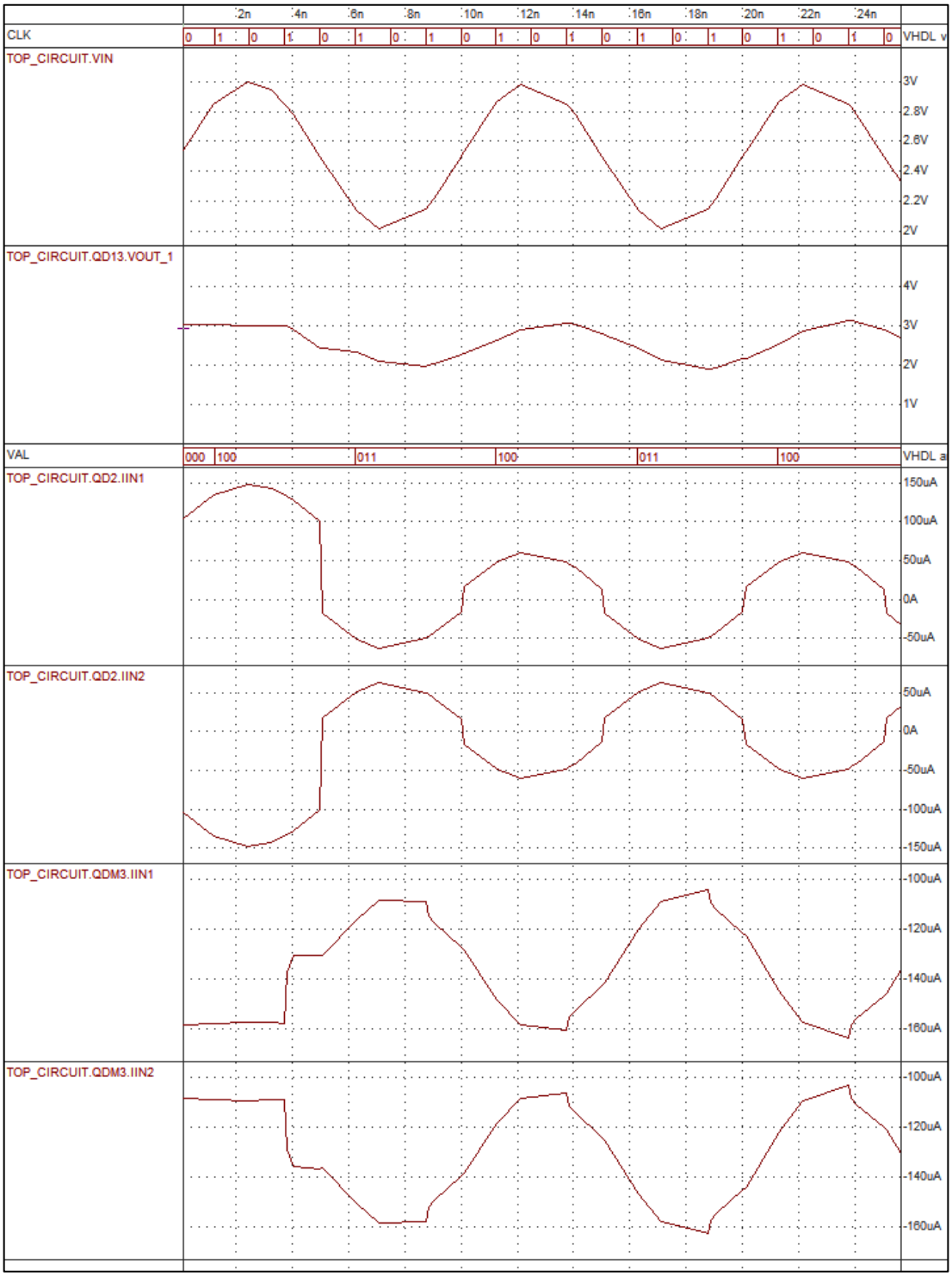


Figure 5.37 2nd order modulator power-up – VHDL-AMS simulation

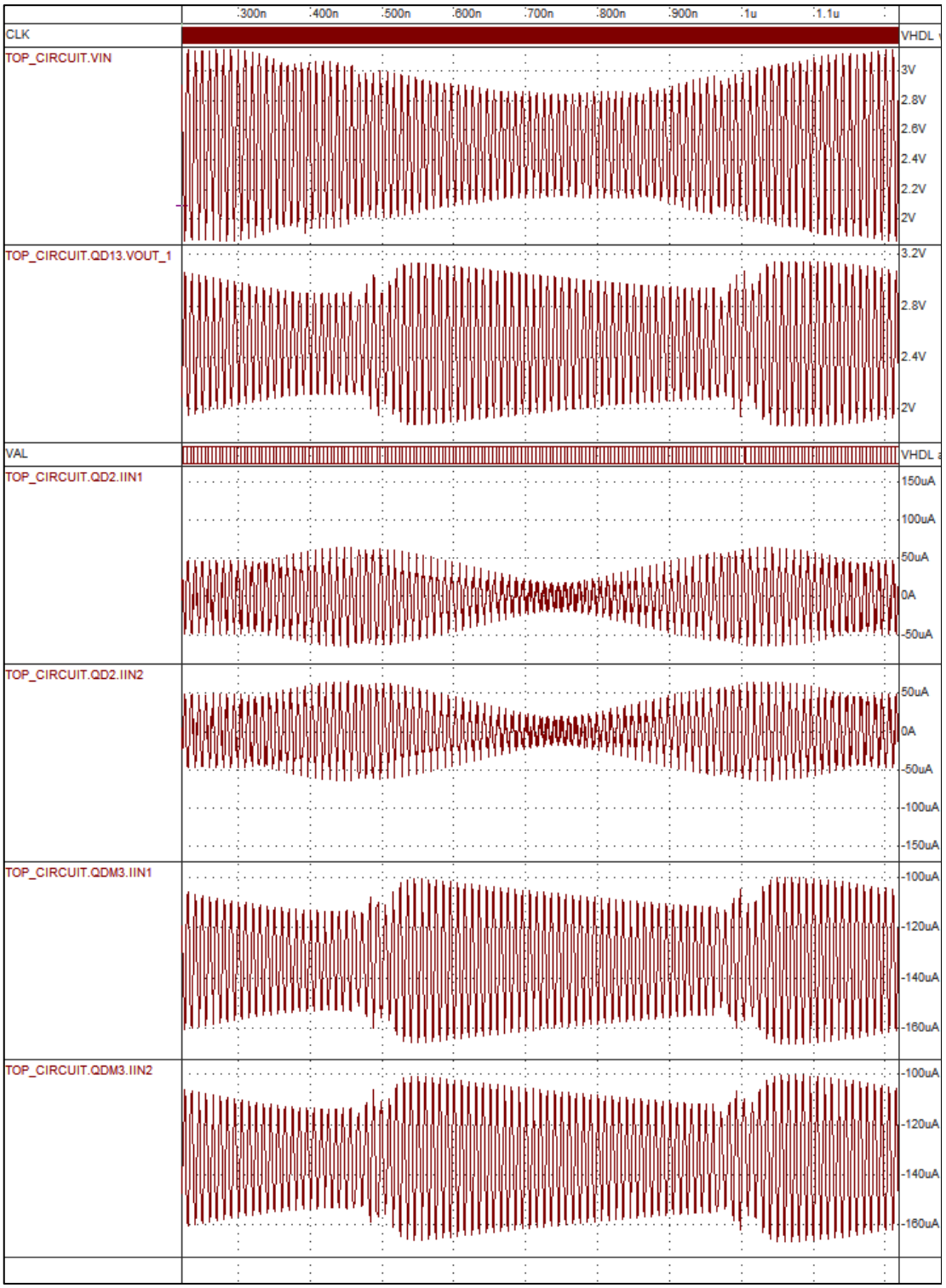


Figure 5.38 2nd order modulator functionality – VHDL-AMS simulation

5.4.2. 6th Order ΣΔ Modulator System-level Design

The 6th order ΣΔ modulator was implemented in Simulink and VHDL-AMS starting from its general system-level architecture in Figure 5.7 and using the previously designed component blocks.

Because the analog filter of this modulator has a higher degree of complexity when compared to the 2nd order one, stage-level evaluations were performed and then the entire modulator was simulated.

The structure in Figure 5.39 represents the high-level implementation in SIMULINK for the first filter stage. The *gm1* is a unipolar to differential trans-conductance amplifier of gain=97μS, *tr_real1* is a differential inverted trans-impedance structure, having a gain of 1940Ω. The LWR circuit is a model for a differential resonator structure with f_r=*100MHz*. The current mirror (gain *out1*=0.52, gain *out2*=0.153, gain *out3*=1) has the design characteristics presented previously.

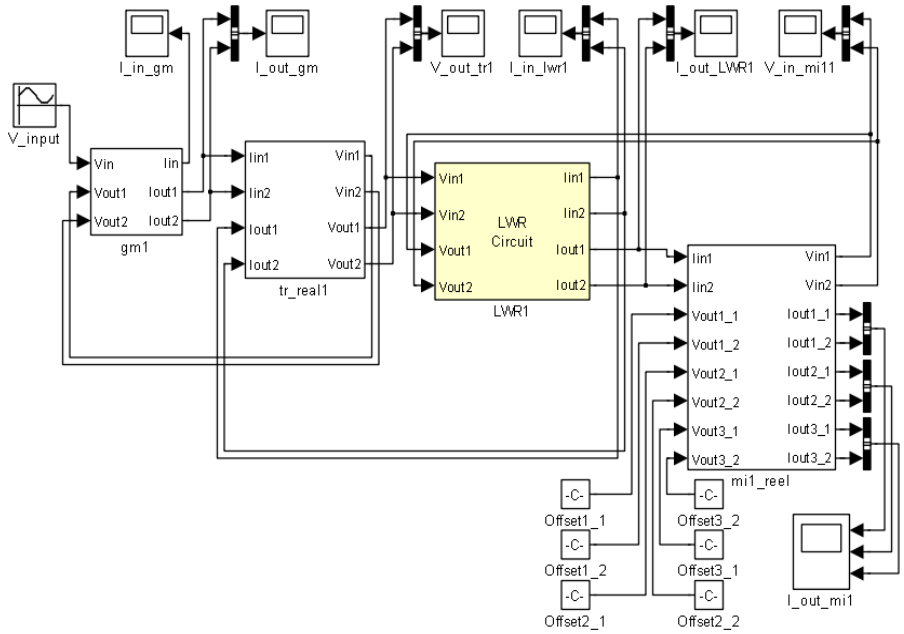


Figure 5.39 6th order Sigma-Delta filter first stage macro-model implementation in SIMULINK.

Figure 5.40 gives a comparative representation of the first stage direct transfer function on output *out3* in SIMULINK for two different orders of macro-models versus the transistor level function.

The simulations results are coherent proving that even small-order block models (e.g. 2) can be used for very fast simulations with reasonable accuracy. Models of order 4-5 are very accurate for the large majority of applications but will increase the simulation time, while orders>6 will exceptionally justify the plus-performance while increasing the system complexity.

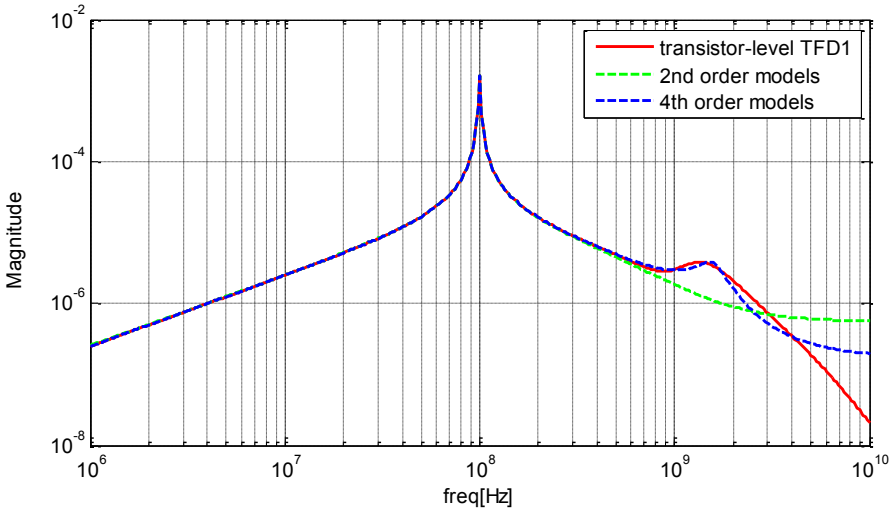


Figure 5.40 First stage direct transfer function (output iout3_1) comparison.

A small signal analysis result for the VHDL-AMS high-level implementation (second order models) is presented in Figure 5.41. The results were verified for coherence with the transistor-level and SIMULINK implementations and were validated.

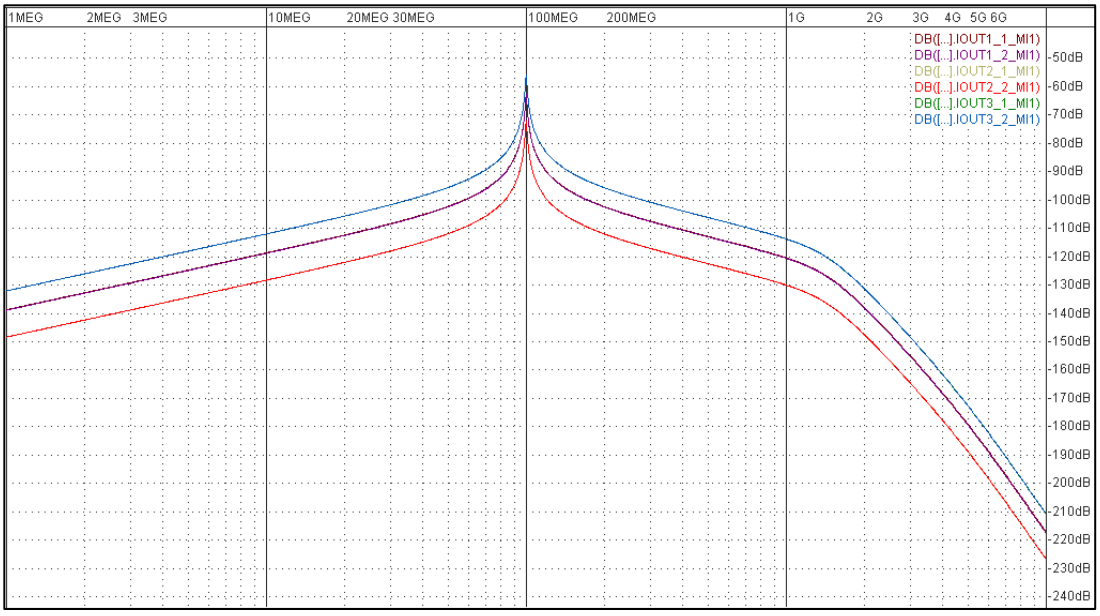


Figure 5.41 First stage direct transfer functions for the 3 outputs (VHDL-AMS high level implementation).

Next, we can verify the stability of the architectures and the transient behavior of the blocks by using transient simulations. As an example, the first filter stage was comparatively simulated in SIMULINK, VHDL-AMS and at transistor-level. The input voltage is *V_input*=0.1V at *Freq_input*=100MHz.

Figure 5.42 presents the SIMULINK results and Figure 5.43 the VHDL-AMS results, all of them being coherent with the transistor level ones.

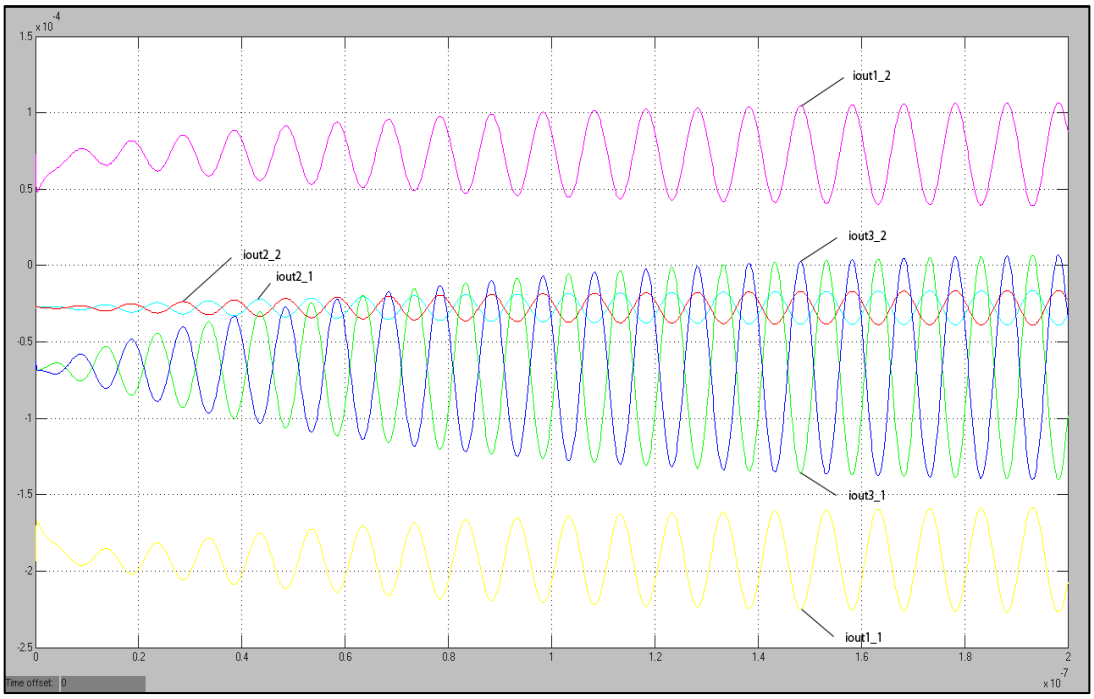


Figure 5.42 SIMULINK transient simulation for the first filter stage with *V_input*=0.1V; *Freq_input*=100MHz.

In SIMULINK the analysis was a CT one using the variable-step ode15s solver.

In VHDL-AMS, the transient analysis was performed as a Guitar analysis for Quartz oscillators in order to have accurate tolerances and default initial values.

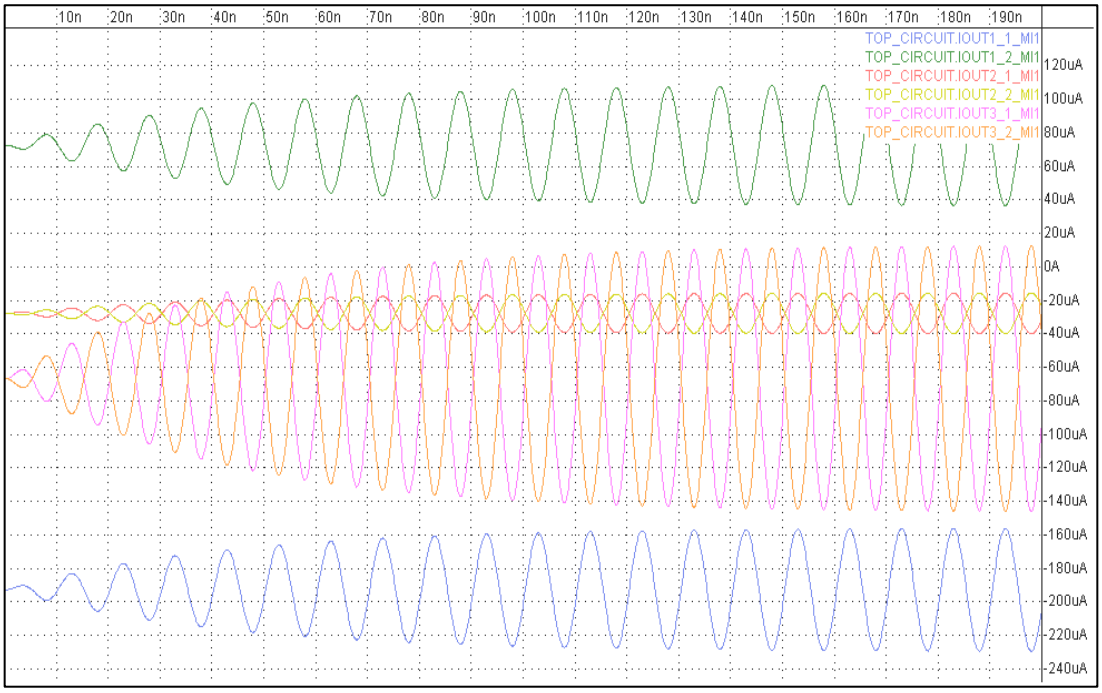


Figure 5.43 VHDL-AMS transient simulation for the first filter stage with *V_input*=0.1V; *Freq_input*=100MHz.

The transient resonant regime was correctly established in the two cases; the two designs are stable and accurate when compared to the transistor-level simulations.

Next, the second filter stage of the 6th order ΣΔ modulator was modeled in Simulink (Figure 5.44) and VHDL-AMS.

The *gm2* is a unipolar to differential trans-conductance amplifier of gain=106μS, *tr_real2* is a differential inverted trans-impedance structure, having a gain of 3630Ω. The LWR circuit is a model for a differential resonator structure with f_r=*99.8MHz*. The current mirror *mi2_reel* (gain *out1*=0.413, gain *out2*=1) is a fully differential converter with low common-mode gain.

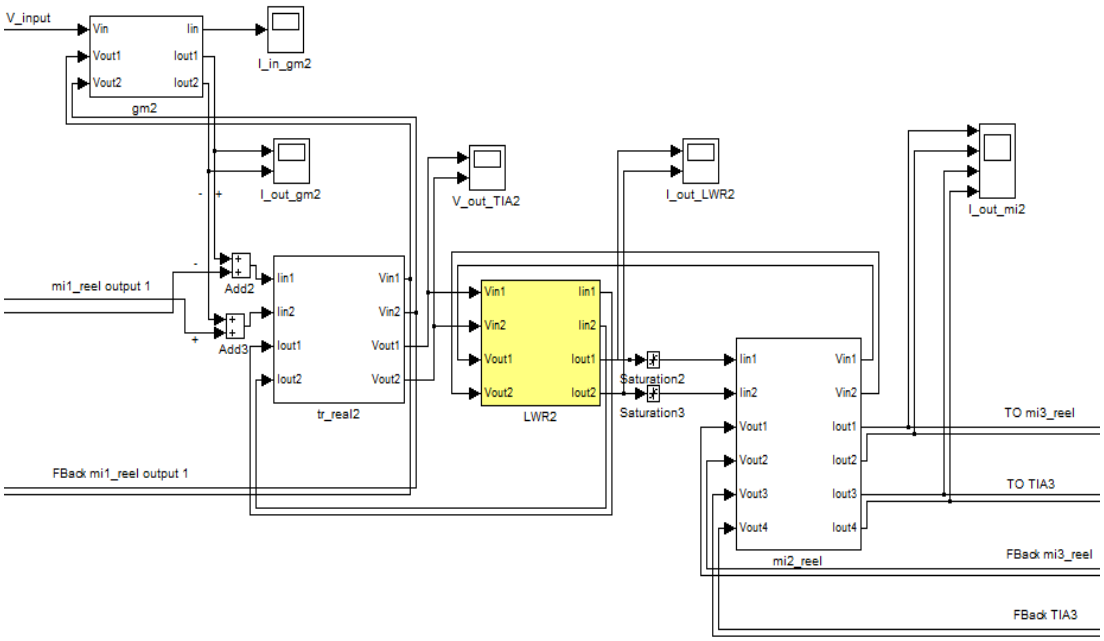


Figure 5.44 6th order Sigma-Delta filter second stage macro-model implementation in SIMULINK.

In this structure, the mirror *mi2_reel* was automatically synthetized as a multi-output model as discussed in Chapter 3, Section 3.5. Its internal structure is presented in Figure 5.45.

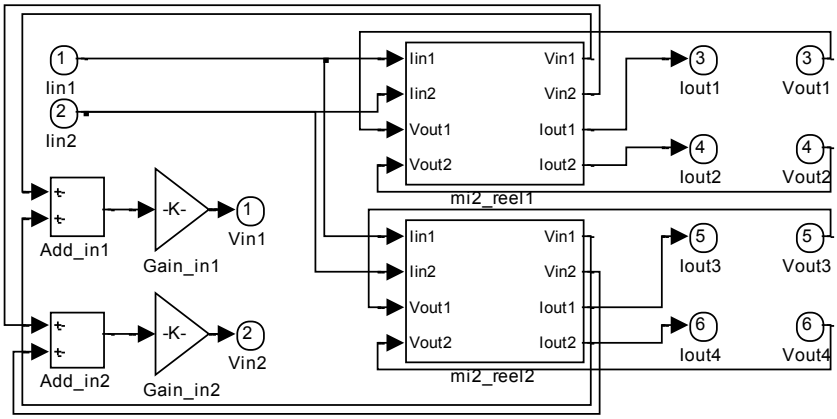


Figure 5.45 Current mirror macro-model for the second stage - implementation in SIMULINK.

In a final step, the third filter stage of the 6th order ΣΔ modulator was modeled in Simulink (Figure 5.46) and VHDL-AMS.

The *gm3* is a unipolar to differential trans-conductance amplifier of gain=74μS, *tr_real2* is a differential inverted trans-impedance structure, having a gain of 2590Ω. The LWR circuit is a model for a differential resonator structure with f_r=*101.2MHz*. For this filter stage, the current mirror is modeled together with *TIA4*, the last trans-impedance amplifier in order to reduce the filter complexity and the overall model order. In the initial design, *mi3* is a differential to unipolar current-to-current converter of gain=4 and *TIA4* is a single-ended trans-impedance converter of gain=4800Ω.

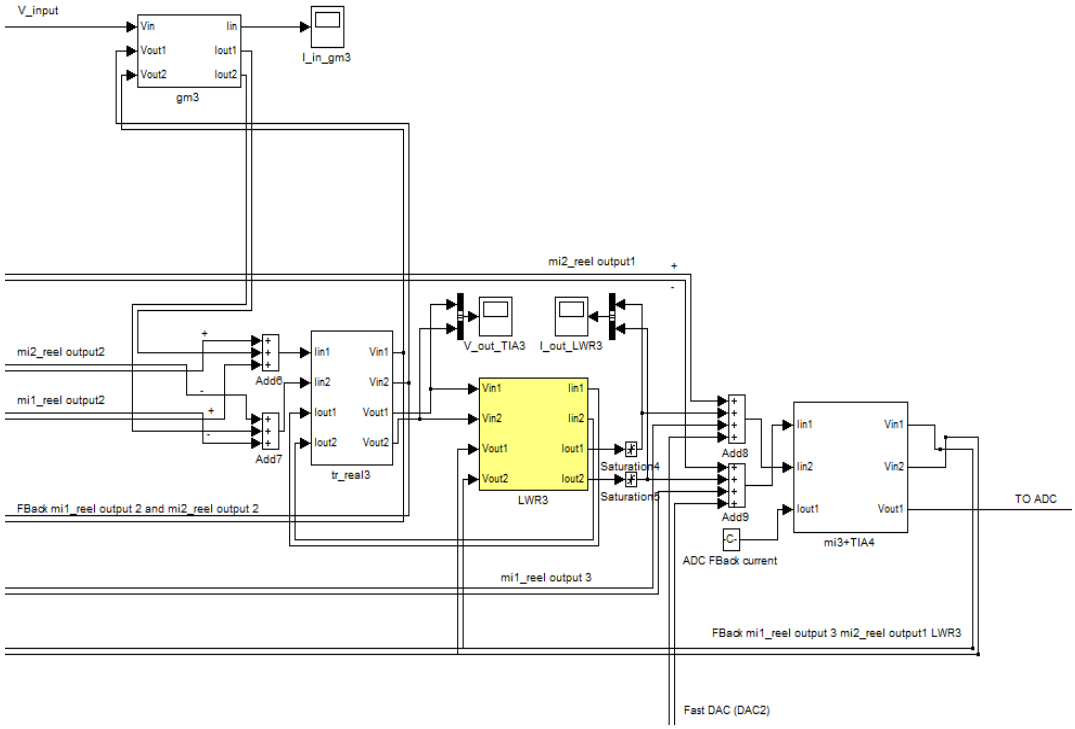


Figure 5.46 6th order Sigma-Delta filter third stage macro-model implementation in SIMULINK.

Once the stage levels were evaluated, the entire modulator was implemented and comparatively simulated with CADENCE Spectre at transistor-level, respectively SIMULINK and Dolphin Integration SMASH at macro-model level.

Figure 5.47 contains the comparison between the transistor-level and the two types of high-level models for the transfer function of the whole Sigma-Delta filter. Like in the 2nd order case, the small-signal results are coherent and more accurate when increasing the models orders. The comparison includes a fourth function representing the filter transfer function when real LWRs functions containing parasitic resonances were employed. This function is used in a further analysis, when studying the ΣΔ modulators design in the presence of micro-mechanical resonators parasitic effects.

The Power Spectral Densities (PSDs) are evaluated on the modulator output in a further step. In Figure 5.48 the output signal power spectral density when normal-amplitude input stimulus is applied and the output power spectral density when zero-amplitude input stimulus is

applied are presented in two cases (green: analog simulator, red: SIMULINK, order 2 macro-models) showing that the transistor-level and macro-model noise shaping proprieties of the modulator are coherent.

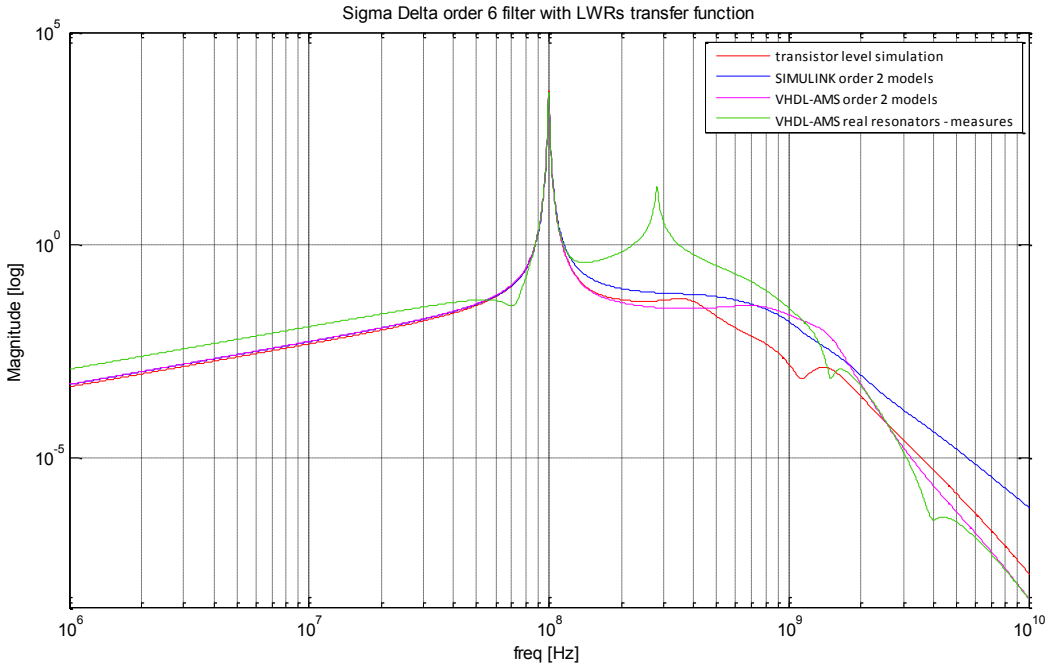


Figure 5.47 6th order Sigma-Delta filter transfer function: transistor-level simulation versus SIMULINK 2nd order models versus VHDL-AMS models

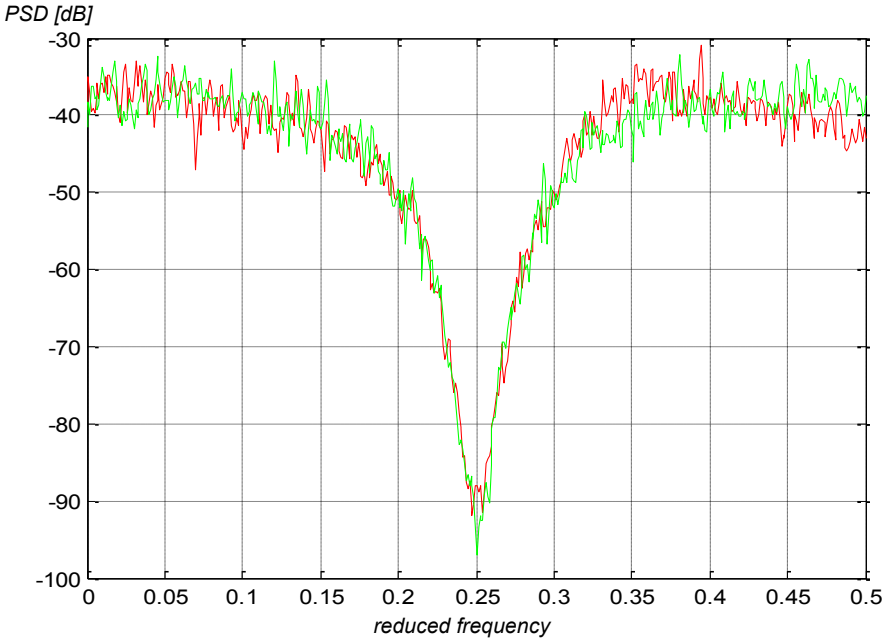


Figure 5.48 Transistor-level and SIMULINK PSDs comparison for the 6th order Sigma-Delta modulator (fc=0.25fs).

All the PSDs of the output when normal amplitude or small amplitude input signal is applied along with the output signals shapes were evaluated in the three cases studied: transistor-level, SIMULINK macro-models and VHDL-AMS behavioral models. We exemplify

only the VHDL-AMS results, the others being conform to these figures of merit, like in the 2nd order case.

The output signal power spectral density when normal-amplitude input stimulus is applied and the output power spectral density when small-amplitude input stimulus is applied for the 6th order modulator in the case of VHDL-AMS simulations are presented in Figure 5.49.

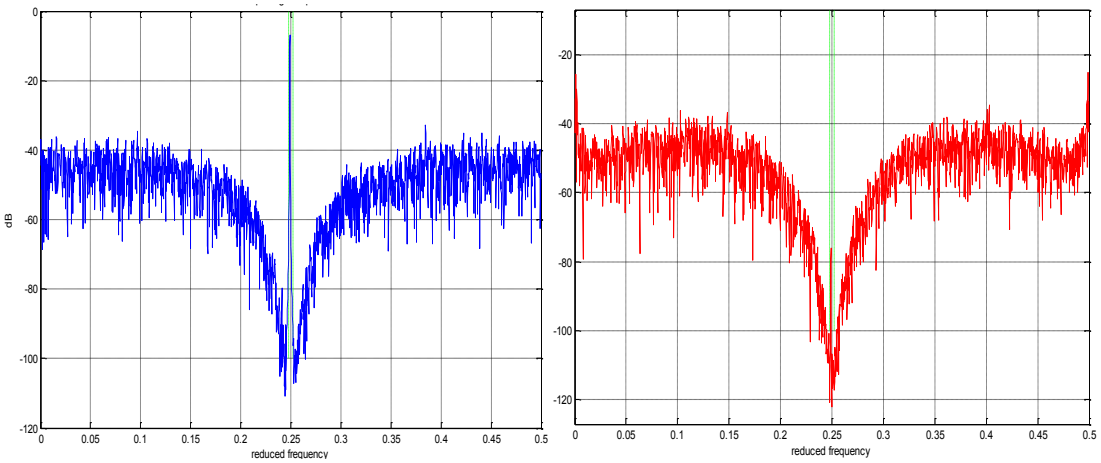


Figure 5.49 6th order modulator VHDL-AMS: *(a)* PSD of the output with normal amplitude stimulus and *(b)* PSD of the output with small amplitude stimulus

In these conditions, a theoretical resolution of 14.35 bits can be achieved for the 6th order modulator.

The temporal evolution of the output in the case of the VHDL-AMS behavioral simulation is shown in Figure 5.50.

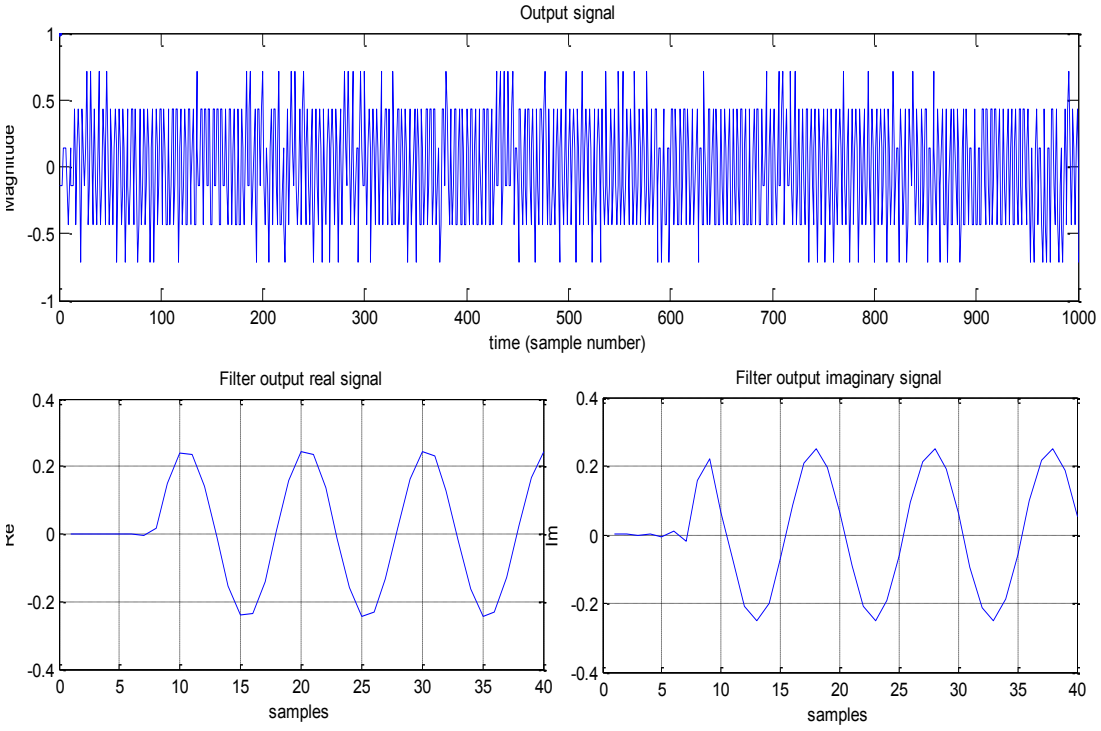


Figure 5.50 Output bits and Filter real and imaginary output signals

Like in the 2nd order case, we present two supplementary evaluations in the transient domain for the VHDL-AMS simulations. The 6th order modulator power-up is shown in Figure 5.51, for an input signal of type *100MHz* sinusoidal. The 6th order functionality for an input of type modulated sine is shown in Figure 5.52. For the two simulations the filter output voltage *Vout_1* is displayed along with the input currents for the *tr_reel1 (QD2)* trans-impedance and *mi_3 (QD12)* current mirror (in these points the feedbacks from DACs are summed).

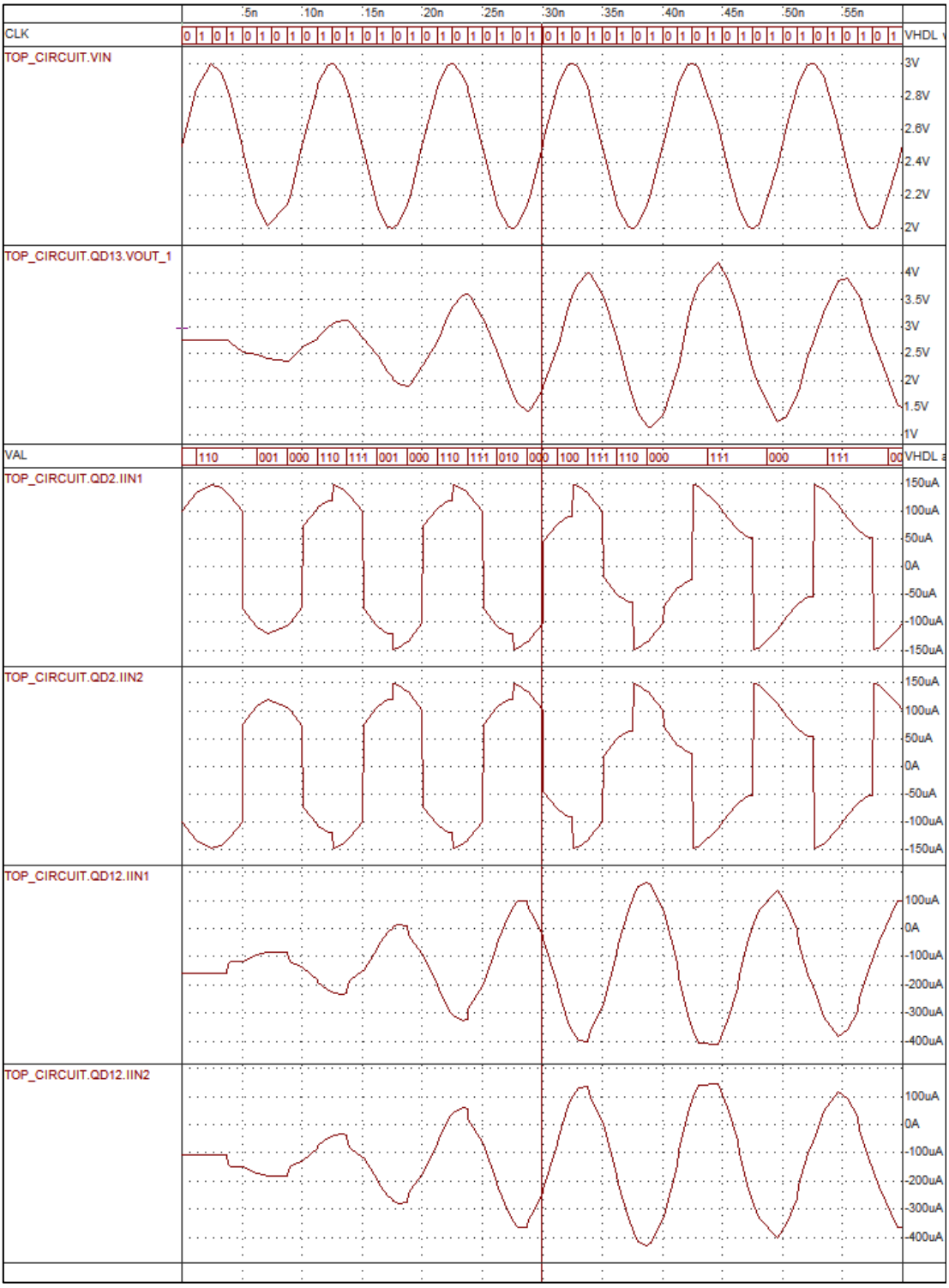


Figure 5.51 6th order modulator power-up – VHDL-AMS simulation

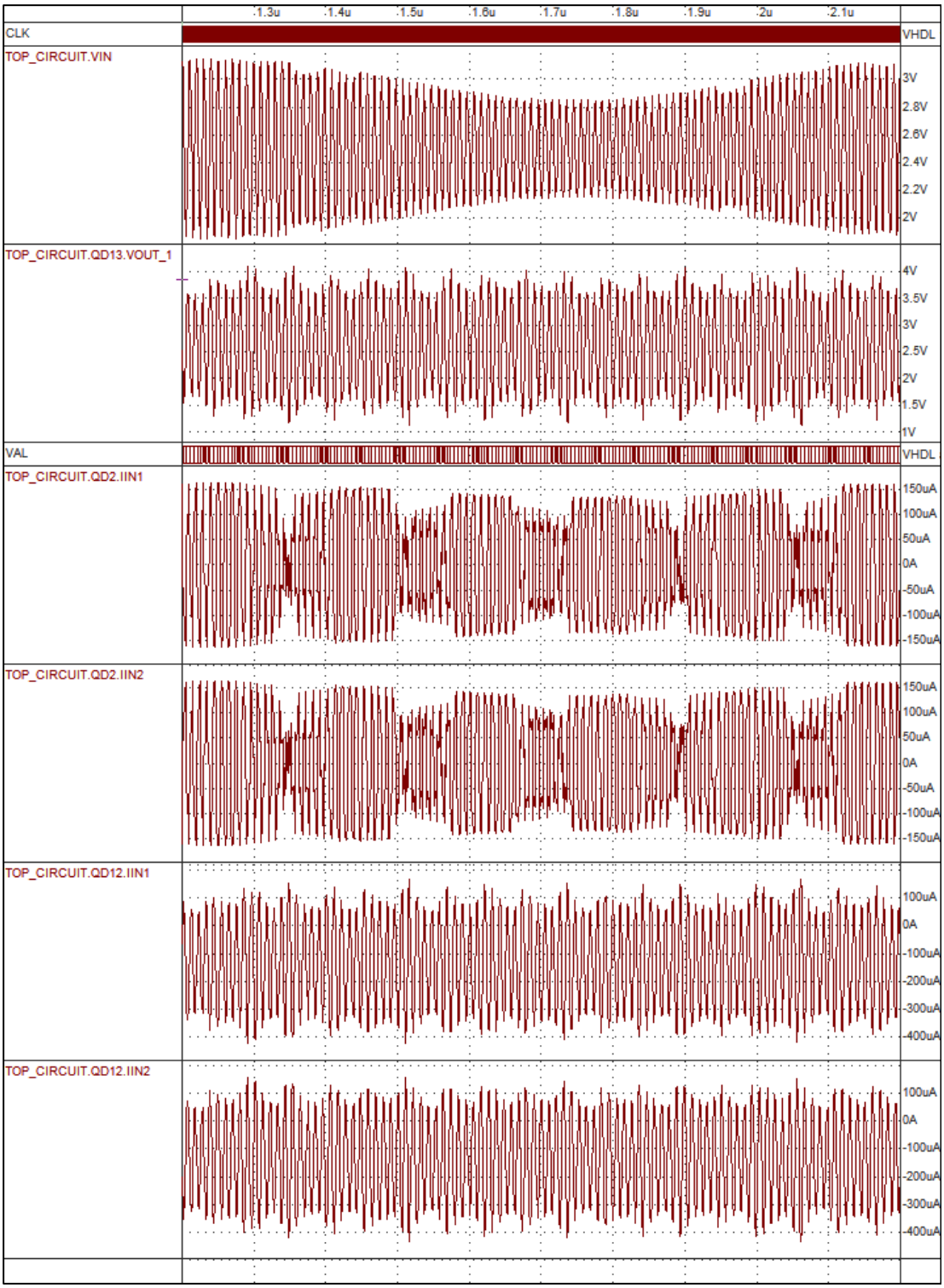


Figure 5.52 6th order modulator functionality – VHDL-AMS simulation

5.4.3. ΣΔ Modulators Design in the Presence of LWRs Parasitic Effects

We study here the effects of introducing the real measured functions of the LWRs and then the parasitic effects on the performance of the studied ΣΔ architectures.

In the 2nd order case, a first study concerns the functionality of the modulator and its performances when the real resonance functions without the parasitic effects are employed. For this purpose, the output signal power spectral density when normal-amplitude input stimulus is applied and the output power spectral density when small-amplitude input stimulus is applied in the case of the SIMULINK simulations were studied. They are presented in Figure 5.53.

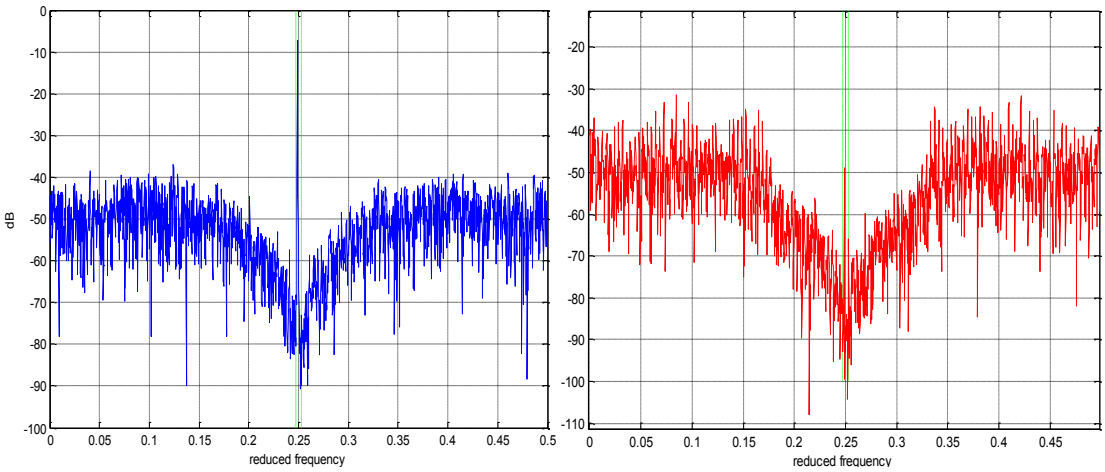


Figure 5.53 2nd order modulator SIMULINK – real functions for LWRs: *(a)* PSD of the output with normal amplitude and *(b)* PSD of the output with small amplitude input

In this case, a theoretical resolution of 10.82 bits can be achieved, resulting in approximately 1 bit resolution degradation when compared with the ideal implementations. This is a realistic figure of performance, as long as the real extracted functions of amplifiers were used, the DAC and ADC were accurately modeled and the real measured functions of resonators were used.

The modulator remains stable and further improvement can be envisaged if amplifiers gains are correctly tuned and potentially the ΣΔ architecture is reshaped function of the new constraints at transistor-level: the ADC/DAC levels should be slightly adjusted to meet the new amplifiers gains, the loop delay and particularly the fraction of the loop delay introduced by the DAC should be correctly adjusted function of the new delay introduced by the analog part.

Next, the 2nd order modulator is designed using the LWRs real functions with parasitic resonances, as shown in Figure 5.29.

Like previously, we studied the performances by computing the output signal power spectral density when normal-amplitude input stimulus is applied and the output power spectral density when small-amplitude input stimulus is applied. In the case of the SIMULINK simulations they are presented in Figure 5.54.

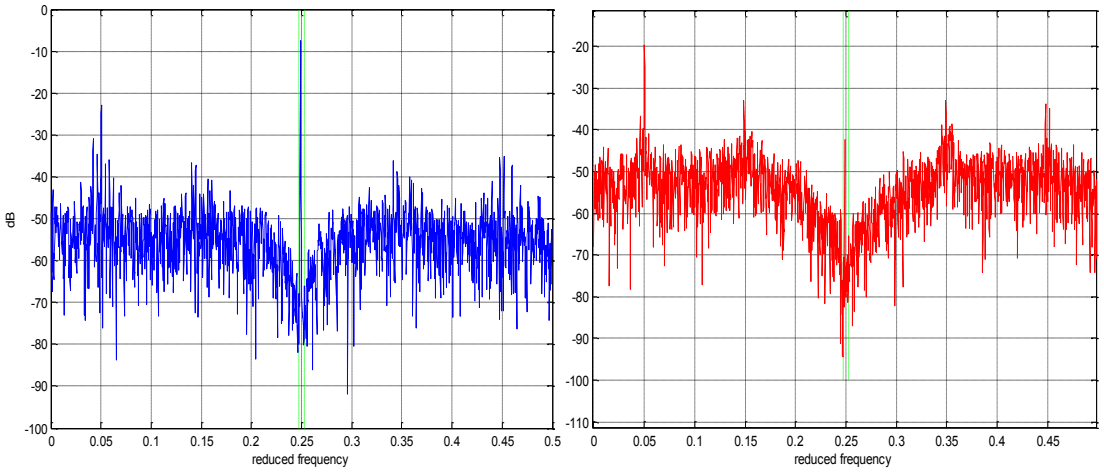


Figure 5.54 2nd order modulator SIMULINK – real functions with parasitic: *(a)* PSD of the output with normal amplitude and *(b)* PSD of the output with small amplitude input

In the case when parasitic resonances are considered, a theoretical resolution of 9.21 bits can be achieved, resulting in approximately 1.5 bits resolution lost when compared with the previous implementation, when the real functions without parasites were considered. Practically, it was determined that the modulator remains stable only if the first parasitic resonance is situated at a frequency equal or higher than $2.4 \cdot fc$ where fc=0.25*fs* is the modulator central frequency. In all other cases, no stable structure was found.

The same approach was used in the study of the 6th order ΣΔ modulator. A first study concerns the functionality of the 6th order modulator and its performances when the real resonance functions without the parasitic effects are employed. The output signal PSD when normal-amplitude input signal is applied and the output PSD when small-amplitude input signal is applied are presented in Figure 5.55 for the SIMULINK simulations case.

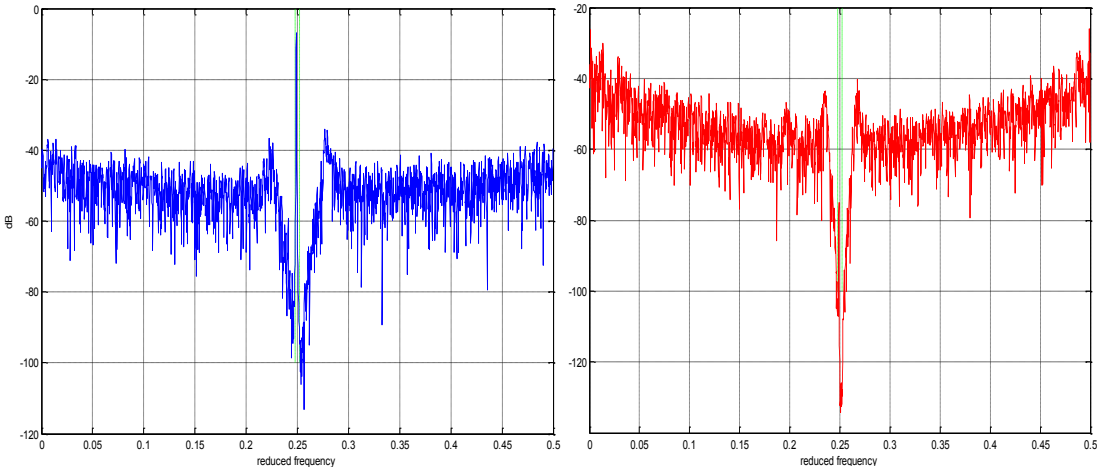


Figure 5.55 6th order modulator SIMULINK – real functions for LWRs: *(a)* PSD of the output with normal amplitude and *(b)* PSD of the output with small amplitude input

In this case, a theoretical maximal resolution of 12.20 bits can be achieved, resulting in more than 2 bits resolution degradation when compared with the ideal implementations. Small variations on the amplifiers gains show that the modulator functions at its stability margin and practical technology implementation could result in a non-stable architecture. One solution for this issue could be the active compensation for LWRs parasitic capacitive-resistive effects, resulting in a different ΣΔ filter structure, probably more complex. Another approach would be to introduce more degrees of freedom (minimum +1) in the filter in order to control better the modulator stability when real functions are employed. This direction needs further investigation and can be a perspective of this work.

Additionally, when the real LWRs functions with parasitic resonances are considered, the 6th order modulator becomes unstable.

Like previously, we tried to study the performances by computing the output signal power spectral density when normal-amplitude input stimulus is applied and the output power spectral density when small-amplitude input stimulus is applied. In the case of the SIMULINK simulations with real LWRs functions with parasitic resonances they are presented in Figure 5.56.

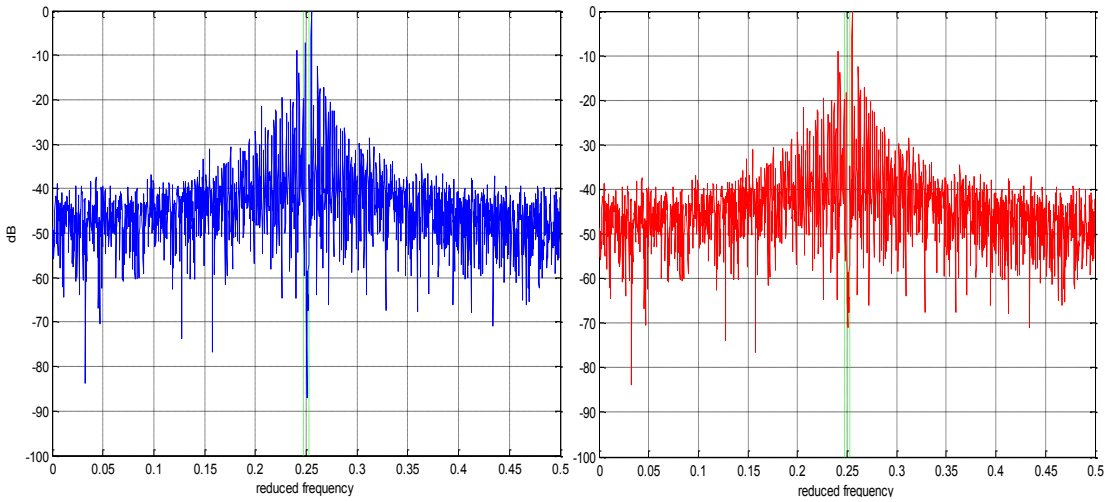


Figure 5.56 6th order modulator SIMULINK – real functions with parasitic: *(a)* PSD of the output with normal amplitude and *(b)* PSD of the output with small amplitude input

No improvement can be found when the parasitic resonances are varied in the frequency band of interest according to the LWRs technology characteristics. The resulting architectures are not stable.

Possible solutions reside in the usage of other high-order architectures (e.g. potentially a 4th order architecture would be a good compromise) or in a different ΣΔ modulator band-pass filter structure.

5.5. Synthetic Results, Simulation Precision and Used Resources

A synthesis of the theoretical resolutions obtained from the calculus of the *SNRs* for the different implementations of the two architectures will permit to compare them and to study the coherence and validity of the proposed methodology.

The Table 5-4 presents synthetically the results obtained in Section 5.4., for comparison. The results are coherent under the same design condition and degrade when realistic figures of merit are considered for the resonators.

Modulator Design	Simulation Scenario	Theoretical Resolution
Band-pass ΣΔ 2nd order	Transistor-level, ideal LWR	11.57 bits
	Macro-model Simulink, ideal LWR	11.44 bits
	Macro-model VHDL-AMS, ideal LWR	11.68 bits
	Macro-model Simulink, real LWR	10.82 bits
	Macro-model Simulink, real + harmonics LWR	9.21 bits
Band-pass ΣΔ 6th order	Transistor-level, ideal LWR	15.5 bits [39]
	Macro-model Simulink, ideal LWR	14.78 bits
	Macro-model VHDL-AMS, ideal LWR	14.35 bits
	Macro-model Simulink, real LWR	12.20 bits
	Macro-model Simulink, real + harmonics LWR	Unstable

Table 5-4 Simulation Scenarios and results for the two Σ Δ modulator architectures

Another important aspect for this methodology concerns the precision of the simulations and the computation times needed to simulate the separate components and an entire ΣΔ modulator at system-level using the different high-level approaches and at transistor-level.

All the small-signal analyses and the PSDs were compared in the transistor-level and macro-model level cases. Additionally, direct transient analyses were compared for simulations at stage-level and different scenarios at system-level.

The maximum total relative errors normalized to the number of points between the transistor-level and macro-models characteristics were used to predict the accuracy of the models. We obtained errors in $[10^{-8}; 10^{-6}]$ for 2^{nd}-6^{th} order components models for small-signal and PSDs. The transient analyses were coherent in the sense that the architectures were functional and presenting similar performances.

A key factor of the methodology resides in the speed improvement for transient analyses: factors of 10×-30× were obtained for specific components and a whole ΣΔ modulator. Table 5-5 shows the comparative computation times needed for 1000 output samples for multiple components of the 6^{th} order modulator and for the entire modulator.

Simulation / Design	Transistor-level (CADENCE Spectre)	Macro-models (SIMULINK)	Behavioral (SMASH: VHDL-AMS)
Tr1	24.5s	2.2s	2.1s
Mirroir1	47.8s	5.7s	4.2s
Tr2	22.1s	2.1s	2.1s
Mirroir2	30.3s	4.9s	2.9s
Tr3	25.1s	2.2s	2.2s
Mirroir3	19.5s	2s	1.9s
TIA4	15.7s	1.5s	1.2s
ADC 3bits	38.8s	2.5s	4.2s
DAC 3bits	40.9s	2.4s	4.1s
$\Sigma\Delta$ 6^{th} order	3h37m1s	7m8s	6m41s

Table 5-5 Computation times for transient analyses (components and $\Sigma\ \Delta$ modulator)

For 1000 output samples of the 2^{nd} order modulator, the simulation required 1h45m12s on the analog simulator, 5m48s on SIMULINK and 5m27s for VHDL-AMS (under Dolphin Integration SMASH), implying a factor of approximately 25x speed improvement.

For particular simulations, when employing the high-level models of type VHDL-AMS in multi-level simulations using the Cadence AMS simulator, convergence problems were noted. Keeping a part of the design at transistor level for in-depth verification and other parts at behavioral level resulted in some cases in non-convergence of the solution or very small simulation steps (e.g. 10^{-15}s). In this case, behavioral models of type Verilog-A were used for the amplifiers, assuring the solvability and the interface problems correction in Cadence.

5.6. Chapter Conclusion

The design methodology for macro-models extraction and high-level modeling of complex CT functions was employed in this chapter for the design of a 2nd order Sigma-Delta modulator and a 6th order modulator architecture.

In order to automate the analysis and synthesis processes, the MATLAB / SIMULINK – CADENCE – VHDL-AMS / VerilogA framework which was developed (SIMECT), was used in the design process. This application can start automatically all kinds of simulations (DC, AC, parametric, transient, Harmonic Balance), and can, from these simulations, obtain macro-models of amplifying functions. The models can be exported as SIMULINK blocks or VHDL-AMS/VerilogA behavioral modules and used for system-level implementation. This framework also allows to automatically optimizing the transistor-level cells based on a single optimization objective and one or more constraints.

As a first approach, the analog active elements of the Sigma-Delta filter were optimized and macro-modeled in SIMULINK, VHDL-AMS or VerilogA.

The automatic optimization techniques for CT amplifiers based on stochastic algorithms were then applied in order to obtain the best cells configurations function of the requirements at system-level. The improved nonlinear characterization in the macro-models and the technology-accurate functions extractions provided accurate high-level models which include the low-level characteristics.

A second approach was to model the micro-mechanical resonators of type LWR used in the Sigma-Delta architectures.

Finally, the mixed-signal nature components of the modulators were modeled: the ADCs and DACs.

Using all these components, the two architectures were implemented at high-level and compared with the transistor-level counterparts. The consistency of the results suggested that the high-level architectures can be used to further study the Sigma-Delta architectures, without employing also the transistor-level simulations, principally time-consuming.

A further research direction consisted in the study of the impact of LWRs real measured functions, potentially including parasitic effects, on the stability and performance of the modulators.

The usage of this methodology assures a considerable speed improvement and consistent results. Comparable results are expected when using high-level macro-models of type Verilog-A exclusively. For this work, the Verilog-A models where used only to solve convergence and interface issues in the Cadence environment.

The proposed application framework and additional conception examples are available via the MathWorks website at [148].

VI

6. Conclusions and Perspectives

6.1. Conclusions

This research work was about the initiation of a novel analog design methodology, the theoretical fundaments, the specific algorithms and the corresponding design tools which can be employed in the dynamic conception of weakly non-linear continuous-time (CT) functions, in the context of an increasing complexity of nowadays electronic systems, both structurally and functionally.

Starting from the existing top-down and bottom-up design paradigms, the existing state-of-the-art EDA tools and analyzing the requirements for the conception of high-performance circuits in the case of band-pass CT Sigma-Delta modulators, an efficient analog design methodology was presented.

This methodology is a top-down combined with a bottom-up design chain based on the automatic optimization of transistor-level analog cells using a modified Bayesian Kriging method and the synthesis of robust analog macro-models in analog and mixed-signal HDLs (e.g. VHDL-AMS, Verilog-A) or simulation environments (e.g. Mathworks Matlab-Simulink).

For this methodology, a first research step consisted in establishing a macro-model extraction technique involving a complete set of analyses (DC, AC, transient, parametric, Harmonic Balance) which are performed on the analog schematics implemented on a specific technology process. It was shown how macro-models of analog functions can be directly synthetized from a multitude of figures of merit extracted in the simulation process: transfer functions and impedances extracted from AC measures, offsets and nonlinearities extracted from DC and parametric simulations and HF nonlinearities and distortions obtained from Harmonic Balance scenarios.

Different aspects for the architectures of macro-models were studied to allow the extraction of any kind of amplification function, differential or unipolar, potentially with multi-output capabilities.

As long as the models are obtained from transistor-level simulations, they bring valuable low-level characteristics at system-level and can be directly regenerated in any stage of the design process.

But it is important the models orders to be correctly imposed and the simulations for models extraction to be performed in the domains of interest (frequency, DC, parametric) and on already optimized cells.

Then, another research direction explored the possibilities of automatic optimization of analog cells or automatic transistor-level sizing for medium-scale circuits.

In a first step, we focused on finding and implementing a simple and intuitive method, which can be rigorously controlled and can bring some optimization results, based mainly on the designer experience and on simulations automation. Thus, a semi-automatic optimization method based on a pseudo-gradient descent algorithm was implemented. This method can be used for initial explorations of the analog designs performances and for fine-tuning the manual optimization results, although it is time-consuming and entirely based on transistor-level simulations. But its main importance is related to de fact that it can be used for validation and coherence approval with any automatic optimization method.

In parallel with an internship, an automatic optimization technique for analog cells was developed. It uses a Bayesian method, where the evaluation space is created by the means of a Kriging surrogate model, and the selection is effectuated by using the expected improvement (EI) criterion subject to constraints.

Our original approach was to consider the optimization objectives along with 2 types of constraints: strong constraints (nonlinearities, distortions) and normal constraints (offsets, impedances, gains, bandwidths), which are adequate for analog characterization, ensuring robust optimal solutions on a limited number of evaluations.

Parallel applications of the two methods proved that the two techniques are coherent and the automatic optimization method is very valuable, with better temporal performances, when its parameters are correctly tuned.

Still, in the current implementation of the Bayesian Kriging optimization method there exists a technical limitation: due to representation of matrix variables on real *double* precision by Mathworks MATLAB, it is not practical to optimize designs with more than 15-20 parametric variables with a normal granularity of the points. This will be further discussed and a solution will be proposed as a perspective of this work.

In parallel with the development of the theoretical work, a conception tool was developed (*SIMECT*), which was integrated as a Matlab toolbox, including all the macro-models extraction and automatic optimization techniques.

The validation of the developed methodology and tools was conducted on a 2^{nd} order and on a 6^{th} order CT Sigma-Delta modulator architecture, based on a 350nm *AMS* technology process and using micro-mechanical resonators of type Lamb Wave Resonator (LWR).

Multiple types of simulations and experiments were conducted in parallel between the transistor-level implementations and the high-level ones, when using the extracted macro-models. Based on the results obtained, the design methodology was validated for coherence with the transistor-level accurate figures of merit and it was shown that it is possible to use the macro-model based implementations for further explorations with a high level of confidence (maximum total relative errors normalized to the number of points between $[10^{-8}; 10^{-6}]$ for 2^{nd}-6^{th} order models, for the small-signal functions and power spectral densities).

Once the methodology was validated, the 2^{nd} order and on a 6^{th} order CT Sigma-Delta modulator architectures were explored in more realistic design conditions, using macro-models and measured responses for the real LWRs provided by CEA-LETI. This allowed a reliable characterization of the stability and expected performances of the resulted Sigma-Delta structures.

It was shown that the 2^{nd} order Sigma-Delta modulator remains stable and achieves a resolution of almost 11-bits when a single-resonance real LWR function is employed, while it can achieve only 9.2-bits and remains stable only in certain configurations when multi-resonance real functions are employed.

For the 6^{th} order Sigma-Delta architecture, it was found that a maximal resolution of 12.20 bits can be achieved, in the presence of a single-resonance real function. But small variations on the amplifiers gains showed that the modulator functions at its stability margin and practical technology implementation could result in a non-stable architecture. The 6^{th} order architecture is found non-stable in the presence of multi-resonance real LWR functions.

Further improvements can be envisaged for these architectures, which will be presented as perspectives of this work.

A final research direction during this PhD thesis concerned a preliminary study for a novel 2^{nd} order band-pass Sigma-Delta architecture, working in an RF band around fc=1.25GHz. Based on barium strontium titanate (BST) adjustable resonators, this application is part of the ARTEMOS European project, currently in progress [13]. For the preliminary architecture, a reference STMicroelectronics 65 nm Low-Power General-Purpose (LPGP) CMOS technology process was used.

6.2. Perspectives

Short-term perspectives and medium/long-term perspectives can be formulated for the research objectives presented in this work.

A first short-term perspective concerns the integration of the HF non-linear effects in the characterization of the analog cells, the implementation of the HF non-linear coefficients extraction and inclusion of these coefficients in the macro-model synthesis method. The theoretical study was already performed and the framework for this operation is already implemented, it remains to find the RF analyses offered by the current EDA tools which could provide the specific non-linearity and distortions coefficients.

Another direction concerns the optimization methods. The automatic optimization method which was demonstrated resides for the moment on a single-objective multi-constraints approach. Practical optimization cases have shown that it is interesting to extend this approach to a multi-objective multi-constraints paradigm. This can be done by using the Pareto criterion or by an appropriate weighting of the multiple objective functions to form a single objective.

Regarding the Bayesian Kriging optimization method, in order to overcome the technical limitation due to *double* precision imposed by Mathworks MATLAB, a C/C++ implementation of the Kriging meta-model and EI predictor functions can be envisaged. In this case, *single* precision real variables would bring sufficient calculus accuracy and the compiled routines would assure a consistent speed improvement of the computations.

One medium-term perspective concerning the Sigma-Delta applications consists in the structural modeling of the micro-mechanical resonators. This kind of modeling was not practicable due to the fact that material and geometrical characteristics of the MEMS were not available during the test period. If CEA-Leti will provide these elements in the following design stages of micro-mechanical resonators, an interesting perspective of this work is to implement the structural models of the resonators and evaluate the system-level results using the behavioral, functional and the structural modeling strategies. In this way, the structural approach explored in this work will also be validated along with the two others.

As medium or long-term perspective, a silicon tape-out and prototype realization of at least one of the studied Sigma-Delta architectures would be a corner-stone validation criterion for the design methodology. But this can be done only after the LWR micro-mechanical resonators achieve at least prototyping stage and the used integration technology is compatible with the deposition of piezoelectric materials in order to implement a monolithic circuit. All the pre-silicon stages would necessarily be achieved with at least a layout-level validation of the real LWRs with the analog and mixed-signal components. The realization of the demonstrators in the ARTEMOS project could bring an achievement in this direction.

Annex

7. ANNEX A

7.1. OCEAN Batch Command Files

The batch files generated by Matlab for OCEAN (in Chapter 3, Section 3.6) contain the following commands used to setup the environment, set design variables, start analog simulations, write results, etc.:

41. The init file (init.ocn) containing the selection of the simulator, the design, results repository and model files:

```
simulator( 'spectre )
design("/Sim/mi1/spectre/schematic/netlist/netlist")
resultsDir("/Sim/mi1/spectre/schematic" )
modelFile(
    '("/usr/AMS_4.0/spectre/s35/mcparams.scs" "")
    '("/usr/AMS_4.0/spectre/s35/cmos53.scs" "cmostm")
    '("/usr/AMS_4.0/spectre/s35/res.scs" "restm")
    '("/usr/AMS_4.0/spectre/s35/cap.scs" "captm")
    '("/usr/AMS_4.0/spectre/s35/vbic.scs" "biptm")
    '("/usr/AMS_4.0/spectre/s35/ind.scs" "indtm")
)
temp( 27 )
```

42. The design variables file (desvar.ocn) containing all the design variables of the schematic to be simulated:

```
desVar( "w3" 1.5e-05 )
desVar( "l3" 1e-06 )
desVar( "r" 1510 )
desVar( "vout1" 2.5 )
```

43. The analyses file (ac.ocn) which contains all the simulations to be performed and other instructions:

```
save( 'i "V30:n" )
analysis('dc ?dev ""  ?param "vin" ?start "-1" ?stop "1" )
analysis('ac ?start "100000"  ?stop "10000000000000"  ?dec "100"  )
run()
delete( 'analysis 'ac)
par1 = paramAnalysis( "vout2" ?start 2.4 ?stop 2.6 ?step 0.01 )
paramRun('par1)
exit
```

8. ANNEX B

We present here the figures of merit extracted from DC, Parametric, AC and transient analyses for the Sigma-Delta components detailed in Chapter 5, Section 5.3.

8.1. Trans-impedance Amplifier Extracted Parameters

The DC and Parametric figures of merit for the *TIA* amplifier:

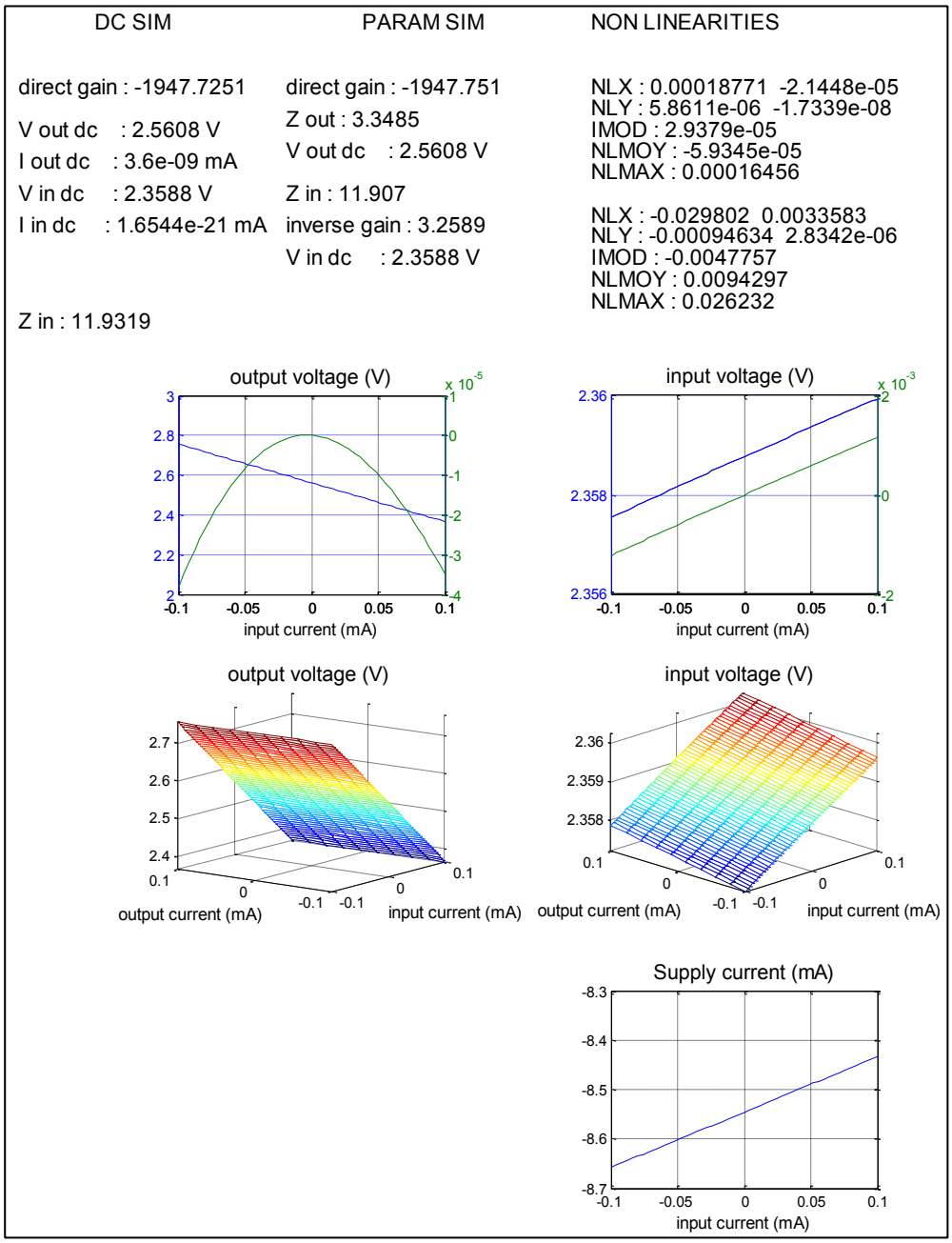


Figure 8.1 TIA Amplifier DC in and DC out responses and extracted parameters

The AC and transient figures of merit for the *TIA* amplifier:

Zout at f0:	3.7782	Rs in :	11.907
		R in :	558.4878
Zin at f0:	85.2356	L in :	1.3236e-07
		C in :	1.0044e-13
FC trans :	1.319e+09	Rs out :	3.3476
		R out :	36.3507
Delay @ f0 (ns):	-0.24083	L out :	2.9227e-09
		C out :	5.7261e-14
Quality factor:	3.2359e-08		

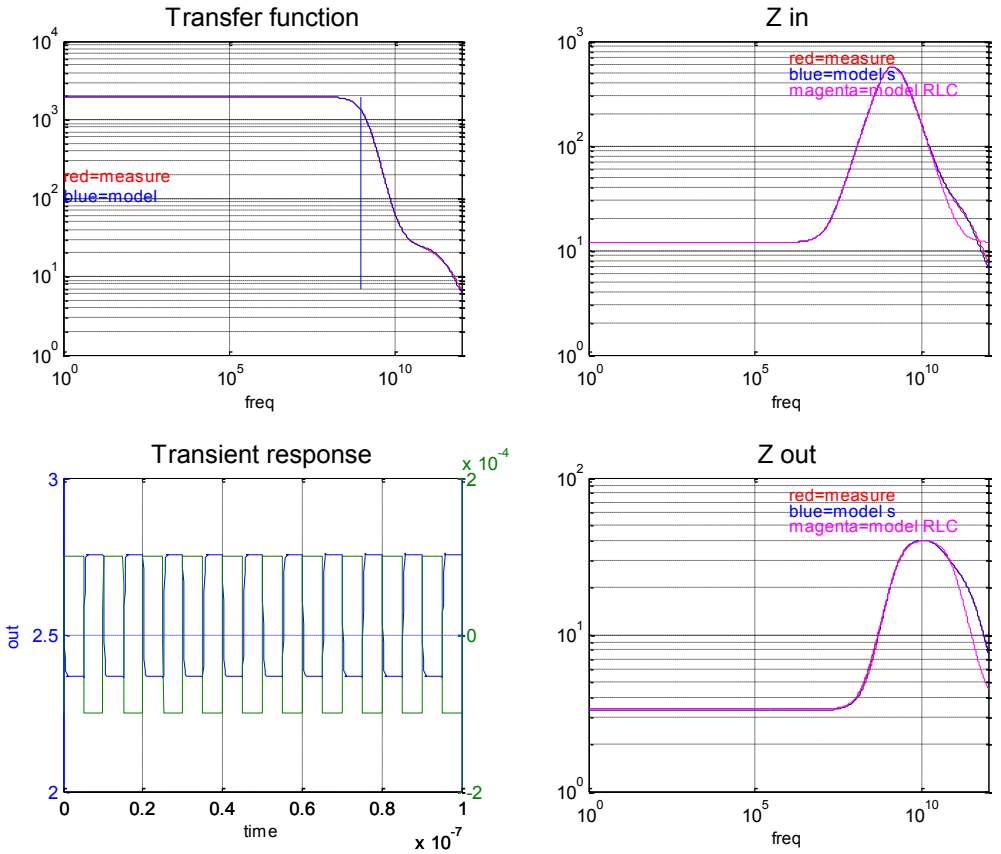


Figure 8.2 TIA Amplifier AC and transient responses and extracted parameters

8.2. Trans-conductance Amplifier Extracted Parameters

The DC and Parametric figures of merit for the *Gm* amplifier in the case of the first output (*Iout-*):

DC SIM

direct gain : -5.7445e-07
I out dc : -0.49557 mA
V out dc : 2.34 V
I in dc : -0.0048499 mA
V in dc : 2.5 V

Z in : -50012116.5327

PARAM SIM

direct gain : -5.7325e-07
Z out : 318777.3532
I out dc : -0.49557 mA
Z in : 50137931.8751
inverse gain : 1.7194e-20
I in dc : -0.0048499 mA

NON LINEARITIES

NLX : -0.031337 0.0033345
NLY : -0.095744 -0.0067492
IMOD : -0.00037233
NLMOY : -0.03573
NLMAX : 0.080133

NLX : 0.023199 0.0040336
NLY : 3.5063e-12 7.6784e-13
IMOD : 3.5468e-13
NLMOY : -0.0072185
NLMAX : 0.016918

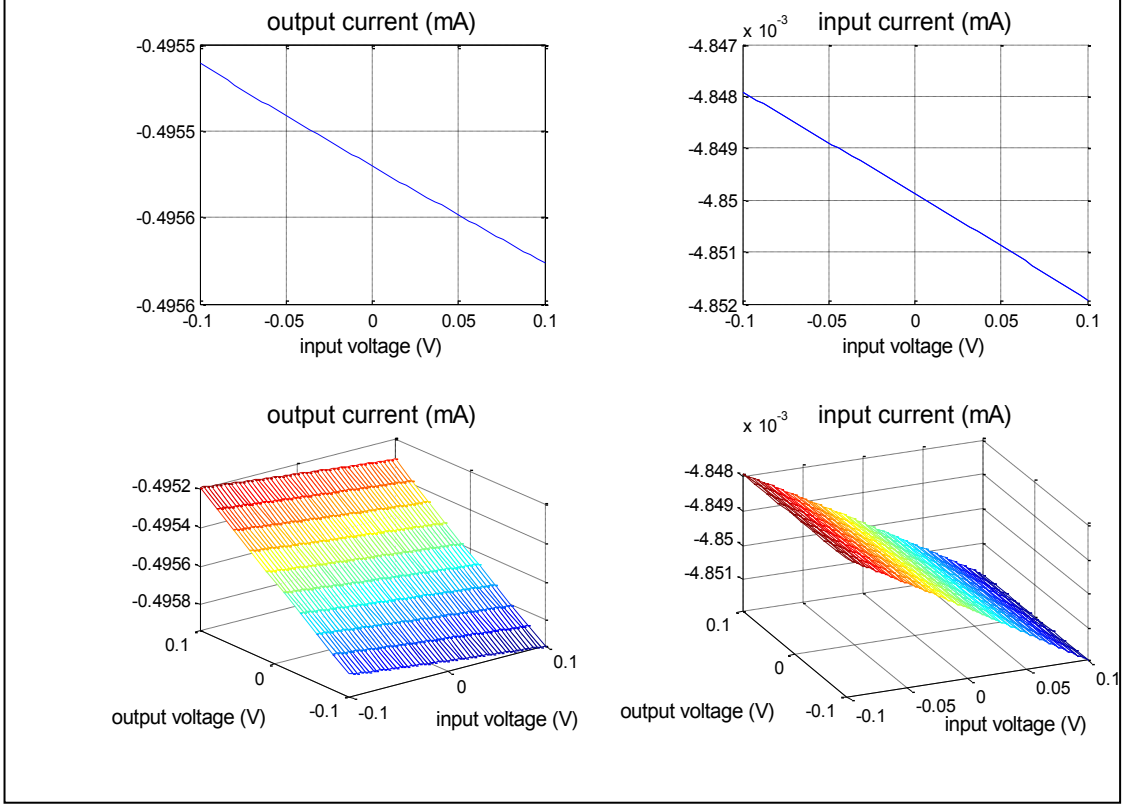


Figure 8.3 Trans-conductance amplifier DC in and DC out for *Iout-* output responses and extracted parameters

The DC and Parametric figures of merit for the *Gm* amplifier in the case of the second output (*Iout*+):

DC SIM

direct gain : 5.7445e-07
I out dc : 0.49964 mA
V out dc : 2.34 V
I in dc : -0.0048499 mA
V in dc : 2.5 V

Z in : -50012116.5327

PARAM SIM

direct gain : 5.7341e-07
Z out : 3847506.3436
I out dc : 0.49964 mA
Z in : 50137931.8752
inverse gain : 1.2923e-20
I in dc : -0.0048499 mA

NLX : 0.023199 0.0040336
NLY : 2.1626e-12 5.7427e-13
IMOD : 1.284e-13
NLMOY : -0.0072185
NLMAX : 0.016918

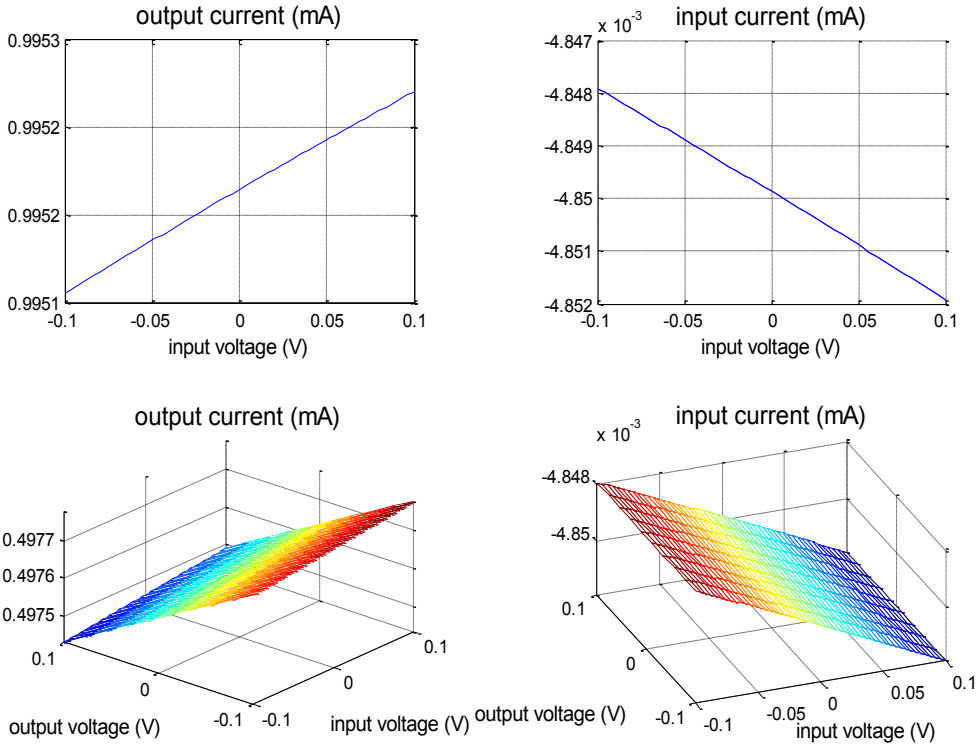


Figure 8.4 Trans-conductance amplifier DC in and DC out for *Iout+* output responses and extracted parameters

The DC and Parametric figures of merit for the *Gm* amplifier in the differential output configuration:

DC SIM

direct gain : -2.8722e-07

I out dc : -0.49557 mA

V out dc : 2.34 V

I in dc : -0.0048499 mA

V in dc : 2.5 V

Z in : -50012116.5327

PARAM SIM

direct gain : -2.8663e-07

Z out : 318777.3532

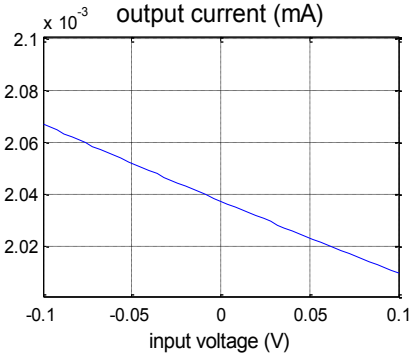
I out dc : -0.49557 mA

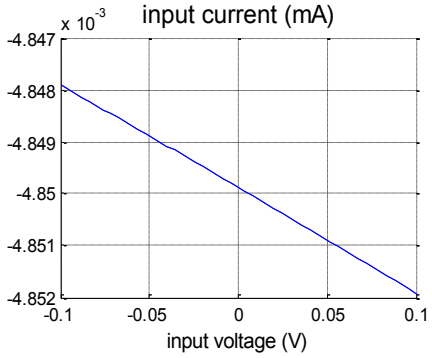
Z in : 50137931.8751

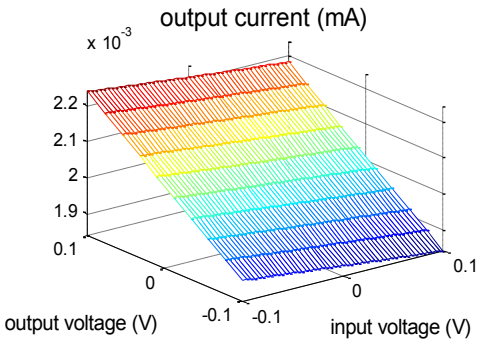
inverse gain : 1.7194e-20

I in dc : -0.0048499 mA

NLX : 0.023199 0.0040336
NLY : 3.5063e-12 7.6784e-13
IMOD : 3.5468e-13
NLMOY : -0.0072185
NLMAX : 0.016918







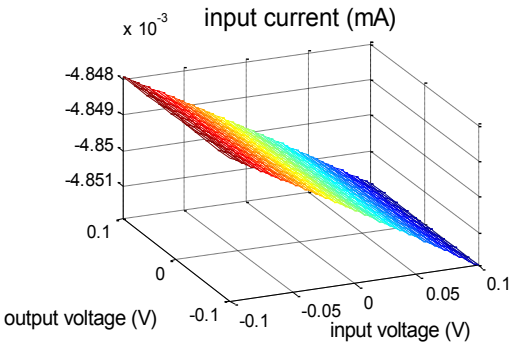


Figure 8.5 Trans-conductance amplifier DC in and DC out in differential mode responses and extracted parameters

The DC and Parametric figures of merit for the *Gm* amplifier in the common-mode output configuration:

DC SIM	PARAM SIM	NON LINEARITIES
direct gain : 1.0049e-12	direct gain : 1.005e-12	NLX : 1.1684e-05 -5.1354e-05
I out dc : 0.49964 mA	Z out : 3847506.3436	NLY : 6067.2243 2437.4044
V out dc : 2.34 V	I out dc : 0.49964 mA	IMOD : -0.00037841
I in dc : -0.0048499 mA	Z in : 50137931.8752	NLMOY : 2227.7003
V in dc : 2.5 V	inverse gain : 1.2923e-20	NLMAX : 4316.4094
Z in : -50012116.5327	I in dc : -0.0048499 mA	NLX : 0.023199 0.0040336
		NLY : 2.1626e-12 5.7427e-13
		IMOD : 1.284e-13
		NLMOY : -0.0072185
		NLMAX : 0.016918

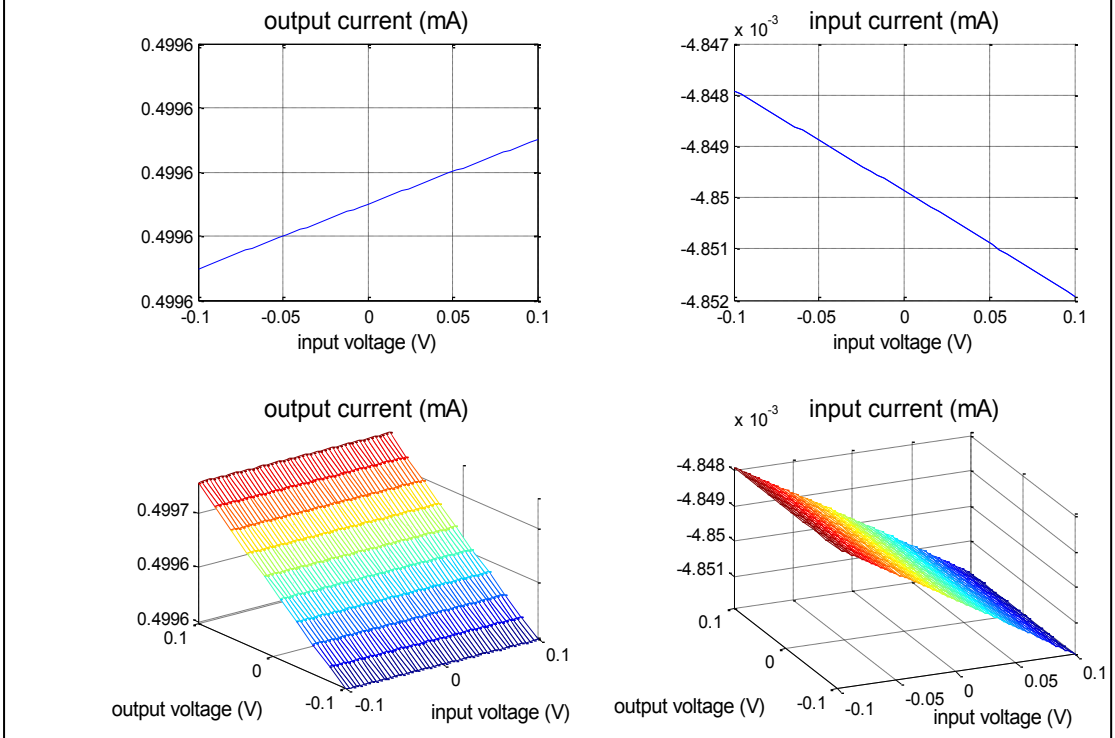


Figure 8.6 Trans-conductance amplifier DC in and DC out in common mode responses and extracted parameters

The AC and transient figures of merit for the *Gm* amplifier:

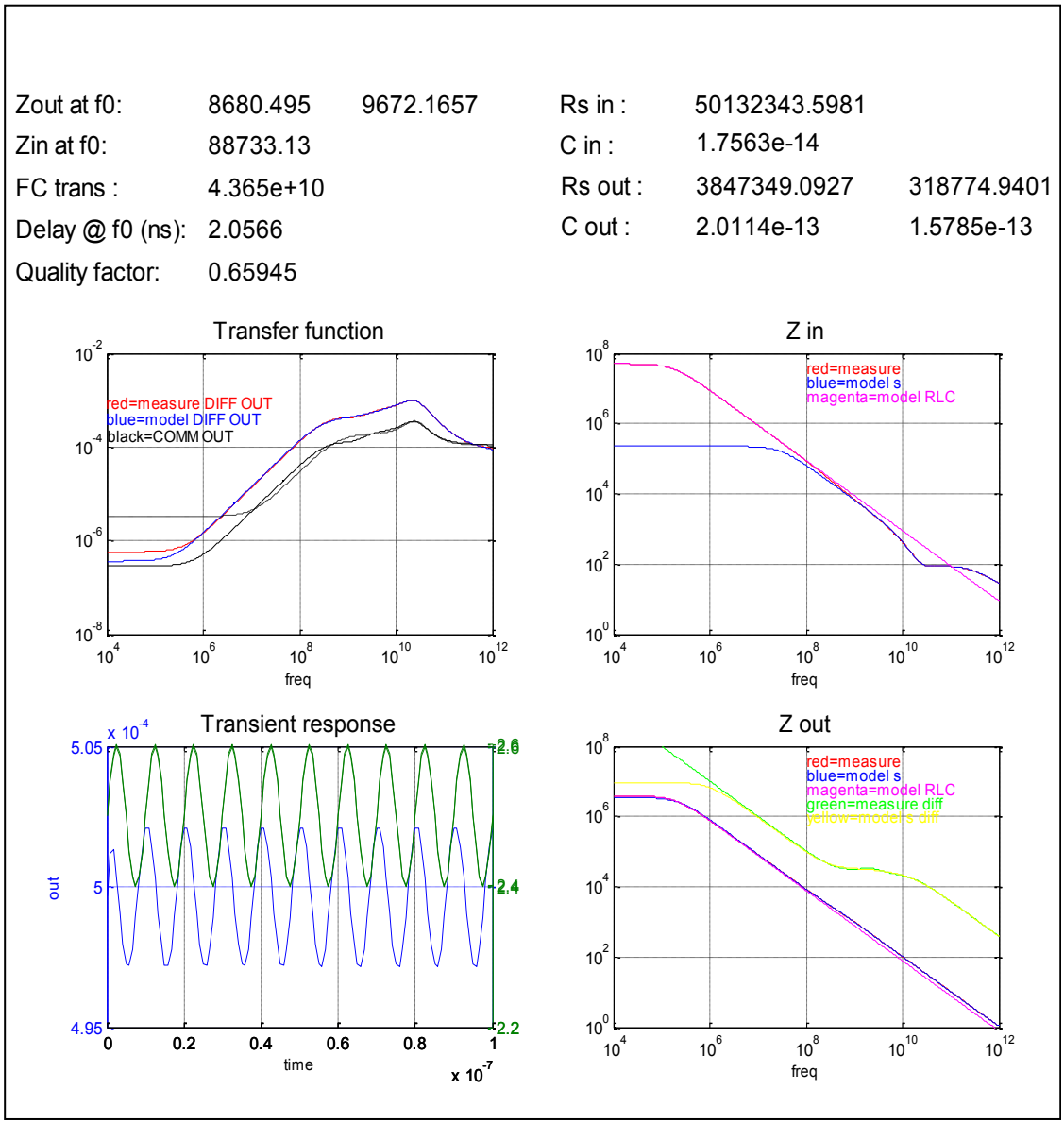


Figure 8.7 Trans-conductance amplifier AC and transient responses and extracted parameters

8.3. Current Mirror Extracted Parameters

The DC and Parametric figures of merit for the *current mirror* in the case of the first output (*Iout-*):

DC SIM	PARAM SIM	NON LINEARITIES
C.M. gain : 0.0076217	C.M. gain : 0.010346	
direct gain : -0.97913	direct gain : -1.0266	NLX : -0.0051049 0.081089
	Z out : 7899.6242	NLY : 0.0045955 0.0046704
		IMOD : 0.023844
I out dc : -0.045669 mA	I out dc : -0.045669 mA	NLTOT : 4.3687e-05
Z in : 15.1299	Z in : 12.23	NLX : -3.4119 4.6496
	inverse gain : -1.4033e-10	NLY : 2.1868e-08 4.1051e-08
		IMOD : -5.7683e-08
V in dc : 2.1399 V	V in dc : 2.1399 V	NLTOT : 0.0026578

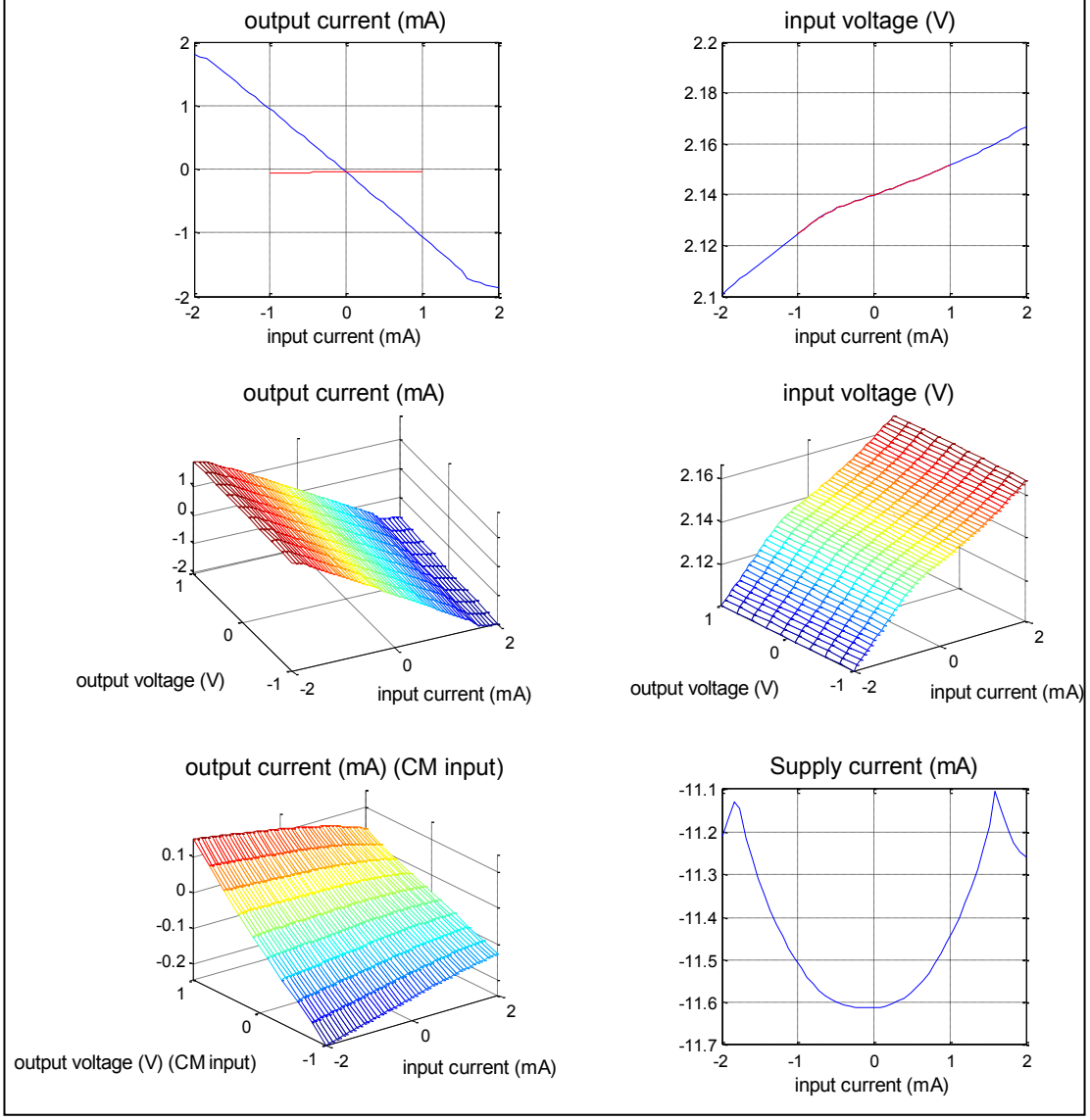


Figure 8.8 Current mirror DC in and DC out for *Iout-* output responses and extracted parameters

The DC and Parametric figures of merit for the *current mirror* in the case of the second output (*Iout+*):

DC SIM	PARAM SIM	NON LINEARITIES
C.M. gain : 0.030754	C.M. gain : 0.029458	
direct gain : 0.97305	direct gain : 1.0189	NLX : -0.059007 -0.078488
	Z out : 8218.8175	NLY : 0.0039294 -0.0049026
		IMOD : 0.025782
I out dc : 0.15435 mA	I out dc : 0.15435 mA	NLTOT : 5.6072e-05
Z in : 15.1299	Z in : 12.23	NLX : -3.4119 4.6496
	inverse gain : -1.4033e-10	NLY : 2.1868e-08 4.1051e-08
		IMOD : -5.7683e-08
V in dc : 2.1399 V	V in dc : 2.1399 V	NLTOT : 0.0026578

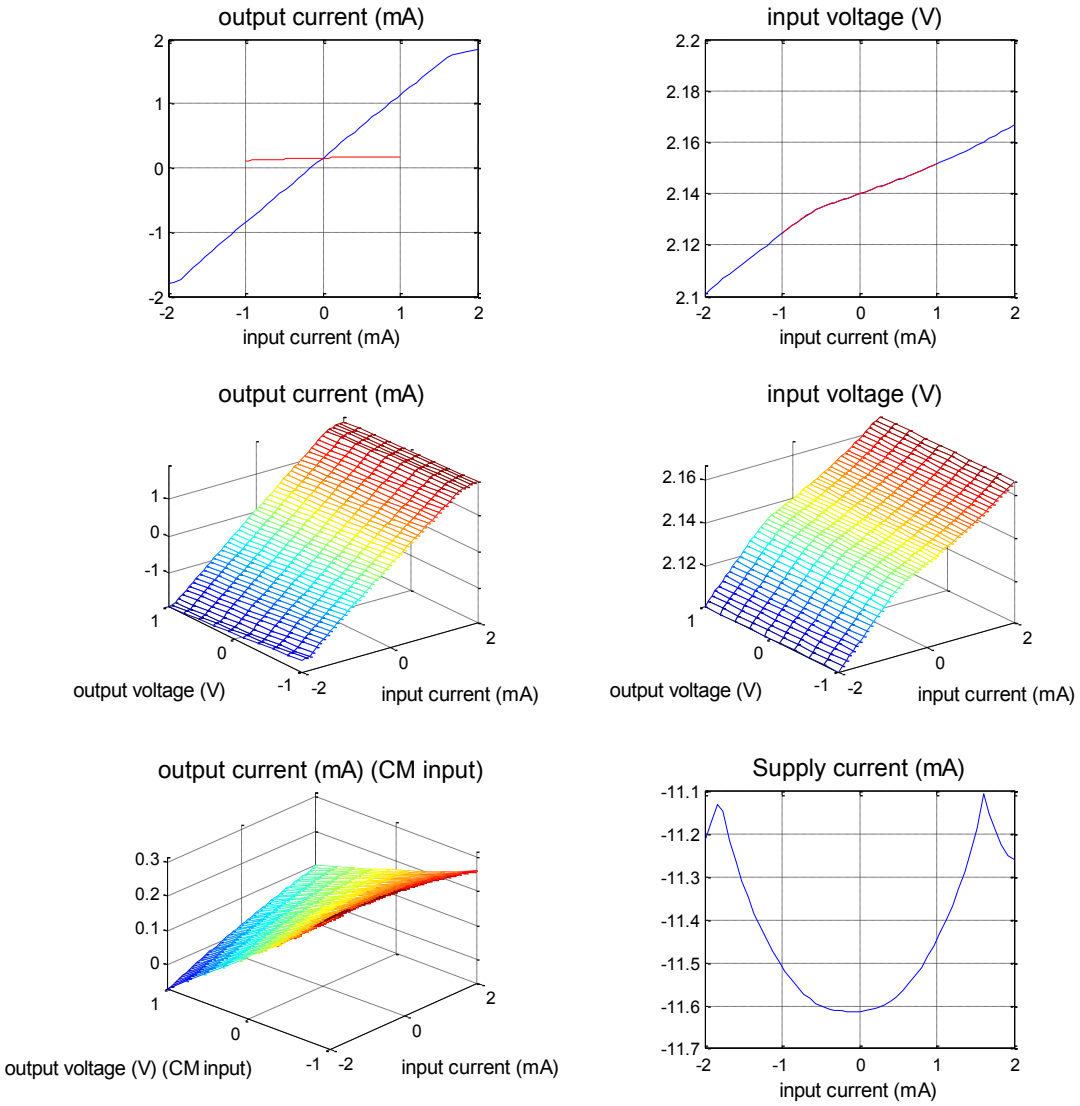


Figure 8.9 Current mirror DC in and DC out for Iout+ output responses and extracted parameters

The DC and Parametric figures of merit for the *current mirror* in the differential output configuration:

DC SIM	PARAM SIM
C.M. gain : -0.011566	C.M. gain : -0.0095564
direct gain : -0.97609	direct gain : -1.0227
	Z out : 7899.6242
I out dc : -0.045669 mA	I out dc : -0.045669 mA
Z in : 15.1299	Z in : 12.23
	inverse gain : -1.4033e-10
V in dc : 2.1399 V	V in dc : 2.1399 V

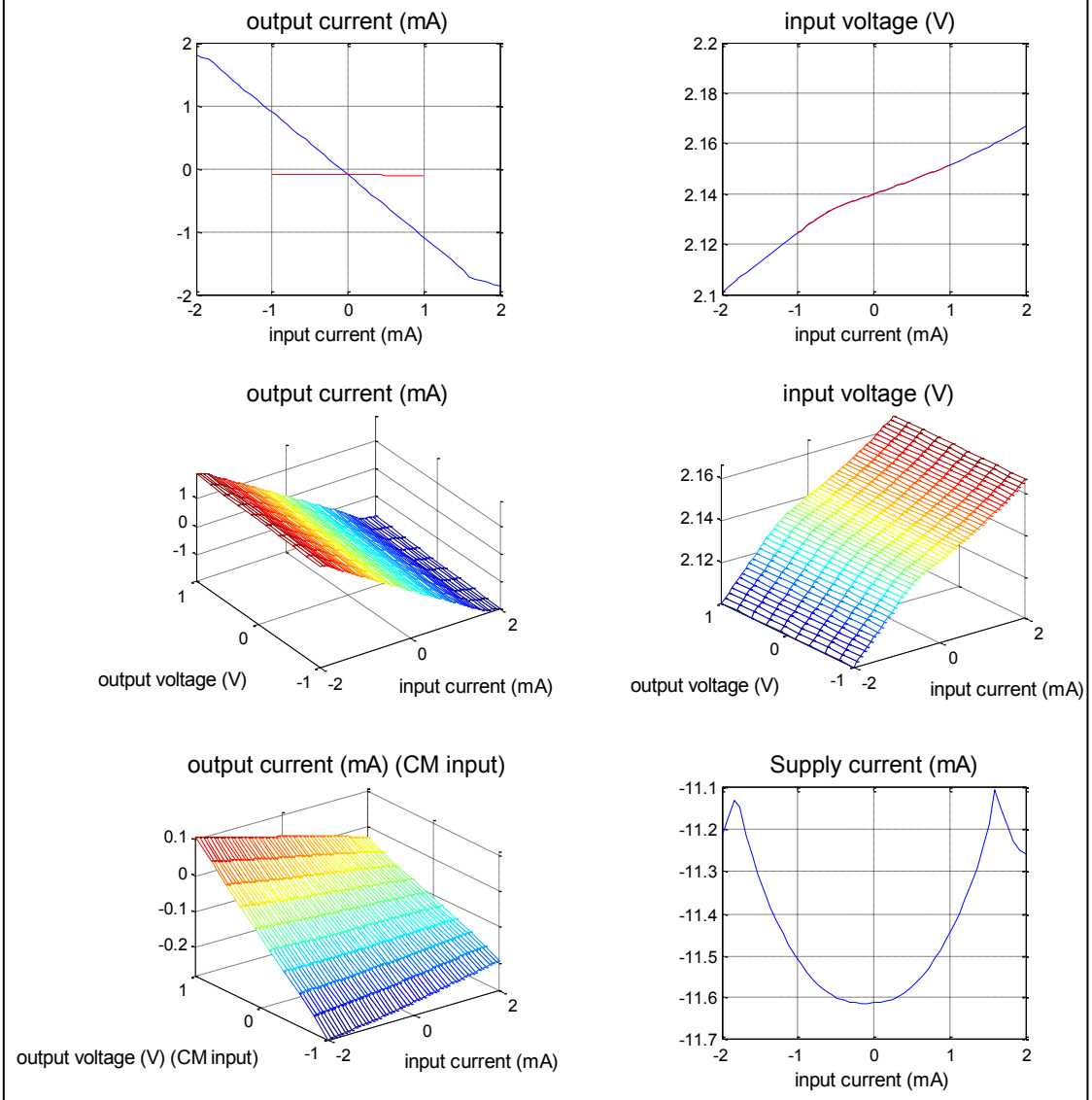


Figure 8.10 Current mirror DC in and DC out in differential mode responses and extracted parameters

The DC and Parametric figures of merit for the *current mirror* in the common-mode output configuration:

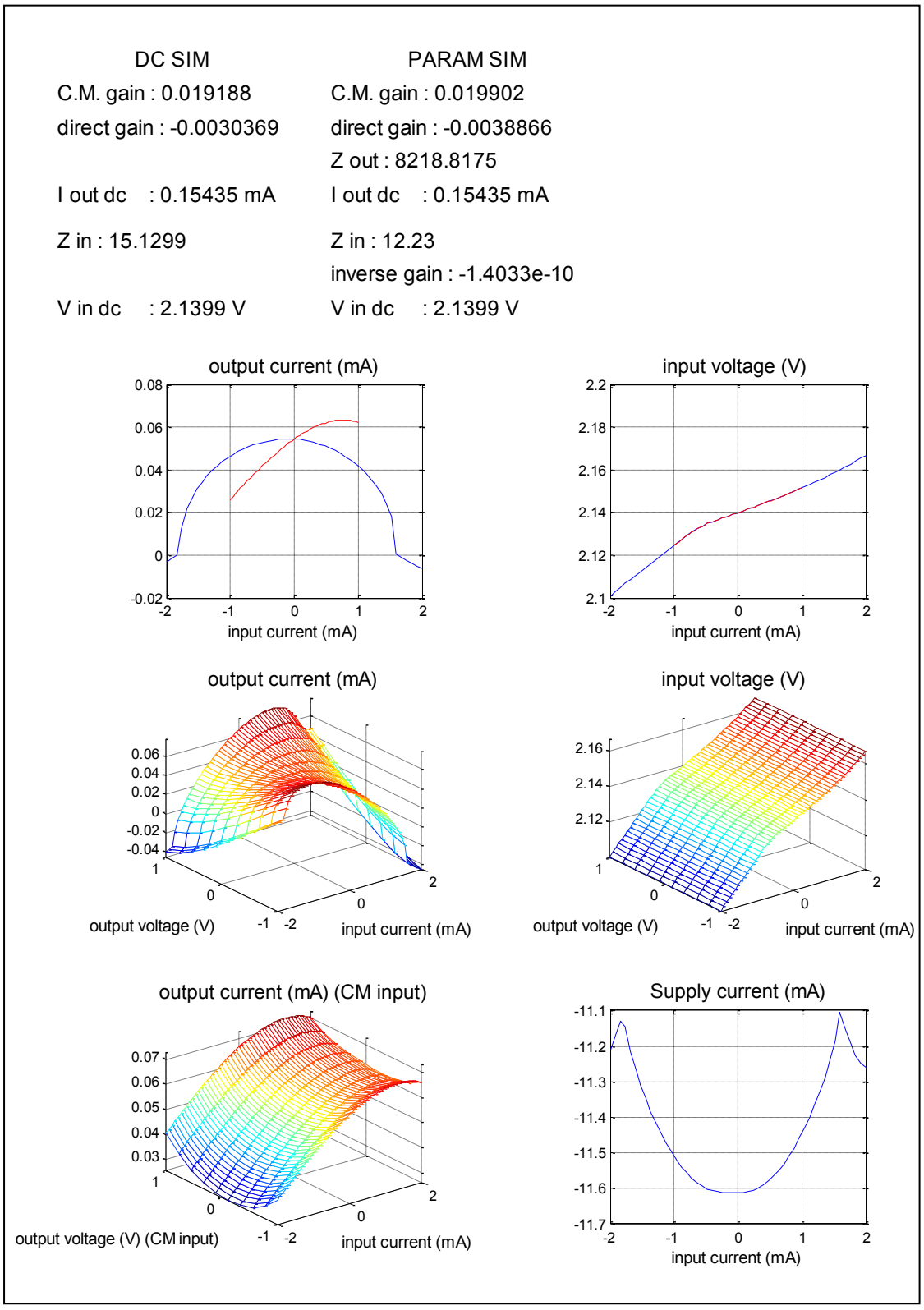


Figure 8.11 Current mirror DC in and DC out in common mode responses and extracted parameters

The AC and transient figures of merit for the *current mirror* amplifier:

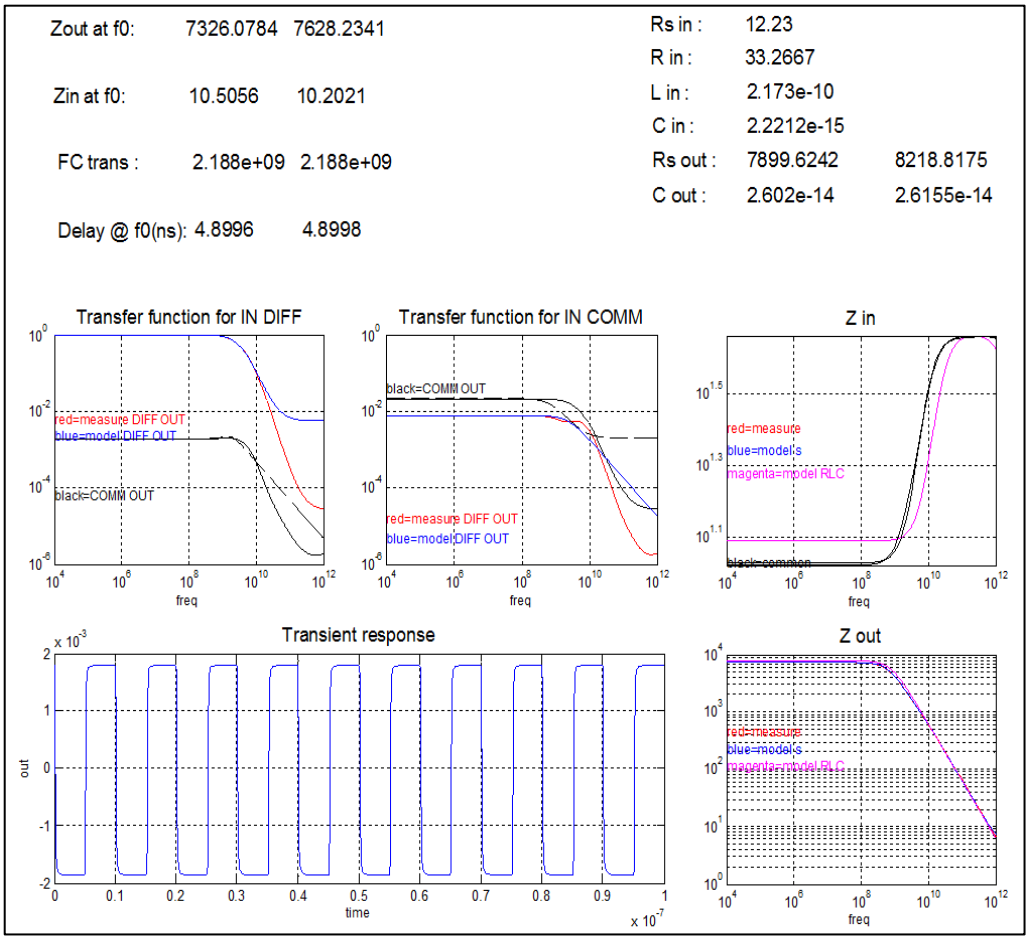


Figure 8.12 Current mirror AC and transient responses and extracted parameters

9. ANNEX C

We present here the HDL behavioral models extracted in the case of the design applications presented in Chapter 3, Section 3.7 and Chapter 5, Section 5.3.

9.1. Trans-impedance Amplifier HDL Models

The VHDL-AMS linear model used for simulations with Dolphin Integration SMASH in the case of the TIA Amplifier is presented:

```
----------------------------------------
-- Generated by SIMECT
-- Generated on: 27-Jun-2012 19:57:25
----------------------------------------

LIBRARY IEEE;
USE IEEE.electrical_systems.all;
use IEEE.math_real.all;
USE work.all;

ENTITY TIA1 IS

  PORT    (TERMINAL V,Iin,Vout: ELECTRICAL);

END ENTITY TIA1;

ARCHITECTURE behavioral OF TIA1 IS

  CONSTANT Out_1_Num_TF_dir1_1 : REAL_VECTOR := (-1.000000000000e+00, 8.392493702775e-
12, 1.606775416888e-22, 1.157481919566e-35, 0.0);
  CONSTANT Out_1_Den_TF_dir1_1 : REAL_VECTOR := (5.134064667246e-04, 1.196219114437e-13,
6.801749787007e-24, 4.351729896451e-36, 0.0);
  CONSTANT Out_1_Num_TF_inv1_1 : REAL_VECTOR := (1.000000000000e+00, 7.990013098239e-10,
2.291242057562e-21, 1.017049124732e-34, 0.0);
  CONSTANT Out_1_Den_TF_inv1_1 : REAL_VECTOR := (3.085910350244e-01, 2.736993035706e-11,
1.200976380483e-22, 5.456265273894e-35, 0.0);
  CONSTANT Out_1_Num_Zin1_1 : REAL_VECTOR := (1.000000000000e+00, 1.131801292676e-08,
3.458296002212e-20, 2.997901757356e-33, 0.0);
  CONSTANT Out_1_Den_Zin1_1 : REAL_VECTOR := (8.387698623110e-02, 1.992439018526e-11,
1.127393415452e-21, 8.861977716360e-34, 0.0);
  CONSTANT Out_1_Num_Zout1_1 : REAL_VECTOR := (1.000000000000e+00, 9.505476387527e-10,
1.967285406589e-21, 7.008572301083e-35, 0.0);
  CONSTANT Out_1_Den_Zout1_1 : REAL_VECTOR := (3.030594256065e-01, 2.412112809841e-11,
9.135374391957e-23, 3.971671256110e-35, 0.0);
  CONSTANT Out_1_OffIn11 : REAL := 1.654361e-24;
  CONSTANT Out_1_OffIn12 : REAL := 2.358771e+00;
  CONSTANT Out_1_OffOut11 : REAL := 2.560804e+00;
  CONSTANT Out_1_OffOut12 : REAL := 3.600000e-12;
  QUANTITY Vin_1 ACROSS Iin_1 THROUGH Iin TO ground;
  QUANTITY Vout_1 ACROSS Iout_1 THROUGH Vout TO ground;
  QUANTITY deltaIin: current;
  QUANTITY deltaIout_1: current;

BEGIN

  deltaIin == Iin_1 - Out_1_OffIn11;
  deltaIout_1 == Iout_1 - Out_1_OffOut12;
  Vout_1 == deltaIin'LTF(Out_1_Num_TF_dir1_1,Out_1_Den_TF_dir1_1)+
deltaIout_1'LTF(Out_1_Num_Zout1_1,Out_1_Den_Zout1_1)+ Out_1_OffOut11;
  Vin_1 == deltaIin'LTF(Out_1_Num_Zin1_1,Out_1_Den_Zin1_1)+
deltaIout_1'LTF(Out_1_Num_TF_inv1_1,Out_1_Den_TF_inv1_1)+ Out_1_OffIn12;

END ARCHITECTURE behavioral;
```

The linear VHDL-AMS model of the TIA amplifier used for simulations with the Cadence AMS:

```
---------------------------------------
-- Generated by SIMECT
-- Generated on: 27-Jun-2012 19:57:49
---------------------------------------

library ieee, std;
use ieee.std_logic_1164.all;
USE ieee.electrical_systems.all;
use ieee.math_real.all;
use work.all;

ENTITY TIA1 IS

  PORT    (TERMINAL V,Iin,Vout: ELECTRICAL);

END ENTITY TIA1;

ARCHITECTURE behavioral OF TIA1 IS

  CONSTANT Out_1_Num_TF_dir1_1 : REAL_VECTOR := (-1.24885750e+01, -9.03086957e-13, 0.0);
  CONSTANT Out_1_Den_TF_dir1_1 : REAL_VECTOR := (1.000000000e+00, 6.471404330e-13, 0.0);
  CONSTANT Out_1_Num_TF_dir1_2 : REAL_VECTOR := (1.248857506e+01, 1.137516996e-10, 0.0);
  CONSTANT Out_1_Den_TF_dir1_2 : REAL_VECTOR := (1.000000000e+00, 9.620859050e-11, 0.0);
  CONSTANT Out_1_Num_TF_dir1_3 : REAL_VECTOR := (1.248857506e+01, -2.19465074e-10, 0.0);
  CONSTANT Out_1_Den_TF_dir1_3 : REAL_VECTOR := (1.000000007e+00, 1.361407722e-10, 0.0);
  CONSTANT Out_1_Num_TF_inv1_1 : REAL_VECTOR := (1.479808661e+00, 6.673602356e-14, 0.0);
  CONSTANT Out_1_Den_TF_inv1_1 : REAL_VECTOR := (1.000000000e+00, 5.142341358e-13, 0.0);
  CONSTANT Out_1_Num_TF_inv1_2 : REAL_VECTOR := (1.479808661e+00, 4.191913439e-12, 0.0);
  CONSTANT Out_1_Den_TF_inv1_2 : REAL_VECTOR := (1.000000000e+00, 4.088900204e-12, 0.0);
  CONSTANT Out_1_Num_TF_inv1_3 : REAL_VECTOR := (1.479808661e+00, 1.178110409e-09, 0.0);
  CONSTANT Out_1_Den_TF_inv1_3 : REAL_VECTOR := (1.000000000e+00, 8.409007852e-11, 0.0);
  CONSTANT Out_1_Num_Zin1_1 : REAL_VECTOR := (2.284471411e+00, 2.039961476e-13, 0.0);
  CONSTANT Out_1_Den_Zin1_1 : REAL_VECTOR := (1.000000000e+00, 7.972545269e-13, 0.0);
  CONSTANT Out_1_Num_Zin1_2 : REAL_VECTOR := (2.284471411e+00, 6.778194002e-12, 0.0);
  CONSTANT Out_1_Den_Zin1_2 : REAL_VECTOR := (1.000000000e+00, 9.080751600e-11, 0.0);
  CONSTANT Out_1_Num_Zin1_3 : REAL_VECTOR := (2.284471411e+00, 2.584869477e-08, 0.0);
  CONSTANT Out_1_Den_Zin1_3 : REAL_VECTOR := (1.000000000e+00, 1.459382204e-10, 0.0);
  CONSTANT Out_1_Num_Zout1_1 : REAL_VECTOR := (1.488757866e+00, 5.398367348e-14, 0.0);
  CONSTANT Out_1_Den_Zout1_1 : REAL_VECTOR := (1.000000000e+00, 5.004779965e-13, 0.0);
  CONSTANT Out_1_Num_Zout1_2 : REAL_VECTOR := (1.488757869e+00, 3.033821553e-12, 0.0);
  CONSTANT Out_1_Den_Zout1_2 : REAL_VECTOR := (1.000000000e+00, 3.462348110e-12, 0.0);
  CONSTANT Out_1_Num_Zout1_3 : REAL_VECTOR := (1.488757862e+00, 1.412047465e-09, 0.0);
  CONSTANT Out_1_Den_Zout1_3 : REAL_VECTOR := (1.000000000e+00, 7.562924746e-11, 0.0);
  CONSTANT Out_1_OffIn11 : REAL := 1.654361e-24;
  CONSTANT Out_1_OffIn12 : REAL := 2.358771e+00;
  CONSTANT Out_1_OffOut11 : REAL := 2.560804e+00;
  CONSTANT Out_1_OffOut12 : REAL := 3.600000e-12;
  QUANTITY Vin1 ACROSS Iin1 THROUGH Iin TO ground;
  QUANTITY Vout1_1 ACROSS Iout1_1 THROUGH Vout TO ground;
  QUANTITY deltaIin1: current;
  QUANTITY deltaIout1_1: current;

BEGIN

  deltaIin1 == Iin1 - Out_1_OffIn11;
  deltaIout1_1 == Iout1_1 - Out_1_OffOut12;
  Vout1_1 ==
 deltaIin1'LTF((Out_1_Num_TF_dir1_1(0),Out_1_Num_TF_dir1_1(1)),
(Out_1_Den_TF_dir1_1(0),Out_1_Den_TF_dir1_1(1)))'LTF((Out_1_Num_TF_dir1_2(0),Out_1_Num_T
F_dir1_2(1)),
(Out_1_Den_TF_dir1_2(0),Out_1_Den_TF_dir1_2(1)))'LTF((Out_1_Num_TF_dir1_3(0),Out_1_Num_T
F_dir1_3(1)), (Out_1_Den_TF_dir1_3(0),Out_1_Den_TF_dir1_3(1)))
 + deltaIout1_1'LTF((Out_1_Num_Zout1_1(0),Out_1_Num_Zout1_1(1)),
(Out_1_Den_Zout1_1(0),Out_1_Den_Zout1_1(1)))'LTF((Out_1_Num_Zout1_2(0),Out_1_Num_Zout1_2
(1)),
(Out_1_Den_Zout1_2(0),Out_1_Den_Zout1_2(1)))'LTF((Out_1_Num_Zout1_3(0),Out_1_Num_Zout1_3
(1)), (Out_1_Den_Zout1_3(0),Out_1_Den_Zout1_3(1)))
 + Out_1_OffOut11;
  Vin1 == deltaIin1'LTF((Out_1_Num_Zin1_1(0),Out_1_Num_Zin1_1(1)),
(Out_1_Den_Zin1_1(0),Out_1_Den_Zin1_1(1)))'LTF((Out_1_Num_Zin1_2(0),Out_1_Num_Zin1_2(1))
,
(Out_1_Den_Zin1_2(0),Out_1_Den_Zin1_2(1)))'LTF((Out_1_Num_Zin1_3(0),Out_1_Num_Zin1_3(1))
```

```
, (Out_1_Den_Zin1_3(0),Out_1_Den_Zin1_3(1)))+
deltaIout1_1'LTF((Out_1_Num_TF_inv1_1(0),Out_1_Num_TF_inv1_1(1)),
(Out_1_Den_TF_inv1_1(0),Out_1_Den_TF_inv1_1(1)))'LTF((Out_1_Num_TF_inv1_2(0),Out_1_Num_T
F_inv1_2(1)),
(Out_1_Den_TF_inv1_2(0),Out_1_Den_TF_inv1_2(1)))'LTF((Out_1_Num_TF_inv1_3(0),Out_1_Num_T
F_inv1_3(1)), (Out_1_Den_TF_inv1_3(0),Out_1_Den_TF_inv1_3(1)))+ Out_1_OffIn12;

END ARCHITECTURE behavioral;
```

The Verilog-A model used for simulations with Cadence AMS for the TIA Amplifier is also presented:

```
//-------------------------------------
// Generated by SIMECT
// Generated on: 27-Jun-2012 19:58:07
//-------------------------------------

`include "constants.vams"
`include "disciplines.vams"

module TIA1 (V,Iin,Vout,gnd);
input V;
electrical V;
input Iin;
electrical Iin;
output Vout;
electrical Vout;
input gnd;
electrical gnd;

parameter real Out_1_Num_TF_dir1_1_1  = -1.000000000000e+00 ;
parameter real Out_1_Num_TF_dir1_1_2  = 8.392493702775e-12 ;
parameter real Out_1_Num_TF_dir1_1_3  = 1.606775416888e-22 ;
parameter real Out_1_Num_TF_dir1_1_4  = 1.157481919566e-35 ;
parameter real Out_1_Den_TF_dir1_1_1  = 5.134064667246e-04 ;
parameter real Out_1_Den_TF_dir1_1_2  = 1.196219114437e-13 ;
parameter real Out_1_Den_TF_dir1_1_3  = 6.801749787007e-24 ;
parameter real Out_1_Den_TF_dir1_1_4  = 4.351729896451e-36 ;
parameter real Out_1_Num_TF_inv1_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Num_TF_inv1_1_2  = 7.990013098239e-10 ;
parameter real Out_1_Num_TF_inv1_1_3  = 2.291242057562e-21 ;
parameter real Out_1_Num_TF_inv1_1_4  = 1.017049124732e-34 ;
parameter real Out_1_Den_TF_inv1_1_1  = 3.085910350244e-01 ;
parameter real Out_1_Den_TF_inv1_1_2  = 2.736993035706e-11 ;
parameter real Out_1_Den_TF_inv1_1_3  = 1.200976380483e-22 ;
parameter real Out_1_Den_TF_inv1_1_4  = 5.456265273894e-35 ;
parameter real Out_1_Num_Zin1_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Num_Zin1_1_2  = 1.131801292676e-08 ;
parameter real Out_1_Num_Zin1_1_3  = 3.458296002212e-20 ;
parameter real Out_1_Num_Zin1_1_4  = 2.997901757356e-33 ;
parameter real Out_1_Den_Zin1_1_1  = 8.387698623110e-02 ;
parameter real Out_1_Den_Zin1_1_2  = 1.992439018526e-11 ;
parameter real Out_1_Den_Zin1_1_3  = 1.127393415452e-21 ;
parameter real Out_1_Den_Zin1_1_4  = 8.861977716360e-34 ;
parameter real Out_1_Num_Zout1_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Num_Zout1_1_2  = 9.505476387527e-10 ;
parameter real Out_1_Num_Zout1_1_3  = 1.967285406589e-21 ;
parameter real Out_1_Num_Zout1_1_4  = 7.008572301083e-35 ;
parameter real Out_1_Den_Zout1_1_1  = 3.030594256065e-01 ;
parameter real Out_1_Den_Zout1_1_2  = 2.412112809841e-11 ;
parameter real Out_1_Den_Zout1_1_3  = 9.135374391957e-23 ;
parameter real Out_1_Den_Zout1_1_4  = 3.971671256110e-35 ;
parameter real Out_1_OffIn11 = 1.654361e-24;
parameter real Out_1_OffIn12 = 2.358771e+00;
parameter real Out_1_OffOut11 = 2.560804e+00;
parameter real Out_1_OffOut12 = 3.600000e-12;

real Vin1 ;
real Iin1 ;
real Vout1_1 ;
real Iout1_1 ;
real deltaIin1 ;
real deltaIout1_1 ;

analog
```

```
begin

  Iin1 = I(Iin,gnd);
  V(Iin,gnd) <+ Vin1 ;
  Iout1_1 = I(Vout,gnd);
  V(Vout,gnd) <+ Vout1_1 ;

  deltaIin1 = Iin1 - Out_1_OffIn11;
  deltaIout1_1 = Iout1_1 - Out_1_OffOut12;

  Vout1_1 =
+laplace_nd(deltaIin1,{Out_1_Num_TF_dir1_1_1,Out_1_Num_TF_dir1_1_2,Out_1_Num_TF_dir1_1_3
,Out_1_Num_TF_dir1_1_4},
{Out_1_Den_TF_dir1_1_1,Out_1_Den_TF_dir1_1_2,Out_1_Den_TF_dir1_1_3,Out_1_Den_TF_dir1_1_4
})
+laplace_nd(deltaIout1_1,{Out_1_Num_Zout1_1_1,Out_1_Num_Zout1_1_2,Out_1_Num_Zout1_1_3,Ou
t_1_Num_Zout1_1_4},
{Out_1_Den_Zout1_1_1,Out_1_Den_Zout1_1_2,Out_1_Den_Zout1_1_3,Out_1_Den_Zout1_1_4}) +
Out_1_OffOut11;
  Vin1 =
+laplace_nd(deltaIin1,{Out_1_Num_Zin1_1_1,Out_1_Num_Zin1_1_2,Out_1_Num_Zin1_1_3,Out_1_Nu
m_Zin1_1_4},
{Out_1_Den_Zin1_1_1,Out_1_Den_Zin1_1_2,Out_1_Den_Zin1_1_3,Out_1_Den_Zin1_1_4})
+laplace_nd(deltaIout1_1,{Out_1_Num_TF_inv1_1_1,Out_1_Num_TF_inv1_1_2,Out_1_Num_TF_inv1_
1_3,Out_1_Num_TF_inv1_1_4},
{Out_1_Den_TF_inv1_1_1,Out_1_Den_TF_inv1_1_2,Out_1_Den_TF_inv1_1_3,Out_1_Den_TF_inv1_1_4
})+ Out_1_OffIn12;

end
endmodule
```

9.2. Trans-conductance Amplifier HDL Models

The VHDL-AMS linear model used for simulations with Dolphin Integration SMASH in the case of the Trans-conductance (*Gm*) amplifier is presented:

```
----------------------------------------
-- Generated by SIMECT
-- Generated on: 28-Jun-2012 12:09:57
----------------------------------------

LIBRARY IEEE;
USE IEEE.electrical_systems.all;
use IEEE.math_real.all;
USE work.all;

ENTITY gm1_reel IS

  PORT    (TERMINAL vin,ioutp,ioutm: ELECTRICAL);

END ENTITY gm1_reel;

ARCHITECTURE behavioral OF gm1_reel IS

  CONSTANT Out_1_Num_TF_dir1_1 : REAL_VECTOR := (1.040545599440e-07, 4.105376303282e-14,
-2.286276626831e-25, -1.401327065230e-36, -3.620748160317e-50, 0.0);
  CONSTANT Out_1_Den_TF_dir1_1 : REAL_VECTOR := (1.000000000000e+00, 4.512610753079e-10,
2.994502966504e-21, 2.016503319230e-32, 2.683037797826e-45, 1.290934814129e-70, 0.0);
  CONSTANT Out_1_Num_TF_dir2_1 : REAL_VECTOR := (-6.068796075955e-06, -1.367472315186e-
13, 1.553175477475e-24, -1.655783188424e-36, 0.0);
  CONSTANT Out_1_Den_TF_dir2_1 : REAL_VECTOR := (1.000000000000e+00, 3.022827544914e-10,
1.993397597661e-21, 1.128781430884e-32, 9.725677466935e-46, 8.031936516939e-59, 0.0);
  CONSTANT Out_1_Num_TF_inv1_1 : REAL_VECTOR := (4.783275926758e-07, 2.066574976766e-16,
-2.831980868699e-27, 4.296594997262e-41, 0.0);
  CONSTANT Out_1_Den_TF_inv1_1 : REAL_VECTOR := (1.000000000000e+00, 7.712399130202e-12,
4.384399363657e-23, 3.373839214471e-36, 5.724405236602e-53, 5.721335486422e-74, 0.0);
  CONSTANT Out_1_Num_TF_inv2_1 : REAL_VECTOR := (8.307483096607e-07, -4.922790138277e-
16, -6.217399445025e-27, -1.608083462546e-38, -3.721809031789e-52, 0.0);
  CONSTANT Out_1_Den_TF_inv2_1 : REAL_VECTOR := (1.000000000000e+00, 1.083575721652e-11,
5.705530870031e-23, 1.057492771540e-34, 1.233784603140e-47, 4.793530575889e-61, 0.0);
  CONSTANT Out_1_Num_Zin1_1 : REAL_VECTOR := (1.000000000000e+00, 7.810574187728e-12,
4.125650035554e-23, 3.391711522248e-36, 2.747394574238e-51, 2.253759458416e-64, 0.0);
```

```
  CONSTANT Out_1_Den_Zin1_1 : REAL_VECTOR := (-4.410984629251e-06, 2.317308564808e-14,
4.512329368446e-25, 2.431160318837e-37, 0.0);
  CONSTANT Out_1_Num_Zout1_1 : REAL_VECTOR := (1.000000000000e+00, 5.473301826773e-10,
1.761999206136e-20, 4.816753479040e-32, 4.032447312237e-45, 7.510980891643e-59, 0.0);
  CONSTANT Out_1_Den_Zout1_1 : REAL_VECTOR := (2.866433707175e-07, 1.877906670098e-13,
8.536231827455e-23, 2.679713232291e-33, 7.196959708127e-45, 2.876682040870e-58, 0.0);
  CONSTANT Out_1_Num_Zout2_1 : REAL_VECTOR := (1.000000000000e+00, 2.846309467993e-11,
1.594800368005e-24, 2.712093705546e-39, 1.513345639249e-52, 0.0);
  CONSTANT Out_1_Den_Zout2_1 : REAL_VECTOR := (1.732989289772e-05, 1.664587611997e-13,
4.466411980985e-24, 0.0);
  CONSTANT Out_1_Num_Zout_diff1_1 : REAL_VECTOR := (1.000000000000e+00, 5.552925942774e-
10, 1.261516582582e-20, 5.008392471933e-34, 1.062881320756e-47, 8.722938486814e-73,
0.0);
  CONSTANT Out_1_Den_Zout_diff1_1 : REAL_VECTOR := (1.070067838657e-07, -
1.656837180522e-14, -4.148581567544e-25, 5.084033420726e-36, 0.0);
  CONSTANT Out_1_Num_Zout_diff2_1 : REAL_VECTOR := (1.000000000000e+00, 8.169850349575e-
11, 7.606077589097e-22, 2.991914079413e-35, 6.890152049288e-49, 2.509833827506e-74,
0.0);
  CONSTANT Out_1_Den_Zout_diff2_1 : REAL_VECTOR := (3.313673240436e-06, -
3.550788887631e-15, -3.824368416127e-27, 3.050805663438e-37, 0.0);
  CONSTANT Out_1_OffIn11 : REAL := 2.500000e+00;
  CONSTANT Out_1_OffIn12 : REAL := -4.849873e-06;
  CONSTANT Out_1_OffOut11 : REAL := 4.996447e-04;
  CONSTANT Out_1_OffOut12 : REAL := 2.340000e+00;
  CONSTANT Out_1_OffOut21 : REAL := -4.955704e-04;
  CONSTANT Out_1_OffOut22 : REAL := 2.340000e+00;
  QUANTITY Vin1 ACROSS Iin1 THROUGH vin TO ground;
  QUANTITY Vout1_1 ACROSS Iout1_1 THROUGH ioutp TO ground;
  QUANTITY Vout2_1 ACROSS Iout2_1 THROUGH ioutm TO ground;
  QUANTITY deltaVin1: voltage;
  QUANTITY deltaVout1_1, deltaVout2_1: voltage;

BEGIN

  deltaVin1 == Vin1 - Out_1_OffIn11;
  deltaVout1_1 == Vout1_1 - Out_1_OffOut12;
  deltaVout2_1 == Vout2_1 - Out_1_OffOut22;
  Iout1_1 == deltaVin1'LTF(Out_1_Num_TF_dir1_1,Out_1_Den_TF_dir1_1)+
deltaVout1_1'LTF(Out_1_Den_Zout1_1,Out_1_Num_Zout1_1)+
deltaVout2_1'LTF(Out_1_Den_Zout_diff2_1,Out_1_Num_Zout_diff2_1)+ Out_1_OffOut11;
  Iout2_1 == deltaVin1'LTF(Out_1_Num_TF_dir2_1,Out_1_Den_TF_dir2_1)+
deltaVout2_1'LTF(Out_1_Den_Zout2_1,Out_1_Num_Zout2_1)+
deltaVout1_1'LTF(Out_1_Den_Zout_diff1_1,Out_1_Num_Zout_diff1_1)+ Out_1_OffOut21;
  Iin1 == deltaVin1'LTF(Out_1_Den_Zin1_1,Out_1_Num_Zin1_1)+
deltaVout1_1'LTF(Out_1_Num_TF_inv1_1,Out_1_Den_TF_inv1_1)+
deltaVout2_1'LTF(Out_1_Num_TF_inv2_1,Out_1_Den_TF_inv2_1)+ Out_1_OffIn12;

END ARCHITECTURE behavioral;
```

The VHDL-AMS model for Cadence AMS in the case of the Trans-conductance amplifier is also presented. The transfer functions decomposition and the vectors splitting techniques are used:

```
----------------------------------------
-- Generated by SIMECT
-- Generated on: 28-Jun-2012 12:10:22
----------------------------------------

library ieee, std;
use ieee.std_logic_1164.all;
USE ieee.electrical_systems.all;
use ieee.math_real.all;
use work.all;

ENTITY gm1_reel IS

  PORT    (TERMINAL vin,ioutp,ioutm: ELECTRICAL);

END ENTITY gm1_reel;

ARCHITECTURE behavioral OF gm1_reel IS

  CONSTANT Out_1_Num_TF_dir1_1 : REAL_VECTOR := (1.79603699e-02, 4.660421294e-16, 0.0);
  CONSTANT Out_1_Den_TF_dir1_1 : REAL_VECTOR := (1.00000000e+00, 4.811467118e-26, 0.0);
```

```
  CONSTANT Out_1_Num_TF_dir1_2 : REAL_VECTOR := (1.79603699e-02, 6.589595318e-14, 0.0);
  CONSTANT Out_1_Den_TF_dir1_2 : REAL_VECTOR := (1.00000000e+00, 1.357339443e-13, 0.0);
  CONSTANT Out_1_Num_TF_dir1_3 : REAL_VECTOR := (1.79603699e-02, -1.663800049e-13, 0.0);
  CONSTANT Out_1_Den_TF_dir1_3 : REAL_VECTOR := (1.00000000e+00, 4.446282085e-10, 0.0);
  CONSTANT Out_1_Num_TF_dir1_4 : REAL_VECTOR := (1.79603699e-02, 7.086197679e-09,
0.00000000e+00, 0.0);
  CONSTANT Out_1_Den_TF_dir1_4 : REAL_VECTOR := (1.00000000e+00, 6.497142650e-12,
4.44571194e-23, 0.0);
  CONSTANT Out_1_Num_TF_dir2_1 : REAL_VECTOR := (-1.82403925e-02, 2.172333861e-14, 0.0);
  CONSTANT Out_1_Den_TF_dir2_1 : REAL_VECTOR := (1.00000000e+00, 2.956698667e-10, 0.0);
  CONSTANT Out_1_Num_TF_dir2_2 : REAL_VECTOR := (1.82403925e-02, 4.110296329e-10, -
4.17871464e-21, 0.0);
  CONSTANT Out_1_Den_TF_dir2_2 : REAL_VECTOR := (1.00000000e+00, 8.621365374e-14,
7.22555096e-27, 0.0);
  CONSTANT Out_1_Num_TF_dir2_3 : REAL_VECTOR := (1.82403925e-02, -1.824039259e-02, -
0.000000000e+00, 0.0);
  CONSTANT Out_1_Den_TF_dir2_3 : REAL_VECTOR := (1.00000000e+00, 6.526674070e-12,
3.759605126901e-23, 0.0);
  CONSTANT Out_1_Num_TF_inv1_1 : REAL_VECTOR := (2.62985230e-02, -3.994359001e-16, 0.0);
  CONSTANT Out_1_Den_TF_inv1_1 : REAL_VECTOR := (1.00000000e+00, 9.99522624e-22, 0.0);
  CONSTANT Out_1_Num_TF_inv1_2 : REAL_VECTOR := (2.62985230e-02, -3.492416041e-13, 0.0);
  CONSTANT Out_1_Den_TF_inv1_2 : REAL_VECTOR := (1.00000000e+00, 1.696977926e-17, 0.0);
  CONSTANT Out_1_Num_TF_inv1_3 : REAL_VECTOR := (2.62985230e-02, 1.171170136e-11, 0.0);
  CONSTANT Out_1_Den_TF_inv1_3 : REAL_VECTOR := (1.00000000e+00, 7.799346999e-14, 0.0);
  CONSTANT Out_1_Num_TF_inv1_4 : REAL_VECTOR := (2.62985230e-02, -2.629852303e-02, -
0.00000000e+00, 0.0);
  CONSTANT Out_1_Den_TF_inv1_4 : REAL_VECTOR := (1.00000000e+00, 7.634388689e-12,
4.32484302e-23, 0.0);
  CONSTANT Out_1_Num_TF_inv2_1 : REAL_VECTOR := (9.40061980e-03, 2.195503189e-16, 0.0);
  CONSTANT Out_1_Den_TF_inv2_1 : REAL_VECTOR := (1.00000000e+00, 3.059334976e-12, 0.0);
  CONSTANT Out_1_Num_TF_inv2_2 : REAL_VECTOR := (9.40061980e-03, 1.165677982e-13,
2.98064848e-25, 0.0);
  CONSTANT Out_1_Den_TF_inv2_2 : REAL_VECTOR := (1.00000000e+00, 1.220245317e-13,
4.84707532e-27, 0.0);
  CONSTANT Out_1_Num_TF_inv2_3 : REAL_VECTOR := (9.40061980e-03, -5.687340781e-12, -
0.00000000e+00, 0.0);
  CONSTANT Out_1_Den_TF_inv2_3 : REAL_VECTOR := (1.00000000e+00, 7.654397708e-12,
3.23257567e-23, 0.0);
  CONSTANT Out_1_Num_Zin1_1 : REAL_VECTOR := (1.00000000e+00, 8.350224553e-14, 0.0);
  CONSTANT Out_1_Den_Zin1_1 : REAL_VECTOR := (-1.64000503e-02, -9.095105725e-15, 0.0);
  CONSTANT Out_1_Num_Zin1_2 : REAL_VECTOR := (1.00000000e+00, 1.613585846e-18,
6.64605076e-29, 0.0);
  CONSTANT Out_1_Den_Zin1_2 : REAL_VECTOR := (1.64000507e-02, -8.616667741e-11, -
1.62989880e-21, 0.0);
  CONSTANT Out_1_Num_Zin1_3 : REAL_VECTOR := (1.00000000e+00, 7.727070328e-12,
4.06111915e-23, 0.0);
  CONSTANT Out_1_Den_Zin1_3 : REAL_VECTOR := (1.64000507e-02, -1.640005037e-02, -
0.00000000e+00, 0.0);
  CONSTANT Out_1_Num_Zout1_1 : REAL_VECTOR := (1.00000000e+00, 2.769666405e-14, 0.0);
  CONSTANT Out_1_Den_Zout1_1 : REAL_VECTOR := (4.91437598e-02, 1.994413296e-15, 0.0);
  CONSTANT Out_1_Num_Zout1_2 : REAL_VECTOR := (1.00000000e+00, 5.812163206e-14, 0.0);
  CONSTANT Out_1_Den_Zout1_2 : REAL_VECTOR := (4.91437598e-02, 1.434275922e-13, 0.0);
  CONSTANT Out_1_Num_Zout1_3 : REAL_VECTOR := (1.00000000e+00, 2.919059613e-12, 0.0);
  CONSTANT Out_1_Den_Zout1_3 : REAL_VECTOR := (4.914375989e-02, 1.510159992e-12, 0.0);
  CONSTANT Out_1_Num_Zout1_4 : REAL_VECTOR := (1.000000000e+00, 3.114729915e-11, 0.0);
  CONSTANT Out_1_Den_Zout1_4 : REAL_VECTOR := (4.914375989e-02, 2.069770443e-11, 0.0);
  CONSTANT Out_1_Num_Zout1_5 : REAL_VECTOR := (1.000000000e+00, 5.131780056e-10, 0.0);
  CONSTANT Out_1_Den_Zout1_5 : REAL_VECTOR := (4.914375989e-02, 3.217354026e-08, 0.0);
  CONSTANT Out_1_Num_Zout2_1 : REAL_VECTOR := (1.000000000e+00, 5.613113981e-14, 0.0);
  CONSTANT Out_1_Den_Zout2_1 : REAL_VECTOR := (2.587807454e-02, 6.963095914e-13, 0.0);
  CONSTANT Out_1_Num_Zout2_2 : REAL_VECTOR := (1.000000000e+00, 2.840695684e-11, 0.0);
  CONSTANT Out_1_Den_Zout2_2 : REAL_VECTOR := (2.587807454e-02, 2.478702716e-10, 0.0);
  CONSTANT Out_1_Num_Zout2_3 : REAL_VECTOR := (1.000000000e+00, 6.696115704e-18,
9.49094719e-29, 0.0);
  CONSTANT Out_1_Den_Zout2_3 : REAL_VECTOR := (2.58780745e-02, -2.587807454e-02, -
0.00000000e+00, 0.0);
  CONSTANT Out_1_OffIn11 : REAL := 2.500000e+00;
  CONSTANT Out_1_OffIn12 : REAL := -4.849873e-06;
  CONSTANT Out_1_OffOut11 : REAL := 4.996447e-04;
  CONSTANT Out_1_OffOut12 : REAL := 2.340000e+00;
  CONSTANT Out_1_OffOut21 : REAL := -4.955704e-04;
  CONSTANT Out_1_OffOut22 : REAL := 2.340000e+00;
  QUANTITY Vin1 ACROSS Iin1 THROUGH vin TO ground;
  QUANTITY Vout1_1 ACROSS Iout1_1 THROUGH ioutp TO ground;
  QUANTITY Vout2_1 ACROSS Iout2_1 THROUGH ioutm TO ground;
  QUANTITY deltaVin1: voltage;
  QUANTITY deltaVout1_1: voltage;
```

```
  QUANTITY deltaVout2_1: voltage;

BEGIN

  deltaVin1 == Vin1 - Out_1_OffIn11;
  deltaVout1_1 == Vout1_1 - Out_1_OffOut12;
  deltaVout2_1 == Vout2_1 - Out_1_OffOut22;
  Iout1_1 ==
 deltaVin1'LTF((Out_1_Num_TF_dir1_1(0),Out_1_Num_TF_dir1_1(1)),
(Out_1_Den_TF_dir1_1(0),Out_1_Den_TF_dir1_1(1)))'LTF((Out_1_Num_TF_dir1_2(0),Out_1_Num_T
F_dir1_2(1)),
(Out_1_Den_TF_dir1_2(0),Out_1_Den_TF_dir1_2(1)))'LTF((Out_1_Num_TF_dir1_3(0),Out_1_Num_T
F_dir1_3(1)),
(Out_1_Den_TF_dir1_3(0),Out_1_Den_TF_dir1_3(1)))'LTF((Out_1_Num_TF_dir1_4(0),Out_1_Num_T
F_dir1_4(1),Out_1_Num_TF_dir1_4(2)),
(Out_1_Den_TF_dir1_4(0),Out_1_Den_TF_dir1_4(1),Out_1_Den_TF_dir1_4(2)))
 + deltaVout1_1'LTF((Out_1_Den_Zout1_1(0),Out_1_Den_Zout1_1(1)),
(Out_1_Num_Zout1_1(0),Out_1_Num_Zout1_1(1)))'LTF((Out_1_Den_Zout1_2(0),Out_1_Den_Zout1_2
(1)),
(Out_1_Num_Zout1_2(0),Out_1_Num_Zout1_2(1)))'LTF((Out_1_Den_Zout1_3(0),Out_1_Den_Zout1_3
(1)),
(Out_1_Num_Zout1_3(0),Out_1_Num_Zout1_3(1)))'LTF((Out_1_Den_Zout1_4(0),Out_1_Den_Zout1_4
(1)),
(Out_1_Num_Zout1_4(0),Out_1_Num_Zout1_4(1)))'LTF((Out_1_Den_Zout1_5(0),Out_1_Den_Zout1_5
(1)), (Out_1_Num_Zout1_5(0),Out_1_Num_Zout1_5(1)))
 + Out_1_OffOut11;
  Iout2_1 ==
 deltaVin1'LTF((Out_1_Num_TF_dir2_1(0),Out_1_Num_TF_dir2_1(1)),
(Out_1_Den_TF_dir2_1(0),Out_1_Den_TF_dir2_1(1)))'LTF((Out_1_Num_TF_dir2_2(0),Out_1_Num_T
F_dir2_2(1),Out_1_Num_TF_dir2_2(2)),
(Out_1_Den_TF_dir2_2(0),Out_1_Den_TF_dir2_2(1),Out_1_Den_TF_dir2_2(2)))'LTF((Out_1_Num_T
F_dir2_3(0),Out_1_Num_TF_dir2_3(1),Out_1_Num_TF_dir2_3(2)),
(Out_1_Den_TF_dir2_3(0),Out_1_Den_TF_dir2_3(1),Out_1_Den_TF_dir2_3(2)))
 + deltaVout2_1'LTF((Out_1_Den_Zout2_1(0),Out_1_Den_Zout2_1(1)),
(Out_1_Num_Zout2_1(0),Out_1_Num_Zout2_1(1)))'LTF((Out_1_Den_Zout2_2(0),Out_1_Den_Zout2_2
(1)),
(Out_1_Num_Zout2_2(0),Out_1_Num_Zout2_2(1)))'LTF((Out_1_Den_Zout2_3(0),Out_1_Den_Zout2_3
(1),Out_1_Den_Zout2_3(2)),
(Out_1_Num_Zout2_3(0),Out_1_Num_Zout2_3(1),Out_1_Num_Zout2_3(2)))
 + Out_1_OffOut21;
  Iin1 == deltaVin1'LTF((Out_1_Den_Zin1_1(0),Out_1_Den_Zin1_1(1)),
(Out_1_Num_Zin1_1(0),Out_1_Num_Zin1_1(1)))'LTF((Out_1_Den_Zin1_2(0),Out_1_Den_Zin1_2(1),
Out_1_Den_Zin1_2(2)),
(Out_1_Num_Zin1_2(0),Out_1_Num_Zin1_2(1),Out_1_Num_Zin1_2(2)))'LTF((Out_1_Den_Zin1_3(0),
Out_1_Den_Zin1_3(1),Out_1_Den_Zin1_3(2)),
(Out_1_Num_Zin1_3(0),Out_1_Num_Zin1_3(1),Out_1_Num_Zin1_3(2)))+
deltaVout1_1'LTF((Out_1_Num_TF_inv1_1(0),Out_1_Num_TF_inv1_1(1)),
(Out_1_Den_TF_inv1_1(0),Out_1_Den_TF_inv1_1(1)))'LTF((Out_1_Num_TF_inv1_2(0),Out_1_Num_T
F_inv1_2(1)),
(Out_1_Den_TF_inv1_2(0),Out_1_Den_TF_inv1_2(1)))'LTF((Out_1_Num_TF_inv1_3(0),Out_1_Num_T
F_inv1_3(1)),
(Out_1_Den_TF_inv1_3(0),Out_1_Den_TF_inv1_3(1)))'LTF((Out_1_Num_TF_inv1_4(0),Out_1_Num_T
F_inv1_4(1),Out_1_Num_TF_inv1_4(2)),
(Out_1_Den_TF_inv1_4(0),Out_1_Den_TF_inv1_4(1),Out_1_Den_TF_inv1_4(2)))+
deltaVout2_1'LTF((Out_1_Num_TF_inv2_1(0),Out_1_Num_TF_inv2_1(1)),
(Out_1_Den_TF_inv2_1(0),Out_1_Den_TF_inv2_1(1)))'LTF((Out_1_Num_TF_inv2_2(0),Out_1_Num_T
F_inv2_2(1),Out_1_Num_TF_inv2_2(2)),
(Out_1_Den_TF_inv2_2(0),Out_1_Den_TF_inv2_2(1),Out_1_Den_TF_inv2_2(2)))'LTF((Out_1_Num_T
F_inv2_3(0),Out_1_Num_TF_inv2_3(1),Out_1_Num_TF_inv2_3(2)),
(Out_1_Den_TF_inv2_3(0),Out_1_Den_TF_inv2_3(1),Out_1_Den_TF_inv2_3(2)))+ Out_1_OffIn12;

END ARCHITECTURE behavioral;
```

The Verilog-A model used for simulations with Cadence AMS simulator for the Trans-Conductance amplifier is also presented:

```
//-------------------------------------
// Generated by SIMECT
// Generated on: 28-Jun-2012 12:39:01
//-------------------------------------

`include "constants.vams"
`include "disciplines.vams"

module gm1_reel (vin,ioutp,ioutm);
```

```
input vin;
electrical vin;
output ioutp;
electrical ioutp;
output ioutm;
electrical ioutm;

parameter real Out_1_Num_TF_dir1_1_1  = 1.040545599440e-07 ;
parameter real Out_1_Num_TF_dir1_1_2  = 4.105376303282e-14 ;
parameter real Out_1_Num_TF_dir1_1_3  = -2.286276626831e-25 ;
parameter real Out_1_Num_TF_dir1_1_4  = -1.401327065230e-36 ;
parameter real Out_1_Num_TF_dir1_1_5  = -3.620748160317e-50 ;
parameter real Out_1_Den_TF_dir1_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Den_TF_dir1_1_2  = 4.512610753079e-10 ;
parameter real Out_1_Den_TF_dir1_1_3  = 2.994502966504e-21 ;
parameter real Out_1_Den_TF_dir1_1_4  = 2.016503319230e-32 ;
parameter real Out_1_Den_TF_dir1_1_5  = 2.683037797826e-45 ;
parameter real Out_1_Den_TF_dir1_1_6  = 1.290934814129e-70 ;
parameter real Out_1_Num_TF_dir2_1_1  = -6.068796075955e-06 ;
parameter real Out_1_Num_TF_dir2_1_2  = -1.367472315186e-13 ;
parameter real Out_1_Num_TF_dir2_1_3  = 1.553175477475e-24 ;
parameter real Out_1_Num_TF_dir2_1_4  = -1.655783188424e-36 ;
parameter real Out_1_Den_TF_dir2_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Den_TF_dir2_1_2  = 3.022827544914e-10 ;
parameter real Out_1_Den_TF_dir2_1_3  = 1.993397597661e-21 ;
parameter real Out_1_Den_TF_dir2_1_4  = 1.128781430884e-32 ;
parameter real Out_1_Den_TF_dir2_1_5  = 9.725677466935e-46 ;
parameter real Out_1_Den_TF_dir2_1_6  = 8.031936516939e-59 ;
parameter real Out_1_Num_TF_inv1_1_1  = 4.783275926758e-07 ;
parameter real Out_1_Num_TF_inv1_1_2  = 2.066574976766e-16 ;
parameter real Out_1_Num_TF_inv1_1_3  = -2.831980868699e-27 ;
parameter real Out_1_Num_TF_inv1_1_4  = 4.296594997262e-41 ;
parameter real Out_1_Den_TF_inv1_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Den_TF_inv1_1_2  = 7.712399130202e-12 ;
parameter real Out_1_Den_TF_inv1_1_3  = 4.384399363657e-23 ;
parameter real Out_1_Den_TF_inv1_1_4  = 3.373839214471e-36 ;
parameter real Out_1_Den_TF_inv1_1_5  = 5.724405236602e-53 ;
parameter real Out_1_Den_TF_inv1_1_6  = 5.721335486422e-74 ;
parameter real Out_1_Num_TF_inv2_1_1  = 8.307483096607e-07 ;
parameter real Out_1_Num_TF_inv2_1_2  = -4.922790138277e-16 ;
parameter real Out_1_Num_TF_inv2_1_3  = -6.217399445025e-27 ;
parameter real Out_1_Num_TF_inv2_1_4  = -1.608083462546e-38 ;
parameter real Out_1_Num_TF_inv2_1_5  = -3.721809031789e-52 ;
parameter real Out_1_Den_TF_inv2_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Den_TF_inv2_1_2  = 1.083575721652e-11 ;
parameter real Out_1_Den_TF_inv2_1_3  = 5.705530870031e-23 ;
parameter real Out_1_Den_TF_inv2_1_4  = 1.057492771540e-34 ;
parameter real Out_1_Den_TF_inv2_1_5  = 1.233784603140e-47 ;
parameter real Out_1_Den_TF_inv2_1_6  = 4.793530575889e-61 ;
parameter real Out_1_Num_Zin1_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Num_Zin1_1_2  = 7.810574187728e-12 ;
parameter real Out_1_Num_Zin1_1_3  = 4.125650035554e-23 ;
parameter real Out_1_Num_Zin1_1_4  = 3.391711522248e-36 ;
parameter real Out_1_Num_Zin1_1_5  = 2.747394574238e-51 ;
parameter real Out_1_Num_Zin1_1_6  = 2.253759458416e-64 ;
parameter real Out_1_Den_Zin1_1_1  = -4.410984629251e-06 ;
parameter real Out_1_Den_Zin1_1_2  = 2.317308564808e-14 ;
parameter real Out_1_Den_Zin1_1_3  = 4.512329368446e-25 ;
parameter real Out_1_Den_Zin1_1_4  = 2.431160318837e-37 ;
parameter real Out_1_Num_Zout1_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Num_Zout1_1_2  = 5.473301826773e-10 ;
parameter real Out_1_Num_Zout1_1_3  = 1.761999206136e-20 ;
parameter real Out_1_Num_Zout1_1_4  = 4.816753479040e-32 ;
parameter real Out_1_Num_Zout1_1_5  = 4.032447312237e-45 ;
parameter real Out_1_Num_Zout1_1_6  = 7.510980891643e-59 ;
parameter real Out_1_Den_Zout1_1_1  = 2.866433707175e-07 ;
parameter real Out_1_Den_Zout1_1_2  = 1.877906670098e-13 ;
parameter real Out_1_Den_Zout1_1_3  = 8.536231827455e-23 ;
parameter real Out_1_Den_Zout1_1_4  = 2.679713232291e-33 ;
parameter real Out_1_Den_Zout1_1_5  = 7.196959708127e-45 ;
parameter real Out_1_Den_Zout1_1_6  = 2.876682040870e-58 ;
parameter real Out_1_Num_Zout2_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Num_Zout2_1_2  = 2.846309467993e-11 ;
parameter real Out_1_Num_Zout2_1_3  = 1.594800368005e-24 ;
parameter real Out_1_Num_Zout2_1_4  = 2.712093705546e-39 ;
parameter real Out_1_Num_Zout2_1_5  = 1.513345639249e-52 ;
parameter real Out_1_Den_Zout2_1_1  = 1.732989289772e-05 ;
```

```
parameter real Out_1_Den_Zout2_1_2  = 1.664587611997e-13 ;
parameter real Out_1_Den_Zout2_1_3  = 4.466411980985e-24 ;
parameter real Out_1_OffIn11 = 2.500000e+00;
parameter real Out_1_OffIn12 = -4.849873e-06;
parameter real Out_1_OffOut11 = 4.996447e-04;
parameter real Out_1_OffOut12 = 2.340000e+00;
parameter real Out_1_OffOut21 = -4.955704e-04;
parameter real Out_1_OffOut22 = 2.340000e+00;

real Vin1 ;
real Iin1 ;
real Vout1_1 ;
real Iout1_1 ;
real Vout2_1 ;
real Iout2_1 ;
real deltaVin1 ;
real deltaVout1_1 ;
real deltaVout2_1 ;

analog
begin

  Vin1 = V(vin,);
  I(vin,) <+ Iin1 ;
  Vout1_1 = V(ioutp,);
  I(ioutp,) <+ Iout1_1 ;
  Vout2_1 = V(ioutm,);
  I(ioutm,) <+ Iout2_1 ;

  deltaVin1 = Vin1 - Out_1_OffIn11;
  deltaVout1_1 = Vout1_1 - Out_1_OffOut12;
  deltaVout2_1 = Vout2_1 - Out_1_OffOut22;

  Iout1_1 =
+laplace_nd(deltaVin1,{Out_1_Num_TF_dir1_1_1,Out_1_Num_TF_dir1_1_2,Out_1_Num_TF_dir1_1_3
,Out_1_Num_TF_dir1_1_4,Out_1_Num_TF_dir1_1_5},
{Out_1_Den_TF_dir1_1_1,Out_1_Den_TF_dir1_1_2,Out_1_Den_TF_dir1_1_3,Out_1_Den_TF_dir1_1_4
,Out_1_Den_TF_dir1_1_5,Out_1_Den_TF_dir1_1_6})
+laplace_nd(deltaVout1_1,{Out_1_Den_Zout1_1_1,Out_1_Den_Zout1_1_2,Out_1_Den_Zout1_1_3,Ou
t_1_Den_Zout1_1_4,Out_1_Den_Zout1_1_5,Out_1_Den_Zout1_1_6},
{Out_1_Num_Zout1_1_1,Out_1_Num_Zout1_1_2,Out_1_Num_Zout1_1_3,Out_1_Num_Zout1_1_4,Out_1_N
um_Zout1_1_5,Out_1_Num_Zout1_1_6}) + Out_1_OffOut11;
  Iout2_1 =
+laplace_nd(deltaVin1,{Out_1_Num_TF_dir2_1_1,Out_1_Num_TF_dir2_1_2,Out_1_Num_TF_dir2_1_3
,Out_1_Num_TF_dir2_1_4},
{Out_1_Den_TF_dir2_1_1,Out_1_Den_TF_dir2_1_2,Out_1_Den_TF_dir2_1_3,Out_1_Den_TF_dir2_1_4
,Out_1_Den_TF_dir2_1_5,Out_1_Den_TF_dir2_1_6})
+laplace_nd(deltaVout2_1,{Out_1_Den_Zout2_1_1,Out_1_Den_Zout2_1_2,Out_1_Den_Zout2_1_3},
{Out_1_Num_Zout2_1_1,Out_1_Num_Zout2_1_2,Out_1_Num_Zout2_1_3,Out_1_Num_Zout2_1_4,Out_1_N
um_Zout2_1_5}) + Out_1_OffOut21;
  Iin1 =
+laplace_nd(deltaVin1,{Out_1_Den_Zin1_1_1,Out_1_Den_Zin1_1_2,Out_1_Den_Zin1_1_3,Out_1_De
n_Zin1_1_4},
{Out_1_Num_Zin1_1_1,Out_1_Num_Zin1_1_2,Out_1_Num_Zin1_1_3,Out_1_Num_Zin1_1_4,Out_1_Num_Z
in1_1_5,Out_1_Num_Zin1_1_6})
+laplace_nd(deltaVout1_1,{Out_1_Num_TF_inv1_1_1,Out_1_Num_TF_inv1_1_2,Out_1_Num_TF_inv1_
1_3,Out_1_Num_TF_inv1_1_4},
{Out_1_Den_TF_inv1_1_1,Out_1_Den_TF_inv1_1_2,Out_1_Den_TF_inv1_1_3,Out_1_Den_TF_inv1_1_4
,Out_1_Den_TF_inv1_1_5,Out_1_Den_TF_inv1_1_6})
+laplace_nd(deltaVout2_1,{Out_1_Num_TF_inv2_1_1,Out_1_Num_TF_inv2_1_2,Out_1_Num_TF_inv2_
1_3,Out_1_Num_TF_inv2_1_4,Out_1_Num_TF_inv2_1_5},
{Out_1_Den_TF_inv2_1_1,Out_1_Den_TF_inv2_1_2,Out_1_Den_TF_inv2_1_3,Out_1_Den_TF_inv2_1_4
,Out_1_Den_TF_inv2_1_5,Out_1_Den_TF_inv2_1_6})+ Out_1_OffIn12;

end
endmodule
```

9.3. Differential Current Mirror HDL Models

The VHDL-AMS linear model used for simulations with Dolphin Integration SMASH in the case of the Current mirror is presented:

```
----------------------------------------
-- Generated by SIMECT
-- Generated on: 25-Jun-2012 14:32:13
----------------------------------------

LIBRARY IEEE;
USE IEEE.electrical_systems.all;
use IEEE.math_real.all;
USE work.all;

ENTITY test_i_diff_i_diff IS

  PORT    (TERMINAL Vin1,Vin2,Vout1,Vout2: ELECTRICAL);

END ENTITY test_i_diff_i_diff;

ARCHITECTURE behavioral OF test_i_diff_i_diff IS

  CONSTANT Out_1_Num_TF_dir1_1 : REAL_VECTOR := (-4.942364798317e-01, 3.619091945274e-
12, -5.910993949171e-24, 0.0);
  CONSTANT Out_1_Den_TF_dir1_1 : REAL_VECTOR := (1.000000000000e+00, 9.209412151044e-11,
1.993695527241e-21, 0.0);
  CONSTANT Out_1_Num_TF_dir2_1 : REAL_VECTOR := (5.132056575945e-01, -3.606639541675e-
12, 6.008142911821e-24, 0.0);
  CONSTANT Out_1_Den_TF_dir2_1 : REAL_VECTOR := (1.000000000000e+00, 9.587804145951e-11,
2.223253237730e-21, 0.0);
  CONSTANT Out_1_Num_TF_dir3_1 : REAL_VECTOR := (5.073520996927e-01, -3.692566018699e-
12, 6.398161330145e-24, 0.0);
  CONSTANT Out_1_Den_TF_dir3_1 : REAL_VECTOR := (1.000000000000e+00, 9.619812843732e-11,
2.312388602691e-21, 0.0);
  CONSTANT Out_1_Num_TF_dir4_1 : REAL_VECTOR := (-4.844657916512e-01, 3.613096677274e-
12, -6.046868793534e-24, 0.0);
  CONSTANT Out_1_Den_TF_dir4_1 : REAL_VECTOR := (1.000000000000e+00, 9.167374877558e-11,
2.028564017957e-21, 0.0);
  CONSTANT Out_1_Num_TF_inv1_1 : REAL_VECTOR := (-3.48616716e-07, 8.38225540e-17, 0.0);
  CONSTANT Out_1_Den_TF_inv1_1 : REAL_VECTOR := (1.000000000e+00, 4.482782965e-12, 0.0);
  CONSTANT Out_1_Num_TF_inv2_1 : REAL_VECTOR := (-2.43353247e-06, 2.406536756e-15, 0.0);
  CONSTANT Out_1_Den_TF_inv2_1 : REAL_VECTOR := (1.00000000e+00, 2.152929623e-11,
1.311806131097e-22, 0.0);
  CONSTANT Out_1_Num_TF_inv3_1 : REAL_VECTOR := (-2.69429481e-06, 2.737893116e-15, 0.0);
  CONSTANT Out_1_Den_TF_inv3_1 : REAL_VECTOR := (1.00000000e+00, 2.2331630748e-11,
1.411564217149e-22, 0.0);
  CONSTANT Out_1_Num_TF_inv4_1 : REAL_VECTOR := (-3.75163404e-07, 8.969411956e-17, 0.0);
  CONSTANT Out_1_Den_TF_inv4_1 : REAL_VECTOR := (1.00000000e+00, 4.470818653e-12, 0.0);
  CONSTANT Out_1_Num_Zin1_1 : REAL_VECTOR := (1.00000000e+00, 6.678475090e-11, 0.0);
  CONSTANT Out_1_Den_Zin1_1 : REAL_VECTOR := (9.39149606e-02, 1.477287883e-12,
1.148437913563e-26, 0.0);
  CONSTANT Out_1_Num_Zin2_1 : REAL_VECTOR := (1.00000000e+00, 7.957696351e-11, 0.0);
  CONSTANT Out_1_Den_Zin2_1 : REAL_VECTOR := (9.67519721e-02, 1.774069010e-12, 0.0);
  CONSTANT Out_1_Num_Zin_diff1_1 : REAL_VECTOR := (0.000000000000e+00, 0.0);
  CONSTANT Out_1_Den_Zin_diff1_1 : REAL_VECTOR := (1.000000000000e+00, 0.0);
  CONSTANT Out_1_Num_Zin_diff2_1 : REAL_VECTOR := (0.000000000000e+00, 0.0);
  CONSTANT Out_1_Den_Zin_diff2_1 : REAL_VECTOR := (1.000000000000e+00, 0.0);
  CONSTANT Out_1_Num_Zout1_1 : REAL_VECTOR := (1.000000000000e+00, 3.167267728687e-13,
1.225675234239e-26, 0.0);
  CONSTANT Out_1_Den_Zout1_1 : REAL_VECTOR := (1.732293643995e-04, 2.522576573002e-14,
6.736633293931e-27, 0.0);
  CONSTANT Out_1_Num_Zout2_1 : REAL_VECTOR := (1.000000000000e+00, 3.144998826705e-13,
1.214650787950e-26, 0.0);
  CONSTANT Out_1_Den_Zout2_1 : REAL_VECTOR := (1.670640056852e-04, 2.521866886640e-14,
6.681625484108e-27, 0.0);
  CONSTANT Out_1_Num_Zout_diff1_1 : REAL_VECTOR := (1.000000000000e+00, 0.0);
  CONSTANT Out_1_Den_Zout_diff1_1 : REAL_VECTOR := (1.432159207110e-06, 0.0);
  CONSTANT Out_1_Num_Zout_diff2_1 : REAL_VECTOR := (1.000000000000e+00, 0.0);
  CONSTANT Out_1_Den_Zout_diff2_1 : REAL_VECTOR := (1.463607998754e-06, 0.0);
  CONSTANT Out_1_OffIn11 : REAL := -1.000000e-04;
  CONSTANT Out_1_OffIn12 : REAL := 2.139866e+00;
  CONSTANT Out_1_OffIn21 : REAL := 1.000000e-04;
  CONSTANT Out_1_OffIn22 : REAL := 2.137801e+00;
  CONSTANT Out_1_OffOut11 : REAL := 4.566877e-05;
  CONSTANT Out_1_OffOut12 : REAL := 2.250000e+00;
  CONSTANT Out_1_OffOut21 : REAL := -1.543486e-04;
  CONSTANT Out_1_OffOut22 : REAL := 2.250000e+00;
  QUANTITY Vin1 ACROSS Iin1 THROUGH Vin1 TO ground;
  QUANTITY Vin2 ACROSS Iin2 THROUGH Vin2 TO ground;
  QUANTITY Vout1_1 ACROSS Iout1_1 THROUGH Vout1 TO ground;
  QUANTITY Vout2_1 ACROSS Iout2_1 THROUGH Vout2 TO ground;
```

```
  QUANTITY deltaIin1, deltaIin2: current;
  QUANTITY deltaVout1_1, deltaVout2_1: voltage;

BEGIN

  deltaIin1 == Iin1 - Out_1_OffIn11;
  deltaIin2 == Iin2 - Out_1_OffIn21;
  deltaVout1_1 == Vout1_1 - Out_1_OffOut12;
  deltaVout2_1 == Vout2_1 - Out_1_OffOut22;
  Iout1_1 == deltaIin1'LTF(Out_1_Num_TF_dir1_1,Out_1_Den_TF_dir1_1)+
deltaIin2'LTF(Out_1_Num_TF_dir3_1,Out_1_Den_TF_dir3_1)+
deltaVout1_1'LTF(Out_1_Den_Zout1_1,Out_1_Num_Zout1_1)+ deltaVout2_1*(1.463607998754e-06)
+ Out_1_OffOut11;
  Iout2_1 == deltaIin1'LTF(Out_1_Num_TF_dir2_1,Out_1_Den_TF_dir2_1)+
deltaIin2'LTF(Out_1_Num_TF_dir4_1,Out_1_Den_TF_dir4_1)+
deltaVout2_1'LTF(Out_1_Den_Zout2_1,Out_1_Num_Zout2_1)+ deltaVout1_1*(1.432159207110e-06)
+ Out_1_OffOut21;
  Vin1 == deltaIin1'LTF(Out_1_Num_Zin1_1,Out_1_Den_Zin1_1)+
deltaIin2*(0.000000000000e+00) +
deltaVout1_1'LTF(Out_1_Num_TF_inv1_1,Out_1_Den_TF_inv1_1)+
deltaVout2_1'LTF(Out_1_Num_TF_inv2_1,Out_1_Den_TF_inv2_1)+ Out_1_OffIn12;
  Vin2 == deltaIin2'LTF(Out_1_Num_Zin2_1,Out_1_Den_Zin2_1)+
deltaIin1*(0.000000000000e+00) +
deltaVout1_1'LTF(Out_1_Num_TF_inv3_1,Out_1_Den_TF_inv3_1)+
deltaVout2_1'LTF(Out_1_Num_TF_inv4_1,Out_1_Den_TF_inv4_1)+ Out_1_OffIn22;

END ARCHITECTURE behavioral;
```

The VHDL-AMS linear model for Cadence AMS in the case of the Current mirror is also presented:

```
----------------------------------------
-- Generated by SIMECT
-- Generated on: 25-Jun-2012 14:32:01
----------------------------------------

library ieee, std;
use ieee.std_logic_1164.all;
USE ieee.electrical_systems.all;
use ieee.math_real.all;
use work.all;

ENTITY test_i_diff_i_diff IS

  PORT    (TERMINAL Vin1,Vin2,Vout1,Vout2: ELECTRICAL);

END ENTITY test_i_diff_i_diff;

ARCHITECTURE behavioral OF test_i_diff_i_diff IS

  CONSTANT Out_1_Num_TF_dir1_1 : REAL_VECTOR := (-7.03019544e-01, 1.728806217e-12, 0.0);
  CONSTANT Out_1_Den_TF_dir1_1 : REAL_VECTOR := (1.00000000e+00, 3.479378225e-11, 0.0);
  CONSTANT Out_1_Num_TF_dir1_2 : REAL_VECTOR := (7.03019544e-01, -3.419118864e-12, 0.0);
  CONSTANT Out_1_Den_TF_dir1_2 : REAL_VECTOR := (1.00000000e+00, 5.730033925e-11, 0.0);
  CONSTANT Out_1_Num_TF_dir2_1 : REAL_VECTOR := (7.16383716e-01, -1.944180484e-12, 0.0);
  CONSTANT Out_1_Den_TF_dir2_1 : REAL_VECTOR := (1.00000000e+00, 3.928471736e-11, 0.0);
  CONSTANT Out_1_Num_TF_dir2_2 : REAL_VECTOR := (7.16383716e-01, -3.090337850e-12, 0.0);
  CONSTANT Out_1_Den_TF_dir2_2 : REAL_VECTOR := (1.00000000e+00, 5.659329514e-11, 0.0);
  CONSTANT Out_1_Num_TF_dir3_1 : REAL_VECTOR := (7.12286591e-01, -2.025864585e-12, 0.0);
  CONSTANT Out_1_Den_TF_dir3_1 : REAL_VECTOR := (1.00000000e+00, 4.70354028e-11, 0.0);
  CONSTANT Out_1_Num_TF_dir3_2 : REAL_VECTOR := (7.12286091e-01, -3.158237402e-12, 0.0);
  CONSTANT Out_1_Den_TF_dir3_2 : REAL_VECTOR := (1.00000000e+00, 4.916272584e-11, 0.0);
  CONSTANT Out_1_Num_TF_dir4_1 : REAL_VECTOR := (-6.96035752e-01, 1.765025397e-12, 0.0);
  CONSTANT Out_1_Den_TF_dir4_1 : REAL_VECTOR := (1.00000000e+00, 3.732482204e-11, 0.0);
  CONSTANT Out_1_Num_TF_dir4_2 : REAL_VECTOR := (6.96035762e-01, -3.425938064e-12, 0.0);
  CONSTANT Out_1_Den_TF_dir4_2 : REAL_VECTOR := (1.00000000e+00, 5.434892674e-11, 0.0);
  CONSTANT Out_1_Num_TF_inv1_1 : REAL_VECTOR := (-3.48616689e-07, 8.382255588e-17, 0.0);
  CONSTANT Out_1_Den_TF_inv1_1 : REAL_VECTOR := (1.00000000e+00, 4.482782960e-12, 0.0);
  CONSTANT Out_1_Num_TF_inv2_1 : REAL_VECTOR := (-2.43353243e-06, 2.406536945e-15,
0.00000000e+00, 0.0);
  CONSTANT Out_1_Den_TF_inv2_1 : REAL_VECTOR := (1.00000000e+00, 2.15292931e-11,
1.31180697e-22, 0.0);
  CONSTANT Out_1_Num_TF_inv3_1 : REAL_VECTOR := (-2.69411423e-06, 2.737896976e-15,
0.00000000e+00, 0.0);
```

```
  CONSTANT Out_1_Den_TF_inv3_1 : REAL_VECTOR := (1.00000000e+00, 2.23316307e-11,
1.41156449e-22, 0.0);
  CONSTANT Out_1_Num_TF_inv4_1 : REAL_VECTOR := (-3.75163404e-07, 8.969411956e-17, 0.0);
  CONSTANT Out_1_Den_TF_inv4_1 : REAL_VECTOR := (1.00000000e+00, 4.470818653e-12, 0.0);
  CONSTANT Out_1_Num_Zin1_1 : REAL_VECTOR := (3.26311672e+00, 2.179264378e-10, 0.0);
  CONSTANT Out_1_Den_Zin1_1 : REAL_VECTOR := (1.00000000e+00, 7.777807279e-15, 0.0);
  CONSTANT Out_1_Num_Zin1_2 : REAL_VECTOR := (3.26311672e+00, 0.0);
  CONSTANT Out_1_Den_Zin1_2 : REAL_VECTOR := (1.00000000e+00, 1.572228131e-11, 0.0);
  CONSTANT Out_1_Num_Zin2_1 : REAL_VECTOR := (1.03357066e+01, 8.224841497e-10, 0.0);
  CONSTANT Out_1_Den_Zin2_1 : REAL_VECTOR := (1.00000000e+00, 1.833625684e-11, 0.0);
  CONSTANT Out_1_Num_Zin_diff1_1 : REAL_VECTOR := (0.000000000e+00, 0.0);
  CONSTANT Out_1_Den_Zin_diff1_1 : REAL_VECTOR := (1.000000000e+00, 0.0);
  CONSTANT Out_1_Num_Zin_diff2_1 : REAL_VECTOR := (0.000000000e+00, 0.0);
  CONSTANT Out_1_Den_Zin_diff2_1 : REAL_VECTOR := (1.000000000e+00, 0.0);
  CONSTANT Out_1_Num_Zout1_1 : REAL_VECTOR := (1.00000000e+00, 4.512818951e-14, 0.0);
  CONSTANT Out_1_Den_Zout1_1 : REAL_VECTOR := (1.31616626e-02, 3.521339981e-15, 0.0);
  CONSTANT Out_1_Num_Zout1_2 : REAL_VECTOR := (1.00000000e+00, 2.715985833e-13, 0.0);
  CONSTANT Out_1_Den_Zout1_2 : REAL_VECTOR := (1.31616626e-02, 1.913088008e-12, 0.0);
  CONSTANT Out_1_Num_Zout2_1 : REAL_VECTOR := (1.00000000e+00, 4.50847231e-14, 0.0);
  CONSTANT Out_1_Den_Zout2_1 : REAL_VECTOR := (1.29253242e-02, 3.430565320e-15, 0.0);
  CONSTANT Out_1_Num_Zout2_2 : REAL_VECTOR := (1.00000000e+00, 2.694151595e-13, 0.0);
  CONSTANT Out_1_Den_Zout2_2 : REAL_VECTOR := (1.29253242e-02, 1.947674759e-12, 0.0);
  CONSTANT Out_1_OffIn11 : REAL := -1.000000e-04;
  CONSTANT Out_1_OffIn12 : REAL := 2.139866e+00;
  CONSTANT Out_1_OffIn21 : REAL := 1.000000e-04;
  CONSTANT Out_1_OffIn22 : REAL := 2.137801e+00;
  CONSTANT Out_1_OffOut11 : REAL := 4.566877e-05;
  CONSTANT Out_1_OffOut12 : REAL := 2.250000e+00;
  CONSTANT Out_1_OffOut21 : REAL := -1.543486e-04;
  CONSTANT Out_1_OffOut22 : REAL := 2.250000e+00;
  QUANTITY Vin1 ACROSS Iin1 THROUGH Vin1 TO ground;
  QUANTITY Vin2 ACROSS Iin2 THROUGH Vin2 TO ground;
  QUANTITY Vout1_1 ACROSS Iout1_1 THROUGH Vout1 TO ground;
  QUANTITY Vout2_1 ACROSS Iout2_1 THROUGH Vout2 TO ground;
  QUANTITY deltaIin1: current;
  QUANTITY deltaIin2: current;
  QUANTITY deltaVout1_1: voltage;
  QUANTITY deltaVout2_1: voltage;

BEGIN

  deltaIin1 == Iin1 - Out_1_OffIn11;
  deltaIin2 == Iin2 - Out_1_OffIn21;
  deltaVout1_1 == Vout1_1 - Out_1_OffOut12;
  deltaVout2_1 == Vout2_1 - Out_1_OffOut22;
  Iout1_1 ==
 deltaIin1'LTF((Out_1_Num_TF_dir1_1(0),Out_1_Num_TF_dir1_1(1)),
(Out_1_Den_TF_dir1_1(0),Out_1_Den_TF_dir1_1(1)))'LTF((Out_1_Num_TF_dir1_2(0),Out_1_Num_T
F_dir1_2(1)), (Out_1_Den_TF_dir1_2(0),Out_1_Den_TF_dir1_2(1)))
 + deltaIin2'LTF((Out_1_Num_TF_dir3_1(0),Out_1_Num_TF_dir3_1(1)),
(Out_1_Den_TF_dir3_1(0),Out_1_Den_TF_dir3_1(1)))'LTF((Out_1_Num_TF_dir3_2(0),Out_1_Num_T
F_dir3_2(1)), (Out_1_Den_TF_dir3_2(0),Out_1_Den_TF_dir3_2(1)))
 + deltaVout1_1'LTF((Out_1_Den_Zout1_1(0),Out_1_Den_Zout1_1(1)),
(Out_1_Num_Zout1_1(0),Out_1_Num_Zout1_1(1)))'LTF((Out_1_Den_Zout1_2(0),Out_1_Den_Zout1_2
(1)), (Out_1_Num_Zout1_2(0),Out_1_Num_Zout1_2(1))) + Out_1_OffOut11;
  Iout2_1 ==
 deltaIin1'LTF((Out_1_Num_TF_dir2_1(0),Out_1_Num_TF_dir2_1(1)),
(Out_1_Den_TF_dir2_1(0),Out_1_Den_TF_dir2_1(1)))'LTF((Out_1_Num_TF_dir2_2(0),Out_1_Num_T
F_dir2_2(1)), (Out_1_Den_TF_dir2_2(0),Out_1_Den_TF_dir2_2(1)))
 + deltaIin2'LTF((Out_1_Num_TF_dir4_1(0),Out_1_Num_TF_dir4_1(1)),
(Out_1_Den_TF_dir4_1(0),Out_1_Den_TF_dir4_1(1)))'LTF((Out_1_Num_TF_dir4_2(0),Out_1_Num_T
F_dir4_2(1)), (Out_1_Den_TF_dir4_2(0),Out_1_Den_TF_dir4_2(1)))
 + deltaVout2_1'LTF((Out_1_Den_Zout2_1(0),Out_1_Den_Zout2_1(1)),
(Out_1_Num_Zout2_1(0),Out_1_Num_Zout2_1(1)))'LTF((Out_1_Den_Zout2_2(0),Out_1_Den_Zout2_2
(1)), (Out_1_Num_Zout2_2(0),Out_1_Num_Zout2_2(1)) + Out_1_OffOut21;
  Vin1 == deltaIin1'LTF((Out_1_Num_Zin1_1(0),Out_1_Num_Zin1_1(1)),
(Out_1_Den_Zin1_1(0),Out_1_Den_Zin1_1(1)))'LTF((Out_1_Num_Zin1_2(0),Out_1_Num_Zin1_2(1))
, (Out_1_Den_Zin1_2(0),Out_1_Den_Zin1_2(1)))+ deltaIin2*(0.000000000000e+00) +
deltaVout1_1'LTF((Out_1_Num_TF_inv1_1(0),Out_1_Num_TF_inv1_1(1)),
(Out_1_Den_TF_inv1_1(0),Out_1_Den_TF_inv1_1(1)))+
deltaVout2_1'LTF((Out_1_Num_TF_inv2_1(0),Out_1_Num_TF_inv2_1(1),Out_1_Num_TF_inv2_1(2)),
(Out_1_Den_TF_inv2_1(0),Out_1_Den_TF_inv2_1(1),Out_1_Den_TF_inv2_1(2)))+ Out_1_OffIn12;
  Vin2 == deltaIin2'LTF((Out_1_Num_Zin2_1(0),Out_1_Num_Zin2_1(1)),
(Out_1_Den_Zin2_1(0),Out_1_Den_Zin2_1(1)))+ deltaIin1*(0.000000000000e+00) +
deltaVout1_1'LTF((Out_1_Num_TF_inv3_1(0),Out_1_Num_TF_inv3_1(1),Out_1_Num_TF_inv3_1(2)),
(Out_1_Den_TF_inv3_1(0),Out_1_Den_TF_inv3_1(1),Out_1_Den_TF_inv3_1(2)))+
```

```
deltaVout2_1'LTF((Out_1_Num_TF_inv4_1(0),Out_1_Num_TF_inv4_1(1)),
(Out_1_Den_TF_inv4_1(0),Out_1_Den_TF_inv4_1(1)))+ Out_1_OffIn22;

END ARCHITECTURE behavioral;
```

The extracted Verilog-A linear model for Cadence AMS in the case of the Current mirror will be:

```
//-------------------------------------
// Generated by SIMECT
// Generated on: 25-Jun-2012 14:32:26
//-------------------------------------

`include "constants.vams"
`include "disciplines.vams"

module test_i_diff_i_diff (Vin1,Vin2,Vout1,Vout2);
input Vin1;
electrical Vin1;
input Vin2;
electrical Vin2;
output Vout1;
electrical Vout1;
output Vout2;
electrical Vout2;

parameter real Out_1_Num_TF_dir1_1_1  = -4.942364798317e-01 ;
parameter real Out_1_Num_TF_dir1_1_2  = 3.619091945274e-12 ;
parameter real Out_1_Num_TF_dir1_1_3  = -5.910993949171e-24 ;
parameter real Out_1_Den_TF_dir1_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Den_TF_dir1_1_2  = 9.209412151044e-11 ;
parameter real Out_1_Den_TF_dir1_1_3  = 1.993695527241e-21 ;
parameter real Out_1_Num_TF_dir2_1_1  = 5.132056575945e-01 ;
parameter real Out_1_Num_TF_dir2_1_2  = -3.606639541675e-12 ;
parameter real Out_1_Num_TF_dir2_1_3  = 6.008142911821e-24 ;
parameter real Out_1_Den_TF_dir2_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Den_TF_dir2_1_2  = 9.587804145951e-11 ;
parameter real Out_1_Den_TF_dir2_1_3  = 2.223253237730e-21 ;
parameter real Out_1_Num_TF_dir3_1_1  = 5.073520996927e-01 ;
parameter real Out_1_Num_TF_dir3_1_2  = -3.692566018699e-12 ;
parameter real Out_1_Num_TF_dir3_1_3  = 6.398161330145e-24 ;
parameter real Out_1_Den_TF_dir3_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Den_TF_dir3_1_2  = 9.619812843732e-11 ;
parameter real Out_1_Den_TF_dir3_1_3  = 2.312388602691e-21 ;
parameter real Out_1_Num_TF_dir4_1_1  = -4.844657916512e-01 ;
parameter real Out_1_Num_TF_dir4_1_2  = 3.613096677274e-12 ;
parameter real Out_1_Num_TF_dir4_1_3  = -6.046868793534e-24 ;
parameter real Out_1_Den_TF_dir4_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Den_TF_dir4_1_2  = 9.167374877558e-11 ;
parameter real Out_1_Den_TF_dir4_1_3  = 2.028564017957e-21 ;
parameter real Out_1_Num_TF_inv1_1_1  = -3.486167165689e-07 ;
parameter real Out_1_Num_TF_inv1_1_2  = 8.382255405588e-17 ;
parameter real Out_1_Den_TF_inv1_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Den_TF_inv1_1_2  = 4.482782965400e-12 ;
parameter real Out_1_Num_TF_inv2_1_1  = -2.433532472643e-06 ;
parameter real Out_1_Num_TF_inv2_1_2  = 2.406536756445e-15 ;
parameter real Out_1_Den_TF_inv2_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Den_TF_inv2_1_2  = 2.152929623931e-11 ;
parameter real Out_1_Den_TF_inv2_1_3  = 1.311806131097e-22 ;
parameter real Out_1_Num_TF_inv3_1_1  = -2.694294811423e-06 ;
parameter real Out_1_Num_TF_inv3_1_2  = 2.737893116976e-15 ;
parameter real Out_1_Den_TF_inv3_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Den_TF_inv3_1_2  = 2.233163074851e-11 ;
parameter real Out_1_Den_TF_inv3_1_3  = 1.411564217149e-22 ;
parameter real Out_1_Num_TF_inv4_1_1  = -3.751634040420e-07 ;
parameter real Out_1_Num_TF_inv4_1_2  = 8.969411956430e-17 ;
parameter real Out_1_Den_TF_inv4_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Den_TF_inv4_1_2  = 4.470818653026e-12 ;
parameter real Out_1_Num_Zin1_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Num_Zin1_1_2  = 6.678475090746e-11 ;
parameter real Out_1_Den_Zin1_1_1  = 9.391496064130e-02 ;
parameter real Out_1_Den_Zin1_1_2  = 1.477287883452e-12 ;
parameter real Out_1_Den_Zin1_1_3  = 1.148437913563e-26 ;
parameter real Out_1_Num_Zin2_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Num_Zin2_1_2  = 7.957696351986e-11 ;
```

```
parameter real Out_1_Den_Zin2_1_1  = 9.675197210992e-02 ;
parameter real Out_1_Den_Zin2_1_2  = 1.774069010995e-12 ;
parameter real Out_1_Num_Zin_diff1_1_1  = 0.000000000000e+00 ;
parameter real Out_1_Den_Zin_diff1_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Num_Zin_diff2_1_1  = 0.000000000000e+00 ;
parameter real Out_1_Den_Zin_diff2_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Num_Zout1_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Num_Zout1_1_2  = 3.167267728687e-13 ;
parameter real Out_1_Num_Zout1_1_3  = 1.225675234239e-26 ;
parameter real Out_1_Den_Zout1_1_1  = 1.732293643995e-04 ;
parameter real Out_1_Den_Zout1_1_2  = 2.522576573002e-14 ;
parameter real Out_1_Den_Zout1_1_3  = 6.736633293931e-27 ;
parameter real Out_1_Num_Zout2_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Num_Zout2_1_2  = 3.144998826705e-13 ;
parameter real Out_1_Num_Zout2_1_3  = 1.214650787950e-26 ;
parameter real Out_1_Den_Zout2_1_1  = 1.670640056852e-04 ;
parameter real Out_1_Den_Zout2_1_2  = 2.521866886640e-14 ;
parameter real Out_1_Den_Zout2_1_3  = 6.681625484108e-27 ;
parameter real Out_1_OffIn11 = -1.000000e-04;
parameter real Out_1_OffIn12 = 2.139866e+00;
parameter real Out_1_OffIn21 = 1.000000e-04;
parameter real Out_1_OffIn22 = 2.137801e+00;
parameter real Out_1_OffOut11 = 4.566877e-05;
parameter real Out_1_OffOut12 = 2.250000e+00;
parameter real Out_1_OffOut21 = -1.543486e-04;
parameter real Out_1_OffOut22 = 2.250000e+00;

real Vin1 ;
real Iin1 ;
real Vin2 ;
real Iin2 ;
real Vout1_1 ;
real Iout1_1 ;
real Vout2_1 ;
real Iout2_1 ;
real deltaIin1 ;
real deltaIin2 ;
real deltaVout1_1 ;
real deltaVout2_1 ;

analog
begin

  Iin1 = I(Vin1,);
  V(Vin1,) <+ Vin1 ;
  Iin2 = I(Vin2,);
  V(Vin2,) <+ Vin2 ;
  Vout1_1 = V(Vout1,);
  I(Vout1,) <+ Iout1_1 ;
  Vout2_1 = V(Vout2,);
  I(Vout2,) <+ Iout2_1 ;
  deltaIin1 = Iin1 - Out_1_OffIn11;
  deltaIin2 = Iin2 - Out_1_OffIn21;
  deltaVout1_1 = Vout1_1 - Out_1_OffOut12;
  deltaVout2_1 = Vout2_1 - Out_1_OffOut22;

  Iout1_1 =
+laplace_nd(deltaIin1,{Out_1_Num_TF_dir1_1_1,Out_1_Num_TF_dir1_1_2,Out_1_Num_TF_dir1_1_3
}, {Out_1_Den_TF_dir1_1_1,Out_1_Den_TF_dir1_1_2,Out_1_Den_TF_dir1_1_3})
+laplace_nd(deltaIin2,{Out_1_Num_TF_dir3_1_1,Out_1_Num_TF_dir3_1_2,Out_1_Num_TF_dir3_1_3
}, {Out_1_Den_TF_dir3_1_1,Out_1_Den_TF_dir3_1_2,Out_1_Den_TF_dir3_1_3})
+laplace_nd(deltaVout1_1,{Out_1_Den_Zout1_1_1,Out_1_Den_Zout1_1_2,Out_1_Den_Zout1_1_3},
{Out_1_Num_Zout1_1_1,Out_1_Num_Zout1_1_2,Out_1_Num_Zout1_1_3}) + Out_1_OffOut11;
  Iout2_1 =
+laplace_nd(deltaIin1,{Out_1_Num_TF_dir2_1_1,Out_1_Num_TF_dir2_1_2,Out_1_Num_TF_dir2_1_3
}, {Out_1_Den_TF_dir2_1_1,Out_1_Den_TF_dir2_1_2,Out_1_Den_TF_dir2_1_3})
+laplace_nd(deltaIin2,{Out_1_Num_TF_dir4_1_1,Out_1_Num_TF_dir4_1_2,Out_1_Num_TF_dir4_1_3
}, {Out_1_Den_TF_dir4_1_1,Out_1_Den_TF_dir4_1_2,Out_1_Den_TF_dir4_1_3})
+laplace_nd(deltaVout2_1,{Out_1_Den_Zout2_1_1,Out_1_Den_Zout2_1_2,Out_1_Den_Zout2_1_3},
{Out_1_Num_Zout2_1_1,Out_1_Num_Zout2_1_2,Out_1_Num_Zout2_1_3}) + Out_1_OffOut21;
  Vin1 =  +laplace_nd(deltaIin1,{Out_1_Num_Zin1_1_1,Out_1_Num_Zin1_1_2},
{Out_1_Den_Zin1_1_1,Out_1_Den_Zin1_1_2,Out_1_Den_Zin1_1_3}) +0.0
+laplace_nd(deltaVout1_1,{Out_1_Num_TF_inv1_1_1,Out_1_Num_TF_inv1_1_2},
{Out_1_Den_TF_inv1_1_1,Out_1_Den_TF_inv1_1_2})
+laplace_nd(deltaVout2_1,{Out_1_Num_TF_inv2_1_1,Out_1_Num_TF_inv2_1_2},
{Out_1_Den_TF_inv2_1_1,Out_1_Den_TF_inv2_1_2,Out_1_Den_TF_inv2_1_3})+ Out_1_OffIn12;
```

```
  Vin2 =  +laplace_nd(deltaIin2,{Out_1_Num_Zin2_1_1,Out_1_Num_Zin2_1_2},
{Out_1_Den_Zin2_1_1,Out_1_Den_Zin2_1_2}) +0.0
+laplace_nd(deltaVout1_1,{Out_1_Num_TF_inv3_1_1,Out_1_Num_TF_inv3_1_2},
{Out_1_Den_TF_inv3_1_1,Out_1_Den_TF_inv3_1_2,Out_1_Den_TF_inv3_1_3})
+laplace_nd(deltaVout2_1,{Out_1_Num_TF_inv4_1_1,Out_1_Num_TF_inv4_1_2},
{Out_1_Den_TF_inv4_1_1,Out_1_Den_TF_inv4_1_2})+ Out_1_OffIn22;

end
endmodule
```

9.4. Second generation Current Conveyer (CCII) HDL Models

The VHDL-AMS linear model used for simulations with Dolphin Integration SMASH in the case of the Second generation current conveyer (CCII) with acoustic resonator is presented:

```
----------------------------------------
-- Generated by SIMECT
-- Generated on: 26-Jun-2012 18:38:25
----------------------------------------

LIBRARY IEEE;
USE IEEE.electrical_systems.all;
use IEEE.math_real.all;
USE work.all;

ENTITY filtre_test_rlc_4res_conveyer3_1v IS

  PORT    (TERMINAL gnd!,Vin,Iout: ELECTRICAL);

END ENTITY filtre_test_rlc_4res_conveyer3_1v;


ARCHITECTURE behavioral OF filtre_test_rlc_4res_conveyer3_1v IS

  CONSTANT Out_1_Num_TF_dir1_1 : REAL_VECTOR := (7.169122790853e-06, 5.237279260008e-15,
-2.979915690365e-26, 3.327631782769e-35, 2.579879882589e-47, -2.197545192937e-59, -
5.113191326721e-71, 0.0);
  CONSTANT Out_1_Den_TF_dir1_1 : REAL_VECTOR := (1.000000000000e+00, 1.182449777199e-10,
4.972570495025e-21, 4.970189001622e-31, 2.004201492397e-42, 4.968257835073e-54,
7.697004798142e-84, 0.0);
  CONSTANT Out_1_Num_TF_inv1_1 : REAL_VECTOR := (-1.493117711085e-07, 2.943084739429e-
17, 4.829145939645e-28, -6.476147784368e-40, 0.0);
  CONSTANT Out_1_Den_TF_inv1_1 : REAL_VECTOR := (1.000000000000e+00, 5.123400417199e-12,
4.169859504916e-23, 5.140582983458e-37, 0.0);
  CONSTANT Out_1_Num_Zin1_1 : REAL_VECTOR := (1.000000000000e+00, 4.284475717757e-11,
3.550663188109e-22, 8.781098115568e-34, 5.423279188483e-59, 0.0);
  CONSTANT Out_1_Den_Zin1_1 : REAL_VECTOR := (2.074001437424e-04, 5.102337159191e-14,
1.292050726761e-24, 6.778138824786e-36, 8.290434413062e-48, 0.0);
  CONSTANT Out_1_Num_Zout1_1 : REAL_VECTOR := (1.000000000000e+00, 4.839044114655e-11,
4.338896607202e-22, 7.455079594181e-34, 2.462587263949e-50, 0.0);
  CONSTANT Out_1_Den_Zout1_1 : REAL_VECTOR := (2.166713552798e-04, 2.599576573783e-14,
9.839197396340e-25, 6.053447358283e-36, 7.916313592994e-48, 0.0);
  CONSTANT Out_1_OffIn11 : REAL := 0.000000e+00;
  CONSTANT Out_1_OffIn12 : REAL := 1.140241e-05;
  CONSTANT Out_1_OffOut11 : REAL := 1.770304e-05;
  CONSTANT Out_1_OffOut12 : REAL := 0.000000e+00;
  QUANTITY Vin_1 ACROSS Iin_1 THROUGH Vin TO ground;
  QUANTITY Vout_1 ACROSS Iout_1 THROUGH Iout TO ground;
  QUANTITY deltaVin: voltage;
  QUANTITY deltaVout_1: voltage;


BEGIN


  deltaVin == Vin_1 - Out_1_OffIn11;
  deltaVout_1 == Vout_1 - Out_1_OffOut12;
  Iout_1 == deltaVin'LTF(Out_1_Num_TF_dir1_1,Out_1_Den_TF_dir1_1)+
deltaVout_1'LTF(Out_1_Den_Zout1_1,Out_1_Num_Zout1_1)+ Out_1_OffOut11;
  Iin_1 == deltaVin'LTF(Out_1_Den_Zin1_1,Out_1_Num_Zin1_1)+
deltaVout_1'LTF(Out_1_Num_TF_inv1_1,Out_1_Den_TF_inv1_1)+ Out_1_OffIn12;
```

```
END ARCHITECTURE behavioral;
```

The VHDL-AMS linear model used for simulations with Cadence AMS in the case of the Second generation current conveyer (CCII) with acoustic resonator is also presented:

```
----------------------------------------
-- Generated by SIMECT
-- Generated on: 26-Jun-2012 18:36:40
----------------------------------------

library ieee, std;
use ieee.std_logic_1164.all;
USE ieee.electrical_systems.all;
use ieee.math_real.all;
use work.all;


ENTITY filtre_test_rlc_4res_conveyer3_1v IS

  PORT    (TERMINAL gnd!,Vin,Iout: ELECTRICAL);

END ENTITY filtre_test_rlc_4res_conveyer3_1v;


ARCHITECTURE behavioral OF filtre_test_rlc_4res_conveyer3_1v IS

  CONSTANT Out_1_Num_TF_dir1_1 : REAL_VECTOR := (5.17447752e-02, -5.686353494e-14, 0.0);
  CONSTANT Out_1_Den_TF_dir1_1 : REAL_VECTOR := (1.000000000e+00, 1.549236181e-30, 0.0);
  CONSTANT Out_1_Num_TF_dir1_2 : REAL_VECTOR := (5.174477522e-02, 3.852304379e-11, 0.0);
  CONSTANT Out_1_Den_TF_dir1_2 : REAL_VECTOR := (1.000000000e+00, 1.120040422e-10, 0.0);
  CONSTANT Out_1_Num_TF_dir1_3 : REAL_VECTOR := (5.174477522329e-02, 9.692270745681e-14,
7.228843896510e-26, 0.0);
  CONSTANT Out_1_Den_TF_dir1_3 : REAL_VECTOR := (1.000000000000e+00, 4.100155123947e-12,
1.040681538111e-23, 0.0);
  CONSTANT Out_1_Num_TF_dir1_4 : REAL_VECTOR := (5.174477522329e-02, -7.618480634802e-
13, 3.229005229162e-22, 0.0);
  CONSTANT Out_1_Den_TF_dir1_4 : REAL_VECTOR := (1.000000000000e+00, 2.068221705397e-12,
4.259806750597e-21, 0.0);
  CONSTANT Out_1_Num_TF_inv1_1 : REAL_VECTOR := (-3.86408813e-04, 4.818132459e-16, 0.0);
  CONSTANT Out_1_Den_TF_inv1_1 : REAL_VECTOR := (1.00000000e+00, 1.234663719e-14, 0.0);
  CONSTANT Out_1_Num_TF_inv1_2 : REAL_VECTOR := (3.864088134457e-04, -7.568323881862e-
14, -1.344119913542e-24, 0.0);
  CONSTANT Out_1_Den_TF_inv1_2 : REAL_VECTOR := (1.000000000000e+00, 5.111053780005e-12,
4.163549072246e-23, 0.0);
  CONSTANT Out_1_Num_Zin1_1 : REAL_VECTOR := (1.00000000e+00, 6.176083124e-26, 0.0);
  CONSTANT Out_1_Den_Zin1_1 : REAL_VECTOR := (5.91929090e-02, 1.061106901e-13, 0.0);
  CONSTANT Out_1_Num_Zin1_2 : REAL_VECTOR := (1.00000000e+00, 3.284967576e-11, 0.0);
  CONSTANT Out_1_Den_Zin1_2 : REAL_VECTOR := (5.91929090e-02, 2.843544212e-13, 0.0);
  CONSTANT Out_1_Num_Zin1_3 : REAL_VECTOR := (1.000000000000e+00, 9.995080580799e-12,
2.673115560817e-23, 0.0);
  CONSTANT Out_1_Den_Zin1_3 : REAL_VECTOR := (5.919290909697e-02, 1.417182853963e-11,
2.747629453457e-22, 0.0);
  CONSTANT Out_1_Num_Zout1_1 : REAL_VECTOR := (1.00000000e+00, 3.303297540e-17, 0.0);
  CONSTANT Out_1_Den_Zout1_1 : REAL_VECTOR := (6.00620982e-02, 1.093975859e-13, 0.0);
  CONSTANT Out_1_Num_Zout1_2 : REAL_VECTOR := (1.00000000e+00, 2.262266914e-12, 0.0);
  CONSTANT Out_1_Den_Zout1_2 : REAL_VECTOR := (6.00620982e-02, 3.240974125e-13, 0.0);
  CONSTANT Out_1_Num_Zout1_3 : REAL_VECTOR := (1.000000000000e+00, 4.612814119954e-11,
3.295338945995e-22, 0.0);
  CONSTANT Out_1_Den_Zout1_3 : REAL_VECTOR := (6.006209823389e-02, 6.772627072450e-12,
2.232747889189e-22, 0.0);
  CONSTANT Out_1_OffIn11 : REAL := 0.000000e+00;
  CONSTANT Out_1_OffIn12 : REAL := 1.140241e-05;
  CONSTANT Out_1_OffOut11 : REAL := 1.770304e-05;
  CONSTANT Out_1_OffOut12 : REAL := 0.000000e+00;
  QUANTITY Vin1 ACROSS Iin1 THROUGH Vin TO ground;
  QUANTITY Vout1_1 ACROSS Iout1_1 THROUGH Iout TO ground;
  QUANTITY deltaVin1: voltage;
  QUANTITY deltaVout1_1: voltage;


BEGIN
```

```
  deltaVin1 == Vin1 - Out_1_OffIn11;
  deltaVout1_1 == Vout1_1 - Out_1_OffOut12;
  Iout1_1 ==
 deltaVin1'LTF((Out_1_Num_TF_dir1_1(0),Out_1_Num_TF_dir1_1(1)),
(Out_1_Den_TF_dir1_1(0),Out_1_Den_TF_dir1_1(1)))'LTF((Out_1_Num_TF_dir1_2(0),Out_1_Num_T
F_dir1_2(1)),
(Out_1_Den_TF_dir1_2(0),Out_1_Den_TF_dir1_2(1)))'LTF((Out_1_Num_TF_dir1_3(0),Out_1_Num_T
F_dir1_3(1),Out_1_Num_TF_dir1_3(2)),
(Out_1_Den_TF_dir1_3(0),Out_1_Den_TF_dir1_3(1),Out_1_Den_TF_dir1_3(2)))'LTF((Out_1_Num_T
F_dir1_4(0),Out_1_Num_TF_dir1_4(1),Out_1_Num_TF_dir1_4(2)),
(Out_1_Den_TF_dir1_4(0),Out_1_Den_TF_dir1_4(1),Out_1_Den_TF_dir1_4(2)))
 + deltaVout1_1'LTF((Out_1_Den_Zout1_1(0),Out_1_Den_Zout1_1(1)),
(Out_1_Num_Zout1_1(0),Out_1_Num_Zout1_1(1)))'LTF((Out_1_Den_Zout1_2(0),Out_1_Den_Zout1_2
(1)),
(Out_1_Num_Zout1_2(0),Out_1_Num_Zout1_2(1)))'LTF((Out_1_Den_Zout1_3(0),Out_1_Den_Zout1_3
(1),Out_1_Den_Zout1_3(2)),
(Out_1_Num_Zout1_3(0),Out_1_Num_Zout1_3(1),Out_1_Num_Zout1_3(2)))
 + Out_1_OffOut11;
  Iin1 == deltaVin1'LTF((Out_1_Den_Zin1_1(0),Out_1_Den_Zin1_1(1)),
(Out_1_Num_Zin1_1(0),Out_1_Num_Zin1_1(1)))'LTF((Out_1_Den_Zin1_2(0),Out_1_Den_Zin1_2(1))
,
(Out_1_Num_Zin1_2(0),Out_1_Num_Zin1_2(1)))'LTF((Out_1_Den_Zin1_3(0),Out_1_Den_Zin1_3(1),
Out_1_Den_Zin1_3(2)), (Out_1_Num_Zin1_3(0),Out_1_Num_Zin1_3(1),Out_1_Num_Zin1_3(2)))+
deltaVout1_1'LTF((Out_1_Num_TF_inv1_1(0),Out_1_Num_TF_inv1_1(1)),
(Out_1_Den_TF_inv1_1(0),Out_1_Den_TF_inv1_1(1)))'LTF((Out_1_Num_TF_inv1_2(0),Out_1_Num_T
F_inv1_2(1),Out_1_Num_TF_inv1_2(2)),
(Out_1_Den_TF_inv1_2(0),Out_1_Den_TF_inv1_2(1),Out_1_Den_TF_inv1_2(2)))+ Out_1_OffIn12;


END ARCHITECTURE behavioral;
```

The Verilog-A linear model used for simulations with the Cadence AMS Simulator in the case of the Second generation current conveyer (CCII) with acoustic resonator is also presented:

```
//-------------------------------------
// Generated by SIMECT
// Generated on: 26-Jun-2012 18:38:47
//-------------------------------------

`include "constants.vams"
`include "disciplines.vams"

module filtre_test_rlc_4res_conveyer3_1v (gnd,Vin,Iout);
input gnd;
electrical gnd;
input Vin;
electrical Vin;
output Iout;
electrical Iout;

parameter real Out_1_Num_TF_dir1_1_1  = 7.169122790853e-06 ;
parameter real Out_1_Num_TF_dir1_1_2  = 5.237279260008e-15 ;
parameter real Out_1_Num_TF_dir1_1_3  = -2.979915690365e-26 ;
parameter real Out_1_Num_TF_dir1_1_4  = 3.327631782769e-35 ;
parameter real Out_1_Num_TF_dir1_1_5  = 2.579879882589e-47 ;
parameter real Out_1_Num_TF_dir1_1_6  = -2.197545192937e-59 ;
parameter real Out_1_Num_TF_dir1_1_7  = -5.113191326721e-71 ;
parameter real Out_1_Den_TF_dir1_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Den_TF_dir1_1_2  = 1.182449777199e-10 ;
parameter real Out_1_Den_TF_dir1_1_3  = 4.972570495025e-21 ;
parameter real Out_1_Den_TF_dir1_1_4  = 4.970189001622e-31 ;
parameter real Out_1_Den_TF_dir1_1_5  = 2.004201492397e-42 ;
parameter real Out_1_Den_TF_dir1_1_6  = 4.968257835073e-54 ;
parameter real Out_1_Den_TF_dir1_1_7  = 7.697004798142e-84 ;
parameter real Out_1_Num_TF_inv1_1_1  = -1.493117711085e-07 ;
parameter real Out_1_Num_TF_inv1_1_2  = 2.943084739429e-17 ;
parameter real Out_1_Num_TF_inv1_1_3  = 4.829145939645e-28 ;
parameter real Out_1_Num_TF_inv1_1_4  = -6.476147784368e-40 ;
parameter real Out_1_Den_TF_inv1_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Den_TF_inv1_1_2  = 5.123400417199e-12 ;
```

```
parameter real Out_1_Den_TF_inv1_1_3  = 4.169859504916e-23 ;
parameter real Out_1_Den_TF_inv1_1_4  = 5.140582983458e-37 ;
parameter real Out_1_Num_Zin1_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Num_Zin1_1_2  = 4.284475717757e-11 ;
parameter real Out_1_Num_Zin1_1_3  = 3.550663188109e-22 ;
parameter real Out_1_Num_Zin1_1_4  = 8.781098115568e-34 ;
parameter real Out_1_Num_Zin1_1_5  = 5.423279188483e-59 ;
parameter real Out_1_Den_Zin1_1_1  = 2.074001437424e-04 ;
parameter real Out_1_Den_Zin1_1_2  = 5.102337159191e-14 ;
parameter real Out_1_Den_Zin1_1_3  = 1.292050726761e-24 ;
parameter real Out_1_Den_Zin1_1_4  = 6.778138824786e-36 ;
parameter real Out_1_Den_Zin1_1_5  = 8.290434413062e-48 ;
parameter real Out_1_Num_Zout1_1_1  = 1.000000000000e+00 ;
parameter real Out_1_Num_Zout1_1_2  = 4.839044114655e-11 ;
parameter real Out_1_Num_Zout1_1_3  = 4.338896607202e-22 ;
parameter real Out_1_Num_Zout1_1_4  = 7.455079594181e-34 ;
parameter real Out_1_Num_Zout1_1_5  = 2.462587263949e-50 ;
parameter real Out_1_Den_Zout1_1_1  = 2.166713552798e-04 ;
parameter real Out_1_Den_Zout1_1_2  = 2.599576573783e-14 ;
parameter real Out_1_Den_Zout1_1_3  = 9.839197396340e-25 ;
parameter real Out_1_Den_Zout1_1_4  = 6.053447358283e-36 ;
parameter real Out_1_Den_Zout1_1_5  = 7.916313592994e-48 ;
parameter real Out_1_OffIn11 = 0.000000e+00;
parameter real Out_1_OffIn12 = 1.140241e-05;
parameter real Out_1_OffOut11 = 1.770304e-05;
parameter real Out_1_OffOut12 = 0.000000e+00;

real Vin1 ;
real Iin1 ;
real Vout1_1 ;
real Iout1_1 ;
real deltaVin1 ;
real deltaVout1_1 ;

analog

begin

  Vin1 = V(Vin,gnd);
  I(Vin,gnd) <+ Iin1 ;
  Vout1_1 = V(Iout,gnd);
  I(Iout,gnd) <+ Iout1_1 ;

  deltaVin1 = Vin1 - Out_1_OffIn11;
  deltaVout1_1 = Vout1_1 - Out_1_OffOut12;

  Iout1_1 =
+laplace_nd(deltaVin1,{Out_1_Num_TF_dir1_1_1,Out_1_Num_TF_dir1_1_2,Out_1_Num_TF_dir1_1_3
,Out_1_Num_TF_dir1_1_4,Out_1_Num_TF_dir1_1_5,Out_1_Num_TF_dir1_1_6,Out_1_Num_TF_dir1_1_7
},
{Out_1_Den_TF_dir1_1_1,Out_1_Den_TF_dir1_1_2,Out_1_Den_TF_dir1_1_3,Out_1_Den_TF_dir1_1_4
,Out_1_Den_TF_dir1_1_5,Out_1_Den_TF_dir1_1_6,Out_1_Den_TF_dir1_1_7})
+laplace_nd(deltaVout1_1,{Out_1_Den_Zout1_1_1,Out_1_Den_Zout1_1_2,Out_1_Den_Zout1_1_3,Ou
t_1_Den_Zout1_1_4,Out_1_Den_Zout1_1_5},
{Out_1_Num_Zout1_1_1,Out_1_Num_Zout1_1_2,Out_1_Num_Zout1_1_3,Out_1_Num_Zout1_1_4,Out_1_N
um_Zout1_1_5}) + Out_1_OffOut11;

  Iin1 =
+laplace_nd(deltaVin1,{Out_1_Den_Zin1_1_1,Out_1_Den_Zin1_1_2,Out_1_Den_Zin1_1_3,Out_1_De
n_Zin1_1_4,Out_1_Den_Zin1_1_5},
{Out_1_Num_Zin1_1_1,Out_1_Num_Zin1_1_2,Out_1_Num_Zin1_1_3,Out_1_Num_Zin1_1_4,Out_1_Num_Z
in1_1_5})
+laplace_nd(deltaVout1_1,{Out_1_Num_TF_inv1_1_1,Out_1_Num_TF_inv1_1_2,Out_1_Num_TF_inv1_
1_3,Out_1_Num_TF_inv1_1_4},
{Out_1_Den_TF_inv1_1_1,Out_1_Den_TF_inv1_1_2,Out_1_Den_TF_inv1_1_3,Out_1_Den_TF_inv1_1_4
})+ Out_1_OffIn12;

end

endmodule
```

10. ANNEX D

In this annex we present the functions for the models using indicial responses of LWRs corresponding to Chapter 5, Subsection 5.3.5.3.

10.1. Embedded Functions and Models for the Indicial Response of LWRs

Matlab embedded Simulink function for the indicial response model:

```
function y = fcn(u,t,repind,trep,prec)
%embedded Matlab function
persistent vt vu na

if (isempty(vu)) || (t==0)
    vt=zeros(1,5000) ;
    vu=zeros(1,5000) ;
    na=1 ;
end

if (t==0)
    vt(1)=t ;
    vu(1)=u ;
    na=2 ;
elseif ((t-vt(1))>trep/100)
    vt=[t vt(1:end-1)] ;
    vu=[u vu(1:end-1)] ;
    na=na+1 ;
else
    vt(1)=t ;
    vu(1)=u ;
end

if (vt(end)==0)
  kmax=min(find(floor((t-vt)/trep)+3<=length(repind), 1, 'last' ),na) -1 ;
else
  kmax=find(floor((t-vt)/trep)+3<=length(repind), 1, 'last' ) -1 ;
end

kmax=find(floor((t-vt)/trep)+3<=length(repind), 1, 'last' );
repind=[3*repind(1)-3*repind(2)+repind(3) ; repind(1:size(repind,1)-1)] ;
repind=[2*repind(1)-repind(2) ; repind] ;
repind=[0 ; repind] ;

k=1 : kmax ;
ech=floor((t-vt(k))/trep);  % integer part of delay
dlt=(t-vt(k))/trep-ech;     % fractional part of delay
d12=(dlt-1).*(dlt-2) ;
dd1=dlt.*(dlt+1) ;
if prec
    y=sum( (vu(k)-vu(k+1)).*(-repind(1+ech)'.*(dlt).*d12/6 +
repind(2+ech)'.*(dlt+1).*d12/2 -repind(3+ech)'.*dd1.*(dlt-2)/2
+repind(4+ech)'.*(dd1).*(dlt-1)/6  ) ) ;
else
    y=sum( (vu(k)-vu(k+1)).*(repind(1+ech)'.*(1-dlt)+repind(2+ech)'.*(dlt))) ;
end
```

C embedded Simulink model for the indicial response model:

```
static void mdlOutputs(SimStruct *S, int_T tid)
{
```

```
    int_T i;
    InputRealPtrsType uPtrs = ssGetInputPortRealSignalPtrs(S,0);
    real_T *y = ssGetOutputPortRealSignal(S,0);
    int_T width = ssGetOutputPortWidth(S,0);

    /* user variables */

    static real_T trep = 0.05;
    static int_T na;
    static real_T repind[500]; /* contains the auto written indicial response */
    static real_T vt[5000];
    static real_T vu[5000];
    static int_T size_vt = 5000;
    static int_T size_repind = 500;
    static int_T floor_prec = 1;
    static int_T prec = 1;

    real_T crt_time;
    real_T aux_val;
    int_T idx_max;
    int_T kmax;
    int_T found_idx;
    int_T A_aux;
    real_T out_calc;
    int_T ech;
    real_T d12;
    real_T dd1;
    real_T dlt;

    crt_time=ssGetTaskTime(S, 0);

    if (crt_time==0.0)
    {
            for (i=0; i<size_vt; i++) {
                 vt[i] = 0;
                 vu[i] = 0;
            }
    na=2;
    }
    else if ((crt_time - vt[0])>trep/100.00)
    {
        for (i=size_vt-1; i>=1; i--) {
            vt[i] = vt[i-1];
            vu[i] = vu[i-1];
        }
        na = na + 1;
    }

    vt[0]=crt_time;
    vu[0]=*uPtrs[0];
    idx_max = 0;
    found_idx = 0;

    for (i=size_vt-1; i>=0; i--) {
        A_aux=(((crt_time-vt[i])/trep)+3.0)*floor_prec;
        if ((A_aux<=size_repind*floor_prec)&&(found_idx==0))
        {
                    idx_max = i;
                    found_idx = 1;
        }
     }

    if (vt[size_vt-1]==0.0)
    {
        if (idx_max<=na)
        {
            kmax = idx_max-1;
        }
        else
        {
            kmax = na-1;
        }
    }
    else
    {
        kmax = idx_max-1;
    }
```

```
    aux_val = repind[0];
    repind[0] = 3.0*aux_val-3.0*repind[1]+repind[2];
    for (i=size_repind-1; i>=1; i--) {
            repind[i] = repind[i-1];
        }

    out_calc=0.0;

    for (i=0; i<kmax; i++) {
        ech=((crt_time-vt[i])/trep);
        dlt=(crt_time-vt[i])/trep-ech;
        d12=(dlt-1.0)*(dlt-2.0);
        dd1=dlt*(dlt+1.0);

        if (prec==1)
        {
            out_calc = out_calc + ((vu[i]-vu[i+1])*(-repind[ech]*(dlt)*d12/6.0 +
repind[1+ech]*(dlt+1.0)*d12/2.0 -repind[2+ech]*dd1*(dlt-
2.0)/2.0+repind[3+ech]*(dd1)*(dlt-1.0)/6.0  ) );
        }
        else
        {
            out_calc = out_calc + ((vu[i]-vu[i+1])*(repind[ech]*(1.0-
dlt)+repind[1+ech]*(dlt)));
        }
    }
    *y = out_calc;
}
```

VHDL-AMS model for the indicial response model:

```
library ieee;
use ieee.electrical_systems.all;
use ieee.math_real.all;

entity freq_model is

  generic(trep : real := 0.05;
          prec : natural := 1;
          Zout : real := 50 );
  port(terminal in_a, out_a : electrical;
       signal clk : in bit := '0');

end entity freq_model;

architecture behav of freq_model is

constant size_repind : positive := 500;
type repind_vector is array (1 to size_repind) of real;

constant size_vt : positive := 5000;
type vt_vector is array (1 to size_vt) of real;

signal v_aux : real := 0.0;

quantity Vin1 ACROSS Iin1 THROUGH in_a TO ground;
quantity Vout1 ACROSS Iout1 THROUGH out_a TO ground;

begin
  Iin1 == 0.0;
  compute_out : process is

    variable repind : repind_vector; --contains the auto-written indicial response
    variable aux_val : real := 0.0;
    variable na : natural := 0;
    variable vt : vt_vector;
    variable vu : vt_vector;
    variable idx_max : natural;
    variable found_idx : positive;
    variable kmax : natural := 0;
    variable ech : positive;
    variable dlt : real := 0.0;
    variable d12 : real := 0.0;
    variable dd1 : real := 0.0;
    variable y : real := 0.0;
```

```
  begin
    loop
        wait until clk = '1';
        if now = 0.0 then
            for i in 1 to size_vt loop
                vt(i) := 0.0;
                vu(i) := 0.0;
            end loop;
            na := 2;
        else if ((now-vt(1))>trep/100.00) then
                for i in size_vt downto 2 loop
                      vt(i) := vt(i-1);
                      vu(i) := vu(i-1);
                end loop;
                na := na + 1;
             end if;
        end if;

        vt(1) := now;
        vu(1) := Vin1;

        idx_max := 0;
        found_idx := 0;
        for i in size_vt downto 1 loop
            if ((floor((now-vt(i))/trep)+3.0)<=real(size_repind)) and (found_idx=0) then
                   idx_max := i;
                   found_idx := 1;
            end if;
        end loop;

        if vt(size_vt) = 0.0 then
            if idx_max<=na then
                kmax := idx_max-1;
            else
                kmax := na-1;
            end if;
        else
            kmax := idx_max-1;
        end if;

        aux_val := repind(1);
        repind(1) := 3.0*aux_val-3.0*repind(2)+repind(3);

        for i in size_repind downto 2 loop
            repind(i) := repind(i-1);
        end loop;

        y:=0.0;
        for k in 1 to kmax loop
            ech:=integer(floor((now-vt(k))/trep));
            dlt:=(now-vt(k))/trep-real(ech);
            d12:=(dlt-1.0)*(dlt-2.0);
            dd1:=dlt*(dlt+1.0);

            if prec=1 then
               y:=y+((vu(k)-vu(k+1))*(-repind(1+ech)*(dlt)*d12/6.0 +
repind(2+ech)*(dlt+1.0)*d12/2.0 -repind(3+ech)*dd1*(dlt-
2.0)/2.0+repind(4+ech)*(dd1)*(dlt-1.0)/6.0  ) );
            else
               y:=y+((vu(k)-vu(k+1))*(repind(1+ech)*(1.0-dlt)+repind(2+ech)*(dlt)));
            end if;
        end loop;
        v_aux <= y;
     end loop;
  end process compute_out;
  Iout1 ==  v_aux/Zout;

end architecture behav;
```

11. ANNEX E

We present here the VHDL-AMS functions for the models of ADCs and DACs corresponding to Chapter 5, Subsection 5.3.7.

11.1. ADC and DAC HDL Models

The ADC model in VHDL-AMS:

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.electrical_systems.all;

entity adc is

    generic (MIN : real := -1.0; -- minimum limit for ADC
             MAX : real := 1.0;  -- maximum limit for ADC
             DC_OUT_FILT : real := 2.5; --TIA4 DC voltage on output
             NBITS : positive := 3
             );

    port (terminal IN_ADC : electrical;
          signal CLK : in bit;
          signal VAL : out bit_vector (NBITS-1 downto 0) := (others => '0');
          signal THERM : out bit_vector (2**NBITS-2 downto 0) := (others => '0')
          );

end entity adc;


architecture behav of adc is

quantity Vin ACROSS Iin THROUGH IN_ADC TO ground;

begin

    Iin == 0.0; --ideal voltage sink

    process

    variable delta_v : real := 0.0;
    variable input_hold : real := 0.0;
    variable count : integer := 0;


        begin

        wait on CLK;

        delta_v := MAX - MIN;

        input_hold := Vin - DC_OUT_FILT - MIN;

        if (CLK'event and CLK = '1') then
            count:=0;

            --bits and count
            for i in NBITS-1 downto 0 loop

                delta_v := delta_v / 2.0;

                if input_hold >= delta_v then
                    VAL(i) <= '1';
                    input_hold := input_hold - delta_v;
```

```
                    count := count + 2**i;
                else
                    VAL(i) <= '0';
                end if;

            end loop;

            --thermometric
            for i in 1 to 2**NBITS-1 loop

                if (i<=count) then
                THERM(i-1) <= '1';
                else
                THERM(i-1) <= '0';
                end if;

            end loop;

        end if;

    end process;

end architecture;
```

The DAC model in VHDL-AMS:

```
library ieee;
library std;

use ieee.std_logic_1164.all;
use ieee.electrical_systems.all;
use std.standard.all;
use ieee.math_real.all;


entity dac is

    generic (
             NBITS : positive := 3;-- number of bits
             TT : REAL := 4.0e-12; -- transition time
             PR_T : time := 1.00ns; -- loop delay = 0.4 Ts DAC + 1Ts introduced by ADC +
0.1 Ts introduced by analog filter
             REF_I_IDEAL : REAL := 230.00e-6;--230.00e-6; --ideal max reference current
= 200 val max + 1 supplementary interval
             REF_I_OFFSET : REAL := 100.00e-6 --ideal offset for current
             );

    port (signal CLK : in bit;
          signal DIGITAL_IN : in bit_vector(NBITS-1 downto 0);
          terminal ANALOG_OUT_P : electrical;
          terminal ANALOG_OUT_M : electrical
         );

end entity dac;


architecture behav of dac is

quantity Voutp ACROSS Ioutp THROUGH ANALOG_OUT_P to ground;
quantity Voutm ACROSS Ioutm THROUGH ANALOG_OUT_M to ground;


signal AOUT,REFV : real:=0.0;


begin


    process is


    function d_to_a(input:bit_vector(NBITS-1 downto 0);max_val:real) return real is

    variable s_out: real:=0.0;
    variable sum:real;
```

```
    begin

        sum:=0.0;

        for i in 0 to NBITS-1 loop

            if input(i)='1' then sum:=sum+max_val/real(2**(NBITS-i));
            else sum:=sum;
            end if;

        end loop;

    s_out:=sum;

    return s_out;

    end d_to_a;


    begin --process begin

    wait on CLK;

    REFV<=REF_I_IDEAL;

    if(CLK'event and CLK='1')then
        AOUT<=d_to_a(DIGITAL_IN,REFV) after PR_T; --conversion
    end if;

   end process;

Ioutp==-AOUT'ramp(TT)+REF_I_OFFSET; --current output +
Ioutm==AOUT'ramp(TT)-REF_I_OFFSET; --current output -

end architecture behav;
```

References

12. Bibliography

1] Cadence Design Systems, "20 Questions on 20nm," Cadence Design Systems, San Jose, CA, USA, 2012.

2] NVIDIA, "Whitepaper NVIDIA's Next Generation CUDA Compute Architecture: Kepler GK110," NVIDIA, Santa Clara, CA, 2012.

3] Intel Corporation, "Intel Xeon Processor E7 Specification," Intel Corporation, Santa Clara, CA, USA, 2011.

4] M. ROBERT, "CAO de Circuits Intégrés Numériques," LIRMM Montpellier, Montpellier, France, 2010.

5] IEEE ISSCC, "Systems-on-a-chip," in *IEEE International Solid-State Circuits Conference*, San Francisco, CA, USA, 1996.

6] B. Pakkenberg and H. J. G. Gundersen, "Neocortical neuron number in humans: Effect of sex and age," *The Journal of Comparative Neurology,* vol. 384, no. 2, p. 312–320, 1998.

7] G. Gielen, "Design methodologies and tools for circuit design in CMOS nanometer technologies," in *The 32nd European Solid-State Circuits Conference*, Bordeaux, France, 2006.

8] Greg Taylor, Intel Corporation, "Future of Analog Design and Upcoming Challenges in Nanometer CMOS," Intel Corporation, Santa Clara, CA, USA, 2010.

9] K. Kundert and H. Chang, "Model-based functional verification," in *47th ACM/IEEE Design Automation Conference (DAC)*, San Jose, CA, USA, 2010.

R. Rutenbar and G. Gielen, "Computer-Aided Design of Analog and Mixed-

10] Signal Integrated Circuits," *Proceedings of the IEEE,* vol. 88, no. 12, pp. 1825-1852, 2000.

Labrecque, C.; SYNOPSYS, "Near-Term Industrial Perspective of Analog CAD,"
11] IEEE ICCAD'06, San Jose, CA, USA, 2006.

G. Gielen, "Design tool solutions for mixed-signal/RF circuit design in CMOS
12] nanometer technologies," in *Asia and South Pacific Design Automation Conference*, Yokohama, 2007.

ARTEMOS, "Agile RF Transceivers and Front-Ends for Future Smart Multi-
13] Standard COmmunications ApplicationS," ARTEMOS, 1 April 2011. [Online]. Available: https://artemos.eu/.

R. Schreier and B. Zhang, "Delta-Sigma modulators employing continuous-time
14] circuitry," *IEEE Transactions on Circuit and Systems-I: Fundamental Theory and Applications,* vol. 43, pp. 324-332, April 1996.

Y. Dupret, "Modelisation des dispersions des performances des cellules
15] analogiques integrees en fonction des dispersions du processus de fabrication," UNIVERSITE PARIS XI, Orsay, France, 2005.

H. Inose and Y. Yasuda, "A unity bit coding method by negative feedback,"
16] *Proceedings of the IEEE,* vol. 51, no. 11, pp. 1524 - 1535, 1963.

J. Candy, "A Use of Double Integration in Sigma Delta Modulation," *IEEE*
17] *Transactions on Communications,* vol. 33, no. 3, pp. 249 - 258, 1985.

M. W. Hauser, "Principles of Oversampling A/D Conversion," *Journal Audio*
18] *Engineering Society,* vol. 39, no. 1/2, 1991.

M. Ortmanns and F. Gerfers, Continuous-Time Sigma-Delta A/D Conversion,
19] Berlin: Springer, 2006.

W. Benett, "Spectra of Quantized Signals," *Bell System Technology Journal,* vol.
20] 27, no. 1, pp. 446-472, 1948.

B. Widrow, "A study of rough amplitude quantization by means of Nyquist
21] sampling theory," *IRE Circuit Theory,* vol. CT3, pp. 266-276, 1956.

N. Ahmed and T. Natarajan, Discrete-Time Signals and Systems, Englewood
22] Cliffs, NJ, USA: Prentice-Hall, 1983.

P. Benabes, "Etude de Nouvelles Structures de Modulateurs Sigma-Delta Passe-
23] Bande," Universite de Paris XI, Orsay, FRANCE, 1994.

M. Vidyasagar, Nonlinear Systems Analysis, Englewood Cliffs: Prentice-Hall,
24] 1978.

P.W.Wong, "Quantization noise, fixed-point multiplicative roundoff noise, and
25] dithering," *IEEE Transactions on Signal Processing,* vol. 38, no. 2, pp. 286-300, 1990.

S. Tewksbury and R. Hallock, "Oversampled linear predictive and noise shaping
26] coders of order n>1," *IEEE Transactions on Circuits & Systems,* vol. 7, pp. 436-447, 1978.

R. Gray, "Oversampled Sigma-delta Modulation," *IEEE Transactions on*
27] *Communications,* vol. 35, no. 05, pp. 481-488, 1987.

M. Javidan, "Design of high-order sigma-delta modulators for parallel analog-to-
28] digital converters," SUPELEC, Gif-sur-Yvette, 2009.

A. Beydoun, "Systeme de numerisation hautes-performances a base de
29] modulateurs Sigma-Delta passe-bande," UNIVERSITE PARIS XI ORSAY, Orsay, France, 2008.

B. Boser and B. Wooley, "The design of sigma-delta modulation analog-to-digital
30] converters," *IEEE Journal of Solid-State Circuits,* Vols. SC-23, pp. 1298-1308, 1988.

R. Schreier and M. Snelgrove, "Bandpass sigma-delta modulation," *Electronics*
31] *Letters ,* vol. 25, no. 23, pp. 1560-1561, 1989.

H. Tang and A. Doboli, "High-Level Synthesis of ΔΣ Modulator Topologies
32] Optimized for Complexity, Sensitivity, and Power Consumption," in *IEEE ISCAS'06*, Island of Kos, Greece, 2006.

P. BENABES, P. ALDEBERT and R. KIELBASA, "A Matlab based tool for
33] bandpass continuous-time sigma-delta modulators design," in *IEEE International Symposium on Circuits and Systems - ISCAS'98*, Monterey, CA, USA, 1998.

R. Schreier and G. Temes, Delta Sigma Data Converters, Piscataway, New
34] Jersey: Wiley, 2005.

A. Gosslau and A. Gottwald, " Linearization of Sigma-Delta Modulator by a
35] Proper Loop Delay," in *IEEE International Symposium on Circuits and Systems*, New Orleans, USA, 1990.

A. Gosslau and A. Gottwald, "Optimization of a Sigma-Delta modulator by the
36] use of a slow ADC," in *IEEE International Symposium on Circuits and Systems*, Helsinki, 1988.

A. Yahia, "Contribution a la Conception Automatise de Convertisseurs
37] Analogique-Numerique Sigma-Delta Passe-Bande Rapides," Universite Paris XI, Orsay, France, 2001.

E. Najafi Aghdam, "NOUVELLES TECHNIQUES D'APPARIEMENT

38] DYNAMIQUE DANS UN CNA MULTIBIT POUR LES CONVERTISSEURS SIGMA-DELTA," Universite Paris XI, Orsay, France, 2006.

39] P. Benabes and M. Javidan, "VersaNUM - Architecture de modulateur sigma delta passe-bande," SUPELEC, Gif-sur-Yvette, France, 2009.

40] P. Benabes and S. Guessab, "Passive Sigma-Delta modulators - Transistor level validation," SUPELEC - Texas Instruments, Gif-sur-Yvette, France, 2003.

41] S. Benabid, "Contribution a la Conception d'un Convertisseur Sigma-Delta Passe-Bande a Temps Continu dans une Technologie Standard CMOS," Universite Paris-Sud XI, Orsay, 2005.

42] E. Avignon, "Contribution a conception d'un modulateur Sigma-Delta Passe-Bande a temps continu pour la conversion directe de signaux radiofrequences," UNIVERSITE PIERRE ET MARIE CURIE, Paris, France, 2007.

43] J. Huijsing, W. Sansen and R. Van der Plassche, Analog Circuit Design, Boston: Kluwer Academic Publishers, 1999.

44] G. Gielen, "Modeling and analysis techniques for system-level architectural design of telecom frontends," *IEEE Transactions on Microwave Theory and Techniques,* vol. 50, no. 1, pp. 360-368, 2002.

45] R. Peruzzi, "Efficient Verification and Virtual Prototyping of Analog and Mixed-Signal IP and SOCs Using Behavioral Models," R.Peruzzi Consulting, Inc., Pennsylvania, USA, 2005.

46] V. Ferragina, "Design of Digital Blocks for CMOS Mixed-signal Integrated Circuits," Universita degli studi di Pavia, Pavia, Italy, 2006.

47] D. Gajski and R. Kuhn, "New VLSI tools," *Computer,* vol. 16, no. 12, pp. 11-14, 1983.

48] H. Bousetta, "Modélisation multi-physiques et simulation globale de systèmes autonomes sur puce," Institut Polytechnique de Grenoble, Grenoble, France, 2010.

49] T. McConaghy, T. Eeckelaert and G. Gielen, " CAFFEINE: template-free symbolic model generation of analog circuits via canonical form functions and genetic programming," in *IEEE Design Automation and Test in Europe conference (DATE 2005)*, Munich, Germany, 2005.

50] W. Daems, G. Gielen and W. Sansen, "Simulation-based generation of posynomial performance models for the sizing of analog integrated circuits," *IEEE Transactions on Computer-Aided Design,* vol. 22, no. 5, pp. 517-534, 2003.

R. J. Duffin, E. L. Peterson and C. Zener, Geometric Programming, New York:

51] John Wiley and Sons, 1967.

52] A. Zaabab, Q.-J. Zhang and M. Nakhla, "Device and circuit-level modeling using neural networks with faster training based on network sparsity," *IEEE Transactions on Microwave Theory and Techniques,* vol. 45, no. 10, pp. 1696 - 1704, 1997.

53] S. Doboli, G. Gothoskar and A. Doboli, "Extraction of piecewise-linear analog circuit models from trained neural networks using hidden neuron clustering," in *Design, Automation and Test in Europe Conference*, Munich, Germany, 2003.

54] C. Borchers, L. Hedrich and E. Barke, "Equation-based behavioral model generation for nonlinear analog circuits," in *IEEE/ACM Design Automation Conference (DAC'96)*, Las Vegas, Nevada, USA, 1996.

55] H. Liu, A. Singhee, R. Rutenbar and L. Carley, "Remembrance of circuits past: macromodeling by data mining in large analog design spaces," in *Proceedings of the DAC Conference*, New Orleans, LA, USA, 2002.

56] J. Phillips, J. Afonso, A. Oliveira and L. Silveira, "Analog macromodeling using kernel methods," in *IEEE/ACM International Conference on Computer Aided Design*, San Jose, CA, 2003.

57] T. Kiely and G. Gielen, "Performance modeling of analog integrated circuits using least-squares support vector machines," in *Design, Automation and Test in Europe Conference (DATE 2004)*, Paris, France, 2004.

58] H. Li, M. Mansour, S. Maturi and L.-C. Wang, "A new sampling method for analog behavioral modeling," in *IEEE NEWCAS 2010*, Paris, France, 2010.

59] Olaf Zinke, Cadence Design Systems Inc., "Mixed-signal simulation in design, verification," EE Times-Asia, 2005.

60] M. Javidan and P. Benabes, "Design of electronic control circuit of piezo-electric resonators for ΣΔ modulator loop in AMS Bi-CMOS 0.35µm," in *8th IEEE International NEWCAS Conference*, Montréal, Canada, 2010.

61] S. Ouzounov, R. van Veldhoven, C. Bastiaansen, K. Vongehr, R. van Wegberg, G. Geelen, L. Breems and A. van Roermund, "A 1.2V 121-Mode CT Sigma-Delta Modulator for Wireless Receivers in 90nm CMOS," in *IEEE International Solid-State Circuits Conference*, San Francisco, CA, 2007.

62] A. Yahia and P. Benabes, "Bandpass Sigma-Delta Modulators Synthesis with High Loop Delay," in *IEEE International Symposium on Circuits and Systems*, Sydney, Australia, May 2001.

A. Maxim, D. Andreu, M. Cousineau and J. Boucher, "A novel SPICE behavioral

63] macromodel of operational amplifiers including a high accuracy description of frequency characteristics," in *IEEE International Symposium on Circuits and Systems*, Orlando, Florida, USA, July 1999.

64] M. Hui Li Mansour, S. Maturi and L.-C. Wang, "A new Sampling Method for Analog Behavioral Modeling," in *IEEE International Symposium on Circuits and Systems*, Paris, France, 2010.

65] M. El-Nozahi and Y. Massoud, "An Integrated Circuit/Behavioral Simulation Framework For Continuous-time Sigma-Delta ADCs," in *ACM 2006*, Philadelphia, PA, USA, 2006.

66] E. Martens and G. Gielen, "Analyzing Continuous-Time SD Modulators With Generic Behavioral Models," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 25, pp. 924-932, 2006.

67] M. Vasilevski, H. Aboushady and M. Louerat, "Automatic Model Refinement of GmC Integrators for High-Level Simulation of Continuous-Time Sigma-Delta Modulators," in *IEEE International Symposium on Circuits and Systems*, Taipei, Taiwan, May 2009.

68] J. Ruiz-Amaya, J. De la Rosa, F. Medeiro, F. Fernandez, R. Del Rio, B. Perez-Verdu and A. Rodriguez-Vazquez, "MATLAB / SIMULINK Based High-Level Synthesis Of Discrete-Time And Continuous-Time SD Modulators," in *Design, Automation and Test in Europe Conference and Exhibition*, Paris, France, 2004.

69] N. Chandra and G. Roberts, "Top-down analog design methodology using Matlab and Simulink," in *IEEE International Symposium on Circuits and Systems*, Sydney, Australia, 2001.

70] P. Benabes and C.-A. Tugui, "A high-level modeling framework for the design and optimization of complex CT functions," in *NEWCAS 2011*, Bordeaux, France, Jun 2011.

71] Plexim, "Plecs Blockset," 2011. [Online]. Available: http://www.plexim.com/products/plecs_blockset.html.

72] Cadence Design Systems Inc., "SLPS - Simulink to PSpice link," 26 June 2011. [Online]. Available: http://www.cadence.com/rl/Resources/datasheets/pcb_pspice_simulation_ds.pdf.

73] Cadence Design Systems, Inc., "OCEAN Reference," Cadence Design Systems, Inc., San Jose CA, USA, 2004.

74] T. D. F. Cambois, "Développement des solveurs à pas variables en c pour la simulation," Journée française Club utilisateur Matalab, France, 2002.

L. F. Shampine, "Numerical Solution of Ordinary Differential Equations," 75] Chapman & Hall, New York, 1994.

M. Rivoire and J. Ferrier, Matlab, Simulink, Stateflow : avec des exercices 76] d'automatique résolus, France: Editions Technip, 2001.

D. Gibson and C. Purdy., "Extracting behavioral data from physical descriptions 77] of MEMS for simulation," *The Journal of VLSI Signal Processing,* vol. 22, no. 2, p. 135–146, 1999.

A. Vachoux, Modélisation de Systèmes Intégrés Analogiques et Mixtes - 78] Introduction à VHDL-AMS, Laussane: Laboratoire de Systèmes Microélectroniques, EPFL, 2002.

K. GHERFI, "Modelisation VHDL-AMS et application a l'integration de 79] puissance," UNIVERSITE MENTOURI-CONSTANTINE - FACULTE DES SCIENCES DE L'INGENIEUR , Constantine, 2005.

P. Ashenden, G. Peterson and D. Teegarden, The System Designer's Guide to 80] VHDL-AMS - Analog, Mixed-Signal, and Mixed-Technology Modeling, San Francisco, CA: Morgan Kaufmann Publishers, 2003.

T. Kazmierski, *Analogue and mixed-signal behavioural synthesis from VHDL-* 81] *AMS,* Toulouse, France: Departement of Electronic and Computer Science, University of Southampton, UK, 2005.

L. Wang, C. Zhao and T. Kazmierski, "An Extension to VHDL-AMS for AMS 82] Systems with Partial Differential Equations," in *Forum on specification and Design Languages*, Barcelona, Spain, 2007.

Cadence Design Systems, "Verilog-A Laplace Transform S-Domain Filters 83] Application Note," Cadence Design Systems, Inc., San Jose, CA, 2005.

Accellera, "SystemC standard," 14 03 2007. [Online]. Available: 84] http://www.accellera.org/downloads/standards/systemc.

Accellera, "SystemC-AMS Standard," 05 03 2010. [Online]. Available: 85] http://www.accellera.org/downloads/standards/systemc/ams.

A. Vachoux, C. Grimm and K. Einwich, "Extending SystemC to support mixed 86] discrete-continuous system modeling and simulation," in *ISCAS*, Kobe, Japan, 2005.

J. Dennis and R. Schnabel, Numerical Methods for Unconstrained Optimization 87] and Nonlinear Equations, NJ: Prentice-Hall: Englewood Cliffs, 1983.

P. Benabes and C.-A. Tugui, "Effective Modeling of CT Functions for Fast 88] Simulations Using MATLAB-Simulink and VHDL-AMS Applied to Sigma-Delta

Architectures," in *IEEE International Symposium on Circuits and Systems (ISCAS 2011)*, Rio de Janeiro, Brasil, 2011.

89] C. Tugui, "High performance converters bank using parallel bandpass Sigma-Delta modulators: transistor level simulation of a Sigma-Delta modulator using lamb waves," SUPELEC, Gif-sur-Yvette, France, 2009.

90] K. Kundert, "Introduction to RF Simulation and its Application," *IEEE Journal of Solid-State Circuits,* vol. 34, no. 9, 1999.

91] M. Ma and R. Khazaka, "Nonlinear macromodeling using model order reduction," in *IEEE 14th Topical Meeting on Electrical Performance of Electronic Packaging*, Park City, 2005.

92] C. W. Ho, A. E. Ruehli and P. A. Brennan, "The modified nodal approach to network analysis," *IEEE Transactions on Circuits and Systems,* vol. 22, no. 6, p. 504–509, Jun 1975.

93] M. Ma and R. Khazaka, "Sparse Macromodeling for Parametric Nonlinear Networks," *IEEE Transactions on Microwave Theory and Techniques,* vol. 54, no. 12, p. 4305–4315, Dec 2006.

94] M. Kanaan and R. Khazaka, "Model Order Reduction for Nonlinear Macromodeling of RF Circuits," in *9th IEEE International NEWCAS Conference (NEWCAS 2011)*, Bordeaux, France, Jun 2011.

95] Cadence Design Systems, Inc., *Virtuoso (C) SpectreRF Simulation Option Theory,* San Jose, CA: Cadence Design Systems, Inc., 2005.

96] Cadence Design Systems, Inc., *Spectre RF Analyses,* San Jose, CA: Cadence Design Systems, Inc., 2002.

97] Cadence Design Systems, Inc., *Spectre/RF Matlab Toolbox Application Note,* San Jose, CA: Cadence Design Systems, Inc., 2006.

98] Cadence Design Systems, Inc., *SpectreRF Workshop - Using the SpectreRF MATLAB Toolbox,* San Jose, CA: Cadence Design Systems, Inc., 2006.

99] P. Benabes and C.-A. Tugui, "A high-level modeling framework for the design and optimization of complex CT functions," in *NEWCAS 2011*, Bordeaux, France, Jun 2011.

100] B. Calvo, S. Celma, P. Martinez and M. Sanz, "High-Speed High-Precision CMOS Current Conveyor," *Analog Integrated Circuits and Signal Processing,* vol. 34, no. 1, pp. 265-269, 2003.

B. Calvo, S. Celma, P. Martinez and M. Sanz, "Novel High Performance CMOS

101] Current Conveyor," in *9th International Conference on Mixed Design (MIXDES 2002)*, Wroclaw, Poland, 2002.

G. Dantzig and M. Thapa, Linear programming, 1: Introduction, Springer, 1997,
102] pp. 1-8.

D. H. Wolpert and W. Macready, "No Free Lunch Theorems for Optimization,"
103] *IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION,* vol. 1, no. 1, pp. 67-82, 1997.

M. F. M. Barros, J. M. C. Guilherme and N. C. G. Horta, "Analog Circuits and
104] Systems Optimization based on Evolutionary Computation Techniques," in *Studies in Computational Intelligence*, vol. 294, Springer, 2010, pp. 51-52.

V. Aggarwal, "Analog Circuit Optimization using Evolutionary Algorithms and
105] Convex Optimization," MASSACHUSETTS INSTITUTE OF TECHNOLOGY, 2007.

L. O. Chua and P.-L. Min, Computer-aided Analysis of Electronic Circuits:
106] Algorithms and Computational Techniques, New Jersey, Englewood Cliffs: Prentice-Hall, 1975.

A. Das, "On the transistor sizing problem," in *IEEE Thirteenth International*
107] *Conference on VLSI Design* , Calcutta, India, 2000.

A. Khawas, A. Banerjee and S. Mukhopadhyay, "A Response Surface Method for
108] Design Space Exploration and Optimization of Analog Circuits," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, Chennai, India, 2011.

X. Li, P. Gopalakrishnan, Y. Xu and T. Pileggi, "Robust analog/RF circuit design
109] with projection-based posynomial modeling," in *IEEE/ACM International Conference on Computer Aided Design (ICCAD-2004)*, San Jose, CA, USA, 2004.

S. Apostol, "Automatic optimization methods for high performance circuits:
110] applications for a continuous time Sigma-Delta architecture," Gif-sur-Yvette, 2011.

J. Kleijnen, Design and analysis of simulation experiments, New York: Springer,
111] 2008.

S. Yesilyurt and A. Patera, "Surrogates for numerical simulations; optimization of
112] eddy-promoter heat exchangers," *Computer Methods in Applied Mechanics and Engineering,* vol. 121, pp. 231-257, 1995.

R. R. Barton, "Simulation optimization using metamodels," in *Proceedings of the*
113] *2009 Winter Simulation Conference*, USA, 2009.

T. Kohonen, "Self-Organized Formation of Topologically Correct Feature Maps,"
114] *Biological Cybernetics,* vol. 43, no. 2, pp. 59-69, 1982.

I. Nastac, "An adaptive forecasting intelligent model for nonstationary time
115] series," *Journal of Applied Operational Research,* vol. 2, no. 2, pp. 117-129, 2010.

S. Mehrotra, P. Franzon and W. Liu, "Stochastic Optimization Approach to
116] Transistor Sizing for CMOS VLSI Circuits," in *31st Conference on Design Automation*, 2004.

R. Rogenmoser, H. Kaeslin and T. Blickle, "Stochastic Methods for Transistor
117] Size Optimization of CMOS VLSI Circuits," in *Proceedings of PPSN IV Conference*, London, 1996.

M. J. Sasena, P. Papalambros and P. Goovaerts., "Exploration of metamodeling
118] sampling criteria for constrained global optimization," *Engineering Optimization,* vol. 34, pp. 263-278, 2002.

W. E. Biles, J. P. C. Kleijnen, W. C. Van Beers and I. Nieuwenhuyse, "Kriging
119] metamodeling in constrained simulation optimization: an explorative study," in *Proceedings of the 2007 Winter Simulation Conference*, New Jersey, 2007.

H. You, M. Yang, D. Wang and X. Jia, "Kriging Model combined with latin
120] hypercube sampling for surrogate modeling of analog integrated circuit performance," in *International Symposium on Quality Electronic Design 2009*, 2009.

J. A. Egea, "New heuristics for global optimization of complex bioprocesses,"
121] University of Vigo, Vigo, 2008.

D. Krige, "A statistical approach to some mine valuations and allied problems at
122] the Witwatersrand," University of Witwatersrand, Witwatersrand, 1951.

S. Lophaven, H. Nielsen and J. Sondergaard, "DACE – A MATLAB Kriging
123] Toolbox," Techical University of Denmark, Copenhagen, 2002.

J. Sacks, W. Welch, T. Mitchell and H. Wynn, "Design and Analysis of Computer
124] Experiments," in *Statistical Science*, vol. 4, 1989, pp. 409-435.

E. Isaaks and R. Srivastava, An Introduction to Applied Geostatistics, New York,
125] USA: Oxford University Press, 1989.

J. C. Spall, Introduction to Stochastic Search and Optimization, Wiley, 2003.
126]

H. Holger Hoos and T. Stützle, Stochastic Local Search: Foundations and
127] Applications, Morgan Kaufmann / Elsevier, 2004.

D. Jones, M. Schonlau and W. Welch, "Efficient global optimization of expensive
128] black-box functions," *Journal of Global Optimization,* no. 13, pp. 455-492, 1998.

R. Benassi, J. Bect and E. Vazquez, "Robust Gaussian process-based global

129] optimization using a fully Bayesian expected improvement criterion," in *International Conference on Learning and Inteligent Optimization (LION 5)*, Rome, Italy, 2011.

M. Bernardo, R. Buck, L. Liu, W. Nazaret, J. Sacks and W. Welch, "Integrated
130] circuit design optimization using a sequential strategy," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 3, no. 1, p. 361–372, 1992.

J. Mockus, V. Tiesis and A. Zilinskas, "The application of Bayesian methods for
131] seeking the extremum," in *Towards Global Optimization*, New York, North Holland, 1978, p. 117–129.

M. Schonlau, "Computer Experiments and Global Optimization," University of
132] Waterloo, 1997.

D. R. Jones, "A taxonomy of global optimization methods based on response
133] surfaces," *Journal of Global Optimization,* vol. 4, no. 21, pp. 345-383, 2001.

P. Benabes and C.-A. Tugui, "SIMECT - A tool for SImulation and Macromodel
134] Extraction of analog CT functions," 2011. [Online]. Available: www.mathworks.com/matlabcentral/fileexchange/26525.

M. McKay, R. Beckman and W. Conover, "A Comparison of Three Methods for
135] Selecting Values of Input Variables in the Analysis of Output from a Computer Code," *Technometrics (American Statistical Association),* vol. 21, no. 2, p. 239–245, 1979.

R. Iman, J. Helton and J. and Campbell, "An approach to sensitivity analysis of
136] computer models, Part 1. Introduction, input variable selection and preliminary variable assessment," *Journal of Quality Technology,* vol. 13, no. 3, p. 174–183, 1981.

M. Javidan and P. Benabes, " Band-Pass Continuous-Time Delta-Sigma
137] Modulators Employing LWR Resonators," in *IEEE International Conference on Electronics, Circuits, and Systems (ICECS'08)*, Malta, 2008.

C.-A. TUGUI, R. BENASSI, S. APOSTOL and P. BENABES, "Efficient
138] Optimization Methodology for CT Functions Based on a Modified Bayesian Kriging Approach," in *ICECS 2012*, Seville, Spain, 2012.

Coventor, "CoventorWare," 2012. [Online]. Available:
139] http://www.coventor.com/products/coventorware/.

M. Desvergne, P. Vincent, Y. Deval and J.-B. Begueret, "RF lamb wave
140] resonators in bandpass delta-sigma converters for digital receiver architectures," in *IEEE Northeast Workshop on Circuits and Systems 2007 (NEWCAS 2007)*, Montreal, 2007.

J. Arcamone, "Integration of Nanomechanical Sensors on CMOS by
141] Nanopatterning Methods," Universitat Autònoma de Barcelona, Barcelona, 2007.

M.-A. Dubois, "Thin film bulk acoustic wave resonators: a technology," in
142] *MEMSWAVE 03*, Toulouse, France, 2003.

K. M. Lakin, "Thin Film Resonators and Filters I," TFR Technologies, Inc., Bend,
143] 2002.

M. Desvergne, E. Defay, D. Wolozan, M. Aid, P. Vincent, A. Volatier, Y. Deval
144] and J.-B. Begueret, "Intermediate frequency lamb wave coupled resonator filters for RF receiver architectures," in *37th European Solid State Device Research Conference, ESSDERC 2007.* , Munich, 2007.

CEA - Leti - List, "Architecture & IC Design, Embedded Software: Annual
145] Research Report 2010," CEA - Leti - List, Grenoble & Gif-sur-Yvette, 2010.

A. Shirakawa, "ÉTUDE, SYNTHÈSE ET RÉALISATIONS DE FILTRES BAW
146] POUR APPLICATIONS MOBILES," Université Bordeaux 1, Bordeaux, France, 2006.

P. Benabes, "Accurate time-domain simulation of continuous-time sigma-delta
147] modulators," *IEEE Transactions on Circuits and Systems-I: Regular Papers,* vol. 56, no. 10, pp. 2248-2258, 2009.

C.-A. Tugui and P. Benabes, "SIMECT - A tool for SImulation and Macromodel
148] Extraction of analog CT functions," 2011. [Online]. Available: www.mathworks.com/matlabcentral/fileexchange/26525..

A. A. MARIANO, D. DALLET, Y. DEVAL and J.-B. BÉGUERET, "VHDL-
149] AMS Behavioral Modeling of High-Speed Continuous-Time Delta-Sigma Modulator," in *International Workshop on ADC Modelling and Testing – IWADC 2007*, Iasi, Romania, 2007.

E. Corrales Lopez, "Analysis and Design of Bulk Acoustic Wave Filters Based on
150] Acoustically Coupled Resonators," Universitat Autonoma de Barcelona, Barcelona, Spain, 2011.

F. Pêcheux, C. Lallement and A. Vachoux, "VHDL-AMS and Verilog-AMS as
151] Alternative Hardware Description Languages for Efficient Modeling of Multi-Discipline Systems," *IEEE Transactions on CAD of Integrated Circuits and Systems,* vol. 24, no. 2, pp. 204-225, 2005.

A. Shirakawa, J.-M. Pham, P. Jarry and E. Kerherve, "Bulk Acoustic Wave
152] Coupled Resonator Filters Synthesis Methodology," in *2005 European Microwave Conference*, Paris, France , 2005.

J. Galliere, P. Papet and L. Lattore, "A 2-D VHDL-AMS Model for Disk-Shape
153] Piezoelectric Transducers," in *IEEE International Behavioral Modeling and Simulation Workshop, BMAS*, San Jose, California, 2008.

T. Ayed, C. Bernier, M. Pelissier, D. Dallet and J. Begueret, "Ultra Low Power
154] Bandpass Sampling Architectures Using Lamb Wave Filters," in *NEWCAS 2010*, Paris, 2010.

A. Arnau, T. Sogorb and Y. Jimenez, "A Continuous Motional Series Resonant
155] Frequency Monitoring Circuit and a New Method of Determining Butterworth-Van Dyke Parameters of a Crystal Microbalance in Fluid Media," *Rev. Sci. Instrum.,* vol. 71, no. 6, pp. 2563-2571, 2000.

A. N. Kostadinov, "USING VHDL – AMS IN VLSI DESIGN LABORATORY
156] EXERCISES," in *ELECTRONICS' 2005*, Sozopol, BULGARIA, 2005.

E. Christen, K. Bakalar, A. M. Dewey and E. Moser, "Analog and Mixed-Signal
157] Modeling using the VHDL-AMS Language," 36th Design Automation Conference, New Orleans, 1999.

L. C. Durgaraju, "Modeling and Simulation of Hierarchically Decomposed R-2R
158] Ladder DAC using VHDL-AMS," UNIVERSITY OF CINCINNATI, Cincinatti, Ohio, USA, 2005.

Declaration of Authorship

I herewith declare that I have produced this paper without the prohibited assistance of third parties and without making use of aids other than those specified; notions taken over directly or indirectly from other sources have been identified as such. This paper has not previously been presented in identical or similar form to any other French or foreign examination board.
The thesis work was conducted from 2010 to 2012 under the supervision of Phillipe BENABES at Department of Signals Processing and Electronic Systems (SSE), SUPELEC, France.

Gif-sur-Yvette, France

Signature:

Publications

- BENABES P., TUGUI C.-A., "A high-level modeling framework for the design and optimization of complex CT functions ", *9th IEEE International NEWCAS Conference (NEWCAS'11)*, pp. 61-64, Bordeaux, France, June 26-29, 2011.

- BENABES, P., TUGUI, C.-A., "Effective modeling of CT functions for fast simulations using MATLAB-SIMULINK and VHDL-AMS applied to Sigma-Delta architectures", *IEEE International Symposium on Circuits and Systems (ISCAS'11)*, pp. 2269-2272, Rio de Janeiro, Brazil, May 15-18, 2011.

- TUGUI C.-A., BENASSI R., APOSTOL S., BENABES P., "Efficient Optimization Methodology for CT Functions Based on a Modified Bayesian Kriging Approach", *IEEE International Conference on Electronics, Circuits and Systems 2012 (ICECS'12)*, Seville, Spain, December 09-12, 2012.

- TUGUI, C.-A., BENABES, P., "Effective Modeling of CT Functions for Fast Simulations Using SIMULINK, VHDL-AMS, VerilogA Applied to Sigma-Delta Architectures", *submitted to IEEE Transactions on Circuits and Systems-I regular papers (TCAS-I).*

- TUGUI, C.-A., BENABES, P., "Novel Analog Design Methodology Based on Bayesian Kriging Optimization and VHDL-AMS, Verilog-A, SIMULINK Macro-modeling", *submitted to Springer -Analog Integrated Circuits and Signal Processing Journal.*